CHALMERS | UNIVERSITY OF GOTHENBURG

StoryPlay

Developing a Book Game Design Tool for the iPad

Master of Science Thesis in the Master Degree Programme, Interaction Design

ANDREAS RÖNNQVIST
KRISTOFFER HOLMBERG

Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
Göteborg, Sweden, November 2011

StoryPlay
Developing a Book Game Design Tool for the iPad

ANDREAS RÖNNQVIST,
KRISTOFFER HOLMBERG,

Examiner: STAFFAN BJÖRK

Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Cover:
The StoryPlay logo. Developed for, and used in the application.

Department of Computer Science and Engineering
Göteborg, Sweden November 2011

Abstract

The smartphone industry has been established on the market over the last few years and the development of smartphone applications has become a hot industry. This thesis describes the proceedings of designing and implementing a tool, StoryPlay, for interactive book game design, developed to gain an advantage on the competitive smartphone application market.

The iPad tool is conceptualized, designed, prototyped and implemented in stages. A final concept is developed and described, together with advantages and drawbacks of using such a tool for the desired purpose.

Sammanfattning

Smartphoneindustrin har etablerats på marknaden under de senaste åren och utvecklingen av smartphoneapplikationer är idag en het industri. Den här tesen beskriver tillvägagångssättet av designen och implementeringen av ett verktyg, StoryPlay, för design av interaktiva bokspel, utvecklat för att få en fördel på den konkurrenskraftiga smartphoneapplikationsmaknaden.

IPadverktyget är konceptat, designat, prototypat och implementerad i olika steg. Ett slutgiltigt koncept är utvecklat och beskrivet, tillsammans med fördelar och nackdelar av att använda ett sådant verktyg för det önskade syftet.

Acknowledgements

We would like to thank our supervisor at Chalmers, Staffan Björk, for his input and feedback on this thesis. We would also like to thank Edward Antov and Mikael Hellqvist at Appmazed for their support during this project. Lastly, we would like to thank our families, friends, colleagues and affiliates we've met for their continuing support.

Andreas Rönnqvist and Kristoffer Holmberg, Göteborg, 9/6/11

Table of Contents

1 Introduction

To be able to understand why this masters thesis is both suitable in time and appropriate for the authors of this thesis, a little background information is needed. Smartphone application development is more on-going than ever. That combined with publishing houses wanting to publish their material on digital media makes this an interesting topic for a masters thesis.

1.1 Background

Development of applications (apps) for Apple's iPhone, iPad and iPod models have increased exponentially during the last few years. A wide range of developers, from one-man teams to entire companies, have developed apps that are available for download through the AppStore. According to Apple (Apple, 2011), there are more than 350 000 apps at the AppStore. The apps are either free or costs a fee. Due to the profit possibilities when developing for iOS, 90% of all developers connected to Appcelerator, an experienced app developing company, are interested in developing for iOS (Appcelerator, 2010). Thus it is interesting from a business standpoint to research a profitable way to develop these apps.

A developer can develop apps for different reasons. The developer might be interested in earning money from their work, this can be done either by selling the app on the AppStore or by including advertising in the app. The developer can also develop an app that is created to strengthen a company's position on the market by different means. Companies can also develop apps for their own use, which is not displayed on the public AppStore, but will be possible to download by the company's employees.

A handful of companies have started development of interactive story-telling books for the mobile market, but it is still expensive to develop for that particular market. Hence, it is interesting to investigate the possibilities for a tool that allows for a smarter design process where not only the developer may develop for several mobile platforms at once, but also can save time by reusing parts from previous projects.

In recent years, an increasing number of publishing houses have started to see the potential of digitalizing their content for a broader audience. However, this is not limited to the classic e-book genre, which is basically only scanned pages of the original book, but also more interesting concepts, such as games.

1.2 Appmazed AB

Appmazed AB is a small company that aims at developing digital entertainment, such as games, films and story driven mobile solutions for multiple platforms. The team behind Appmazed are two well qualified men. Edward Antov, who is the CEO of Appmazed, has experience from large parts of the digital entertainment industry, including film, TV and games development. Peder Kallin is the Creative Director of Appmazed. With extensive knowledge within the digital entertainment industry, he is leading the production of Appmazed's projects.

Appmazed has a business model where they offer a "games in book games" solution for book

publishing companies. The idea is to create computer software, games, that are based on books, thus book games, but involves both going through the original story from a protagonistic, or multi-protagonistic, view point and playing mini-games that either moves the story forward or are just for fun. One aspect of this is to make game playing a more intellectual stimulation activity, while keeping the fun factor intact.

1.3 Problem Description

The idea for this master thesis is to conceptualize the basics of a tool for designing interactive book games. In its finished state beyond this thesis, the tool is supposed to be used by developers at Appmazed, as well as a tool for writers and publishers at the publishing houses. In both cases, the files produced by the tool will contain a design, a framework, in a way that writers, producers and players understand. The vision for the tool is to produce a design that an in-house or out-sourced programming team may use to develop a finished version.

Today, the development of games is similar to the procedure as that of movie making. This include things like storyboarding and casting, which may be expensive, to say the least. Appmazed has a vision to create high quality book games at a low cost and wants to move away from the old ways of producing games. This new tool is going to help that cause by introducing a way for people with low or no programming skills to express their ideas to all different kinds of people throughout the development hierarchy.

The biggest difference of this tool from software like Microsoft PowerPoint or traditional methods like storyboarding is that this tool will feature a large amount of predefined features, such as mini-games and templates. The tool will be focused on giving the user freedom in creation and variation, thus aiding him in designing book games at a high speed and low cost, without sacrificing quality.

In the scope of this master thesis lies of course not all of the above, but a much more narrow perspective. We will look upon what functionality is needed in a software to support conveying design ideas to other people, both decision makers and programmers. We have a vision for this software to be useful in a book game creation environment, where it may serve as an important tool in the development process. We want to look upon how to make this software user-friendly and intuitive and will realise our result as a prototype.

This master thesis will take a look at the tool set needed for such a software, as well as the interaction, user experience and design. We wish to find out what functionality is needed to make such a tool for a book game design environment. This tool will be called StoryPlay.

1.3.1 Problem Formulation

Is it possible to develop a software that support the book game creators to convey their ideas to both decision makers at one end and programmers at the other?

2 Background

Previous work in the area includes research papers, prototypes from lab environments and commercial products. The work that was considered related was work that had to do with creativity in general, creativity on tablets, book games and other kinds of games that were close to what Appmazed had in mind for their project.

2.1 Related Work

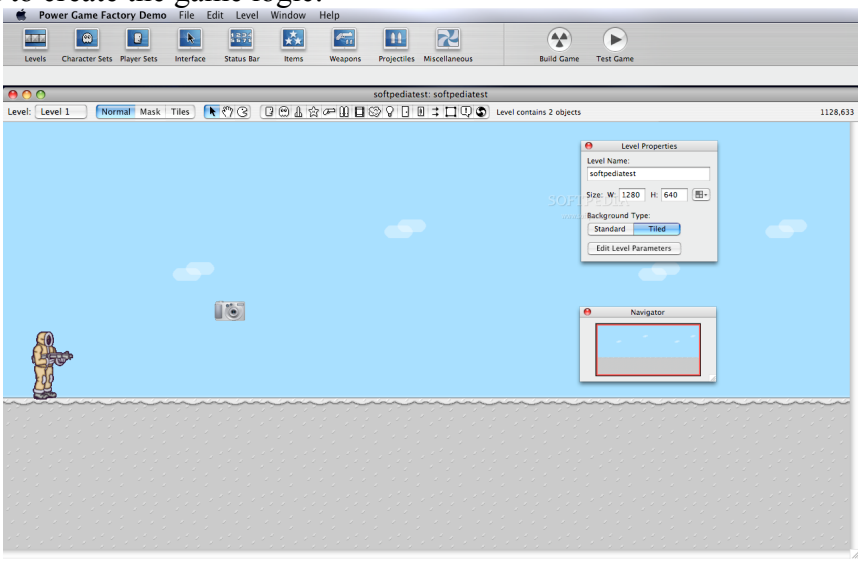
Research was conducted in order to find out how well this area had been explored before. Some related software and games that were looked into during the pre-study gave some inspiration and ideas for both StoryPlay itself but also the book games that are supposed to be developed by the software. Already on the market, there are several apps that allows for creativity within the iPad or iPhone interface. By having a look at these, we expected to get inspiration for our own project.

2.1.1 EDGE

EDGE, An Easy Design Tool of Game Event for Rapid Game Development (Cho, Kim and Yoo, 2010), is an attempt to make the design process of game development faster. The software is supposed to be usable by people without any programming skill to create sprites, game effects and game events. The user links these together and the software automatically generates C code for a preview.

2.1.2 The Games Factory 2 / Multimedia Fusion 2

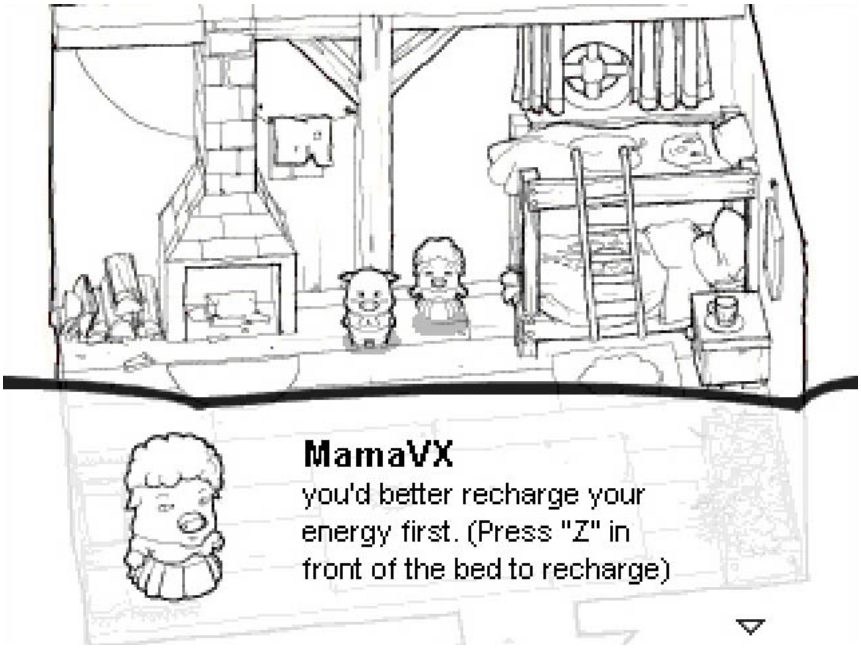
Developed by Clickteam, the software Multimedia Fusion 2 and its less advanced sister The Games Factory 2 are software for game development. The programs are developed for PC and allows the user to design games using a storyboard mode and single frame editing. They implement storyboards, made up of single frames, as the foundation. In any frame objects are placed and events are added to create the game logic.



(Picture 2.1) Screenshot of The Games Factory.

2.1.3 ButaVX: Justice Fighter

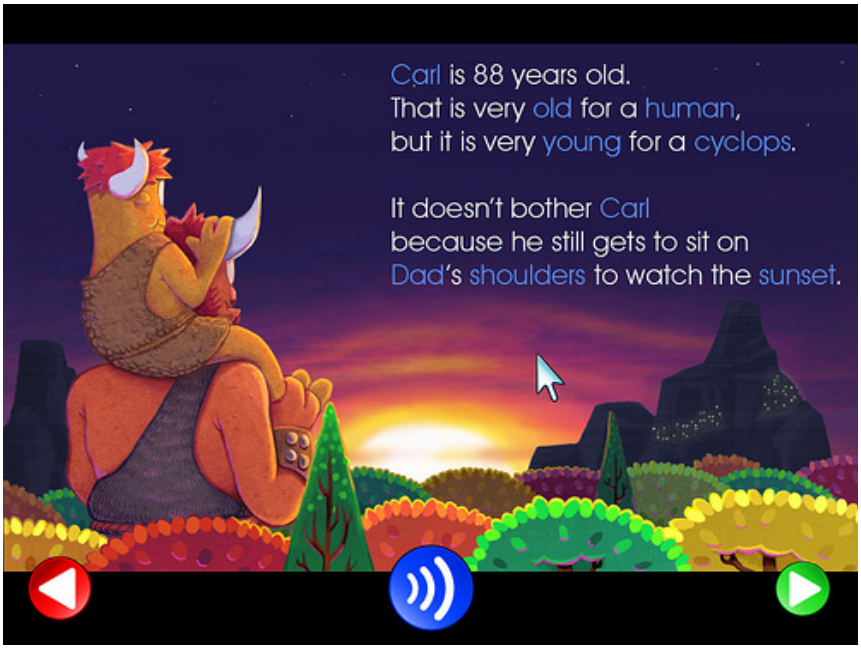
ButaVX is developed by the small team Nekomura Games. It is a short and cute RPG game, clearly inspired by the popular Final Fantasy series. What's interesting about this game from this masters thesis standpoint is the way the story unfolds. The story is told in three different ways throughout the game, through cut scenes, through the actual game play (moving around and interacting with NPCs, Non-Player Character) and through a nifty in-game picture diary that summarizes what has happened so far. The combination of these enables the player to take part of the story without doing a lot of reading and is inspiring.



(Picture 2.2) Screenshot of ButaVX.

2.1.4 What's Bothering Carl

Developed by Story Fort LLC, this game is a story book that tells the story of the young cyclops Carl. The games comes with a voice over and pictures drawn in a cute style. One of the key elements of the game is the ability to click some highlighted words and thus get to see a picture of it and have the narrator pronounce it.



(Picture 2.3) Screenshot of What's Bothering Carl.

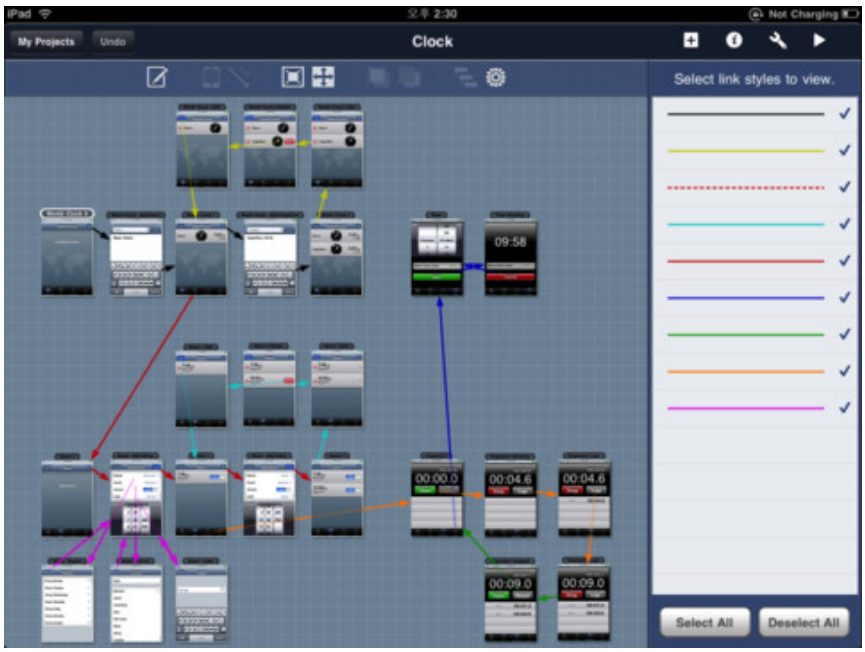
2.1.5 SketchFlow

SketchFlow is a program in the Microsoft Expression Blend 3 suite. The program is used to create rough sketches that are fully functional. The tool may be used instead of, or together with, paper prototypes for software user interfaces. The tool is very effective in user tests, where the user gets feedback from the button he is pressing at once, instead of paper sheets need to be changed.

SketchFlow have a large library with buttons, scroll bars, pictures and different clickables of all sorts to make it easy for the user to use it. The graphics contained in this library does not correspond to how the software should look like in a finished state, but it is rather sketchy looking. The sketchy graphics helps the user who might be testing the software to focus on the experience of where the buttons are placed and what they do, instead of focusing on the graphics and looks of the software.

2.1.6 Blueprint

A linked screenshot maker that allows the user to construct screenshots out of common iPad/iPhone objects such as labels, text boxes, buttons, etc. These screenshots may be linked through actions on buttons to create a demo.



(Picture 2.4) Screenshot of Blueprint.

2.1.7 Keynote

Keynote for iPad is a PowerPoint presentation program. Users are able to create the presentations as well as showcase the presentations with the software. The software has a few standard tools to add and manipulate pictures and text. It also has some built-in tools to make the page transitions. It has a clean user interface with only a few icons; the rest of the tools is accessed via pop-ups when pressing the icons.

2.1.8 GO!GO! Paint

Developed by takami corporation, GO!GO! Paint is a small program that allows for quick drawing on the iPhone. It features 10 different colors for the brush, a slider to decide the size of the brush, an eraser and the possibility to save pictures to the photo roll. Unfortunately, it redundantly makes use of the “shake” gesture to erase the entire document, without any possibility to confirm or undo.

2.2 JerryMaya’s Detective Agency - The Chameleon Strikes Back

The Chameleon Strikes Back is the second computer game produced with the JerryMaya theme, and it is based on the movie with the same name. The game is a point-and-click adventure game, where the player is helping Jerry and Maya with the numerous tasks the citizens of Valleby requests help with. A big part of the game is piecing together different clues hidden in the game’s environment, to be able to solve the different mysteries.

During the masters thesis, Appmazed was in discussions about having a collaboration with the founders of JerryMaya. Because of this, it was decided that JerryMaya should be used as a theme for the prototypes developed during the masters thesis. The reason behind this decision came of the insight of not having to produce all graphical material for the prototypes from scratch, but rather used existing JerryMaya graphics.

3 Theory

This chapter describes the theory researched for this master thesis. It covers general design principles as well as specific design principles for hand-held devices. More specifically the design of iPad applications.

3.1 Usability Principles

Nielsen (2010) has defined an easy-to-understand list of properties for usabilities. These points are definitions that should be kept in mind for all designers and developers, as following them will most likely lead to a better product being made.

The five components defined are:

- "Learnability - How easy is it for users to accomplish basic tasks the first time they encounter the design?"
- "Efficiency - Once users have learned the design, how quickly can they perform tasks?"
- "Memorability - When users return to the design after a period of not using it, how easily can they re-establish proficiency?"
- "Errors - How many errors do users make, how severe are these errors, and how easily can they recover from the errors?"
- "Satisfaction - How pleasant is it to use the design?"

These points are important to keep in mind during all parts of the design process. According to Nielsen, spending about 10% of the project budget on improving usability should result in about a doubled profit.

3.2 Interaction Design Principles

The usability principles are used to gauge an already existing prototype's usability, which may lead to important input for the next iteration. But for the design process, there are several additional principles to follow, which focus on keeping the designer on the right track.

Norman and Nielsen (2010) define these principles as follow:

- "Visibility - All actions the user may take should be visible to him."
- "Feedback - Whenever the user performs an action, he should get a response that confirms what action he took."
- "Consistency - Throughout the applications, the feeling and way to perform actions should

be the same."

- "Non-destructive operations - The user should always be able to recover from an error, if he can’t, he should be notified before taking the action."
- "Discoverability - All actions the user may take should be found in menus or such."
- "Scalability: An operation should be performable and work the same way, no matter the screen size."
- "Reliability: Operations should work. Nothing that feels random should happen."

3.3 Affordances and Design

Norman (1999) explains perceived affordances in a simple manner and gives four principles for how to design to allow new users understand how to use an unfamiliar system. These principles are:

- "Follow conventional usage, both in the choice of images and the allowable interactions."
- "Use words to describe the desired action (e.g. "click here" or use labels in front of perceived objects)."
- "Use metaphors."
- "Follow a coherent conceptual model so that once part of the interface is learned, the same principles apply to other parts."

Perceived affordances is all about how a user understands what actions he takes will result in a desirable reaction from the system. Also, as important is "perceived non-affordances", i.e. the user understands that an action will not result in a reaction from the system.

4 Methodology

Methodology research includes methods for the design phase, the implementation phase and the usability testing phase. To be able to get off on a good start for the project, we needed to research established methods of design, implementation and testing. We were already aware of many different methods, but still felt the need to freshen up both our memory and knowledge by reading up on methods new to us and relate that to our already existing knowledge. Thus, this chapter both includes methods completely new to us, but also methods that we had heard of before.

4.1 Methods for Design

To be able to come up with a great design of a product, the designers need to use different methods. It might sound easy to design a product, but the reality is that designers need methods that inspires him to create good looking and user friendly designs. There are a number of methods a designer can use, but what they have in common is that they make the designer think outside the box, makes him come up with solutions that he wouldn't have come up with on his own.

4.1.1 Speed Sketching

Sketching is a widely used method in the developing process. It is used to demonstrate ideas developed in a quick and easy manner. Sketching is also a way for the designer to reason with himself. According to Wentzel (2009), “When a designer sketches this is done based on current knowledge, but once the sketch is on paper the designer will get new knowledge by reading and interpreting what has just been created.”

Speed sketching takes into account that sketching is a quick way to express ideas, and thus sets a limit on the time available for each sketch. Each sketching phase also has a theme, established by a process leader. Wentzel (2009) used stories from a probe study as the theme of each phase, but there should be no constrain in using focus points in the design object, or random keywords as themes for the different phases.

4.1.2 Brainstorming

Brainstorming is a general name for a group of methods that centers around the activity of spawning many new ideas in a short amount of time. There are different ways to do this, but all of them focuses on stimulating new thinking, be it by setting a theme to the thinking process or force participators to share all ideas, good or bad.

4.1.3 Conceptual Tool for Applying Intuitive Interaction

The principles for applying intuitive interaction developed by Blackler, Popovic and Mahar (2006) are based on several years of research of how humans experience new interfaces. Their research has concluded that following three principles when designing, use familiar symbols from the same domain, use familiar symbols and metaphors from other domains and make use of redundancy and keep internal consistency, will lead to a more intuitive interaction.

The tool developed includes guidelines for the designer in a simple manner. It covers the process from researching the user group and it's familiarities, to designing each feature's function, appearance and location. Experimental studies using this method has revealed that a lot of work during the user studies before the designing starts is required in order for the principles to be successful, but will most likely lead to a more intuitive design.

4.1.4 Five Sketches

The Five Sketches method (Five Sketches, 2009) is a relatively new method for coming up with divergent solutions to a problem set. The participants begin by individually drawing five solutions to an identified problem. These solutions are then shared and combined, but not analyzed. From this set of solutions, each participant then continues to draw one good solution each, which then are critiqued and analyzed. One strong point about this exercise is to quickly come up with a divergent solution set. Another is that as the ideas are shared and combined early on, nobody can or has to claim ownership of an idea, which reduces tendency for defending one's "own" idea and opens up for better critique.

4.2 Methods for Implementation

Implementation methods include different ways to develop software. There are a ton of these, mentioned here are a few that are relatively common. Apart from these the Hermes, the V-Model and the Waterfall models are fairly common.

4.2.1 Extreme Programming

Extreme Programming is a very popular Agile Process which development started in the mid-'90s. The key of Extreme Programming is that its focus is on customer satisfaction instead of pumping in functionality into the software. Extreme Programming emphasize discussion with the customer, and changing the software to the customers will.

According to a website about Extreme Programming (Wells, 2009) "Extreme Programming improves a software project in five essential ways; communication, simplicity, feedback, respect, and courage." In short, Extreme Programming accent communication with the costumers and programmer colleagues as well as keeping a clean and simple design. It also emphasize getting feedback by testing the software substantially during the whole project and by delivering the software to the customer at an early stage to implement the changes they have suggested. By gaining respect for each success along the way, the team gain the courage needed to handle every new assignment.

4.2.2 Scrum

Scrum is a software development process, often used in Agile development processes. Scrum has its core in two types of cycles; daily scrums and sprints. A sprint is a period of software development. Prior to each sprint there is a sprint planning meeting. According to Agarwal (2011), "a 'sprint planning meeting' is held to define the list of features to be included in the sprint, their estimated time for completion and feasibility". A sprint is typically about 30 days long.

The daily scrums are short meetings held each day. Discussed on these meetings are the daily work on the project; what was done yesterday and what is going to be done today. The time limit on the meetings are 15 minutes, and if there has arisen any problem during the meeting, there will be a new meeting where only the concerned team-members participate.

4.2.3 Agile

Agile software development is the main frame for lots of other software development processes. Processes like Extreme Programming and Scrum uses Agile as a base for their processes.

In the manifesto for Agile Software Development, the Agile Alliance has enacted the Agile principles. At their homepage (Agilemanifesto, 2001) they summarize the principles with four bullet points:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

4.3 Methods for Usability Testing

To be able to create a good looking and user friendly product, the product has to be rigorously usability tested. There are several methods to use when conducting usability testing on products, all of them fitted for different kind of products and different stages in the development phase. By user testing the products in the different stages of the development phase the development team ensures that the product maintain a high degree of quality and they are also able to change the things on the product that the usability testing has showed doesn't cut it. This is completely essential in the world we live in today. If a product is released without the high design quality the market is used to, it just won't sell.

4.3.1 Video Filming the Use of a Paper Prototype

The traditional use of a paper prototype is to use it in user tests, but it also may serve as a powerful tool in presentations and demonstration. By video filming the use of the paper prototype and then applying editing, it will look like the prototype actually works. A functional video like this may demonstrate the idea in an even better way than the paper prototype itself did.

4.3.2 State Transition Diagram

A state transition diagram is a method to sketch the interaction of a software. Each state of the software is sketched and below each of these a diagram explaining possible ways to get to another state and the state's position in the whole picture is drawn. With this combination, it is possible to

follow the logic of the software, without having to actually make a digital prototype of it. It is easy to see how the software's logic looks like and navigate through it.

4.3.3 User Observation

User observation means that the observer watches the user while he is using the interface. There are two kinds of user observation methods, *Unobtrusive observation* and *Obtrusive observation*. In unobtrusive observation the observer doesn't interact with the user while he is doing the test, and in obtrusive observation the observer does interact with the user. The interaction may be anything from explaining the system, asking questions about the user's behaviour to engaging in a discussion with the user.

Neither method is right or wrong, but each uncovers different things about the software. InteractionArchitect.com writes that "With unobtrusive observation you learn whether the system is easy to use. [...] With obtrusive observation you learn more about the usefulness and acceptance of the system." (D'Hertefelt, 1999).

4.3.4 Interviews and Questionnaires

Interviews and questionnaires are easy ways to elicit information about the product. Interviews takes longer time per interviewee, but the result is better than with questionnaires. To elicit most valuable information from interviews, the interviewer should have created a manuscript in beforehand. This is important to be able to ask all interviewees the same questions, and not forget anything. Interviews may be recorded with a tape recorder in order to listen through the interview later.

Questionnaires takes some time and some competence to compile properly, but when that is done a big part of the work is finished. After sending the questionnaires to suitable people and receiving it back from them, putting the result together and interpreting it is all that is left.

5 Planning

The project was planned to be carried out in different phases, where each phase would consist of specific tasks. While the different phases will be further explained in the time plan, there is a general time line of the project which starts with background research, followed by conceptual design and prototyping, and ends with a demonstration in the form of a mock-up.

There are lots of tools on today's market that are similar to what this project was aiming at. These tools needed to be studied in order to get a general picture of the design. In the concept design phase, suited functionality was going to be elicited from the tools that had been studied, and implemented in the design of the software.

Prior to the work on StoryPlay could start, a game model to aim at needed to be chosen and designed. This game model needed to be specific enough to have an impact on StoryPlay's interface, but also generic enough to allow freedom in design for the user.

In the prototyping phase, the work was planned in short sprints. The iterations consisted of a deadline and a new or improved prototype developed. Extreme Programming was intended to be used in each sprint, in order to fast and effectively find and solve design issues.

At the end of the master's thesis, a final mock-up with limited functionality was going to be demonstrated.

5.1 Delimitations

The project itself was limited to 20 weeks of full time work carried out by two students. Further more, lots of work time was to be used for planning, administrative work, project report writing and presentation preparation. The remaining 13 weeks was to be dedicated for the goal of the project. Deciding on what functionality we would need in such a software, designing the interface of the software and implementing the design to create a functioning prototype.

The result of the project would to a big extent focus on usability for the user. With an end user group which has little to no programming experience, it would be critical to keep the software at an intuitive and self explaining level, without for that matter make too many drawbacks on the functionality.

The platform we were going to develop for has its own special properties, for better and for worse. The project would have to make use of the platform's qualities, and circumvent the obstacles to be able to create a well-developed software.

A tool set would be needed to be able to focus the work on important functionality. Even though the design phase of the project was going to house numerous ideas, concepts and sketches, there would be very limited functionality in the implemented software, due to time limitations. However, it was important to remember that the ideas conveyed in the design phase would have an important role in future work.

We believed that a tool to create a sketch for a generic game was a bit too complicated for a mobile

platform, thus we would develop StoryPlay with a specific interaction model in mind. This interaction model would be in line with the work that Appmazed was doing at the time, since we would be able to get maximum support that way, in the form of guidance and materials.

As the finished product was to be aimed towards the iPad, and the tool delivered in this thesis work was expected to be relatively simple, the possible interactions for the end-user to produce had to be narrowed down. It was never intended for the end-user to produce 3D FPS games with advanced controls through StoryPlay, but rather interactions with simple controlling, such as point-and-click.

5.2 Expected Result

We expected to develop a software prototype for designing interactive book games on the iPad. We expected to be able do simple presentations of an idea from scratch, using only graphics from external sources. These presentations, or walkthroughs, of the idea were not to be playable versions, but rather convey a concept.

The interface for StoryPlay was intended to be intuitive and user-friendly.

StoryPlay was supposed to be used on foot by people from the publishing houses and thus we were to put some focus on making it powerful in showcasing an idea for a book game. This means that in addition to using StoryPlay for being able to put together ideas for book games, it also needed features for sales pitching.

5.3 Time Plan

This was the time plan used during the masters thesis:

Week 1-2:
The first few weeks will be needed to set up the project as a whole. This includes finishing up the project description, setting up the work space and doing other administrative work, such as plan meetings with a supervisor. This is also the time to do a more detailed plan of the project.

Week 3-4:
After the initial setup, the research phase begins. This includes finding related material and works, but also freshen up the necessary knowledge for carrying out the project. This is where the first few ideas are spawned. The first set of tools needed to be supported by StoryPlay will need to be defined, and using that will help in sketching a GUI.

Week 5:
At this stage, a paper prototype will give necessary feedback on many aspects of the work flow. The paper prototype will be constructed based on specifications given by Appmazed and research from earlier.

Week 6:
The evaluation of the first prototype will be the basis for the coming work and as this stage unveils, the entire project might need to be redefined quite a bit.

Week 7-10:
This is a learning phase for the MacOS environment and Objective-C. In order to go digital, we need to broaden our knowledge about app development. A few weeks will be spent improving our skills in this area, as well as improving the GUI, based on the evaluation of the paper prototype.

Week 11:
After having developed a first digital prototype, it needs to be evaluated and from that point a plan for the remaining work can be devised. This is where we will decide upon additional functionalities as well as in what ways the user experience can be improved.

Week 12-13:
At this stage a demo is going to be implemented, to showcase what kind of products can be expected from StoryPlay. It will serve as a basis for a first template for StoryPlay, which will improve the usability of it.

Week 14-15:
Additional time for writing the project report is reserved here.

Week 16-17:
At this last stage, a final concept will be developed and a mock-up of this will be produced.

Week 18-20:
The last few weeks are dedicated for report writing and presentation preparations. Also, final adjustments to the final concept are made.

5.4 Delivering the Result

For delivering the result, we were to present three things:

- The implementation of the application, the Objective-C source code
- A Design Document, describing the application for other developers so that they can continue the work. This will be delivered to the developers at Appmazed.
- A final concept and a mock-up of this.

5.5 Objective-C Frameworks

The choice of programming language fell on Objective-C naturally as the target platform for StoryPlay is the iPad. As we were new to Objective-C when this project started, we decided to use a framework called Cocos2D to easier get a prototype going.

Cocos2D is a framework that allows for easier image and sound management. It's specialized for use in game development, but, due to it's straight-through API, it's a good starting point for novices at Objective-C.

5.6 The Users

The intended target users for the application were not programmers. The attention was rather focused on writers, publishers and concept artists. This category of people do not usually have professional knowledge of programming, even though they have at least basic computer skills and are able to use word processors and drawing tools. They are familiar with the iPad, but maybe not using creative applications. In order to develop the application to be a satisfactional experience for this user group, we had to specify the needs and basic knowledge of the members of it.

5.7 Context of Pitching

The main use of the application was to visualize ideas and concepts for a book game design to decision makers and programmers. This put some requirements on the application to have a presentation mode where the creator may demonstrate in such a way that the general concept is conveyed to his audience. This includes the ability to cheat or skip in order to cover all functionality.

5.8 JerryMaya

A focus on JerryMaya's Detective Agency was to be used in this thesis. This was a necessity in order to focus the thesis work.

JerryMaya's Detective Agency is a children's book series for kids ages 6-9. It has positioned itself numerous times on the top ten list at a Swedish children's book award called Bokjuryn (Bokjuryn, 2009). This far, a total of 20 books have been written by Martin Widmark and illustrated by Helena Willis (Bonnier Carlsen, 2010). The story is about the two friends, Jerry and Maya, who lives in the fictional town of Valleby. They are very fond of mysteries and are therefore running a detective agency together.

In each book they are coming across a mystery which needs to be solved (e.g. hotel impostors or circus thieves). They are solving the mysteries with cleverness and ingenuity, to the joy of the citizens in Valleby.

In addition to the book series, there has been two computer games, one movie, one card game and one season in the annual Swedish television series called The Christmas Calendar produced.

6 Prestudy

In the theory chapter of this thesis, the authors went through lots of articles, principles and methods. To be able to do this masters thesis the authors had to elicit information from the theory chapter that were useful for this thesis. In this chapter the authors reproduce that information and adapt it to make use of it when designing and implementing their own solution.

6.1 Design Method

Simultaneously to designing, we developed our own way of working for this project. By relying on established principles and conventions, we decided upon the following policy regarding design:

- The design should be as intuitive as possible, we want the user to recognize his environment even the first time he uses the application.
- All possible actions should be visible, if not directly then through a menu.
- For each gesture implemented (except for tap), there should exist a visible button somewhere with the same function.
- Critically analyze Apple's conventions for the iPad and make use of those we find reliant, intuitive and user-friendly.
- The user should not forget how to use the program within a reasonable time period.
- The user should be able to undo any action while using the program. If not, he should need to confirm his action before completing it.
- The user should be able to learn most of the program without consulting help files or tutorials.

The design method carried out in this project is similar to *Speed Sketching*, but has it's own unique features. First of all, a list of focus points had to be produced. Items in the list was extracted by brainstorming. Once the list of focus points was set, the designers began sketching.

Each focus point in the list was to be handled separately, and one sketch was to be developed for each focus point and each designer. For each sketch, the focus point was the main point of interest, and the sketched interface should be all about that particular feature.

When all of the focus points had been gone through, the designers sat down to explain, compare and discuss their sketches. The designers elicited functionality from the sketches that are to be implemented in some manner. Based on the elicited functions and previous sketching, the designers began on a new sketching phase. They sat down together and used *"Extreme Sketching"* to develop a few new sketches, based on the functions that should be implemented. *"Extreme Sketching"* is our own name for both designers sketching together on one piece of paper in iterations, where each

iteration results in a complete sketch.

6.2 Designing for the iPad

Designers can't expect to design iPad applications in the same way that an iPhone application is designed. The iPad is a new tool, which require special treatment.

A study by Nielsen (2010) shows that even though websites work rather well on the iPad (in contrast to the iPhone), it suffers from something called *read-tap asymmetry*. Read-tap asymmetry means that when the text is just big enough to read, it is too small to touch (which is needed in links etc.).

According to the study, the iPad has a really nice looking *etched-screen aesthetic*, which means that designers move away from "visual distractions or nerdy buttons" (Nielsen, 2010). The drawback for the users is quite clear, they don't know where they can click. And even worse, once they find out how to navigate in one application, they can't transfer that knowledge to the next application. According to Nielsen, a simple task as touching a picture in different applications could produce any of the following five results:

- "Nothing happens."
- "Enlarging the picture."
- "Hyper-linking to a more detailed page about that item."
- "Flipping the image to reveal additional pictures in the same place (metaphorically, these new pictures are "on the back side" of the original picture)."
- "Popping up a set of navigation choices."

Using as many standards, conventions and perceived affordances as possible when designing applications and interacting with the applications will make the usability and joy for the users a great deal better.

Nielsen (2010) compiled a few guide points to follow, when designing a user-friendly interface.

- Add dimensionality and better define individual interactive areas to increase discoverability through perceived affordances of what users can do where.
- To achieve these interactive benefits, loosen up the etched-glass aesthetic. Going beyond the flatland of iPad's first-generation apps might create slightly less attractive screens, but designers can retain most of the good looks by making the GUI cues more subtle than the heavy-handed visuals used in the Macintosh-to-Windows-7 progression of GUI styles.
- Abandon the hope of value-add through weirdness. Better to use consistent interaction techniques that empower users to focus on your content instead of wondering how to get it.

- Support standard navigation, including a Back feature, search, clickable headlines, and a homepage for most apps.

6.2.1 Usability of iPad Apps and Websites

According to Usability of iPad Apps and Websites (Budiu, et al., 2011), many iPad apps lack in usability and are contradicting each other in the ways users interact with them. The ways computers are interacted with were not developed over a night and by that logic we should think the same about apps. Gestures, for example, are far from being well-established, especially in the way they are visualized.

That being said, Apple has a lot of guidelines for developing apps, but does not enforce it on the programmer. As they shouldn't. By not following the guidelines, even better usability may emerge, but that is rarely the case. It results in apps' interactions varying a lot from each other and forcing the user to remember all kinds of different interactions, some being unique to a single app.

6.2.2 Principles for Designing for the iPad

Unique principles were developed for this masters thesis by studying those by Norman and Nielsen and consulting the Apple iOS Human Interface Guidelines:

- Discoveraibility: Every function should be found within menus or something like that, not relying on invisible gestures or the like.

- Redundancy: For each gesture or hidden feature, for example pinching or swiping, there should be an equivalent visible button that performs the same action. This allows a novice user to find the feature he seeks, but also allows an advance user to work fast by using gestures.

- Learning Curve: The software should be easy and intuitive, thus having a low learning curve.

- Support standard navigation: Have Undo buttons, back buttons, tabbed navigation, menus.

- Make the user feel in control: Always let the user initiate an action, but guide him through the experience.

- Don't make the user feel lost: Always visually show the user where in the program and process he is at the moment.

- No cluttering: Make use of the iPad's large screen and use tabbed navigation and menus to categorize and hide features.

- Use a “wizard” for certain functions: Guide the user through certain complex features.

- When the user has to use a gesture, make it obvious to him how to do it.
- Use metaphors (desktop, bench...)

6.3 E-books

We found it necessary to analyze different kind of games that are available today and try to find elements that are suitable for e-books. We did this by breaking down the games into isolated game elements, which in turn can be combined in different ways to create unique experiences:

- **Plain:** A digital version of a regular book. Page by page of text.
- **Voice-Over:** An audio book; a digital version of a regular book, accompanied by an audio track reading the text out loud.
- **Effects:** Animations, sounds and visual effects throughout the story.
- **Simple Interactions:** Added visual effects and sounds that activates through user input.
- **Story-changing Choices:** Choices made within the story directly or indirectly changes it.
- **Built-in Activity:** Within the book there are more complex interactions, games and puzzles.
- **Geographical World:** A traversable world is available.

These points can be combined to create different kinds of e-books and thus a different experience.

6.4 Content Handling

In order for the user to be creative on his iPad, he needs access to graphics and documents from his computer. This section covers possible ways to upload content from a computer to the iPad.

- **File Sharing**
  The probably most comfortable way for a user to upload files to his iPad is through File Sharing in iTunes. This is only possible if the app has support for it, but will enable the user to freely control the content the app has access to. The downside of this method is that the computer needs to have iTunes 9.1 or later installed.

- **Sync with Account on Server**
  Another way to upload files to the iPad is to keep it synced with a folder on a remote server, much like Dropbox currently do. The downsides of this approach are many, for example the need of server space, accounts, connection to the internet and a lack of security.

- **Use iPad as FTP and Upload from WiFi**
  A third way is to set the iPad into a FTP mode and use it as a FTP server on the network. This allows for uploading files onto it from computers on the network. The downside to this

method is the configuration needed on the network, as well as lack of security.

- **HTTP Download**
  A forth and last way of getting contents into the iPad is to enter a URL while inside the application and thus downloading a file into it. This method lacks severely in security and is cumbersome after only a short while.

When discussing content handling, it is important to keep the intended business model in mind. In a situation where a user would use StoryPlay for his personal use, entirely on his own, using iTunes would probably be the most convenient solution. The user would only need to sync up his own work and use his own contents. No outside party would need to be shared his work.

Now, this isn't the situation for Appmazed. Appmazed intends to offer to be part of the development process during all stages. Also, it is likely that more than one person will work on the same project at any time, thus data needs to be synced between several devices. The only way this can be done conveniently is to keep a server online with that data, which can be accessed by all authorized devices. Whenever a new customer is taken in and a new project starts, Appmazed will set up a shared space on the server. Both the customer and Appmazed will have access to this server space from within the StoryPlay application on both iPads and workstations. The process of syncing and keeping backups will preferably be done seamlessly, at least from the customer's point of view.

One downside of keeping data stored on a server is that servers can be hacked and information thus compromised. Also, laptops and iPads can be stolen and data thus purged. This will be important to keep in mind when setting up the servers and informing the customers about how to minimize the risks.

6.5 Related Work

From the background research, we could extract both good and bad points, which ended up affecting and inspiring our design choices. In this section we will briefly cover how we ended up with the information we did and what conclusions we could draw from it.

6.5.1 EDGE

The EDGE paper was found when we were looking for game design tools. We found it interesting since it demonstrated one solution to a similar problem as ours. It affected our project in an inspiring way as it gave credibility to our approach.

6.5.2 The Games Factory 2 / Multimedia Fusion 2

These two pieces of software were previously known to us, but as it had been a few years since we last used it we needed to take another look at it. The one point really outstanding for these software programs is that they actually produce something executable. The user has full control of everything from events to timers, from sprites to animations. Our conclusion from having a look at them was that an approach similar to theirs would be too complex for our relatively short project. It made us take another look at our goals and take on a more realistic approach.

6.5.3 ButaVX: Justice Fighter

ButaVX actually came up by chance when we randomly searched for interesting games that had story-telling as a major component. This game gave us inspiration for a possible game genre for StoryPlay to produce. It seemed relatively simple, but even so it was immensely fun to play.

6.5.4 What's Bothering Carl

This game came up when looking for simple book games with interaction. One major fun aspect of this game is when the user may interact somewhere on each page and see a short animation of what was just told in the story. This gave us an idea of what a book game could look like, even though it matched what we already believed it would look like. The concept of this game is one of those that was taken into consideration as a potential product to be created through StoryPlay.

6.5.5 SketchFlow

For this masters thesis, SketchFlow was interesting to look at since the outcome of the user's input created a working prototype. It gave us an idea of what tools would be needed for StoryPlay and also a hunch of the complexity of such a software.

6.5.6 Blueprint

Blueprint was interesting as it was implemented on the iOS devices and also produced something interactable. One good point of it was that it was really simple to use and that inspired us in the design phase.

6.5.7 Keynote

For this master thesis, it was interesting to research the user interaction of such a creative commercial iPad application. In our experience, many Apple applications are not that intuitive to use, but grows on the user and are quite effective to use when learned. Keynote is not an exception. Some of its features are hidden, but when the user has learned about them, they are actually pretty well-placed. Still, we aimed for not hiding any functionality from the user, as we find it important to keep the interface intuitive.

6.5.8 GO!GO! Paint

This app has a terrible interaction that clears the current painting when the device was shaken. The majority of its tools are hidden and there's no Undo option. This acted as a daunting example of how not to design an interface.

7 Concept Development

The concept development phase is the largest part of this masters thesis. In the concept development phase the authors do everything from extracting a tool set needed in the software, creating paper prototypes used for user tests, to develop the software and evaluate it.

7.1 Planning

The time before the actual start of the project was filled with idea spawning and talking back and forth with our supervisors at Chalmers and Appmazed. We wanted to find some common ground, something that would be both of academic value and of value for the company.

When we eventually had narrowed it down to a tool for book game development for publishers, we could move forward and start digging into that problem area. Already at this point, some issues had come up. How can the book game development be made pedagogical? Given that many book games are to be made from this tool, how can the diverseness between them be guaranteed? Also, should the tool be more of a presentation tool, or should it output some actual code?

An early issue that arose was that of platforms. Should the tool be developed for a computer or for a tablet? As the iPad is such an exciting platform to work on, we decided to aim to develop for it. Apart from the excitement, the iPad was a natural choice as it is a great tool to use for presentations, it can be used during travel and the book games that will be developed are supposed to work on it.

Having got access to our office, we started researching previous work and related software. We tried to look for software that did something similar that we wanted to achieve, but also for games that fit the desired model.

We were at this point allowed to use JerryMaya as content for our prototypes in the future, which would spare us a lot of time not having to make our own content, to the same extent at least. We started to get a grasp on in what ways the book games made should look and feel like. The book games should consist of two parts, one book-reading part and one game-playing part. These two parts should co-exist and especially the book-part should give clues when playing the game-part.

We specified the project loosely in these categories:

- Mobility: The tool is supposed to be used on-foot by the user.
- Interactions: The user has access to predefined modules, such as games and puzzles.
- Storyline: The player should be guided in the game-part from the book-part.
- Usability: The tool should be user-friendly.
- User Background: The user most likely has no programming background.
- Simplicity: The tool implements a what-you-see-is-what-you-get philosophy.

We decided upon a simple first tool set that the user might need to have access to when working with the tool. This tool set was based on the scenario that the user would need to build scenes and link these in order to present an idea. This was a rather big simplification of the tool, but we wanted to try out the work flow of the user, using a paper prototype. As our idea of how the final book game made from the tool would look like, but not how the user would actually make it, we wanted to try out some ideas. The first tool set included simple interactions such as adding and moving objects on the scene, resizing objects, removing objects from the scene, define what scene a tap on the object would lead to and creating new, empty scenes.

We sketched on few ways of implementing this tool set and decided upon two which would be prototyped.

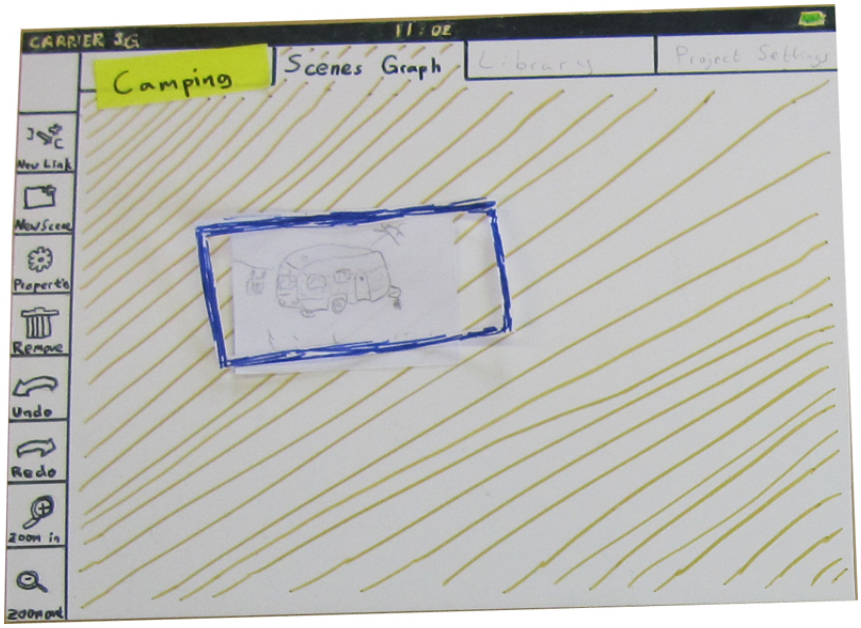
7.2 Paper Prototype

When we were designing the paper prototype we decided to keep it simple. Implementing the basic tool set we had compiled. Key points for our prototype was to keep the interface simple, user friendly and intuitive. One of the most suitable game genres for the iPad is the “Point-And-Tap” genre, where basically the only user inputs are taps. As long as the user gets quick and accurate feedback on his actions, the lack of tangibility is not critical. With that said, the target game we had in mind when designing the first few sketches was that of a simple “Point-And-Tap” game where tapping somewhere lead to something happening on the scene or a scene change.

We needed views, scenes, tools, objects, menus and a way for the user to get from nothing to a working flow of the game. The general idea we had was for the user to go through the building process in different steps, in each step creating a specific part of the book game. We created different tabs that accomplished this. In picture 7.1, the tabs can be seen at the top (called "Stuga", "Scenes Graph", "Library" and "Project Settings").

(Picture 7.1) Paper prototype with focus on the first tab.

Our thought was that the user should start in the first tab, creating every scene that the user wanted to be in the book game. To the left in picture 7.1 is the toolbar. This toolbar is specific to this step of the process, including tools like "Move Object", "Hide Object" and "Object Properties". To the right in picture 7.1 is the object library. By pressing or dragging "Objects" a box with different objects added to the projects pops out. These objects are easily added to the workspace by dragging.



(Picture 7.2) Paper prototype with focus on the second tab.

When the user has added and created all the scenes he would like to use in his book game, he continues to the second tab, as seen picture 7.2. While doing so there is a visual change at the top, indicating that the scene graph now is in focus. The toolbar to the left changes the tools to tools needed in this tab. What the user does in the scene graph tab is to connect all the different scenes that was created in the first tab, with each other. This is done by dragging links from one scene to another. By doing so a pop up will appear asking the user what object in the first scene triggers the jump to the second scene.

7.2.1 User Testing of the Paper Prototype

The creation of the paper prototype had many purposes. First of all, we wanted to try out our concept, see the flaws and strengths of it. Another thing we wanted to do was to do user tests the concept. The user testing was done on people from Appmazed, but without specific knowledge in our project. The method used was *User Observation*, where the user sat down in front of our paper prototype. We gave the user a list of tasks he was going to perform, without any knowledge of the specific functions of the prototype. The user described his actions and what he experienced throughout the user test. We, as observers, never hinted on what to do, but we swapped items on the paper prototype as would have happened if it was a digital prototype. Hence the user understood that his actions made the prototype do things.

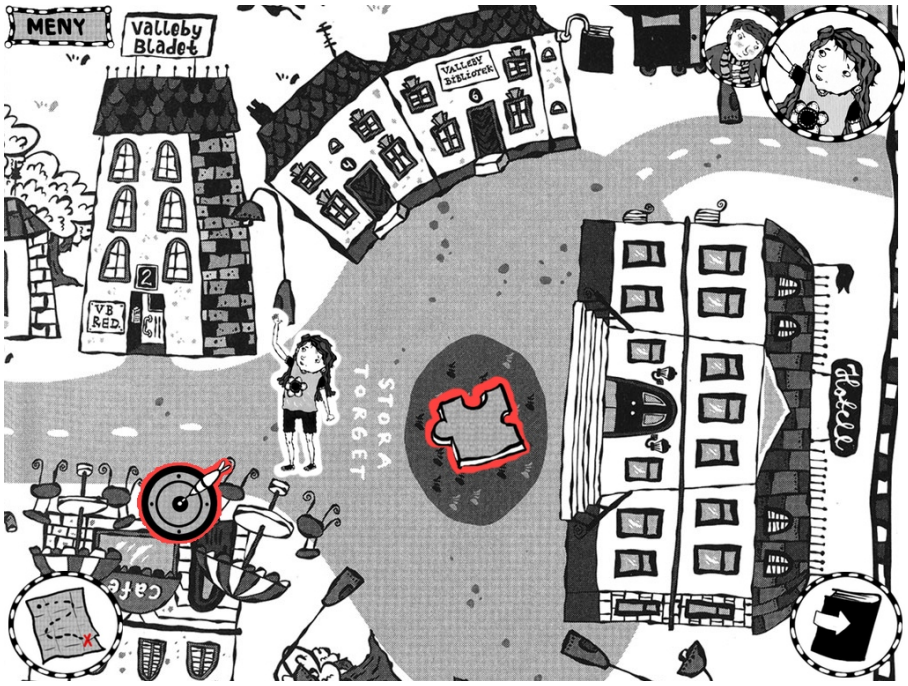
The result of the user testing was positive. We understood what the user found easy to do, and what he was unable to do. These results was taken into consideration when creating the first digital mock-up

7.3 Digital JerryMaya Book Game Mock-up

At this point in time Appmazed had booked a meeting with potential clients in Stockholm. To be able to showcase the strength of the software at the same time as show what kind of book games the software is able to produce, we wanted to make a mock-up of the kind of book game that can be produced.

(Picture 7.6) Book view of the digital JerryMaya book game mock-up.

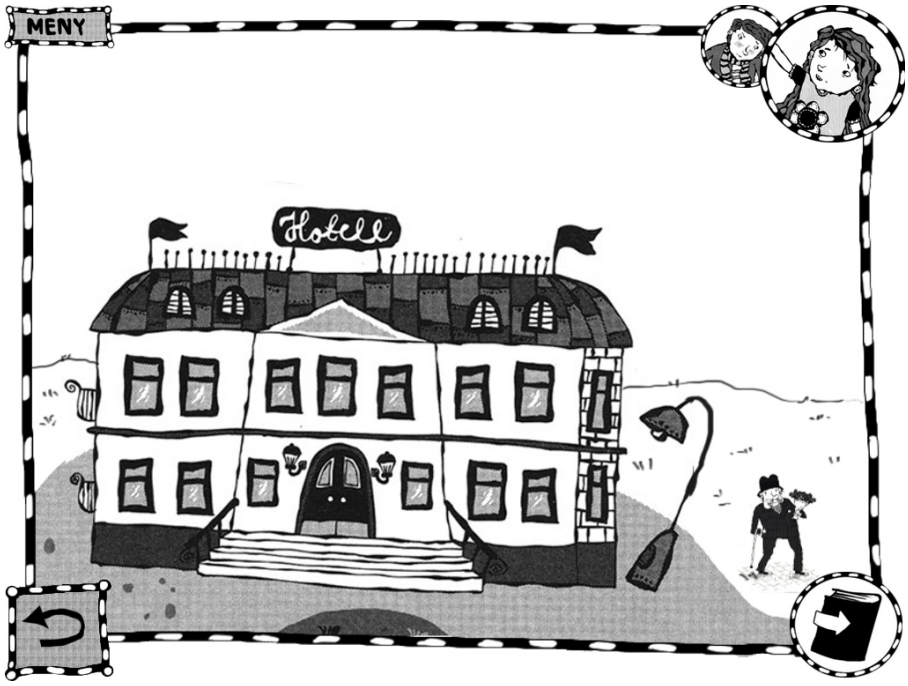
We created the mock-up in Photoshop, with the help of a graphical artist who made lots of the graphics. We also used graphics scanned in from existing JerryMaya books. Our mock-up consists of two views, one book view and one world view. Picture 7.6 shows the book view, in where the user can read the complete JerryMaya book. In the book view the user can not only read the book, but play games by pressing on the different pictures in the book. At the top there is a menu button, and at the bottom are a navigation bar. From this navigation bar the user can go back to page one of the book, see where he is in the book and go to the world view of the book.



(Picture 7.7) World view of the digital JerryMaya book game mock-up.

Picture 7.7 shows the world view of the book game. In the world view the user can navigate Maya by pressing on the map to indicate where she should go. The world view has, just like the book view, different buttons. On the top left is the menu button. The top right has two pictures of Jerry and Maya, respectively. By pressing any of them, the user will be interchanged to Jerry's position on the map, and able to move Jerry instead of Maya. At the bottom left there are a map icon. If the user presses it, a map of Valleby will appear. Pressing it again after that will remove the map. At the bottom right is the button to go to the book view.

The icons with the red border are games that the user is able to play. Clicking them will make Maya walk over to the icon, and then swap scene to that particular game. The thought is that most items in the map should be clickable. For example, clicking on the hotel will make Maya walk over to the hotel, and then a scene swap will occur, changing the scene to the exterior of the hotel.



(Picture 7.8) The exterior of the hotel in the digital JerryMaya book game mock-up.

Moreover, clicking on the little guy standing to the right in picture 7.8 makes a new scene swap occur, making a dialogue with the guy appear. Of course, by clicking on the door of the hotel makes Maya walk into the hotel, where new games and mysteries await.

This digital mock-up of a typical book game created by our software helped us to conceptualize what kind of apps our software should be able to create. With this in mind we knew what functionality was needed in the first version of StoryPlay.

7.4 First Presentation in Stockholm

The digital JerryMaya mock-up and some of the paper prototypes were showcased for a potential customer in Stockholm. The purpose of this meeting was for the customer to see what strengths StoryPlay has, when it comes to transferring existing hard-covers to digital materials to be sold on the tablet market.

7.5 Programming Phase 1

When we began programming the first digital prototype, we soon found out many things. First of all, programming something that complex in a, for us, completely new language was a challenge. Thankfully, we had some assistance from a more experienced programmer at the office that could help us with the most basic issues.

We used a package called Cocos2D in order to arrange graphical object with ease. This proved to be very appropriate for the task at hand. Cocos2D is developed for game programming and suitable for handling graphics.

We began implementing the features of the mock-up, starting with the basic layout of StoryPlay. We implemented a tabbed view at the top, a toolbar on the left and a workspace for the rest of the space. None of this was particularly complex. Next, we created a few objects that the user could move freely in the book editor scene. We believed one intuitive way for the user to create new objects in the book was to drag the category of the object he wanted to create onto the workspace, finally choosing the object type from within a pop-up. Another way to do this would've been to have the categories as menus and have the type be chosen from within those. This would allow the user to quicker browse the different categories, but the created object would have to be illogically placed somewhere on the workspace, for example in the middle. We decided the first solution would allow the user to decide for himself where the created object would end up on the scene.

We lined up the object-creation tokens on the toolbar on the left and started implementing a method that would allow objects to be created when these were dropped on the work space. Additionally, a sister-object was to be created in the scene view at the same time. This would save the user time in creating activities that were going to be available from both the book view and the game view.

Lastly, we allowed the user to swipe in the scene view to get to different areas of the world. The objects he placed in the world remembered their own positions and were removed when their sister-object in the book view was destroyed.

One thing we understood while working on and testing this prototype, was that this way of working really put a lot of requirements on the template used. A template allows the user to get started faster and do less work overall, but heavily restricts his freedom of creation. A template-less solution would give the user (almost) all the freedom he wants, but requires a very sophisticated programmed software. Such a software will need a more complex user interface, which in turn might not be that intuitive to use. We really wanted to keep it simple for the user, given that the target group has so little experience in programming and game design, but we realise that it might not be possible to get enough freedom without making some sacrifices in simplicity.

7.6 Digital Mock-up

One of the things that came out of the paper prototype testing sessions was that the process of designing a whole game is probably too complex for the intended users. Based on this, we decided to try a more didactic approach. By guiding the (new) user, we believe she may learn a lot faster and thus be able to create more complex games. Tutorials and guides are fine in themselves, but we believe implementing a best practice within the interface is going to be more suitable in this case, at least for the most common actions.

Thus, we changed up the interface from the paper prototype quite a bit. We introduced a work flow for the user to follow. The user would choose a template, import material, work with the book view, the map view and finally the scene view. The point of this way to work was to guide the user through StoryPlay. In this iteration, we really wanted to focus on the user-friendliness and making the interface intuitive. The penalty for this, though, was a loss of complexity and variation, as we would soon see.

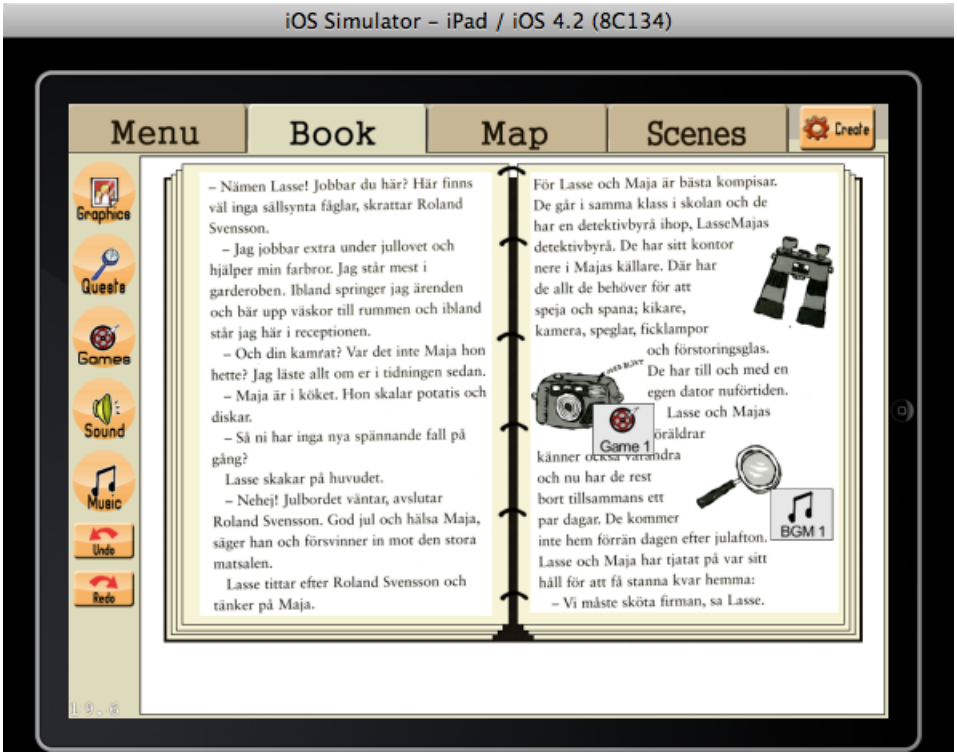
The first digital prototype was to be made for the iPad, and as none of us had any previous knowledge of Objective-C, we had to take some time to learn it properly. This was done by using

our previous knowledge in programming languages, especially C and Java, and adapt that when reading tutorials and a book on Objective-C.

7.6.1 Walkthrough of the Digital Mock-up

When turning on the software the user sees a start screen. In this first prototype the start screen is just a picture and a button that says “Start Editor”. When pressing that button the user comes to the book view. The book view consists of four tabs at the top (two of the tabs are clickable, the “Book” tab and the “Scenes” tab). A create button is placed at the top as well, but in the first prototype it doesn’t do anything.

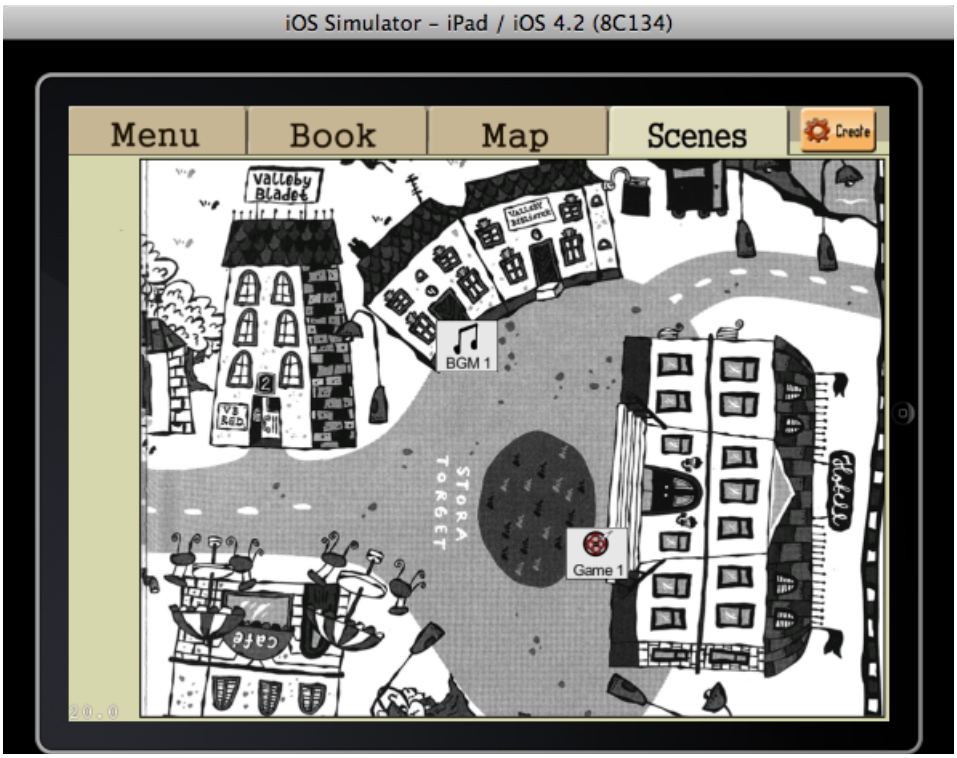
There is a toolbar to the left, including five dragable icons and two buttons for undo and redo, as seen in picture 7.3. The dragable icons lie on a coaster with a name attached to it. When pressing on a dragable icon it will become slightly larger as to indicate that it has been pressed. It is possible to drag the icons to the workspace and drop it on a desired position. When dropped, a pop-up will appear prompting for a name for the object and, depending on type of object, other possible options (such as type, track, file or clip). When done, the object will appear on the workspace with its icon, name and a gray background (for contrast difference). The objects on the workspace is movable if the user decides to change their position. To remove an object, the user will drag the object off the workspace.



(Picture 7.3) The book view of the software in prototype 1.

By pressing the “Scenes” tab the user comes to the scene view, as seen in picture 7.4. The scene view consists of a workspace in the form of a 2D-world. This workspace is swipeable in X- and Y-directions, and by doing so the region will change (the map picture will be interchanged with the adjacent one).

As well as in the book view, there is a toolbar to the left in the scene view. This toolbar doesn't contain the ability to create new objects, but the objects created in the book view will lie on this toolbar until placed in the workspace. Removing object (by dragging them off the workspace) in the scene view doesn't delete them, as in the book view. Removed objects will re-appear in the toolbar, so that the user can add them to another region of the 2D-world. To completely delete an object, the user has to go back to the book view and delete it from there.



(Picture 7.4) The scene view of the software in prototype 1.

7.6.2 Issues

- **How can the user preview tracks, mini-game types, pictures and sound clips?** In short, she cannot, other than by name. As of now, the user only gets to choose music tracks, mini-game type, graphics and sound clips by their name from a list. We would like to add a preview area for that purpose later on. As the user scrolls through the available options, she gets a preview in an adjacent preview area.

- **How do the user see a difference between the object he has created and the ones he uses to create an object?**
  In the first iteration of the prototype, there was no visual difference. That was not a good solution, as it made it feel like the same actions would work on both. The user creates a new object by dragging the token of choice off its coaster and onto the work space, creating a new object at the location where the token was dropped. We believe this to be a good solution as it implements the metaphor of taking a physical object, putting it somewhere and

get a copy of it.

The created object, on the other hand, does not look the same as the token it was created from, thus implying that the same actions are not possible for it.

(Picture 7.5) The created object versus the token used to create it on its coaster.

- **Variation, the lack of it**
  The prototype strictly follows a template and guides the user through filling it up with contents. The user has little to say about the game logic, other than in what way the world looks like. We were fine with this for that time, but it was something we wanted to change in the future. As the focus was on user-friendliness, it didn't matter much that the variation of the end-product was lacking.

- **Mirrored content**
  One of the features of the prototype was the mirrored content between the book view and the scene view. Any content added to the book (pictures, games, puzzles, etc.) was to be added in the book world as well. The idea behind this was that the player should be able to choose to only read the book and still get all contents, but also maybe just play the game and still get the same amount of contents. This was interesting as the player was by no means forced to do either or both. She could freely jump between reading and playing. The biggest fault of this was that some contents didn't have a natural place in both views, thus the user was forced to create a place for it.

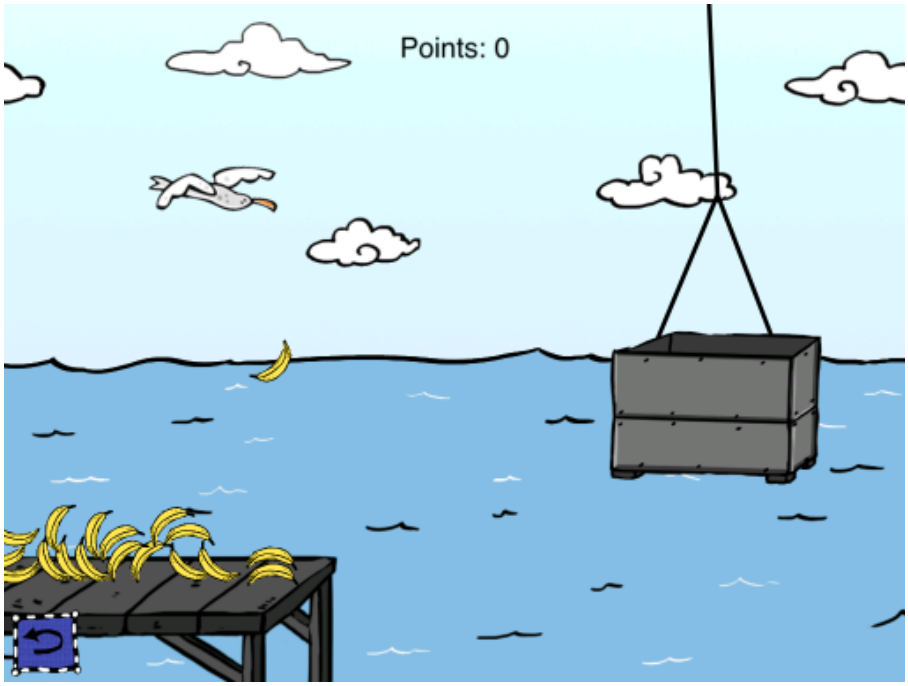
7.7 Digital Prototype of JerryMaya Book Game

After creating the first prototype of the software, we wanted to make a prototype of the JerryMaya book game, to help us further develop the software. This was also done to be able to showcase both the software and the book game in Stockholm at another meeting with the potential customer.

The book game implemented was inspired by the digital mock-up we created earlier. It was a "Point-And-Tap" kind of game, where the user moved Maya in a 2D-map. There were also a book view where the user could read the JerryMaya book.

(Picture 7.9) Digital Prototype of the JerryMaya book game. World view.

Picture 7.9 shows the world view of the digital prototype. It is quite similar to the mock-up done earlier. Maya is fully movable by clicking in the world. By swiping in any of the four cardinal directions, Maya would run in that direction, and the picture would pan over to the next piece of terrain.



(Picture 7.10) Digital Prototype of the JerryMaya book game. World view.

In picture 7.10, a game is showed. The purpose of this fully functional game was to through bananas into the box, and by doing so, scoring points. The game is an example of the kind of games

that might be implemented in the JerryMaya world.

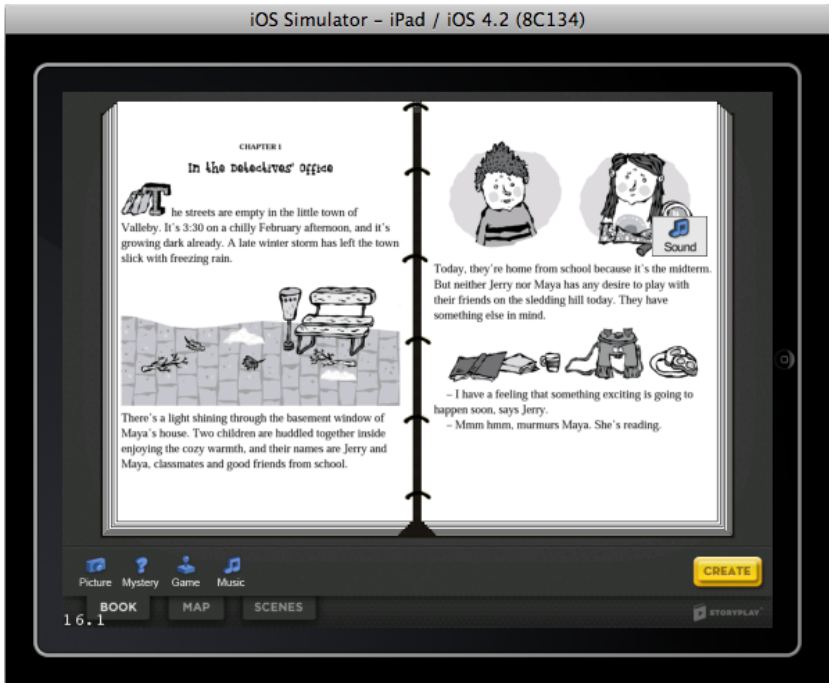
7.8 Second Presentation in Stockholm

On the second meeting in Stockholm, both the prototype of StoryPlay and the JerryMaya prototype book game were showcased. The potential customer got a good feeling of the power of StoryPlay, and what it could be used to create.

7.9 Re-designing StoryPlay

After the presentation in Stockholm, we got some assistance from a professional who pimped the graphics of StoryPlay. The usability basically remained the same, but it got a different look, as seen in pictures 7.11 and 7.12.

The biggest change of the interface was the relocation of the toolbar and navigation bar. By moving it to the bottom, we improved on the usability as the user most likely won't be occluding the work space with his hands anymore. This was an issue with the previous design, something that we didn't realise until just recently.



(Picture 7.11) The book view of the pimped interface.

(Picture 7.12) The scenes view of the pimped interface.

8 Final Concept

The final concept puts together all our knowledge from the research and prototyping and explains what StoryPlay is. We'll describe the business model, the book game concept and basics of StoryPlay.

After evaluating StoryPlay, we realized that it needs more functionality than it has. We want StoryPlay to be able to create more kind of games than just point-and-click games. After studying some target games, and a few pieces of PC software that are created to develop games without any programming skills, we had a fair idea of what functionality was needed to be implemented in StoryPlay.

We realised that to be able to create more diverse game types, we had to cut down on user-friendliness. We implemented two new main features, *Triggers* and *Events*. With the help of these features, the user is able to create a vast spread of game types.

8.1 Business Process

Appmazed has big visions for StoryPlay and thus asked us to define a theoretical business model for it. We felt this would give grounds for the masters thesis as a whole, and give the reader a better view of how StoryPlay fits into the bigger picture.

1. **A publishing house contacts Appmazed**
   - When a publishing house decides to create an application for a tablet, they contact Appmazed to begin the process.

2. **The publishing house sends material to Appmazed**
   - When the contact is established and Appmazed has agreed to create the application, the publishing house sends material to Appmazed. This material is text and pictures wanted in the application.

3. **Appmazed compiles and sends a tailor-made template to the publishing house**
   - After Appmazed has received the material, they package it into the iPad and PC/Mac software and sends it to the publishing house.

4. **The publishing house uses the iPad app and the PC/Mac software with the support of Appmazed**
   - Appmazed supports the publishing house with knowledge about how to use the software while the publishing house develops a draft of the book game.

5. **Appmazed uses its knowledge of game development to iteratively refine the publishing house's draft**
   - The publishing house iteratively sends their work to Appmazed, who gives feedback on it. The publishing house has great knowledge in story telling, while Appmazed has great knowledge in game development.

6. **When a final draft is done, Appmazed develops an app**

- When both the developing house and Appmazed is happy with the draft, Appmazed will take over and develop the application.

7. **The app is published on the market**
   - When the application is fully developed, it will be published on different platforms' market (Android, iOS, Bada, Blackberry, Symbian, etc.).

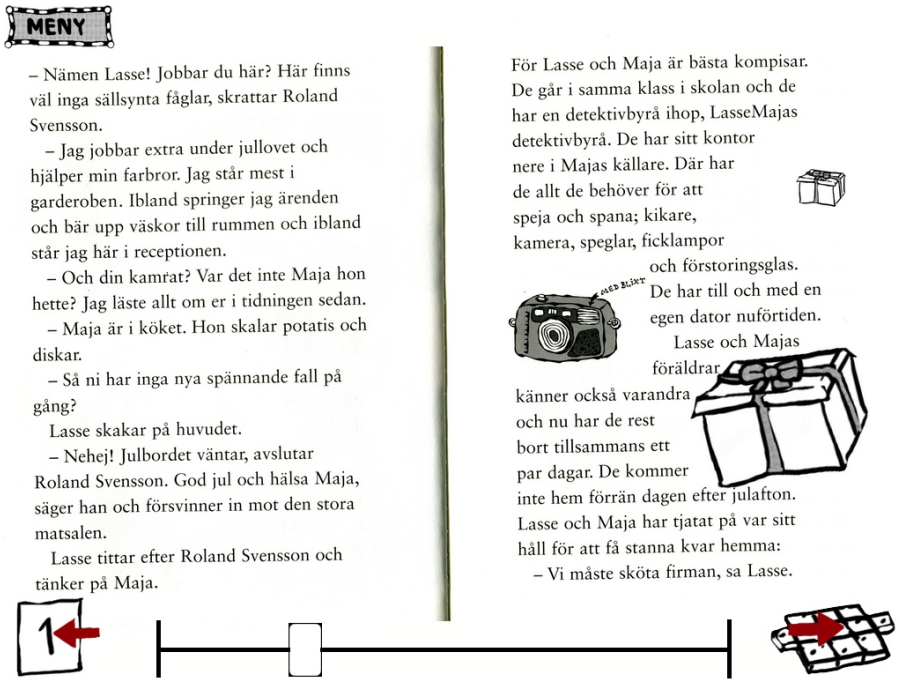
8.2 JerryMaya's Book Game

The book game of JerryMaya's Detective Agency is based on two parts, as seen in pictures 8.1 and 8.2. The main part is the chronological book view and the other part is a view where the user traverse the JerryMaya world as Jerry or Maya. Both the book view and the world view are dependent on the chronological story, but not to the same extent. The book view will follow the content of the book, but extra content in the form of games and quests are added. The world view on the other hand, will follow the content of the book, but the player will be able to utilize parts of the content in a non-chronological order. Content like games and talking to characters will be decoupled from the chronological story, and able to utilize whenever the player want to.

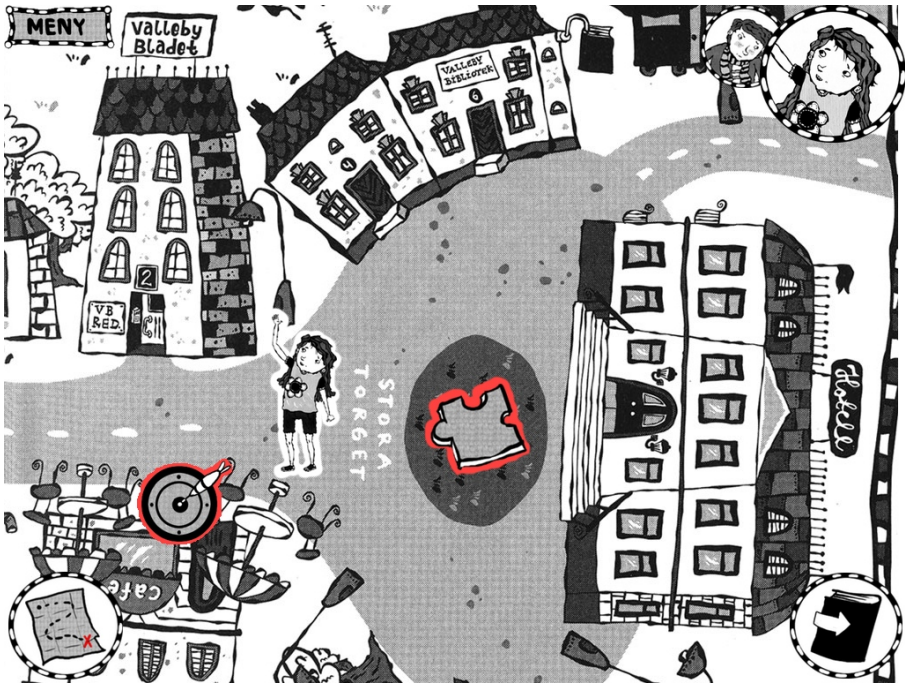
Both the book view and the world view could contain the same story content and the same games. The games in the book view are connected to the pictures on each page, and therefore the player will play the games in the order they come. As a complement to the chronological story and the games, there are quests in the book. Quests are larger games, based upon the stories in the book. Those quests need to be completed in order to unlock new games and quests.

The world view contains of a large map-like world where the player moves around with Jerry and Maya. The world is divided into several smaller parts, and the player pans through these parts with swiping motions on the iPad or iPhone. In each part of the world there are numerous interaction points. These interaction points can be everything from games and interaction with characters to clickable buildings which you enter when you click them. As the player are free to traverse the world as he or she seem fit, the player is welcome to play the games in any order. However, different parts of the map may be closed until the player completes a specific quest, to progress the story.

Unlike the quests, the games are not story-driven, but only thematically coupled to the main story. The player scores high score on the games, and the games themselves may be connected to each other in a manner where gaining a high enough high score in a game unlocks a trivial Easter egg in another game.



(Picture 8.1) The book view of the JerryMaya's book game.



(Picture 8.2) The world view of the JerryMaya's book game.

8.3 The Concept of StoryPlay

The final concept of StoryPlay differs a great deal from the implemented prototype. Some features has been rationalized away and lots of new features has been added. Most notably is the removal of the map tab. The map tab was considered obsolete after the decision was made to bundle the software sent to the author with the complete book, and a semi-complete map, already built in.

Lots of new features has been added as well. The implementation of triggers and events are major improvements of the software. To be able to create things like specific occurrences or happenings, the event handler was implemented. This event handler is able to take care of actions like objects moving in the scene, sounds starting to play on different times, happenings starting on different times and so on.

The triggers and events are not the only new thing implemented. To be able to create a wide variety of games, a number of additional new features has been implemented as well.

8.3.1 Events

When creating a book game, there are a number of events that the designer might want to implement. In the last design of StoryPlay, those events weren't able to be created at all. In this final concept of StoryPlay, we take into consideration how a user might want to implement them. By being able to create those events, the number of game types able to be created is also extended a great deal.

Different events may be:

- **Change the scene**
  To be able to change scenes is most likely very important in any game.

- **Start and stop animation**
  A few types of animations is defined below.

  - **In-place animations**
    These animations are any animations that do not require the object to change coordinates.

  - **Moving animations**
    Moving animations are animations that changes an objects coordinates. The user will need to specify how, where and how fast the object will move.

    - **To finger position**
      The object moves to the position given by the finger tap.

    - **To static position**
      The object moves to a predefined static position on the screen.

    - **In a specified path**
      The object moves in a predefined path on the screen, defined as a list of positions.

    - **To other object's position**
      The object moves to another object's position.

    For all of the above, the user may specify an offset to the position.

- **Add, change or remove object**
  The user will want objects to appear and disappear at any time. Also, the user may want to change an object's size and shape.

- **Starting a timer**
  Timers are useful for making a game feel alive and the user may want to script events according to a schedule.

- **Pop-up**
  Pop-ups may be information boxes, chat bubbles or anything else that pops up at top of everything else.

- **Music and sound**
  Music and sound events are very basic events that the user will want to control.

Most of these events needs to be specified to occur at a specific time, and therefore a timeline for each scene is implemented as well.

8.3.2 Triggers

Triggers are actions that starts the events. Triggers can start the events in different ways, but most triggers comes from the sensors of the iPad, but may also occur when events start or finish. Different actions on the iPad may trigger different events. Actions may also be combined to trigger other events, an example of this is that hold and drag is a different trigger than just hold or just drag.

Different triggers may be:

- **Tap**
  Tapping anywhere may lead to the player interacting with the environment, for example pushing buttons or moving a character.

- **Swipe**
  Swiping may lead to a scene change or toggle a level.

- **Shake**
  Shaking the device may be implemented to cause some effect on the game.

- **Tilt**
  Tilting the device may be used to control a vehicle inside the game.

- **Drag**
  Dragging on the screen may result in panning the view and explore the scene.

- **Hold**
  Holding may be used to pick up items or hold an object in place.

- **Hold and Drag**
  Hold and Drag may be used to move objects on the scene.

- **Timer**
  Timers are useful for scripting events.

- **Collision Detection**
  Two colliding objects may result in an event, for example a sound effect.

8.3.3 Additional New Features

Additional features that will be needed at some time includes, but is not limited to, the following.

- **Object collisions**
  The user will need to specify what objects may collide and thus cause a collision trigger and an event.

- **Specify the z-value for each object**
  The user may want to specify and change an object's z-value anytime during the game.

- **Game specific game elements**
  The user may want to add game elements, such as a menu, health bars, score and so on.

- **Game specific game logic**
  The user will want to specify game logic. This is for example NPCs (Non-Player Characters) or an inventory to keep items in.

- **Edit text**
  The user will want to add and edit text object inside StoryPlay, both for the book world and book view.

- **Edit pictures**
  As with the text, the user will want to add, change, delete and move pictures in the book view.

- **Variables**
  The user may need object, scene and game variables to describe the game flow.

- **Create object types**
  In order to speed up the process, the user may want to specify object types, which he can use as templates when creating new objects.

- **Randomizer**
  If the user wants anything random to occur, he needs to be able to specify that.

8.3.4 Book Mode

The book mode of StoryPlay has the same function as in the implemented prototype. Although there are lots of new functionality to use. The way the content is added to the scene has also been changed due to the amount of new functionality. Having a few categories of objects that is dragable is not ideal when working with the amount of functionality StoryPlay needs. For the final concept, a substantial category system has been elaborated. The objects are divided into categories and sub-categories where they belong. By doing this, a user can easily find the object he is looking for, even though there are an immense amount of objects.

8.3.5 Scene Mode

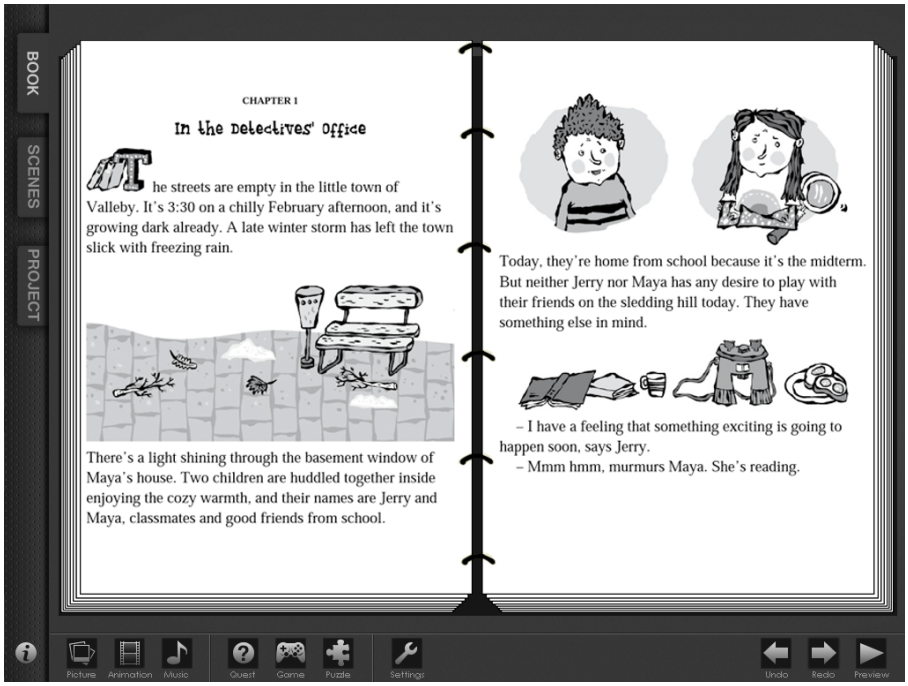
The scene mode is similar to how it has been. The objects added in the book mode is transferred to the scene mode, and adding them is as simple as dragging them into the scene. In this mode, just as in book mode, there are lots of new functionality. Where to find these options depends on what they do. Some functions will be hidden in menus, while other (like move and re-size picture) will be visible only when the affected object is tapped (in this case a picture).

8.3.6 Preview Mode

A new mode added in the final concept is the preview mode. In this mode the user inspects how the book game will look and feel in a finished state. The game mechanics in the preview mode works just like they would do in the finished application. The preview mode will strip away the menus and the tool interface, to only show the book game like its meant to be played.

8.4 Proof of Concept

The proof of concept for this masters thesis is made in Photoshop, and therefore not implemented in Objective-C. For the proof of concept, much of the graphics used in the re-design was used here. The concept only show parts of StoryPlay, but the parts showcased well describes the basic thought of it.

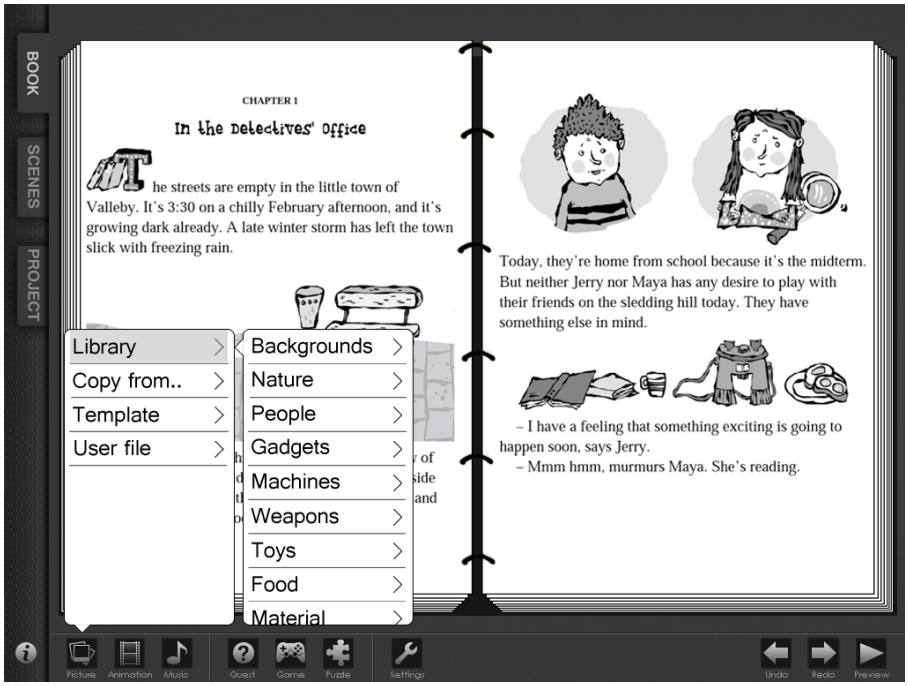


(Picture 9.1) The book view of the Final Concept of StoryPlay.

Picture 9.1 displays the looks of the Final Concept. In order to utilize as much of the workspace as possible, the tabs was moved from the bottom to the left side of the screen. This way, the 4:3 ratio of the workspace can be used without sacrificing too much space. In contrast to the earlier prototypes of StoryPlay, the Final Concept has the same toolbar throughout the whole tool. Changing tab view will change the workspace, but not the toolbar. This is displayed by connecting the tabs to the workspace, and cutting the toolbar off with a dividing line.

The different views of StoryPlay now consists of the Book view, Scenes view and the Project view. The Book view and the Scenes view is the same as before, used for building the book and the world of the game. The Project view is a new view, containing everything from general settings like save and load, to project specific settings.

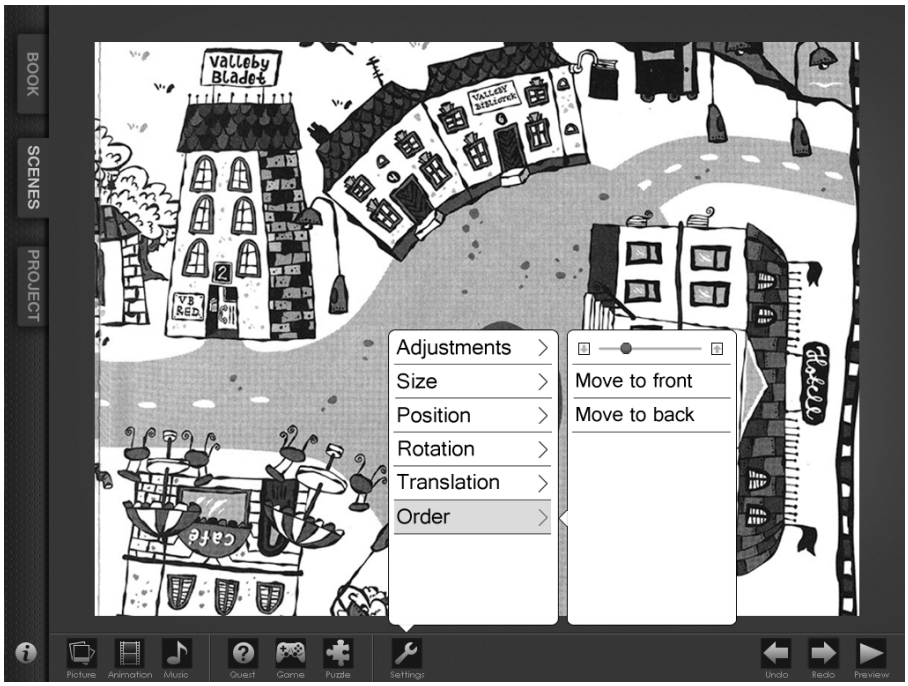
The toolbar has been completely redesigned. It now consists of different categories. The categories are: picture, animation, music, quest, game, puzzle and settings. By pressing each and any of them, a pop-up will appear with more options from that specific category.



(Picture 9.2) The book view of the Final Concept of StoryPlay.

The pop-up for pictures is seen in picture 9.2. There are different categories of pictures, and each step will narrow down the categories. In the last step (not shown in the picture) is the pictures contained in that specific category. Pressing a picture will make it appear in the middle of the workspace. This is the same for each of the buttons in the toolbar, except for the last three.

The last three buttons, undo, redo and preview has specific functions. Undo and redo is rather self-explanatory, they go one step back or forward in the work flow. The preview button plays the book game as it is in its current form. By stripping away the toolbar and the tabs, and giving the implementation is game-play functionality, the user can try out how the book game acts and feels. The preview function is a function that the user will use quite often, therefore the preview button is placed in the lower right corner, close to the users right thumb.



(Picture 9.3) The book view of the Final Concept of StoryPlay.

Picture 9.3 shows the scenes view of StoryPlay. The toolbar stays the same as in the book view, and the settings button is pressed. In the settings button object specific settings can be made. Everything from changing the size of the object, to setting the z-value of it. The z-value, or the order of a object specifies at what depth the object is at. This is important when two or more object is on top of each other, and the user wants to decide what object should be on top of the other. This can be done either by stepping the object one step up or down, or pushing it all the way to the furthest back, or pulling it to the front.

9 Discussion

In this section, we will bring up some points that we felt were important to discuss. We will reflect upon, in hindsight, what we could have done different and if that might have ended up better or worse.

9.1 Design Method

The main advantage of this method was that each important feature of the application had it's own focus session where its particular requirements were taken into consideration. However, this led to a lack of consistency between most of the sketches, as the focusing on one feature made the designs very distinct from each other. This would be the main drawback of the method, which overall felt rewarding as many important points were distinguished through the process.

The second phase of the design method was a bit harder to execute in a formal way, as it largely depended on the result of the first phase. We had a lot to discuss from the first phase, and as it turned out, we managed to combine and subtract ideas from each of the sketches to come forward with two solutions for the problem.

9.2 Implementation Method

The implementation method we used is not a standardized method, but rather a method that worked well for us. By using one programmer instead of two, we could work in a faster pace since one man focuses on the programming, and the other man made the graphics and lots of side work. The drawback with this method is that the work comes to a standstill if the programmer is ill or absent from work for another reason.

9.3 Usability Testing Method

When performing the usability test on the paper prototype, we let the users first try it out for themselves without any assistance. When they had struggled enough, the last actions they had not been able to perform were briefly explained to them. Lastly, they were asked to perform a set of task without any assistance.

Another way to do usability testing is to give the users a more detailed guiding before they get to attempt to use the prototype themselves. This would simulate the training any user would receive prior to using the finished software professionally. The point of this is to measure the efficiency of the design, rather than its intuitivity. As we were not interested in efficiency at that point, we decided to do a rather standardized observation usability test.

9.4 Programming and Design Issues

During the thesis, a few issues arose that we felt was interesting to discuss further:

- **Prototype coding gives rise to non-optimal solutions**

As we added more and more functionality during a very short period of time, the code is far from as optimal as it could be. Quite a few "hacks" has been used, in the sense that it could be solved in a much more efficient and elegant way should the code be restructured. We look at this as the first of many iterations and these steps will serve as a good base for a sharp project. The code issue is not a problem in itself, as it does not give rise to any performance issues or usability issues.

- **Visualize that swiping is available**
  When in the scene view, the user may swipe the screen to change the background, thus navigate in the world. This is, at the moment, not hinted in any way, and that's a problem. We have two solutions for this, both will probably need to be implemented in order to satisfy our visibility, discoverability and redundancy requirements.

  The first solution is to put all areas in the world in a list on the left. We've planned to make the left list a tabbed interface, allowing the user to switch from viewing a list of dragable objects to viewing a list of areas to navigate to. This solution allows the user to directly jump to an area of her choice, not having to swipe through the world, increasing the speed of which she can work.

  The second solution is more of a cry for help than the first one. The solution is to include subtle hints on the edges of the work area that a swiping motion is available and that it will change the background of the screen. This would be done by using a semi-transparent animation that hasn't been decided yet. We would prefer it to include some arrows to show the direction of gesture, but as that is inversed to the direction of movement, it's complicated. We would like to just hint to the user that swiping is available and urge her to try it out, there's no danger in trying, and as the animation is clear on what happens after a swipe, the user would learn it quickly. Also, as after the first solution, swiping is not required in order to navigate the world.

- **Removing objects by dragging it outside the work space, how intuitive is that?**
  The only way to remove an object, as of now, is to drag it outside the work space. We would argue this is a good enough solution, as it is the reverse action of creating it. The user once created the object by dragging the chosen token to the work space, we feel it is intuitive at this stage to drag it off the work space to remove it.

  An alternative solution would be to put a trash can somewhere on the toolbar, but not require the user to drag it to it to remove an object. The purpose of the trash can would be to hint to the user that she can drag an object there to remove it (although she can drag it to any location outside the work space and still get the same result). The downside of this would be that the user might be confused about the trash can and try to tap it before or after an object to see if it does anything. Another use of the trash can is that it's a way to store removed objects and allow the user to restore them at a later point, thus creating a temporary storage location for object the user hasn't decided upon where to put yet.

  Our take on this is to leave as it is, doing some more research on whether it is a good enough solution or not. We don't like the trash can idea as it might confuse the user and that it will take up additional space, something that we don't have on the limited screen. As for restoring removed object, we plan on implementing an "undo" feature anyway, thus

fulfilling the requirement of non-destructive operations.

- **How can the user edit a created object in the prototype?**
  Once again, in short, she cannot. If she named an object wrong, she would have to remove it and try again. This is not a good solution, and not something that we intend to leave as is. We don't have a perfect idea of how to solve this as of now, but this problem is of the same type of, for example, copying of the object. We realise there is a simple solution of letting a option frame pop-up next to the object when it is tapped, but we don't find this an elegant solution and will take a look at what other options we have.

9.5 Design Motivation

The first major design choice the authors had to make was to choose between a desktop computer software, and a mobile software. Due to the rising mobile market of today, the choice fell on making an app for the Apple iPad.

There a many ways to construct a tool that aids the author when designing a book game. The solution created is one of them, and the reason we made a iPad software is that such a tool is easily accessible wherever the author is situated. He can work on the book game when he travels and he can also showcase his work easily for colleagues and friends. Another big reason we chose the iPad platform is that the author gets to work with his book game on the platform it is meant to be used on. What this means is that he can try to play it with the input methods it is meant to be played with, which is a huge advantage against trying the book game on for example a PC with a mouse.

9.6 Final Concept

The final concept has been made much more general than the previous versions of StoryPlay. By making it more general, many more kind of games can be created with it. The drawback is that it is not as user-friendly as before, but a tool of this magnitude needs functionality over user-friendliness.

9.7 Copyright Issues

The project has it's origin in the new opportunities that are arising with new technology, in this case the iPad and similar Android tabs. Publishing houses see this new technology as an opportunity, as they can transfer their analogue material to digital versions. Hard covers are made into digital books to be read on the tabs, but this is not all. To give the reader extra features when buying the digital version, the publishing houses want to release book games, from their most popular titles.

The crux is that according to Swedish law, only the author of the IP (Intellectual Property) can alter or develop new material to the IP. Since the authors in many cases aren't very experienced in software development, they need an easy way to be able to create and edit these book games, and most importantly to express their thoughts to their publishing houses and the companies that are to develop their creations.

Not only the authors needs a way to communicate their ideas, but the publishing houses as well. The publishing houses are often the ones who knows the market best, and it is their responsibility to introduce the author to new market opportunities. Because of this, the publishing houses need a way to communicate ideas with the authors, hence are in need of such a tool.

9.8 Different Ways to Carry Out the Project

The goal for this masters thesis was to answer the question *In what ways is it possible to support the book game creators to convey their ideas to both decision makers at one end and programmers at the other?*

This goal could have been accomplished in a number of different ways. Although we chose to implement a prototype of a software for the iPad, the outcome could have been completely different. There are two sides to the question about different ways to carry out the project. First there are a number of different ways to support the book game creators. Secondly, to every one of these ways, there are a number of different ways to carry out this masters thesis.

Instead of a iPad software, this masters thesis could just as well have developed the computer software that is meant to complement the iPad software. The computer software would probably have had more functionality than the iPad software, thus being more advanced.

Both the iPad software and the computer software could have stopped at concept design stages. Leaving more room in the masters thesis for developing concept ideas, testing those ideas in user tests and developing a fine tuned concept, instead of implementing StoryPlay.

9.9 Prototyping Issues

When making a prototype, is it better to leave out graphics that has no function, or is it better to add it to get a more complete feeling of the interface? Leaving out a lot of graphics make the prototype feel empty, but adding a lot of non-functional buttons will confuse the user. In the end, the user will find out that those buttons didn't do anything, and the prototype will feel as empty anyway. We thus believe that it is better to only include things that work, or at least fakes functionality, when testing the prototype on a user. With this said, the designer should still plan for scalability, although not showing it to the testers.

Another issue we had when prototyping, was that in this sort of creativity application, the user expects some degree of freedom in creation. We couldn't satisfy that expectation, as we couldn't possibly cover for everything the user might want to create. We could, for example, only offer two different sizes for each object, thus rendering the pinch-to-zoom test almost obsolete.

9.10 Deadline Issues

There were some pros and cons regarding working towards the presentations in Stockholm. After deciding upon how the interface of the software should be designed initially, a few things changed. The company had a few meetings with potential customers for the software we are developing. For the company to be able to continue the discussions with the potential customers, the project had to develop a prototype that could be used as showcase material for the customers.

9.11 Other Platforms

When deciding upon platform of choice, both computers and mobile platforms were discussed. A software designed for a computer could become generic and very powerful, but as the target user group would benefit a lot from being able to share ideas with all parts of the company hierarchy, we decide upon focusing on the mobile market instead. As of 2011, when you say mobile market one thinks of the iPad, from Apple. Popular as it is among people in the target group, the choice fell upon it rather easily.

As for Android, we recognize that the Android pads will soon flood the market, but as we had access to an iPad early on, we decided not to consider the Android case for this study. The android pads will differ a lot between each other, both in performance and screen size. This will be a challenge when developing applications for them.

9.12 PC/Mac Software

At the beginning of the project, we believed we would end up with a computer prototype at the end of the thesis. After a while, it became clear that it was much more interesting to develop for the iPad, as that was the target medium for the book games produced by StoryPlay. This, however, did not stop all thoughts of a computer software as well. We think some actions, especially extremely detailed work, are better suited for the computer. Dragging, moving and even resizing objects are pretty easy to do on the iPad, but for example drawing is a hassle. The iPad just can't offer the same precision as a computer and a mouse can. Maybe implementing a mouse driver and a mouse pointer for the iPad would solve this, but it would perhaps take away the point of using the iPad in the first place.

9.13 Future Work

This masters thesis made its way from a mere idea to a software prototype of a iPad program. The way from this point to a functioning software that may be released on the market is huge. The prototype made in this masters thesis is fairly poor coded since it was the first objective-C program the authors of this thesis ever created. Also, as with all first versions of software programs, lots of functionality and structure gets added along the way, thus it is hard to structure the program correctly from the beginning. This is not the only reason for why the code needs to be re-written, integration in the system architecture (PC software and servers) are big parts of the complete system, and the iPad software needs good integration with those components.

For future work, it might be a good idea to take a look at implementing scripting. It would be wise to have one part of the application to be “what-you-see-is-what-you-get”, but also allow the user to take a look “under-the-hood” and edit the scripting code. This is popular in, for example, several blog services available online. This allows the advanced user to adjust fine details at any point in the process. This requires the construction of a scripting language, preferably based on any of the widely used ones, for example Lua.

This thesis has not focused on the small details of the applications functions, but rather the big picture. Lots of functionality remains to be designed, especially when it comes to event handling and construction of actual game logic.

10 Conclusion

Even though this thesis set out to find out ways to support developers to convey their ideas to their surroundings, it soon changed into to find out a good way for book writers to convey their ideas to programmers. This change in subject came upon us subtlety, without us noticing. In hindsight, maybe we should have taken a step backwards and evaluated the progress that we had at the time.

So, why did this change come up in the first place? In lines with Appmazed's wishes for a tool to shorten the production time of new book games, we briefly changed our approach, little by little. We were perhaps a bit too engulfed by Appmazed's business progress to keep an unbiased distance. Thus, the concept for StoryPlay changed from just conveying ideas to actually implement them, but this does still fulfil the initial goal for the project.

We have concluded that it is possible to create a tool for designing interactive book games on the iPad by creating prototypes, and finally developing a proof of concept. The tool may be used for conveying ideas to other people, with desired level of complexity. It should preferably be made to give full freedom to the creator and using an iPad should not be the limiting factor. By implementing a preview mode, the user may share his ideas to others and by collecting the development at a central server many people may work at the same project.

We explored a way for a book game creator to convey his idea to his surroundings through making digital mock-ups of it. We explored the concept of digitalizing children's books, making it interactable and including a book game. The author and the creator are guided through the process, but they are also allowed plenty of freedom.

Lastly, we realise that we have only dug just a little bit on this subject. Also, we realise that the focus of the thesis changed a bit about half-way, making our conclusions a fuzzy. A lot more remains to be investigated, but we have no doubt in that there are good ways to speed up the implementation of and saving money in book game creation for the iPad. Based on this thesis, we have a firm belief that creating a tool for book game creation on the iPad, for the iPad, will allow even people inexperienced in programming to convey and share their ideas in a detailed way. It will also allow for several people to jointly work on the same project, thus speeding up the process, saving money and possibly result in a higher quality product.

As a conclusion to this thesis, we would like to say that it has been rather successful, given our previous knowledge and experience. We predict that a tool in the line of what we have researched may emerge sometime the next three years and possibly have a decent to great impact on the market.

11 References

Agarwal, P. 2011. Continuous SCRUM: Agile Management of SAAS Products. *India Software Engineering Conference*.

Agile manifesto, 2001. *Manifesto for Agile Software Development,* [online] Available at: <http://agilemanifesto.org/> [Accessed 2 October 2011].

Apple, 2011. *Apps for iPhone*, [online] Available at:<http://www.apple.com/iphone/apps-for-iphone/> [Accessed 23 May 2011].

Appcelerator, 2010. *Mobile Developer Survey, June 2010*, [online] Available at: <http://www.appcelerator.com/mobile-developer-survey-June-2010/> [Accessed 9 June 2011].

Beck, K. et al., 2001. *Manifesto for Agile Software Development*, [online] Available at:<http://agilemanifesto.org/> [Accessed 5 May 2011].

Blackler, A.L. Popovic, V. and Mahar, D.P., 2006. Towards a Design Methodology for Applying Intuitive Interaction. In: Friedman, Ken, Love, Terence, Côrte-Real, Eduardo, & Rust, Chris, *Wonderground: 2006 Design Research Society International Conference.* Lisbon, 1-4 November 2006.

Bokjuryn, 2009. Vilka böcker vann, [online] Available at:<http://barnensbibliotek.se/Bokjury/Arkiv/tabid/571/Default.aspx> [Accessed 2 October 2011].

Bonnier Carlsen, 2010. *LasseMajas detektivbyrå*. [online] Available at:<http://www.lassemaja.net/> [Accessed 9 May 2011].

Budiu, R., Nielsen, J., 2010. *Usability of iPad Apps and Websites*. [online] Available at:<http://www.nngroup.com/reports/mobile/ipad/> [Accessed 17 February 2011].

Cho, O-H. Kim, S-W. and Yoo, H-S., 2010. *EDGE: an easy design tool of game event for rapid game development*.

D'Hertefelt, S., 1999. Observation methods and tips for usability testing. [online] Available at:<http://interactionarchitect.com/knowledge/article19991212shd.htm> [Accessed 2 October 2011].

Five Sketches, 2009. *The Five Sketches Method* [online] Available at:<http://www.fivesketches.com/> [Accessed 9 June 2011].

Nielsen, J. 2010. *iPad Usability: First Findings From User Testing*, [online] Available at:<http://www.useit.com/alertbox/ipad.html> [Accessed 11 February 2011].

Norman, D.A. *Affordances and Design*, [online] Available at:<http://jnd.org/dn.mss/affordances_and_design.html> [Accessed 9 June 2011].

Norman, D.A., 1999. Affordances, Conventions and Design (Part 2). *Interactions,* May Issue, pp.38-43.

Norman, D.A. and Nielsen, J., 2010. Gestural Interfaces: A Step Backward In Usability. *Interactions,* September + October, pp.46-49.

Svenska barnboksintitutet, 2010. *Bokjuryn*. [online] Available at:<http://www.barnensbibliotek.se/Bokjury/Arkiv/tabid/571/Default.aspx> [Accessed 9 May 2011]

Wentzel, J. and Holmlid, S., 2009. Speed Sketching With Designers: User Inspired Brainstorming. *International Conference on Designing Pleasurable Products and Interfaces*.

Wells, D. 2009. Extreme Programming. [online] Available at:<http://www.extremeprogramming.org/> [Accessed 10 May 2011].