MPP405 Report

Design, Play, Release: A Conceptual Model for Creating Publisher-Independent Cross-Platform Casual Games in Small Teams

Submitted in partial fulfilment of the Master of Arts/Master of Science Professional Practice at SAE Institute.

Student name:	Nicolas Zanotti-Schudel
Date of submission:	January 2013
Academic advisor:	Dr. Fares Kayali
Word count:	10'838

Abstract

Purpose: Advancements in technology, such as mobile devices and social networks, have introduced new audiences to video games. Thanks to newly available educations and software toolsets, it is possible for a small team or a single person to create games for a casual gaming audience. The generally low income of independent casual game developers creates the need to increase their return of investment. This report addresses this need by suggesting improvements to the production workflow, while extending the reach of the game across multiple platforms.

Objective: To develop and validate a conceptual model for creating publisher-independent cross-platform casual games in small teams based on a framework of ideas from the Spiral Model of Software Development and the Agile concepts, Scrum and Feature Driven Development.

Design: Three cycles of Participatory Action Research and three group interviews conducted on-site.

Participants: The researcher is a participant in the Action Research process. Six professional game developers with expertise in game design, graphic design, animation, audio-engineering and programming are interviewed.

Results: A system, owned and operated by a small development team or single person, to create a cross-platform game for casual gamers. It visualizes optimizations to the creative process of casual game development by incorporating concepts such as prototyping, iterative feature implementation and frequent assessment. The step-by-step workflow also helps avoid mistakes that lead to a technologically unviable or prematurely released product.

Conclusions: The conceptual model is a practical tool for creating publisher-independent cross-platform casual games. Further real-world applications of the model are needed for its evaluation.

Contents

1. Introduction to the Research

1.1. Research Problem

The rise of mobile devices, such as the smartphone and the tablet computer, and the advancement of software technologies, such as social networking websites and digital distribution platforms, have introduced new audiences to video games. Combined with matured software toolsets and the means for learning them, it is now possible for a small team or single person to produce a game for these audiences independently and within a short time frame (MochiMedia, 2011, p. 20; Thoa, 2012). Upon being asked about the driving factors behind independent [indie] video game development, Chris Swain, director of USC's Innovation Lab answered:

> *"Two factors are driving this. One is that there are student programs training guys to create original systems and to differentiate on gameplay […]. The second factor is digital distribution is finally here, thanks to large-scale broadband penetration and ubiquitous tools like Flash."* (Swain 2008)

However, even with these new possibilities, indie game developers seem to earn a low income. In the "Mochi Media 2012 Flash Games Market Survey", over half of the 1359 participants stated they earn less than $500 USD per month for developing games (2012, p. 29). Of 252 surveyed *Apple iOS* game developers, half stated that they make less than $3000 USD in all their app sales (Goss, 2011). Of 102 surveyed mobile game developers, over half stated they do not earn enough money to break even with development costs (Yentin et al., 2012, p. 4).

> *"So for a single developer, to make an average living requires 10-15 games a year."* (Braben, 2012)

There are many measures that can improve the return of investment. Two of which will be suggested in this paper: one is to optimize the production process, in order to lower the time it takes to get a quality product to market (reduce effort). Another, is to develop once and release directly to multiple distribution channels, thus extending the reach to possible customers (increase income). Emeric Thoa, indie game developer and creator of "Squids", has chosen this approach:

> *"SQUIDS will very soon release on PC, Mac, and Android, which was part of the plan from the beginning. In my mind, being multi-platform is really where the indie developer has a future as a studio. As for the money itself, even though SQUIDS hasn't made us rich so far, revenues from the iOS version have almost covered our development costs and we are confident that its upcoming release on other platforms will make the game profitable […]."* (Thoa, 2012)

An increase in income, due to development experience, shows in the results of Goss' survey. The average income rises from $500 USD for the first game to almost $16,000 USD for the tenth game (Goss, 2011).

> *"[...] the more games developers had released, the more per-game average revenue they seem to generate."* (Goss, 2011)

The factors for increasing revenue, otherwise gained by personal development experience, will be researched. Rather than setting focus on improving skills in creative and technical disciplines, it will be set on the process as a whole, and, in order to express this process, a conceptual model [CM] will be created.

Software development processes like Scrum and Test Driven Development already exist, as I will present on upcoming pages. While they are not directly applicable to the specific problem area, as I will argue, they do consist of years of improvements and are thus valuable resources. Comparisons with further processes throughout the paper also allow the questioning of conventional wisdom when applied in current circumstances.

1.2. Hypothesis

As a tangible outcome of the research process, the "Design, Play, Release Model" is a practical tool for creating publisher-independent cross-platform casual games in small teams.

1.3. Scope of the Research

The following decisions were made, in order to narrow the scope of the research:

1. The research will only target publisher-independent game developers. External involvement would require a different decision making process.
2. The outcome of the model should be a *casual* game for multiple platforms. More elaborate games would require larger teams and different forms of funding.
3. Only small teams with a maximum of four actors will be addressed, in order to avoid the necessary complexities of communication and role allocation in larger teams.
4. The model will not describe *how* to design, program or market a game, but rather give an overview of *when specific steps* in the process are recommended.
5. Three game prototypes will be developed in order to test the model. Developing complete games is not possible within the time frame of the research.

I will now describe relevant concepts, before proceeding to discuss the introduced topics in detail.

2. Background Information

2.1. Research Methods

Action Research

Action Research [AR] is a research methodology for solving problems (Oates, 2006, p. 154; Griffiths, 1998, p. 21). It includes an iterative process with multiple steps. Although variable, these steps are based on the process of plan, act and reflect (Blaxter, Hughes & Tight, 2001, p. 70; Oates, 2006, p. 157). The origins of AR are unclear; A description made by Kurt Lewin in the 1940s is credited as being the first (Masters, 1995, p. 1). Lewin described AR as *"proceeding in a spiral of steps, each of which is composed of planning, action and the evaluation of the result of action"* (Kemmis & McTaggert, 1990, p. 8, cited in Masters, 1995, p. 1).

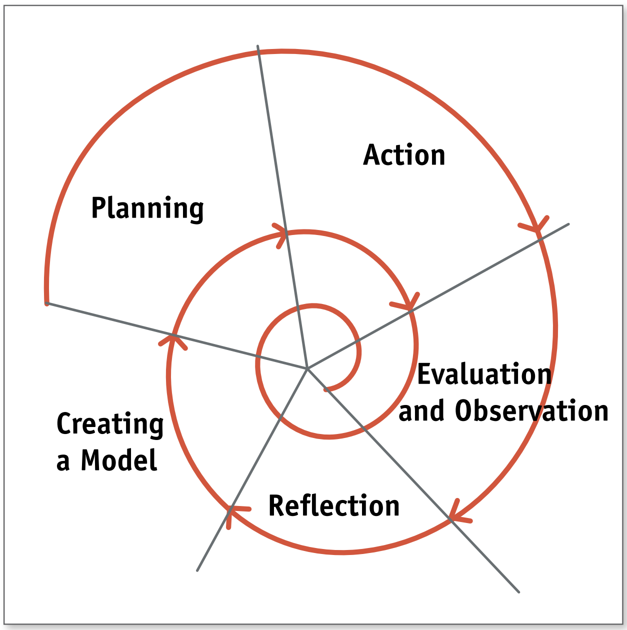


Figure 1: An Action Research Cycle (Adapted from: Routio 2005)

Peter Checkland, an influential action researcher and creator of SSM, suggests that researchers should conceptualize AR by using the FMA model, where *"A particular set of ideas, F, are used in a methodology, M, to investigate some area of investigation, A."* (Checkland, 1991, p. 398, cited in Kock et. al, 2007, p. 145).

Framework:	A framework of ideas acting as the theory base in an AR project;
Methodology:	A problem solving methodology that embodies the theory base;
Area:	A real world problem situation (Oates, 2006, p. 156).

The framework helps action researchers define how knowledge will be created and expressed (Oates, 2006, p. 156).

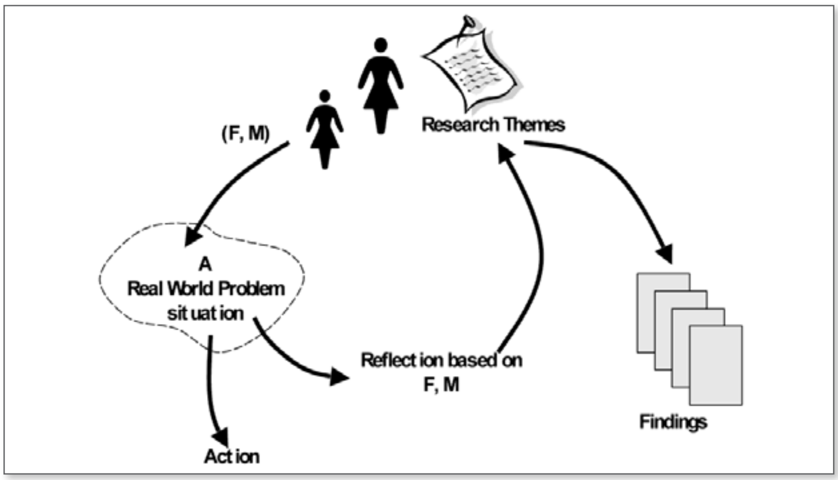


Figure 2: An Action Research Cycle using the FMA framework (Checkland, 1991, cited in Kock, 2007)

As an emergent research strategy, AR has leaned towards becoming participatory in recent years (Oates, 2006, p. 161). Reason and Bradbury describe this 'new' action research as a *"participatory, democratic process"* (Reason & Bradbury, 2001, p. 1). Participation is in alignment with the objective of AR — increasing knowledge on how to improve practice — since the researcher may be part of the situation in need of improvement. A negative aspect of participation is that the researcher may be too close to the subject matter and prone to self-delusion (Oates, 2006, p. 160). It is, therefore, recommended for an action researcher to report the efforts for avoiding self-delusion (Oates, 2006, p. 161).

Soft Systems Methodology

Soft Systems Methodology [SSM] is *"an organized way for tackling messy situations in the real world"*. (Checkland & Scholes, 1999, p. 1). SSM situates itself at the 'soft' end of the spectrum of real-world problem solving, where it is not quite clear what the exact problem is (Wilson, 2001, p. 6). SSM can be characterized by the steps in figure two:

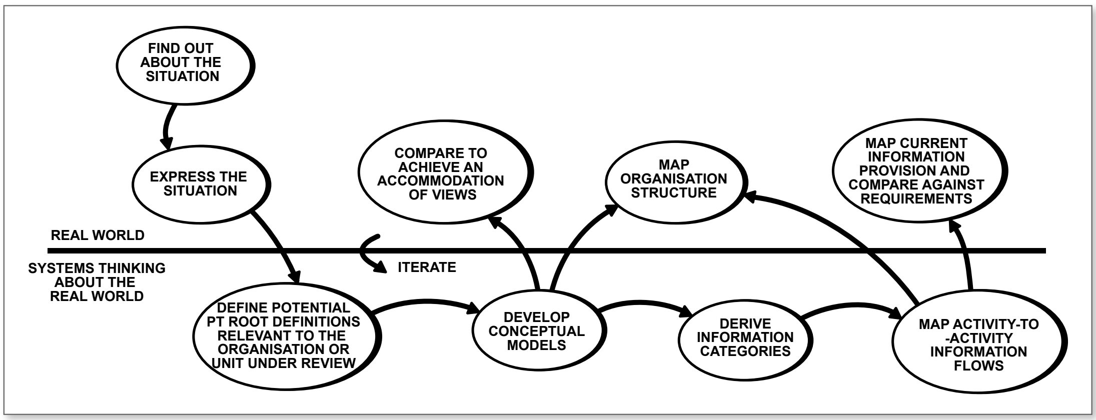

FIND OUT ABOUT THE SITUATION
EXPRESS THE SITUATION
COMPARE TO ACHIEVE AN ACCOMMODATION OF VIEWS
MAP ORGANISATION STRUCTURE
MAP CURRENT INFORMATION PROVISION AND COMPARE AGAINST REQUIREMENTS
REAL WORLD
SYSTEMS THINKING ABOUT THE REAL WORLD
ITERATE
DEFINE POTENTIAL PT ROOT DEFINITIONS RELEVANT TO THE ORGANISATION OR UNIT UNDER REVIEW
DEVELOP CONCEPTUAL MODELS
DERIVE INFORMATION CATEGORIES
MAP ACTIVITY-TO -ACTIVITY INFORMATION FLOWS

Figure 3: A Wilson methodology for information requirements analysis and information audit. (Wilson, 2001, p. 8)

With these steps, SSM contains a process that creates new knowledge on how to solve a problem. For instance, SSM was used to analyze the design of social software (Bouman et. al, 2008).

2.2. *Software Development Concepts*

A Spiral Model of Software Development

In his book on game design, Jesse Schell (2008, p. 82) suggests the use of the "Spiral Model of Software Development", as defined by Barry Boehm in 1986, in order to reduce risk. This spiral model consists of the phases: design, assess risks, build prototypes, test, and reflect (Boehm, 1986, p. 64, cited in Schell, 2008, p. 83).

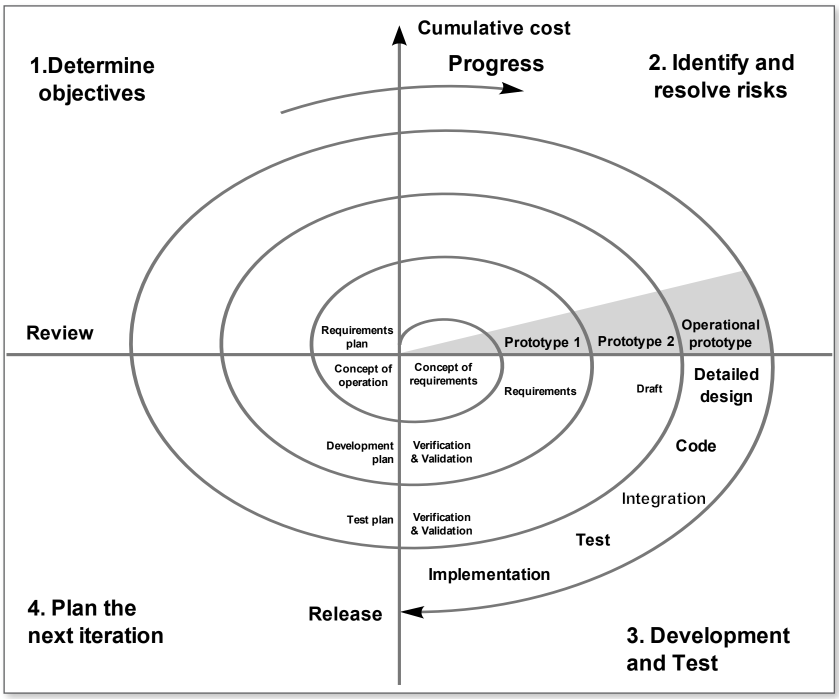


Figure 4: Spiral model of the software process (Boehm 1988, p. 64).

Boehm's model presents a method that potentially optimizes the practice of creating software.

Agile Software Development

"Agile" is a collection of values and principals for software development (Shore & Warden, 2008, p. 9). The inventors of Agile expressed this way of thinking about software development through a manifesto, in which they state the following core concepts:

> *"**Individuals and interactions** over processes and tools;*
> ***Working software** over comprehensive documentation;*
> ***Customer collaboration** over contract negotiation;*
> ***Responding to change** over following a plan."*
> (Beck et al., 2001 a)

Multiple software development methods are based on these principals: "Extreme Programming" (Marchesi & Succi, 2003, p. 9), "Scrum" (Shore & Warden, 2008, p. 9), "Feature Driven Development" (De Luca, 2005), and "Lean Software Development" (Poppendieck & Poppendieck, 2003, p. 11).

Scrum

Based on a term used in rugby, where a team *"tries to go the distance as a unit, passing the ball back and forth"* (Takeuchi & Nonaka, 1986, p. 1), Scrum is a *"framework for developing and sustaining complex products"* (Schwaber & Sutherland, 2011, p. 3). The term was first used in this manner in 1986

Figure 5: A scrum in rugby (Ephram 2008)

by Takeuchi and Nonaka and has since gained popularity for optimizing business processes (West, Gilpin & D'Silva, 2009), including the development of software (Kieth, 2010).

Since the mid nineties, Ken Schwaber and Jeff Sutherland, both active within the computer software industry, have been collecting and contributing new advancements to the Scrum framework:

> *"Scrum is probably a collection of best ideas of what a number of people in our profession have come up with over the years."*
> (Schwaber, 2006)

The key concepts of the framework are organized as artifacts, roles, events, and rules (Schwaber & Sutherland, 2011, p. 3).

Artifacts:

Backlog: A list of features ordered by priority (Kieth, 2010, p. 41). It is a bullet-point style description of what the product should become. Updating the backlog is its own separate process: *"The Product Backlog evolves as the product and the environment in which it will be used evolves."* (Schwaber & Sutherland, 2011, p. 12)

Roles:

There is a distinction between customer and production roles, allowing a clear definition of each party's needs (Keith, 2010, p. 44).

Development Team: All professionals required to complete the project goals (Keith, 2010, p. 46).

Scrum Master: In charge of overseeing the Scrum process (Keith, 2010, p. 46).

Product Owner: In charge of maximizing the product's return of investment (Keith, 2010, p. 51) and managing the backlog (Schwaber & Sutherland, 2011, p. 5).

Stakeholders: Usually the financiers of the project. They are in charge of defining product features (Keith, 2010, p. 54).

Events: Scrum uses the planning techniques of timeboxing and events for determining when certain phases begin and end (Schwaber, 2006).

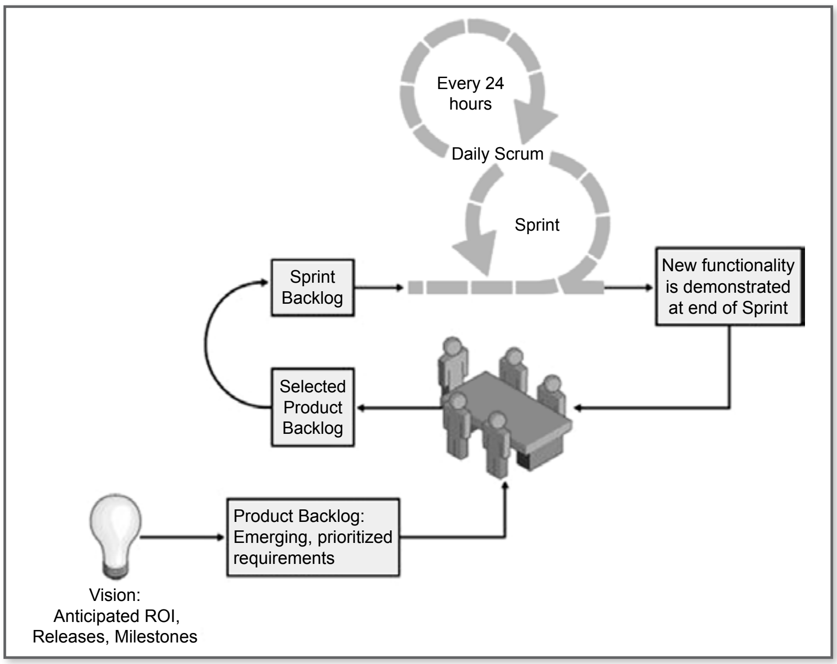

Every 24 hours
Daily Scrum
Sprint
Sprint Backlog
New functionality is demonstrated at end of Sprint
Selected Product Backlog
Product Backlog: Emerging, prioritized requirements
Vision: Anticipated ROI, Releases, Milestones

Figure 6: Scrum process overview (Schwaber, 2009, p. 9)

Sprints: *"A Scrum-developed project makes progress in sprints"* (Keith, 2010, p. 42). During a predefined time-period, the team produces a feature from the backlog. At the end of a sprint, the team shows the new feature to the project's stakeholders, the backlog is updated and a new sprint is commenced.

Releases: A release happens after a set of sprints. At that point, the product should be in a near-shippable state, meaning that it should be fully functional with the current features (Kieth, 2010, p. 43).

Meetings: Project members meet at predetermined intervals for meetings with predefined goals. For instance: planning a sprint (Schwaber & Sutherland, 2011, p. 9), synchronizing work efforts (Kieth, 2010, p. 74), demonstrating product functionality (Kieth, 2010, p. 76), or reflecting on the production process (Kieth, 2010, p. 79).

Rules: The rules bind together events, roles, and artifacts (Schwaber & Sutherland, 2011, p. 5).

Feature Driven Development

Feature Driven Development [FDD] was designed by Jeff De Luca, Peter Coad, and their team during a large IT project in 1997 (Anderson, 2004, p. 2). During the writing of the "Agile Manifesto" in 2001, FDD became one of the recognized sets of Agile software methods (Anderson, 2004, p. 2). As the name suggests, the feature is the central part of the method:

> *"Each feature in FDD reads as a requirement which is understandable by the sponsor – it has true business meaning and describes true business value."* (Anderson, 2004, p. 4)

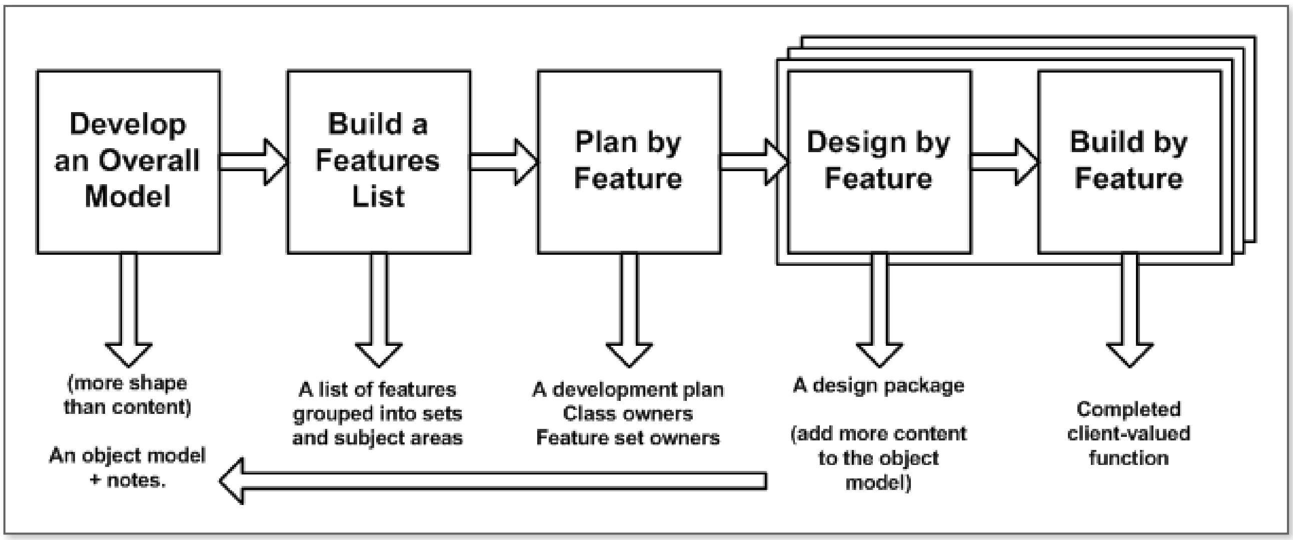


Figure 7: The FDD process. (Palmer & Felsig, 2002)

Jeff De Luca describes the FDD process in the following five steps (2012).

Develop an overall model: The domain area is modeled by domain and development experts (2012, p. 1).

Build a Features List: All the features to support the requirements are identified (2012, p. 3).

Plan by Feature: A development plan is created (2012, p. 5).

Design by Feature: Features are submitted to a chief programmer that then organizes the development (2012, p. 7).

Build by Feature: The items necessary to support the design of the feature are implemented (2012, p. 9).

2.3. Information Technology Concepts

Soft and Hard Systems

Where SSM situates itself on the 'soft' end of the problem spectrum, technological tools and techniques are at the opposing 'hard' end.

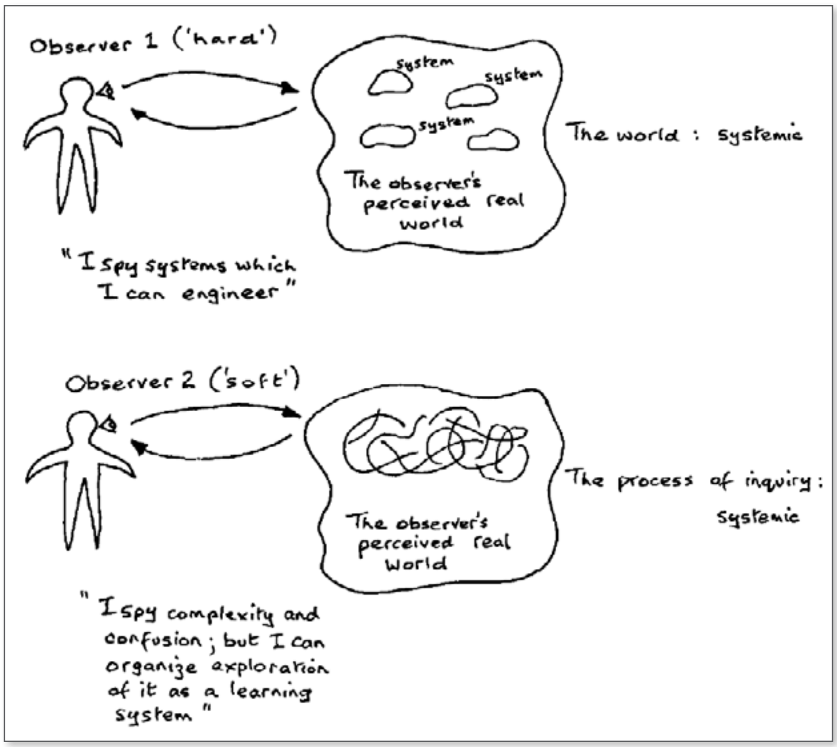

Observer 1 ('hard')
system
system
system
The world : systemic
The observer's perceived real world
"I spy systems which I can engineer"
Observer 2 ('soft')
The process of inquiry: systemic
The observer's perceived real world
"I spy complexity and confusion; but I can organize exploration of it as a learning system"

Figure 8: The hard and soft system stances (Checkland 1999, p. 18)

These two ends are linked, meaning that the availability of new technology can create new real world problems. Ken Schwaber talks about this effect in a tech talk about Scrum:

> *"So it's an idea that if we didn't have the technology [Smalltalk], it [Scrum] would have gone nowhere."* (Schwaber, 2006)

Schwaber suggests that the availability of a new programming language (hard end) allows for a new development method (soft end).

Computing Platforms and Software Frameworks

According to Ralph Johnson of the University of Illinois, Department of Computer Science, a software framework *"is a reusable design for all or part of a software system"* (1997). As described in the paper, "Frameworks on the Rise", software frameworks can shorten and simplify the production process:

> *"The architecture and code reuse provided by [software] frameworks can thus make development quicker and less error prone for applications in the target domain."* (Bierhoff et al., 2009)

A computing platform is typically a combination of such a framework with computer hardware (Microsoft, 2011). Computing platforms for mobile devices, for example, are *Apple iOS* and *Google Android*.

Cross-Platform Software Development

Software frameworks can be abstracted from a specific computing platform, allowing them to compile or run on multiple types of computing hardware. The definition of the term cross-platform by *Sun Microsystems*, the former creators of *Java*, is "*Pertaining to heterogeneous computing environments.*" (Sun Microsystems, 1999). A cross-platform application is written once and compiled for multiple computing platforms (Sun Microsystems, 1999).

2.4. *Gaming Terminology*

In his book, "Theory of Fun for Game Design", Raph Koster defines games as:

> *"[…] concentrated chunks ready for our brains to chew on. Since they are abstracted and iconic, they are readily absorbed. Since they are formal systems, they exclude distracting extra details."*
> (Koster, 2010, p. 36)

Games are classified into different genres. The research targets "casual" and "indie" computer games.

Casual Game

In his 2008 keynote of the "Casual Games Summit", John Welch states that "*casual games can only be loosely defined as those titles that are friendly to new or occasional users and are intuitive and accessible.*" (Welch, 2008). A video game could thus be considered 'casual' if it does not require an extended commitment to learn and play.

Indie Game

Indie art –like music, films, and games– distinguishes itself from mainstream art, in that it is created without corporate financing (Andrews, 2006). The degree of independence may vary: While an indie game may simply mean that it was funded without the help of a publisher, it can also mean that there was little to no outside influence (Gnade 2010). As Mike Gnade, editor of "Indie Game Magazine", writes in his article "What Exactly is an Indie Game?":

> *"Cliff Harris […] of Positech Games not only funds his own projects, but he also self-publishes them on his own website, handles all the customer-support questions and emails, markets, codes his website, manages his forums, and writes his own blog. Positech Games is a one-man shop that embodies the work, risk, and effort it takes to be an indie developer."* (Gnade 2010)

Target Domain

In the context of this research, the target domain is not only the computing platform a game will run on, but, more generally, the environment in which a game will be played. For instance, when a game is displayed on a handheld device while in transit, or projected on a wall during an art exhibition.

Fun

The research is based on the assumption that a game is supposed to be fun. Like a unit of measurement: A game that is not fun does not get played, whereas a game that is a lot of fun gets played repeatedly. Koster explains this link in detail:

> *"Fun from games arises out of mastery. It arises out of comprehension. It is the act of solving puzzles that makes games fun."*
> (Koster, 2010, p. 40)

Throughout this paper, the term 'fun' will merely be used as the *reason* a game is played. A greater debate on the definition of fun, e.g. from a psychological standpoint, is beyond the scope of the research.

3. Research Design

The problem situation is that indie game developers need to optimize their production processes and extend the reach of their games, in order to improve their return of investment.

Participatory AR will be conducted. Checkland's FMA model, as described on page 6, will be applied as follows:

Framework of ideas:	The Spiral Model and the Agile methods Scrum and FDD (see section 2.2 on page 8);
Methodology:	SSM (see section 2.1 on page 6);
Area:	Publisher-independent cross-platform game creation in small teams.

3.1. Data Gathering

The research takes place within the community of indie game developers and is targeting the 'soft' side of the problem area: the researcher's understanding of how indie developers create digital games. The 'hard' side of the problem area is being addressed by computing platform engineers, by providing the technology to run games on different devices (see section 2.1 on page 6).

Two qualitative methods will be used for collecting primary data:

Observation:	As desk work, the process is logged during three AR cycles. The intended outcome is a detailed understanding of the different factors involved in creating a casual game.
Interviews:	The intended outcome of the interviews, held on-site, is to compare the production process with that of the interviewee. The questions will be based on the current CM, as well as the researcher's understanding of the topic area.

Brian Wilson, author of "Soft Systems Methodology: Conceptual Model Building and Its Contribution", writes that "*Undertaking interviews and/or reading documents is still the means of assembling data but this activity needs to be preceded by the intellectual planning*" (Wilson, 2001, p. 9). The questions for the interview will be gathered, based on the observations and outcomes of the AR cycles.

Both the soft and hard sides of the problem area will be documented and discussed in the reflective logbook accompanying this report (appendix on page 50). It should be read during the action phases of the AR process starting in the next chapter.

3.2. Developing the Model

The CM will be developed according to Wilson's "Human Activity Systems "(2001). It will be expressed visually as a step-by-step process.

> *"[…] the models represent a description of what has to be done (as a set of interlinked instructions) to achieve some prescribed purpose."* (Wilson, 2001, p. 12)

As the name suggests, a CM represents a model of a concept (Wilson, 2001, p. 14). It is not the single way for creating a game, rather a suggested approach for creating a game in an organized manner.

Root definition of the model:

Plan, create and release a casual game for multiple computing platforms.

The process will take place in between the inception of a game idea and end when the game is released and ready to be marketed.

Transformation process (T) of the model:

Inception → T → Marketing

For example: The idea of making a casual horse riding game gets transformed (T) into the game "Horse Rider". The game gets released to the *Apple App Store* and the *Google Play Store*. After that, it is ready to be marketed to horse enthusiasts.

CATWOE definition of the model:

A system, owned (O) and operated by (A) a small development team or single person, to create (T) a cross-platform (E) game for casual gamers (C), from the initial inception through to the marketing phase (W).

3.3. Research Participants

The researcher is the sole participant of the AR process. The interviews will be conducted among six professional game developers:

Jann Sigrist and Jörg Sigrist of DigiDingo

DigiDingo is *"a Swiss based studio that focuses on creating games and applications for iPhone, iPad and Android devices"* (2012). They are the creators of the indie game "Grooh". It topped the Swiss *iPad* game charts in February 2012 and was purchased over 12'000 times.

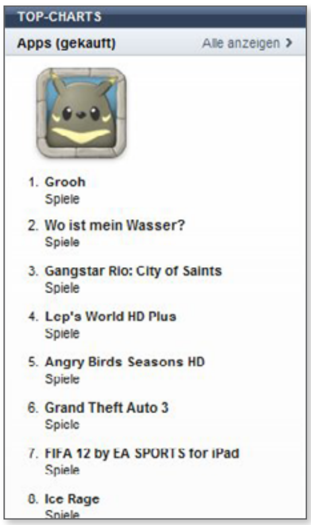

TOP-CHARTS
Apps (gekauft) Alle anzeigen >
1. Grooh
Spiele
2. Wo ist mein Wasser?
Spiele
3. Gangstar Rio: City of Saints
Spiele
4. Lep's World HD Plus
Spiele
5. Angry Birds Seasons HD
Spiele
6. Grand Theft Auto 3
Spiele
7. FIFA 12 by EA SPORTS for iPad
Spiele
8. Ice Rage
Spiele

Figure 9: Screen shot of the *Apple App Store* charts for *iPad* games (DigiDingo 2012)
Figure 10: Screen shot of Grooh running on the *Apple iPad* (DigiDingo 2012)

Moritz Laass and Simon Siegenthaler of Jomoho

Besides being the creator of "Gridflower", an online tool for creating games in collaboration (2011) and being a regular participator of the "Ludum Dare" contest (2012), Moritz Laass is the maker of the game "Franky Flies High". His colleague, Simon Siegenthaler, is the graphic artist and creator of the game character "Franky".

Figure 11: Intro screen of Franky Flies High (Jomoho 2012)

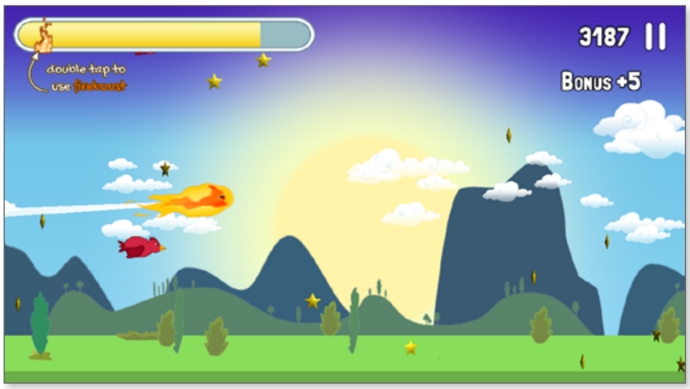


Figure 12: Game screen of Franky Flies High (Jomoho 2012)

Bastiaan van Rooden and Mark Gruber of Nothing Interactive

Nothing Interactive has been creating advergames, social games and serious games for over a decade. They have produced games such as "Elch Express", "Login Flipper", "Cash Explosion", "Groox", and "Plobb" (Nothing Interactive, 2012). The company is also an active participant in the International Game Developers Association (IDGA 2012).

Figure 13: Screen shot of the game "Groox" (Nothing, 2012)

Figure 14: Screen shot of the game "Login Flipper" (Nothing, 2012)

3.4. Ethical Considerations

No information from the interviewees will be published without their permission. The interviewees will receive the recording and the transcript/ translation for their own use upon completion. The CM will not be shown to the interviewees beforehand, in order to avoid influencing any opinions or observations.

3.5. Considerations for Avoiding Self-Delusion

According to the 2011 Flash Games Market Survey, over half of the participants stated that they produce a casual game within one to three months (MochiMedia, 2011, p. 23). Eighty hours of production time has been allotted per game prototype. The shorter production time needs to be extrapolated in order to represent a full production process. Time can be saved by not creating levels, detailed art work and sound effects. However, some details of the production process may be lost. As a measure for avoiding self-delusion, I will ask questions specific to the missing parts of the process during the interviews.

A flaw in the execution of the AR cycles is that the games are only created by a single person. The CM could thus lack attention to the communication between parties. This missing information, however, can also be gathered from the interviews and projected onto the model.

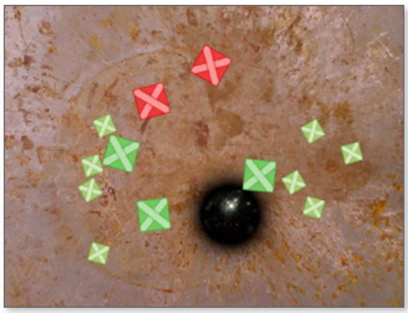
4. *First Cycle of the Action Research Process*

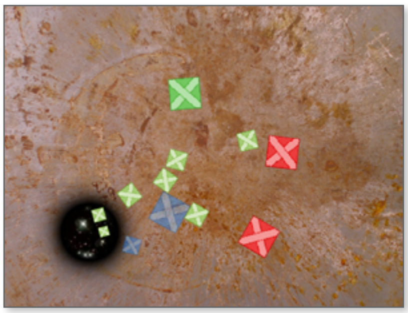
4.1. *Planning*

Intended action:	Develop a prototype of a casual game and create releases for three platforms.
Time frame:	Eighty hours of total design and development time.
Participants:	The researcher.

4.2. *Action*

A casual game named "Proximity" was created. The goal is to place miniature black holes that attract all the needed items, in order to advance to the next level.





Figures 15-17: Screen shots of the game "Proximity".

For a description of the production process, please view the research logbook (appendix on page 50).

4.3. *Key Observations*

Differences Across Platforms

The game and the level graphics were created in a 4:3 ratio. While this works for an *Apple iPad* and a web release, a further version for an *Android* tablet would need to be adapted. In order to keep the game interaction simple (tap to open a black hole), no possibilities were added to zoom or scale the map. By displaying the map at 100% and at a fixed position, however, the ability was lost to display it dynamically through a view-port[1]. For an *Android* tablet version, the layout of the levels would need to be changed manually. Kevin McCluskey, developer on the *Mozilla Firefox* team, notes that "*anyone who has written cross-platform software knows, the early ports are the most challenging*" (Logan 2008). Lakshmi Narasimhan, Senior Applications Engineer at Intel Corporation, states, during his presentation on cross platform game development, that platform capabilities should be considered during the game design phase (Narasimhan, 2009). Platform differences, such as the screen dimensions, will thus be considered at an earlier stage during the next AR cycle.

1 The area in which a scene is displayed. Similar to the boundaries of a camera (W3C, 2012).

Also, it was observed that cross-platform development was not the central issue in the process. Since the technical problems of cross-platform development are covered fairly well by the development tools, efforts could be focused elsewhere.

Tools

The production time was double the expected amount. Much time was lost early in the production process, while getting the physics and graphics software frameworks to function together. In the next AR cycle, more measures should be taken to cope with technological difficulties. The proof of concept demos needs to be more thorough, in order to avoid problems during production.

4.4. Creating a Model

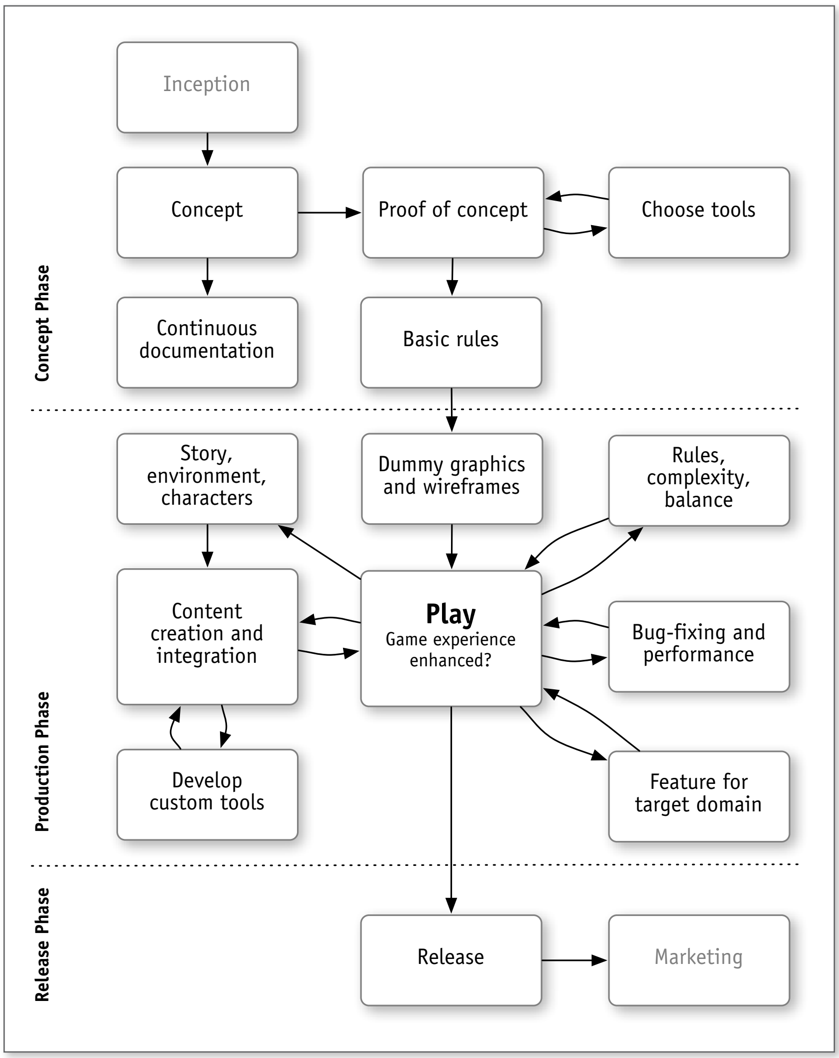

Concept Phase
Production Phase
Release Phase
Inception
Concept
Proof of concept
Choose tools
Continuous documentation
Basic rules
Story, environment, characters
Dummy graphics and wireframes
Rules, complexity, balance
Content creation and integration
Play
Game experience enhanced?
Bug-fixing and performance
Develop custom tools
Feature for target domain
Release
Marketing

Figure 18: A conceptual model during the first AR cycle.

The transformation process shown in this first version of the CM, as described in section 3 on page 15, starts with the game concept at the top left and ends with a release at the bottom. For brevity, I will refer to this conceptual model as the 'first CM' throughout this section.

The process is divided into three phases: the concept phase, the production phase and the release phase. One could argue that a game design phase is missing. In the first CM, creating and adapting the game design is a constant process overlapping both the concept and the production phases.

Concept Phase

In order to reduce the risk of having a game that is not fun or that is not technologically viable, it is beneficial to have a playable version of the idea as soon as possible (Perrin, 2012).

Before reaching a playable version, the overall concept and core mechanics of the game should be designed. Also, the toolset should be chosen and tested by means of a proof of concept. This is the first point that involves iteration: testing software and frameworks in the form of demos until the correct tools for the job are set. The concept phase is concluded when the very first version of the game can be played.

Production Phase

During the creation of Proximity, the actions always lead to the same point: playing the game. Every change that was made would lead to having a playable portion of the game. For instance, creating and integrating an animation, then playing the game to asses the visuals. Or, a new type of game object is introduced, i.e. boxes that need to be gathered, then assessing the gameplay. Often having a playable version of the game has had the following benefits during production:

- It motivates to continue one's work;
- The game can be presented to other players in its current state;
- Snapshots of the game can be saved separately;
- It allows the creator to reflect on that current iteration.

Each time the game is changed, the creator can asses if the game experience has improved. The experience of a game is a crucial factor (Schell, 2008, p. 10; Crawford, 1984; 2011, p. 3). Narasimhan, states that *"The overall design objective should be to maximize game play experience on any given platform"* (Narasimhan, 2009). Assessing the game experience with each new feature, rather than after developing a major portion of the game, reduces the risk of making a game that is not fun to play. The iterations do not necessarily need to be an increase in complexity or features. To remove elements or to simplify the game during an iteration would also be possible.

Andrew Brownsword, Senior Software Engineer at Electronic Arts, spoke about the importance of rapid iteration during an interview:

> *"The key to building a good game is rapid iteration. There is no magic formula that makes a great game. What you want to be able to do is try things rapidly. By trying things rapidly, that means you get to experiment with what works better. The rule of thumb says the more times that you can iterate, the better. That means speeding up iteration times."* (Brownsword, 2011)

The player does not necessarily need to be the developer. It would be beneficial to have the target audience play a prototype of the game. For instance, it would be reasonable for a child to give feedback on a children's game early in the production process. Proximity was shown to coworkers after the completion of the prototype.

Release Phase

The release phase begins after the creator has determined that the game is of good quality for its audience, target domain, and computing platform. The overall game production does not necessarily end at that point. Game development may continue, if, for instance, bugs are found after the initial release. At such a point, the process returns to the production phase. In the release phase, the game is published to different channels such as app stores or social media networks. Also, the game is marketed with gameplay videos, a website, a press kit, and so forth.

Since the game can only be considered a prototype, it was not released to any distribution channels. The release phase for Proximity was limited to installing the prototype on colleagues' *iPads*.

I will now compare the first CM to the software development practices described in section 2.2.

Comparing the First CM with the Spiral Model

The Spiral Model suggests creating prototypes before developing and testing, in order to identify and resolve risks (see section 2.2). This approach was adapted for use in the first CM: Demos were created as proof of concept. The Spiral Model identifies risk once per iteration, whereas the CM evaluates risk after each feature implementation.

Comparing the First CM with Scrum

The Backlog artifact is a list of prioritized features used in Scrum (see section 2.2). A backlog was also made during the creation of Proximity. It was primarily used for documentation purposes, since there were no extra stakeholders involved in the process. Like Scrum, updating the backlog in the first CM is decoupled from the rest of the process.

Scrum's incremental approach to optimize predictability and control risk (Schwaber & Sutherland, 2011, p. 4) has also been adapted to be used in the first CM. Rather than demonstrating the functionality at the end of a sprint, the game is played with each additional feature.

Comparing the First CM with FDD

Developing an overall model is the first phase of FDD (see section 2.2). This can loosely be compared to creating the game concept at the beginning of the first CM. The feature list, as part of the second phase of FDD, is similar to the backlog artifact used in Scrum. As described, the documentation of the project was decoupled from the rest of the process, as this should happen continuously. The final phase of FDD is building the product feature by feature (see section 2.2). The production phase of the CM is in the same manner. However, the model is more specific about the types of features that can be implemented.

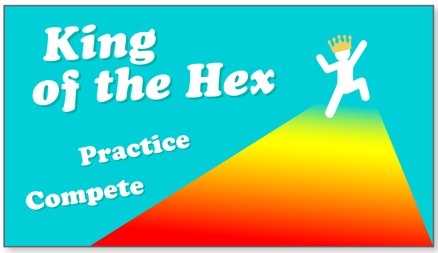
5. *Second Cycle of the Action Research Process*

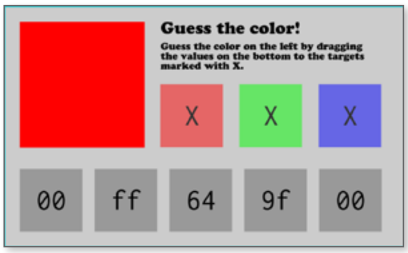
5.1. *Planning*

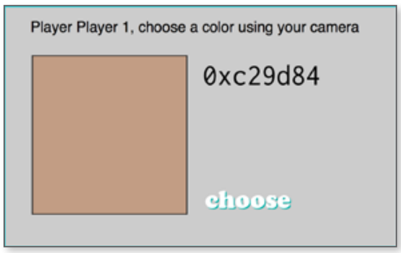
Intended action:	Develop a prototype of a casual game and create releases for three platforms.
Time frame:	Eighty hours of total design and development time.
Participants:	The researcher.

5.2. *Action*

A casual game named "King of the Hex" was created. The goal of the game is to guess the hexadecimal value of a displayed color. A multiplayer option allows a player to record a color, using a camera for the opponent to guess.




Guess the color!
Guess the color on the left by dragging the values on the bottom to the targets marked with X.
X
X
X
00
ff
64
9f
00



Figures 19-21: Screen shots of the game *King of the Hex*.

For a documentation of the production process, please view the research logbook (appendix on page 50).

5.3. *Key Observations*

Proof of Concept

Upon sketching the concept, two core elements of the software were specified. As was the case with Proximity, demos were made to prove that the chosen technology is viable. The first demo was to record a color, using the device's camera. The second was to test the user interface for choosing a color. Completing the complicated parts of the application first, reduced the risk of having to abandon an idea later during production. For instance, it would be better to check if the camera can be accessed correctly on a smartphone during the concept phase, rather than during a release.

Documentation

Unlike the first AR cycle, no backlog or separate documentation was written. Instead, the documentation was added where it was needed: clear comments in the code, logical directory structure and filenames, and so forth.

Refactoring

Reorganizing code in a logical manner becomes necessary with added complexity. This is known as refactoring. Martin Fowler defines it as *"the process of changing a software system in such a way that it does not alter the external behavior of the code yet improves its internal structure"* (Fowler et al., 1999).

The code structure needed to change upon creating the multiplayer version of the game. To avoid duplicate code and different game behavior, the game flow was written to allow variable amounts of players and rounds. This was an unforeseen amount of work with no direct player benefit.

5.4. *Creating a Model*

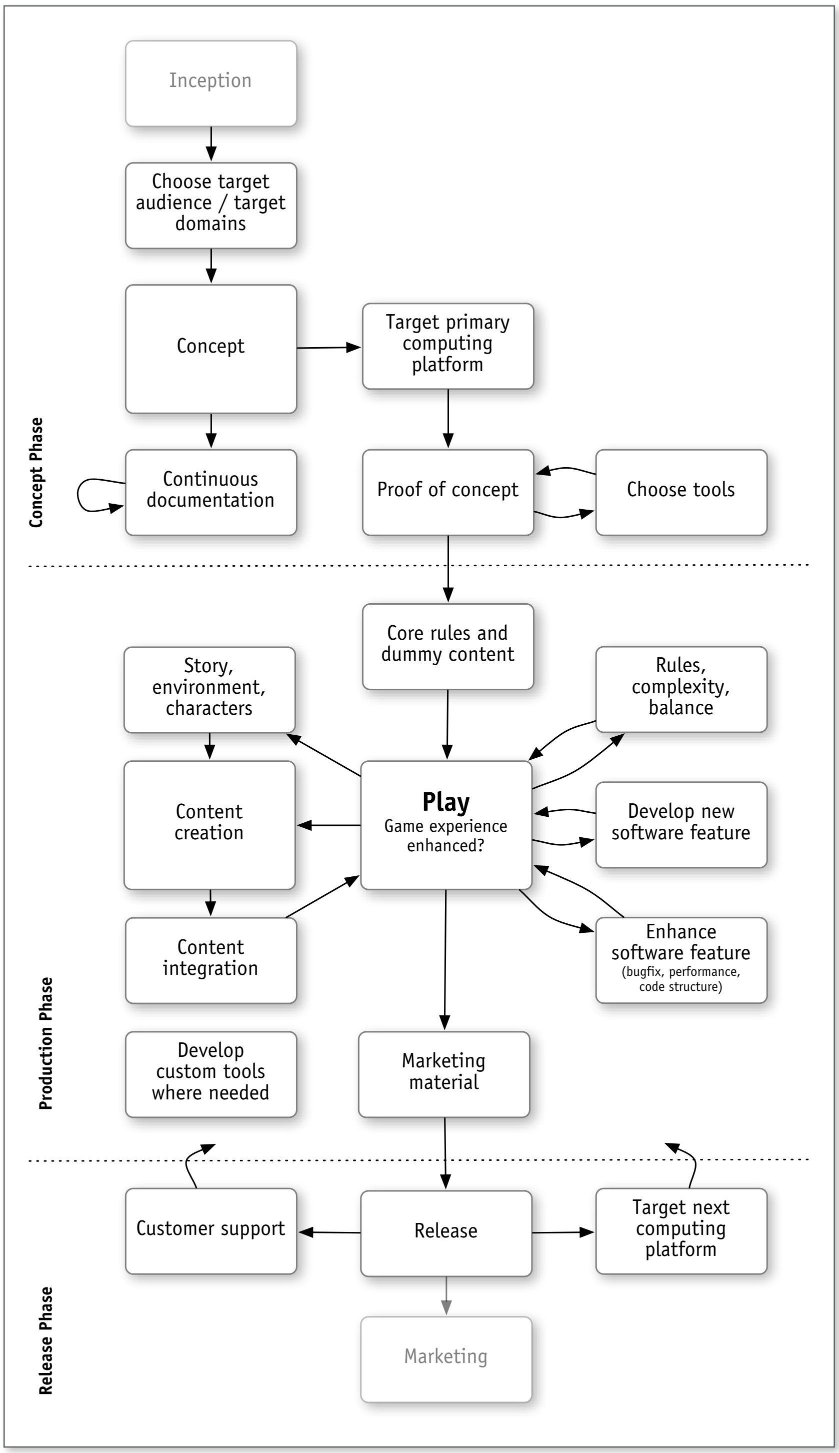

Inception
Choose target audience / target domains
Concept
Target primary computing platform
Continuous documentation
Proof of concept
Choose tools
Concept Phase
Core rules and dummy content
Story, environment, characters
Rules, complexity, balance
Content creation
Play
Game experience enhanced?
Develop new software feature
Content integration
Enhance software feature
(bugfix, performance, code structure)
Develop custom tools where needed
Marketing material
Production Phase
Customer support
Release
Target next computing platform
Marketing
Release Phase

Figure 22: A conceptual model during the second AR cycle.

Concept Phase

Extending the CM from the first AR cycle, there are further steps that require attention during the concept phase.

Audience:	The target audience should be defined. A specific grouping of people may aid in designing the game. If the game is for children, for instance, the user interface, language and game rules should be kept

simple. If the game is targeting a group of professionals, as is the case with King of the Hex, the game mechanics may contain game rules relating to the trade.

Domains: The target domains define the surroundings in which the player will potentially be playing. For example, will the game be played while in transit or will it be played in an art exhibition. This decision influences the choice of the hardware and computing platforms. It also influences the choice of players and the way people interact with the game. King of the Hex can be played by multiple people on the same device – outside or inside an office setting. Thus, the choice of hardware is mobile devices and desktop computers.

Platform: A primary computing platform should be chosen, since this dictates what tools can be used. The CM from the first AR cycle did not present this concept clearly. The approach in the second CM is to develop the game for a single target platform first, and then introduce new platforms at a later stage.

Production Phase

As observed, no backlog was utilized. Instead, the choice of the next feature was made upon completing the previous one. The concept of deferring a choice until the last possible moment is part of the Agile methods Lean Software Development (Poppendieck & Poppendieck, 2003, p. 59) and Extreme Programming (Wells, 1999).

> *"**Defer implementation of future capabilities:** Implement only the simplest code that will satisfy immediate needs rather than putting in capabilities you 'know' you will need in the future. You will know in the future what you really need then, and simpler code will be easier to extend when necessary."*
> (Poppendieck & Poppendieck, 2003, p. 59)

The opposite, although luring, would not be recommendable. For instance, creating the cutscenes for a game all at once may reduce production time. However, the efforts may be lost, depending on later factors, e.g. a new deadline, a change to the story, or a change to the game design. In such a case, it is suggested to treat each cutscene as a feature and extend the game as it progresses.

Creating content and integrating content are two separate steps. The content creator should produce assets in high quality. Later, these assets can be optimized for use in the game. In Proximity, the animations were first created in Adobe Flash, then later exported for use within the game development framework.

Custom tools were not needed during the creation of King of the Hex. Unlike the production of Proximity, such tools are not bound to content creation only. A custom tool that checks a game's performance with many on-screen enemies would be a possibility. The link between custom tools and content creation has been removed in the second CM.

Release Phase

As mentioned in the concept phase, a new computing platform can be introduced after completing a release. Producing a game for multiple computing platforms at the same time is possible, but may lead to unnecessary complexity. The flow of the CM discourages this behavior by placing the step towards the end of the process. This was not sufficiently visualized in the first CM. A loop that returns to the production phase was added after the release.

Comparing the Second CM with the Spiral Model

The Spiral Model presents a cyclic nature of development throughout its different stages: The concepts are a cycle, prototypes one and two are cycles, and the operational prototype is a cycle. What the Spiral Model does not express, however, is iteration on a feature-by-feature scale. The features are added in chunks between the cycles. The second CM includes iteration during the concept phase, where the proof of concept is created, and the production phase, where the game gets played.

Another quality of the Spiral Model is the emphasis on planning and reflection. Determining objectives, identifying risks, and planning the next iteration make up three quarters of each iteration, leaving the action portion fairly small. The core element of the second CM, i.e. playing the game, serves multiple purposes including identifying risks. From that point on, there is –in comparison– more focus on action. Planning the next iteration is a choice that is made after having a playable version.

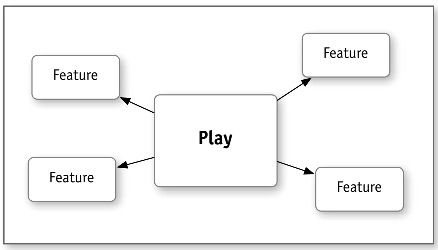


Figure 23: Choosing the next feature right after playing the game.

Comparing the Second CM with Scrum

No backlog was made during the production of King of the Hex. With no other stakeholders and no additional developers involved, communicating the features in writing was not necessary. Making a simple to-do list as a reminder would, instead, suffice. Generally, when compared with Scrum, all the communication events are missing. In a small or one person team that can cover the entire production, there is no need to put mandatory meetings into place.

Comparing the Second CM with FDD

Building a features list is the second step in FDD (see section 2.2), similar to creating a backlog in Scrum. As argued, this is missing in the second CM.

Rules for creating a development plan are displayed in the third step of the FDD process (see section 2.2). The second CM does not present any planning of this sort: No deadline is set during the concept phase. Considering that one of the goals of the CM is to reduce overall production time, this would be an obvious inclusion:

Target a date → Production → Release on set date

However, to release during a specific time frame does not mean that there was less production time. It was simply fitted into that time space (e.g. producing during evenings and weekends). In the second CM, I propose to lift this constraint. Following the CM, the creators should instead concentrate on reaching the desired goal of having a fun game ready for release. It could be the case, however, that a game needs to be shipped at a certain date: when, for example, a time frame was defined for marketing. This would be possible, because each iteration should end in a nearly shippable version of the game before the next feature is implemented. If the CM is followed correctly, a version of the game would be ready at that moment.

6. Third Cycle of the Action Research Process

6.1. Planning

Intended action:	Develop a prototype of a casual game and create releases for three platforms.
Time frame:	Eighty hours of total design and development time.
Participants:	The researcher.

6.2. Action

A casual game named "Flick Kick Soccer" was created. The goal of the game is to shoot goals with a soccer ball by swiping it from one screen to the next.

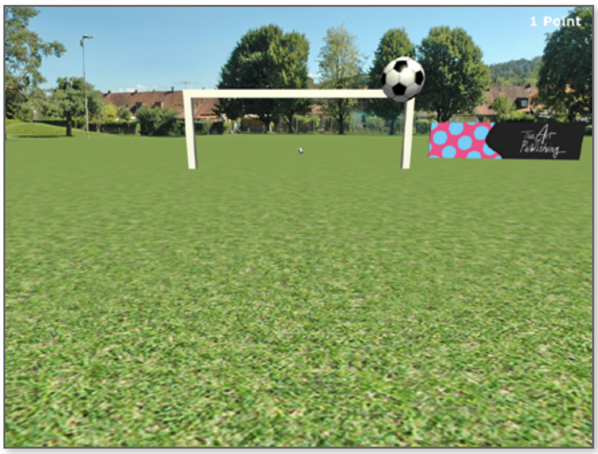


Figure 24: Screen shot of the game *Flick Kick Soccer*.

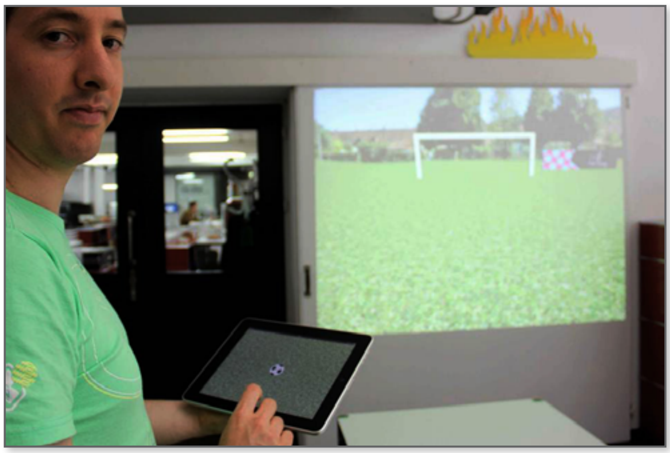


Figure 25: *Flick Kick Soccer* being played on an iPad and video projector.

For a documentation of the production process, please view the research logbook (appendix on page 50).

6.3. Key Observations

Game Design

The game design is simple, because the mechanics are based on a well known sport. This is an advantage, since all the players are already familiar with the rules. Some additional mechanics were added later. For instance, hitting the billboard next to the goal makes it fall over.

Proof of Concept

Due to some technological hurdles, such as real-time communication between devices, the proof of concept stage took up a major portion of the overall production time. Three separate demos were necessary to prove that the game would work.

Release Phase

A fixed deadline could be met, because each new feature required a working version of the game. On the day of the release, the latest functioning version was ready for use.

A later release on the open web took a surprising amount of effort, due to complex server configurations. Unlike Proximity and King of the Hex, the release phase took up another large portion of the overall production time.

6.4. *Creating a Model*

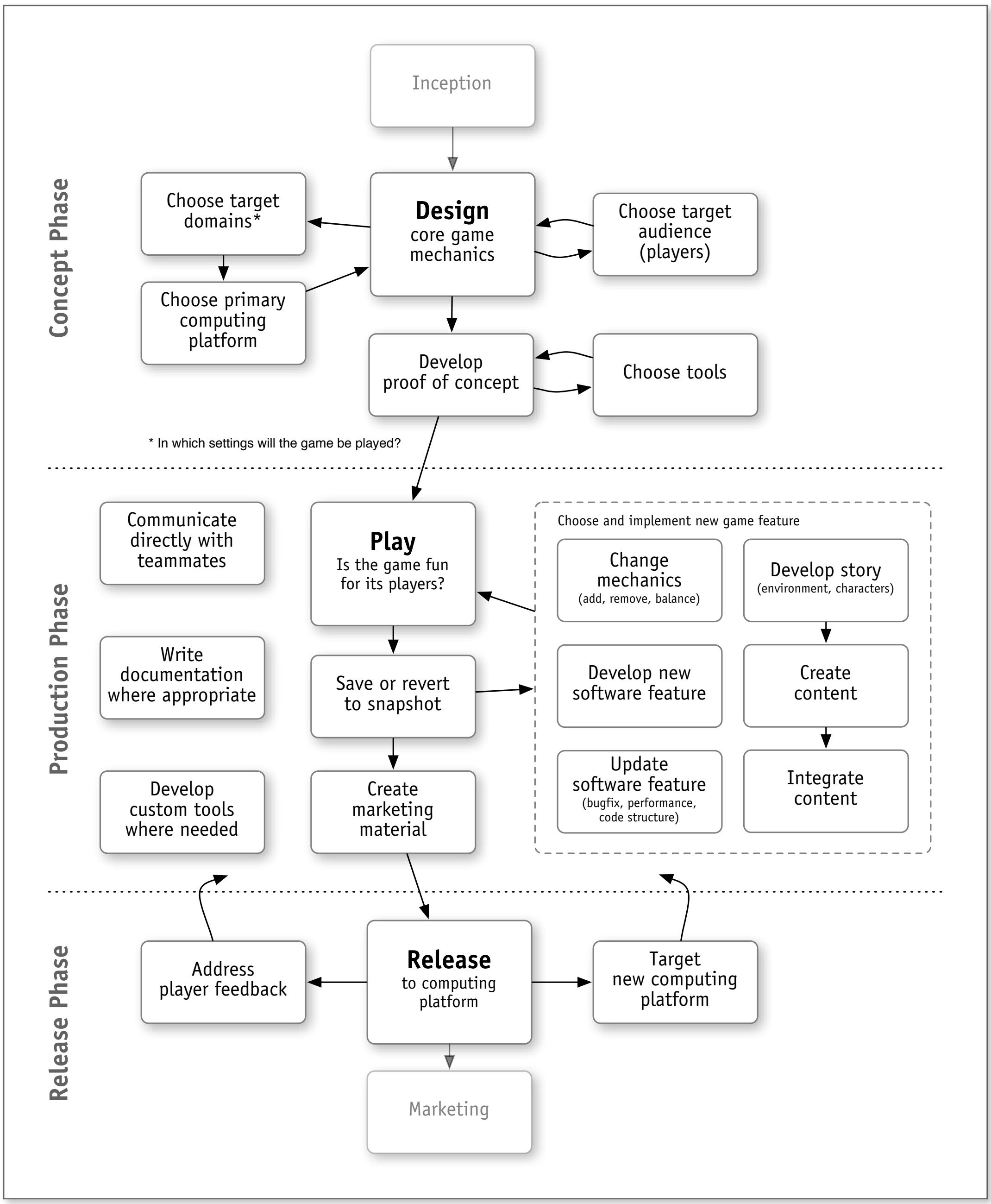

Concept Phase
Inception
Choose target domains*
Design core game mechanics
Choose target audience (players)
Choose primary computing platform
Develop proof of concept
Choose tools
* In which settings will the game be played?
Production Phase
Communicate directly with teammates
Play Is the game fun for its players?
Choose and implement new game feature
Change mechanics (add, remove, balance)
Develop story (environment, characters)
Write documentation where appropriate
Save or revert to snapshot
Develop new software feature
Create content
Develop custom tools where needed
Create marketing material
Update software feature (bugfix, performance, code structure)
Integrate content
Release Phase
Address player feedback
Release to computing platform
Target new computing platform
Marketing

Figure 26: A conceptual model during the third AR cycle.

The descriptions were changed to verbs expressed in the imperative, as suggested by Wilson (2001, p. 12).

Concept Phase

Three decisions should take place while designing the core game mechanics: the choice of the target domains, computing platform and target audience. The choice of the target domains influences the choice of the computing platform. For instance, with Flick Kick Soccer, the target domain is an office or party setting. Players can join in with their own smartphones. The target computing platform is the web. Therefore, no applications need to be installed beforehand. The target audience are visitors, aged 25 to 65. These decisions also dictate the game mechanics. Not all clients are avid gamers. Thus, the game rules should be kept simple.

Upon making these choices, the technological aspects become important. The correct tools are chosen while creating proof of concept demos. Also, the core functionality of the software needs to be addressed at an early stage.

As opposed to the previous two models, the documentation was moved to the production phase. During the concept phase, any game design scribbles and proof of concept demos are documenting the current status of the work.

Production Phase

There is a correlation between the increase in complexity of the game mechanics and the increase of complexity of the game production. What games often have in common, is that they get more intricate as the game progresses. In Proximity, for instance, new game objects are introduced as the game progresses. In King of the Hex, color values get more complicated. At the same time, the development effort increases, in order to implement these changes. More software needs to be written, more art assets are added, and so forth. An efficient way to produce a game is to complete its components in roughly the order in which they would be played. As was previously argued, there is no need to add a feature if the player is not able to reach it. If the production time is stopped short, as was the case with King of the Hex, the game will simply be less complex and takes less time to play through.

> *"The definition of a good game is therefore 'one that teaches everything it has to offer before the player stops playing.'"*
> (Koster, 2010, p. 46)

Through playing the game upon implementing a feature, there is some guarantee that it will be available to the end user. Unless, the change is not considered fun and is removed from the game. This is a new addition to the third CM: After the players of the game asses the newly added feature, the production should either be saved completely in its current state or rolled back to its previous state. With a version control system, such as *Git* (2012) or *Subversion* (Apache, 2012), this is a simple task.

Communication was a missing element that was discussed during the comparison of the second CM to Scrum and FDD on page 29. A direct method of interaction between the game creators is suggested. This can be achieved by sitting together in an office or having an open Voip[2] channel, rather than sending documents and holding meetings. In the documentary, "Indie Game: The Movie", direct communication within teams of two is presented throughout the film (Pajot & Swirsky, 2012). Tommy Refenes, one of the developers interviewed in the movie, commented on the constant interaction with his team mate: *"Edmond [the team mate] and I talk all the time about everything."* (Refenes, 2012).

The aspect of communication will be revisited when analyzing the data from the interviews.

Release Phase

Apart from some formal changes, the release phase has not been changed from the second CM.

Comparing the Third CM with the Spiral Model

As described in the comparison of the second CM to the Spiral Model on page 28, there is more emphasis on planning, determining objectives and identifying risks than on actual development. Upon saving or reverting to a snapshot in the third CM, the visuals for choosing the next feature have been simplified.

Comparing the Third CM with Scrum

There is a correlation between user stories, a feature written from the perspective of a user (Shojaee 2012), and the Play element of the CM. When the players of the game (the customers) give feedback, new user stories are created. This feedback can directly influence the choice of the next game feature or the removal of an existing one.

During a sprint, the production team implements a given set of features from the backlog (see page 9). At the end of the sprint, a product should be available in a near shippable state. In the third CM there is no such grouping. Every feature implementation should lead back to a playable game. This could be a problem if a single feature takes a long time to complete. Abandoning such a work intensive feature or breaking it into smaller chunks should be taken into consideration. An overly complex game mechanic, for instance, may not appeal to the casual gaming audience.

2 Abbreviation for Voice Over Internet Protocol. A transmission technique similar to telephony (FCC, 2012, p. 1).

Comparing the Third CM with FDD

At its core, the third CM is very similar to FDD: The game is planned and built by feature with a completed client-valued function at the end of each iteration. The main difference lies in the level of detail. FDD has abstracted the process of software development to five steps. The third CM is tied closely to the problem area.

7. Analysis of Data from the Interviews

7.1. Interview with DigiDingo

The interview with Jann Sigrist and Jörg Sigrist took place on the 23rd of April 2012 in their office near Zurich, Switzerland. A translation of the interview can be found in the appendixes on page 51.

Concept Phase

DigiDingo showed a clear interest in releasing to multiple domains. The target domains of their initial release were handheld and tablet devices. Jörg Sigrist commented on the differences in interaction between the two devices:

> *"On the iPad, the game is played much differently than on the iPhone. […] That is an important thought: does one create different versions with the same underlying gameplay, but with a different handling. Or does one try to make it universal […]."*
> – Jörg Sigrist

The definition of the target audience and the target platforms were grouped together as part of the game concept when the CM was first created. Based on the inputs from DigiDingo, they were separated into single steps.

DigiDingo evaluated different tools before starting the development phase. Jann Sigrist stated that they tested frameworks for over two weeks before reaching a decision.

The game design document was continuously updated throughout the process. However, according to Jann Sigrist, not much effort was put into further documentation.

Production Phase

The production of Grooh was divided among three people. Jann Sigrist did the programming, Jörg Sigrist created the levels, and their graphic artist (name not stated) designed the game assets. While the third CM does not dictate assignment, it does present the main types of features. This way, single features can be assigned to team members. Jörg Sigrist implemented the features regarding rules, complexity, and balance. The graphic artist created the content. The programming and asset integration was assigned to Jann Sigrist.

Jann Sigrist stated that the team tried to reach a playable prototype as early as possible.

> *"We tried to reach that stage [a playable version] early, because we think it is important to play, in order to find out what can be improved. It is the most important thing to play one's own game. [...] We had to redo a lot of smaller things because we noticed while playing, that it would not work."* – Jann Sigrist

As Jann Sigrist stated, playing their own game was an important part of the process.

In agreement with the 'custom tools' element of the CM, a tool was made to import level data from *Microsoft Excel* into the game.

Release Phase

DigiDingo initially created Grooh for the *iPhone*, *iPod Touch* and *iPad*. After that, they expanded to other computing platforms. The *Android* release was held back, due to technical issues with the framework (hard side of the problem area).

The intended transformation process of the CM is to have a released game (see section 3 on page 15). After that point, there is further work that needs to be done – such as marketing. DigiDingo explained that they made a mistake by releasing the game without marketing material. The process in the first CM was thus corrected, so that marketing material, such as a demo video and a press kit, must be finished prior to a release.

In addition, Jann Sigrist explained that they received feedback from players asking for help in a level, requesting a new feature, or pointing out bugs. The instances in which a return to the production phase is possible will be displayed clearer in the CM.

7.2. Interview with Jomoho

The talk with Moritz Laass and Simon Siegenthaler took place on the 10th of October, 2012 in their office in Basel, Switzerland. The full transcript can be found in the appendixes on page 61.

When referring to the CM, I shall refer to the final result of the AR cycles, unless otherwise noted.

Concept Phase

With regard to the concept phase, Laass commented on the importance of getting a proof of concept done quickly, and then building upon it:

> *"You start off with an idea. What I like to do is prove it quickly. There is a core mechanic of the game that we'd like to see if it feels good. Then we start to build around it."* – Laass

The steps from Laass' statement are in agreement with the concept phase in the CM. The goal of creating a proof of concept, in order to see *"if it feels good"*, correlates with the step of playing the game for fun. In the CM, this step is not expressed in the concept phase. However, it is the first step of the production phase. Building around the proof of concept demo is the key element of the production phase.

According to Laass, the primary computing platform of Franky Flies High, is a desktop computer. The optimizations for a tablet computer happened at a later time, but before the initial release of the game. I disagree with this method. It would be more efficient to first complete and release to the primary computing platform and then return to the game for any further device adaptations. This way, the initial release could start generating income while the next computing platform is targeted.

Production Phase

Assessing a game for its fun factor was expressed by Laass as follows:

> *"I try to be really critical about it. To say 'what could I do more?'. The whole balancing process, I do that by myself. I see what feels different if I change a variable."* – Laass

With the sentiments of 'being critical' and 'feeling different', Laass describes the artistic elements of the process. It is suggested that these parts of the CM should not be predetermined. The CM does not intrude in the artistic process: Designing the game and playing the game are merely steps. How it should be played and how it should *feel* must be expressed by the creator.

When asked about communication between team members, Laass replied that he works in the same place with his graphic artist, Simon Siegenthaler. During the interview, Siegenthaler, who was working close by, would participate in the discussion. This demonstrated direct communication within the team.

Laass writes to-do lists, similar to the backlog that was written during the production of Proximity. Feature lists were removed from the second CM. Instead, the definition of features happens before it is implemented.

At the time of the interview, Laass was working on a promotional website for the game. In the CM, this step is expressed as creating marketing material.

Release Phase

A release channel, such as *Google Play* or the *Apple App Store*, allows updates to already released software. Laass explained how "*a game can grow*" after its first release. Based on sales statistics and player feedback, updates can be made after the initial release.

At the time of the interview, Laass had not yet decided if he would do the marketing himself or appoint the task to a third party.

Game support after the initial release would depend on customer feedback:

> *"If there are many people that like the game and want more. It's an obvious choice to invest in it, because you don't want to let those people down. Also, it's good if something grows."* – Laass

The CM allows returning to the production phase for an update, based on player feedback and sales statistics.

7.3. Interview with Nothing Interactive

The interview with Bastiaan van Rooden and Mark Gruber took place in their office in Bern, Switzerland, on the 12th of October, 2012. The translation of the interview can be found in the appendixes on page 51.

During the interview, Agile and Scrum concepts were mentioned as components of their development process. The workflow also contains their own methods. For instance, a method for detailed time tracking and analysis. Van Rooden stated that his company has put much thought into the workflow over the years.

Concept Phase

During the AR cycles, the CM was created by a single person. What is missing in the conceptual phase, is the assembly of a team. Van Rooden describes his team setup as follows:

> *"The normal set up is: one designer, one developer and someone that does the concept. Then, during the test phase, we commission someone to test the game so we do not have to do it ourselves." – van Rooden*

This corresponds to the maximum team size of the CM.

Production Phase

The team creates a prototype, in order to evaluate the game's fun factor:

> *"You can make a scribble but then you cannot know if it will work – if it is going to be fun. If the prototype is not fun, then you can forget about the project."* – van Rooden

> *"[...] we would create a simple prototype that was made for the sole purpose of interaction. This would allow you to test the fun factor."* – Gruber

The proof of concept created in the CM is a demo and not yet a playable prototype. It is created for testing the technological possibilities of the chosen computing platform and tools. The prototype suggested by van Rooden and Gruber is an early snapshot in the production phase of the CM. All three games created during the AR cycles are considered to be prototypes. They are playable early versions of the game in a nearly releasable state.

The following statements substantiate the position that a game needs to be fun. At the same time, they show how the term is perceived differently:

> *"A brilliant idea does not determine if the game is going to be fun to play. And in the end, the game has to be fun."* – van Rooden

> *"To me, fun is when the consumer reaches that attention span you were hoping to achieve from your game."* – Gruber

There was no mention of the rate in which fun is evaluated. In hindsight, a question regarding the frequency of assessment should have been posed.

The interviewees create versions of their game and consider returning to it in some circumstances:

> *"But sometimes there are changes of direction where you have to decide if you want to go back two steps and start fresh."* – van Rooden

While custom software-tools are not often created, the interviewees try to reuse game code via custom frameworks and web services. They also try to improve their tool chain with each new project.

Release Phase

Nothing Interactive does not market their own games. They leave this to other agencies. Van Rooden acknowledges the importance of marketing after a release.

It is noteworthy, that the company prefers to release their games to a single platform first. The reason for this, is that there is usually a bug-fixing period that needs to be attended to immediately after the release:

> *"You have to schedule a minimum of one week extra after a product launch where somebody has to be available at all times."*
> – van Rooden

Apart from this, there was a discussion about security aspects of the game and how cheating can be avoided. Although this is basically a software update, security will be added to the CM as an element in the release phase.

8. Conclusions

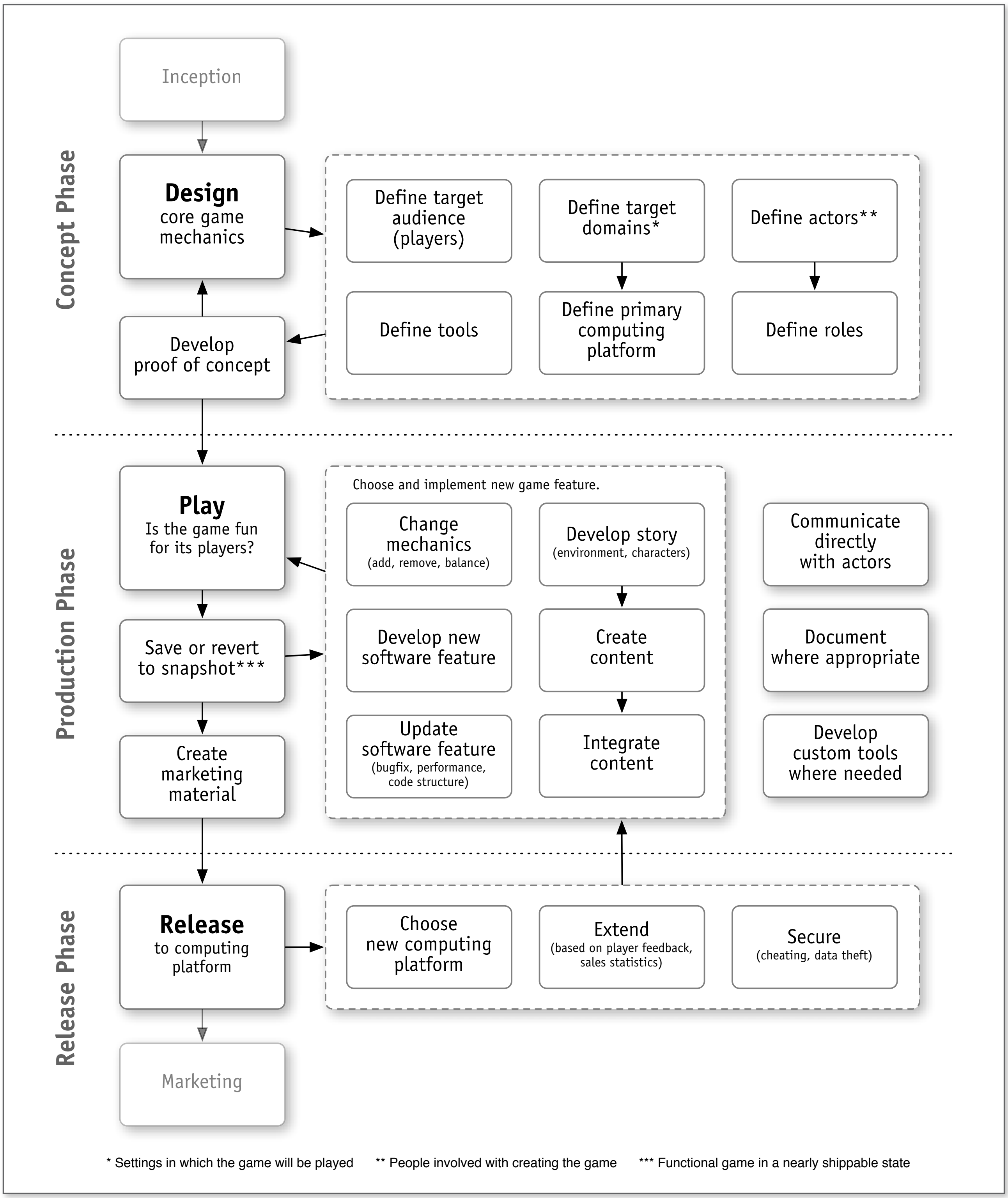

Concept Phase
Inception
Design core game mechanics
Define target audience (players)
Define target domains*
Define actors**
Develop proof of concept
Define tools
Define primary computing platform
Define roles
Production Phase
Play Is the game fun for its players?
Choose and implement new game feature.
Change mechanics (add, remove, balance)
Develop story (environment, characters)
Communicate directly with actors
Save or revert to snapshot***
Develop new software feature
Create content
Document where appropriate
Create marketing material
Update software feature (bugfix, performance, code structure)
Integrate content
Develop custom tools where needed
Release Phase
Release to computing platform
Choose new computing platform
Extend (based on player feedback, sales statistics)
Secure (cheating, data theft)
Marketing
* Settings in which the game will be played
** People involved with creating the game
*** Functional game in a nearly shippable state

Figure 27: The Design Play Release Model.

The model displays an organized approach to creating a cross platform casual game. Wilson, while describing a model he created, states that it *"represents a concept, the manifestation of which in the real world would work"* (Wilson, 2001, p. 17). This model, if followed correctly, would work in the real world. Further applications of the model are needed, as I will explain shortly. First, I will describe the model and present my key findings.

8.1. Model Description

The CM corresponds to the Agile Manifesto (see section 2.2):

> *"**Individuals and interactions** over processes and tools;*
> ***Working software** over comprehensive documentation;*
> ***Customer collaboration** over contract negotiation;*
> ***Responding to change** over following a plan."*
> (Beck et al., 2001 a)

Individuals and interactions:
Small teams can keep a direct communication channel open, as to reduce the overhead of controlling the information flow or managing the team. It is not recommended to predefine how the interactions in the team should occur.

Working software: Snapshots of a functional game in a nearly shippable state are the desired outcome of each iteration during the production phase.

Customer collaboration:
The game players are the customers. Collaboration arises from having a game play-tested by the target audience during production, and listening to feedback after a release.

Responding to change:
A new game feature is only chosen after the completion of the previous one. A new computing platform is only chosen after a game release. The deferral of choices allows quicker response to external influences.

Further attributes of the model are:

One game per process:
The model should be used on a per game basis. However, the model can be applied repeatedly for creating multiple games in parallel. With the Digi Dingo team, for instance, a new game concept was made during the release phase of their current game.

Creative freedom: Artistic expression, as stated by Laass in the Jomoho interview (see page 61) is a fundamental part of creating games. The CM avoids an intrusion of the artistic process. Descriptions, such as "Create Content", are thus kept simple, in order to display *when* a certain step should be taken, and not *how*.

Safeguards:	Steps were added to the process, so that common pitfalls such as technological limitations, unnecessary features and premature game releases can be avoided.

8.2. *Key Findings*

Cross Platform Development

During the proposal of the research, the believed solution for creating cross platform games was to develop towards the lowest common denominator first, then to release to multiple platforms. Yet, this view changed already during the execution of the first AR cycle. From a business perspective, it is advisable to have a version of the game on the market as soon as possible. The game starts generating income and possibly reaches players ahead of the competition. From a production perspective, it would be advisable to separate releases (as discussed in the Nothing Interactive interview), so that enough human resources are available for customer support.

Reducing Risks

The production time can be shortened by avoiding unnecessary work:

Creating game demos as a proof of concept helps avoid technological dead ends later in the process. With King of the Hex, for example, the risk of not being able to record color with the smartphone's camera was eliminated before commencing production.

The tie between the increase of game complexity and production complexity also reduces risk. With Flick Kick Soccer, a snapshot of the game was ready to be played prior to the deadline. New features were continuously added and tested. Deferring the decision of the next feature until the last possible moment, a concept known from Agile practices (see section 5.4 on page 26), helps avoid work that may never be used.

Marketing material, such as a website or press kit, must be created before the game is released. As discussed in the Digi Dingo interview (see page 51), this reduces the risk of creating a faulty perception of the game.

Focus on Fun

Creating better games was not a reason for making the CM, yet an approach was recognized while participating in AR: The rapid addition and removal of game features based on the notion of fun increases the chance of making a good game. This flexibility is a potential advantage which indie game developers have over large production studios. Small teams enjoy more creative freedom: they can "*differentiate on gameplay*" (Swain, 2008), and there is less incentive to keep a feature because of its production cost.

Making a game playable as quickly as possible, and continuously improving on it, seems to be a common approach among game developers. The interviews with Digi Dingo, Jomoho and Nothing Interactive all contain statements of this nature (see section 11 on page 50). Implementing this approach worked well during the creation of Flick Kick Soccer, where a game version was ready well before the presentation deadline.

8.3. ***Conclusions Regarding the Research Process***

In order to validate the hypothesis, sufficient primary data could be gathered by means of participatory AR and on-site interviews. Creating and applying the model repeatedly, exposed inconsistencies and allowed for optimization.

The order in which the AR cycles and interviews where conducted was not optimal. The original design of having an interview follow an AR cycle did not work, because the interviews could not be timed appropriately. Two interviews took place after the AR cycles were completed. Thus, the questions for these two interviews were similar, reflecting the current understanding of the subject matter.

As only became apparent during the talk, the third group of interviewees, in some cases, acted outside the scope of the research. While the company was not dependant on publishers, some of its projects was funded externally for advertising and educational purposes. Because the CM is geared towards game creators without external stakeholders, this aspect was avoided during the discussion. As the interviewees have years of experience in casual game development and are active IDGA members, the data is considered valuable nonetheless.

During both the planning and the data analysis of the interviews, personal bias was avoided, by evaluating contrasting views. Changes were made to the model where appropriate.

The time schedule from the research proposal could not be kept. This is mainly due to the misestimation of the workload.

8.4. Recommendations for Further Research

The recommendations for further research are ordered by relevance:

1. The game should be applied to a full game production so that it can be tested in a real world scenario.
2. To test if the concepts are represented effectively, the CM could be presented to indie game developers in the form of a questionnaire.
3. Comparing the CM with development methods other than the Spiral Model, Scrum and FDD may expose flaws.
4. The CM could be applied while participating in Ludum Dare or similar competitions, in order to test if the model is applicable during a very short time frame.

8.5. Conclusions Regarding the Hypothesis

The Design, Play, Release Model is a tangible outcome of the research process. This report and the accompanying logbook have shown how games were created by implementing the model. The hypothesis can thus be verified, that the model is a practical tool for creating publisher-independent cross-platform casual games in small teams. However, the model needs to be applied further for it to become an accepted method within the community of independent game developers.

> *"[...] only a small percentage of indie game developers can make a living with indie game development. Some people just enjoy creating games until everyday life creeps up behind them."* – van Rooden

It remains to be seen if the model can help achieve the overall goal that was described in the beginning of this report: That the production time can be decreased through optimization and, at the same time, the reach of the game can be extended across multiple distribution channels. This will hopefully result in an improved return of investment for indie game developers, making the creation of casual games not only an artistic experience but also a lucrative business.

9. Bibliography

- Booth C., Colomb G. & Williams M. (2003), *The Craft of Reseach*. Chicago: The University of Chicago Press.

- Checkland P. (2012), *The Origins of SSM*. Lancaster: Lancaster University.
  Available at: http://www.youtube.com/watch?v=XA2i1n-o9L0 [Accessed 27th December 2012]

- Cryer P. (2000), *The research student's guide to success*. Milton Keynes: Open University Press.

- De Luca J. (2005), *Agile Software Development using Feature Driven Development (FDD)*. Available at: http://www.nebulon.com/fdd/index.html [Accessed 20th April 2012]

- Dibona C. et al. (2006), *Open Sources 2.0: The Continuing Evolution*. Sebastopol: O'Reilly Media Inc.

- Fullerton T. (2008), *Game Design Workshop: A Playcentric Approach to Creating Innovative Games*. Boca Raton: CRC Press, Taylor & Francis Group.

- Gyger D. (2004), *Feature-Driven Development*. Department of Informatics, University of Zurich. Available at: https://files.ifi.uzh.ch/rerg/amadeus/teaching/seminars/seminar_ws0304/08_Gyger_Fdd_Ausarbeitung.pdf [Accessed 26th Decebmer 2012]

- High Noon Studios (2006), *A Portrait – Scrum*. Available at: http://www.youtube.com/watch?v=UT4giM9mxHk [Accessed 8th August 2011]

- Levinson P. (1999), *Digital McLuhan: a guide to the information millennium*. London: Routledge.

- Maxwell Chandler H. (2010), *The Game Production Handbook*. 2nd ed.
  Sudbury: Jones and Bartlett Publishers LLC.

- Middlesex University (2011), *Referencing*. Available at: http://libguides.mdx.ac.uk/content.php?pid=58962&sid=1894149 [Accessed 24th August 2011]

- Oxford Ditionary (2011), *Definition of method*. Available at: http://oxforddictionaries.com/definition/method?rskey=uvhGpA&result=1#m_en_gb0515140 [Accessed 9th April 2011]

- Raymond E. (2001), *The Cathedral and the Bazaar*. Sabastapol: O'Reilly Media Inc.

- Reineke B. (2012), *Feature Driven Develoment*, Munich: Grin.

- Riel, M. (2010), *Understanding Action Research*.
  Center For Collaborative Action Research, Pepperdine University.
  Available at: http://cadres.pepperdine.edu/ccar/define.html [Accessed 10th August 2011]

- University of Ballarat (2010), *Research Goal*. Available at: http://www.ballarat.edu.au/ard/ubresearch/rgso/goal.shtml [Accessed 28th February 2011]

- Unsworth J. (2005), *New Methods for Humanities Research*. Available at http://www3.isrl.illinois.edu/~unsworth/lyman.htm [Accessed 14th March 2011]

10. References

- Anderson D. (2004), *Feature-Driven Development: towards a TOC, Lean and Six Sigma solution for software engineering.* Microsoft Corporation. Available at: http://www.agilemanagement.net/AMPDFArchive/Feature_Driven_Development_-_towards_a_TOC__Lean__Six_Sigma_solution_v1_0.pdf [Accessed May 1st 2012]

- Andrews C. (2006), *If it's cool, creative and different, it's indie*. Available at: http://articles.cnn.com/2006-09-19/entertainment/indie.overview_1_term-indie-ryan-schreiber-label [Accessed 4th April 2012]

- Apache (2012), Subversion: Enterprise-class centralized version control for the masses. Available at: http://subversion.apache.org/ [Accessed July 4th 2012]

- Beck et al. (2001) a, *Manifesto for Agile Software Development*. Available at: http://www.agilemanifesto.org [Accessed 14th June 2011]

- Beck et al. (2001) b, *Principles behind the Agile Manifesto*. Available at: http://agilemanifesto.org/principles.html [Accessed 29th August 2011]

- Bierhoff et al. (2009), *Frameworks on the Rise*. Available at: http://www.cs.cmu.edu/~kbierhof/papers/frameworks.pdf [Accessed 15th Feb. 2012]

- Blaxter L., Hughes C. & Tight M. (2001), *How to Research, Second Edition*. Berkshire: Open Universtiy Press.

- Boehm B. (1986), *A Spiral Model of Software Development.* TRW Defense Systems Group. Available at: http://www.cs.umd.edu/class/spring2003/cmsc838p/Process/spiral.pdf [Accessed 24th August 2011]

- Bouman W. et al. (2008), *The Realm of Sociality: Notes on the Design of Social Software*. University of Amsterdam. Amsterdam: Department of Information Management.

- Braben D. (2012). *Has Game Development Gone Full Circle?*, Presentation at Game 12. Imperial College London.

- Brownsword A. (2011) *Episode 175: Game Development with Andrew Brownsword*. Interviewe by Markus Völter for SE Radio, 6th May. Available at: http://www.se-radio.net/2011/05/episode-175-game-development-with-andrew-brownsword/ [Accessed 19th 2012]

- Checkland P. & Scholes J. (1999), *Soft Systems Methodology: A 30-Year Retrospective*. Chichester: John Wily and Sons.

- Crawford C. (1984;2011), *The Art of Computer Game Design: Reflections of a Master Game Designer.* Emeryville: McGraw-Hill Osborne Media

- De Luca J. (2012), *FDD Process*. Available at: http://www.nebulon.com/articles/fdd/latestprocesses.html [Accessed May 1st 2012]

- DigiDingo (2012), *Grooh Presskit*. Available at: http://grooh.digidingo.com/downloads/presskit.zip [Accessed June 12th 2012]

- Ephram J. (2008), *Picture of Scrum*. Flickr (Creative Commons license). Available at: http://www.flickr.com/photos/ephramjames/2661080677/ [Accessed 25th August 2011]

- FCC (2012), *Consumer Guide: Voice Over Internet Protocol (VoIP)*. Available at: http://transition.fcc.gov/cgb/consumerfacts/voip.pdf [Accessed 23rd December 2012]

- Fowler M. et al. (1999), *Refactoring: Improving the Design of Existing Code*. Reading: Addison Wesley Longman, Inc.

- Git (2012), *Open Source Distributed Version Control System*.
  Available at: http://git-scm.com/ [Accessed 4th July 2012]

- Gnade M. (2010), *What Exactly Is An Indie Game?* Available at:
  http://www.indiegamemag.com/what-is-an-indie-game/ [Accessed 6th April 2012]

- Goss O. (2011), *The iOS Game Revenue Survey*. Available at: http://www.streamingcolour.com/blog/2011/09/19/the-ios-game-revenue-survey/ [Accessed 16th February 2012]

- Griffiths M. (1998), *Educational Research for Social Justice: Getting off the Fence*. Buckingham: Open University Press.

- IDGA (2012), International Game Developers Association, Switzerland Chapter: Community. Available at: http://igda.ch/community [Accessed 7th December 2012]

- Johnson R. (1997), *Frameworks Home Page*. Available at:
  http://st-www.cs.illinois.edu/users/johnson/frameworks.html

- Jomoho (2012), *Franky Flies High Presskit*. Email 12.11.2012, <moritz.laass@gmail.com>

- Kemmis S. & McTaggert R. (1990), *The Action Research Planner*. Geelong: Deaking University Press.

- Kieth C. (2010), *Agile Game Development with Scrum*. Boston: Pearson Education.

- Kock N. (2007), *Information Systems Action Research*. New York: Springer Science+Business Media LLC.

- Koster R. (2010), *A Theory of Fun for Game Design*. Phoenix: Paraglyph Press.

- Laass M. (2011), *Gridflower*. Available online at: https://github.com/jomoho/gridflower
  [Accessed 14th December 2012]

- Laass M. (2012), *Moritz Laas*. Available online at: http://moritzlaass.com/ [Accessed 14th December 2012]

- Logan S. (2008), *Cross-Platform Development in C++*. Boston: Pearson Education, Inc.

- Marchesi M. & Succi G. (2003), *Extreme Programming Perspectives*. Boston: Addison-Wesley.

- Masters J. (1995), 'The History of Action Research'
  in I. Hughes (ed) *Action Research Electronic Reader*, The University of Sydney, Available at:
  http://www.behs.cchs.usyd.edu.au/arow/Reader/rmasters.htm [Accessed 16th August 2011]

- McMillen E. (2012), Interviewed by Pajot M. and Swirsky J. for *Indie Game: The Movie*. Canada: Blink Works Media.

- Microsoft (2011), *What is a Platform?* Available at: http://www.microsoft.com/applicationplatform/en/us/About-Platforms/Default.aspx [Accessed 13th February 2012]

- MochiMedia (2011), *2011 Flash Games Market Survey Results*. Available at:
  http://wiki.mochimedia.com/w/page/49099894/2011%20Flash%20Games%20Market%20Survey%20Results
  [Accessed 16th February 2012]

- MochiMedia (2012), *2012 Flash Games Market Survey Results*. Available at:
  http://mochiland.com/articles/2012-flash-games-market-survey-results [Accessed 21st December 2012]

- Narasimhan L. (2009). *Challenges of Cross Platform Game Development*. Presentation held on June 29, 2009. Available at: http://software.intel.com/en-us/videos/cross-platform-game-devt-part1/ [Accessed 13th February 2012]

- Nothing Interactive (2012). *Games that Made Our Customers Happy*. Available at: https://www.nothing.ch/works/games [Accessed 17th December 2012]

- Oates B. (2006), *Researching Information and Computing Systems*. London: Sage Publications.

- Pajot L. & Swirsky J. (2012), *Indie Game: The Movie*. Canada: Blink Works Media.

- Palmer R. & Felsig M. (2002), *A Practical Guide to Feature-Driven Development*. New Jersey: Prentice Hall.

- Perrin R. (2012), *Ludum Dare 25 Keynote*. Available at: http://www.ludumdare.com/compo/2012/12/08/welcome-to-ludum-dare-25/ [Accessed 9th December 2012]

- Poppendieck & Poppendieck (2003), *Lean Software Development: An Agile Toolkit*. Boston: Addison-Wesley Professional.

- Reason P. & Bradbury H. (Eds.) (2001), *Handbook of Action Research: Participative Inquiry and Practice*. Thousand Oaks: Sage.

- Refenes T. (2012), Interviewed by Pajot M. and Swirsky J. for *Indie Game: The Movie*. Canada: Blink Works Media.

- Routio, P. (2005), *Process of Action Research*. University of Art and Design Helsinki. Available at: http://www.uiah.fi/projects/metodi/120.htm [Accessed 19th August 2006]

- Schell J. (2008), *The Art of Game Design – A Book of Lenses*. Carnegie Mellon University, Burlington: Morgan Kaufmann Publishers.

- Schwaber K. (2006), *Scrum Et Al*. Google Tech Talks. Available at: http://www.youtube.com/watch?v=IyNPeTn8fpo [Accessed 15th August 2011]

- Schwaber K. (2009), *Agile Project Management with Scrum*. Sebastopol: Microsoft Press, O'Reilly Media Inc.

- Schwaber K. & Sutherland J. (2011), *The Scrum Guide*. Available at: http://www.scrum.org/storage/scrumguides/Scrum%20Guide%20-%202011.pdf [Accessed 8th August 2011]

- Shojaee H. (2012), *Intro to Agile Scrum in Under Ten Minutes*. Available at: http://www.youtube.com/watch?v=XU0llRltyFM [Accessed 23. December 2012]

- Shore J. & Warden S. (2008), *The Art of Agile Development*. Sebastopol: O'Reilly Media Inc.

- Sun Microsystems (1999), *Glossary: cross-platform*. Available at: http://java.sun.com/products/jlf/ed1/dg/higq.htm [Accessed 15th February 2012]

- Swain C. (2008), *Indie Game Developers Rise Up*. Interview with Chris Swain. Interviewed by Mary Jane Irwin for *Forbes*. Available at: http://www.forbes.com/2008/11/20/games-indie-developers-tech-ebiz-cx_mji_1120indiegames.html [Accessed 16th February 2012]

- Takeuchi H. & Nonaka I. (1986), 'The New New Product Development Game', *Harvard Business Review*. Available at: http://hbr.org/product/new-new-product-development-game/an/86116-PDF-ENG [Accessed 10th April 2011]

- Thoa E. (2012), *Money And The App Store: A Few Figures That Might Help An Indie Developer*. Available at: http://thegamebakers.com/money-and-the-app-store-a-few-figures-that-might-help-an-indie-developer.html [Accessed 21st February 2012]

- Tynjala J. (2011), *Indie Flash Game Development: 2010*. Available at: http://joshblog.net/2011/01/17/indie-flash-game-developer-2010-revenue/ [Accessed 16th February 2012]

- W3C (2012), *Visual Formatting Model: The Viewport*. Available at: http://www.w3.org/TR/CSS2/visuren.html#viewport [Accessed 30th December 2012]

- Welch J. (2008), *The Promise of Casual Games.* Keynote session. San Francisco: Casual Games Summit 2008.

- Wells D. (1999), *Extreme Programming Lessons Learned: Never Add Functionality Early.* Available at: http://www.extremeprogramming.org/rules/early.html [Accessed 30th Devember 2012]

- West D., Gilpin M. & D'Silva D. (2009), *Ensure Success For Agile Using Four Simple Steps*. Forrester Research. Available at: http://www.forrester.com/rb/Research/ensure_success_for_agile_using_four_simple/q/id/54037/t/2 [Accessed 10th April 2011]

- Wilson B. (2001), *Soft Systems Methodology: Conceptual Model Building and its Contribution*. Chichester: John Wiley & Sons.

- Yentin et al. (2012), *The Necessity of Mobile App Marketing: What It Really Takes to Succeed with a Mobile App*. App-Promo. Available online at: http://app-promo.com/whitepaper1/ [Accessed May 5th 2012]

- Zanotti-Schudel N. (2011), *A Comparison of Action Research and Scrum*. Available at: http://nicolas.zanotti.me/sae/cmp-402_theoretical_perspectives_rc2_zanotti.pdf [Accessed September 9th, 2011]

11. Appendixes

The following appendixes can either be found on the CD accompanying this report or can be downloaded using the listed URLs.

A. The Design Play Release Model

A PDF-File of the model is located at:
http://nicolas.zanotti.me/sae/dpr_model.pdf

B. Project Proposal

The project proposal is part of the learning agreement located at:
http://nicolas.zanotti.me/sae/cmp-401_learning-agreement_schudel.pdf

C. Reflective Logbook

The reflective logbook is located at:
http://nicolas.zanotti.me/sae/mpp-405_logbook_zanotti-schudel.pdf

D. Presentation

A 10 minute presentation of the research is located at:
http://nicolas.zanotti.me/sae/mpp-405_presentation_zanotti-schudel.pdf
http://nicolas.zanotti.me/sae/mpp-405_presentation_zanotti-schudel.mp4

E. Game Source Files

The assets and source code for the game prototype *Proximity* is located at:
http://nicolas.zanotti.me/sae/proximity.zip

The assets and source code for the game prototype *King of the Hex* is located at: http://nicolas.zanotti.me/sae/king-of-the-hex.zip

The assets and source code for the game prototype *Flick Kick Soccer* is located at: http://nicolas.zanotti.me/sae/flick-kick-soccer.zip

F. Interviews

Interview with DigiDingo

The interview with Jann Sigrist and Jörg Sigrist of DigiDingo can be found at: http://nicolas.zanotti.me/sae/interview_digidingo.mp3

The interview was held in Swiss-German and translated to English by Nicolas Zanotti:

Nicolas Zanotti: My name is Nicolas Zanotti and I am here with DigiDingo, Jann and Jörg Sigrist. Hello everybody.

Jann Sigrist: Hello.

Jörg Sigrist: Hello.

Nicolas Zanotti: We will discuss the production process of game development. First, could you please introduce Grooh?

Jann Sigrist: We pronounce it 'Groo'. Grooh is a puzzle game. We have four elements: water, wind, earth and fire. They are displayed as symbols and need to be activated to open the door at the end of the level. The goal for the player is to initiate a chain reaction that makes the fields explode. As soon as he freed up all the elements the door is opened and he can proceed to the next level. The goal is to do this in as little steps as possible. To find the best solution possible. The player is awarded with a gold medal. There are about a hundred levels to play. One can place boxes on a field to stop the chain reaction. There are fields that explode when stepped on. There are further variations such as beaming pads. That is the basic setup.

Nicolas Zanotti: So the game gets more complex from level to level?

Jann Sigrist: Exactly, the difficulty increases. The first levels are easy. Levels 80 until 100 get pretty tough.

Nicolas Zanotti: We will return to game design in a short while. First the business aspect: What are your target markets and audience?

Jann Sigrist: We wanted the target audience to be as broad as possible. So also the casual gamer that is 50 years old, not just the 15, 16 year olds. We knew it would be hard to target children, because we think the game is too complex. Not a simple form of entertainment. 20 to 50 is our main audience.

Jörg Sigrist: It matters where it is played and not who plays it. It's a game that is played in between. It's not a game that is played in one go. It's more of a game that is picked up in the train for 10, 15 minutes, playing a level and the satisfaction it gives to reach the end of that level.

Nicolas Zanotti: So a typical casual game, a game that is played in short chunks?

Jann Sigrist: Yes, not in a 4 hour session.

Nicolas Zanotti: Yes, I played through to level 12 in my first round and I'm looking forward to the next session. I have seen on your Facebook fan page that players have reached high levels.

Jan Sigrist: Yes, we probably have about 10% that reached higher levels.

Nicolas Zanotti: Do you have any numbers on how many players there are world wide?

Jann Sigrist: We are officially at 12'000 units sold. Of course there are many that download the game illegally. We presume that there are about 20'000 illegal downloads.

Nicolas Zanotti: So you have a problem with stolen software?

Jann Sigrist: For jail-broken iPhones, there are sadly sites where you can find the game. Mainly Russian sites, where there are sometimes a thousand downloads per day.

Nicolas Zanotti: How long did it take you to develop the game?

Jann Sigrist: It took us 3 1/2 months. I was developing full time and we had a graphic artist with us working 80%.

Jörg Sigrist: I was operating in the background. I was assigned to creating the levels and was also part of the general idea process. However, I did not do any programming, since I worked on separate projects. So, I was more on the creative side like algorithms, game play, and experience. But I didn't code a single line other than maybe a prototyping or an algorithm.

Jann Sigrist: I did all of the development.

Nicolas Zanotti: So the graphic artist was involved at 80%?

Jann Sigrist: Yes, he also was involved with the concept. He did all the animations and sound.

Nicolas Zanotti: Including audio?

Jann Sigrist: Yes. Including audio and visuals.

Nicolas Zanotti: The game was released for the iPad?

Jann Sigrist: Including the iPhone and iPod Touch, yes. It was a universal build.

Nicolas Zanotti: Do you have any other platforms in sight at the moment?

Jann Sigrist: We are working on the Android version. We work with a framework that currently has some problems with Android, mainly performance issues. It’s not running as smoothly as we want it to. It becomes choppy sometimes, and we are working on optimizing it. But then we plan on releasing for all Android based devices, like tablets from Amazon, Kindle Fire, Nook and all of those.

Nicolas Zanotti: I saw you are using the Corona framework.

Jann Sigrist: Exactly.

Nicolas Zanotti: So any supported release platform, do you plan on publishing to it?

Jann Sigrist: Yes, that is what we are planning.

Nicolas Zanotti: Is a complete port planned in the future, such as a web release or social release?

Jann Sigrist: It hasn’t been a planned as of yet. We are keeping it in mind. For instance to create a Facebook application, because we have a Flash background. Or maybe to create a Windows Mobile version, like a port to Windows Mobile 8. At the moment we prefer to work with the framework, because we know we have one solution and can release it to multiple platforms. A complete port is work intensive, because we would basically restart development from scratch. To recreate the entire game in C#… it depends how successful the game becomes.

Nicolas Zanotti: I guess the fun-factor of rehashing previous work would also be low?

Jann Sigrist: Exactly.

Nicolas Zanotti: And platforms such as Steam or Xbox Live?

Jann Sigrist: Steam would be up for debate, because it is possible to release Flash games. We are keeping it in mind. For the moment the target market is iPhone and iPad because they are devices we can release to directly. It’s the solution we have chosen.

Nicolas Zanotti: Moving on to the topic area of game design, in the beginning how did you do the concept? Did you create a game design document?

Jann Sigrist: The very first phase was brain-storming, independent of a document: Gathering ideas, sitting together and discussing. A lot of back and forth amongst ourselves. Then we began to elaborate and started documenting. The graphic artist started making sketches. The idea of how it should work, what the goal is. This documentation was used throughout the process. It is the basis from the start.

Nicolas Zanotti: So you always returned to the documentation?

Jann Sigrist: Yes, we returned. But I mussed confess that we didn’t put much effort into updating it. It was never up to date, but it was always the basis. If one were to read it now, one could say it has become the game. With some elements that needed to be removed.

Jörg Sigrist: Mainly changes in the graphics, in the beginning it looked entirely different. But basically the game has stayed the same. We added new elements and made some changes, but the concept has always stayed the same. We completely redid the graphics at one point.

Jann Sigrist: It was important that we had the document as a basis so we could develop the product. It was clear that changes will be necessary, but that's the normal process.

Nicolas Zanotti: Did you start with wireframes during development? Or did you tell the graphic artist exactly what you needed from the start?

Jann Sigrist: Some mock-ups, game figure variations, putting it together in a scene. Then we decided quickly how we wanted to do it. I would then already start with coding the prototypes. And he would refine the graphics.

Nicolas Zanotti: Did you have the target domain in mind when you designed the game?

Jann Sigrist: Yes, we defined the platform at the start. Also what the target audience should be. Those aspects were known initially.

Jörg Sigrist: What I learned is how differently the iPad and the iPhone are used interactively. On the iPad, the game is played much differently than on the iPhone. The iPad is usually placed on a table and there is more room. It is easier to point at what you want. With the iPhone you need to add controls. One notices if a game was created specifically for the iPhone or the iPad. That is an important thought: does one create different versions with the same underlying gameplay, but with a different handling. Or does one try to make it universal and try to make it work on both and have the users accept it. Those are considerations one needs to make and needs to test.

Nicolas Zanotti: Did you make these decisions during the design phase or did you try this during development?

Jann Sigrist: Those considerations fell a little short. At some point during the process we noticed that it would not quite work. We needed to switch to using a virtual keypad, because that way it would be possible to play it on all the devices. The player would not be to keen on needing to pick up the iPad and move it around. That is definitely something we will need to put more focus on in future releases. Initially one thinks to just do it, and then one is in the middle of the process, it is running on the device, wants to play and then notices it does not quite work as wished. Like navigating everything with simple touches. So it is an important aspect.

Jörg Sigrist: To return to your question: We did think about interaction but we underestimated its importance and also its effects. We had ideas on how to navigate the game, but we did not quite realize how large the differences are between the different device sizes.

Nicolas Zanotti: Did this become apparent with the prototype version?

Jörg Sigrist: Yes, we picked it up and started playing on the iPhone and then we thought we will maybe need a virtual keypad. On the iPad it would have worked perfectly, because the finger has a certain size. The gameplay is the most important thing. That it can be played intuitively and that the player doesn't get the feeling the game is unresponsive. Even if there is one button more, the user having to press it becomes tedious. That user then decides to give a one star rating even though the game is otherwise OK.

Nicolas Zanotti: So the game fails just because of its usability.

Jörg Sigrist: Exactly. There are quite a few reasons a player rates the game with one star, even though they played through the entire game and think it is cool. That happens. The end sequence, we found out, frustrated some players.

Jann Sigrist: Because they didn't understand what happens. It comes out in the end that the game continues with free bonus levels. They though it is boring. For the ending one has to collect gold everywhere and be good at the game. The ending isn't a final level but another part of the story. The game character finds, or doesn't find, something. Some players became very frustrated and then gave a one star rating. Before that, they thought it is the best game.

Jörg Sigrist: The player was not awarded for getting all the gold medals, but they would after playing the bonus levels. Maybe it was not very smart on our behalf.

Nicolas Zanotti: On to the topic of storytelling: Did you integrate the story as part of the initial design or did you continue the story during development?

Jann Sigrist: Design?

Nicolas Zanotti: During the concept phase, did you have the story in mind?

Jann Sigrist: To be honest, we did have a story in mind in the beginning, but then we scratched it – also graphically. First the idea was to put Grooh in a skyscraper with a paint brush. He would paint the floor.

Jörg Sigrist: First he was not a monster but a young painter that has to paint the floor with his brush. That was the setting.

Nicolas Zanotti: OK, so he would go from level to level with the elevator?

Jann Sigrist: We noticed at one point that it did not make any sense to paint a floor. Also the kid seemed a little boring to us. Having a monster with a paintbrush-tail was also a bit strange. Then we discarded the entire setting and switched to a classic setup with a castle and the search for treasure. We helped ourselves to better known themes. That is how the story changed. We had to pack the new setting into a story. That the user has a story-line to follow and that they are driven to proceed with the game.

Nicolas Zanotti: Did you have the entire story, or did you extend the story after level 50?

Jörg Sigrist: Story is maybe a bit exaggerated. The story is basically that he is searching for something in the castle. And he works himself through each level. Creating an intricate story around that is almost not needed. Sometimes he talks and says: “I am getting closer”, or: “Now I got far”, or wonders what is inside the treasure chest. But other than that there is no big story.

Nicolas Zanotti: So the focus is more on puzzles?

Jörg Sigrist: Since it is a casual game, one would forget it if the game is not played for a while. It does not fit in the concept to have a complicated story. Also, it is easier not to have to tell a complex story.

Nicolas Zanotti: So there is less content that needs to be made?

Jörg Sigrist: Yes. Special skills and time are needed to create a story line.

Nicolas Zanotti: Moving on to the development process, when did you decide to use certain tools?

Jann Sigrist: At the beginning we evaluated. We wanted to do something for mobile. Then we tried to find out what the best framework is or if we needed to develop native. I checked Flash because it offers the possibility to release native. We tested for two weeks, then we decided that the performance was too bad to be put to use. Then by coincidence we found the Corona framework, evaluated it, and where excited quickly, because the performance was good and provided us with what we needed. We did not want to do things in 3D. Then we would have chosen Unity as a framework. For 2D games it is great, because it saves us so much time. We can program to OpenGL. Doing so natively would take us a hundred lines of code and we could do it in three, by creating an object with the engine built in. Perfect for us for casual games.

Jörg Sigrist: The licensing costs were also moderate.

Nicolas Zanotti: So it was also a question of price?

Jörg Sigrist: If we would have needed to make a large investment we would have thought about it twice.

Nicolas Zanotti: Like Unity, releasing to an additional platform like Android would have cost more.

Jann Sigrist: Yes, with Corona we can release to Android, iPhone, iPad with one license.

Nicolas Zanotti: How fast did you then start play-testing?

Jann Sigrist: What shall I say. As soon as it was playable. That was about half way in the development process. A little before even. We tried to reach that stage early, because we think it is important to play in order to find out what can be improved. It is the most important thing to play one’s own game. The earlier this is done, the earlier one finds out. We had to redo a lot of smaller things because we noticed during playing that it would not work. Something gets in the way or it is not intuitive. So I would say before half way through. After about one and a half months.

Nicolas Zanotti: Who decided at that moment if it is fun? Did you both check if it is fun for you or did you pass the game on to other people?

Jann Sigrist: Mostly we decided. We showed the game to two or three people, like girlfriends and so forth. But at that point only we played. The three of us.

Nicolas Zanotti: So you reflected upon the gameplay?

Jann Sigrist: Yes, reflective, the three of us. We discussed it between ourselves. Clearly, as soon as we have something playable, it would be great to give it to a hundred people and get feedback, but we do not have the resources.

Nicolas Zanotti: After having the first playable version, did you start creating you own tools? For instance for generating levels? Or did you have something built in to Corona already?

Jörg Sigrist: We are thinking about creating a level editor. Originally I put something together in Excel, where it is possible to assemble the levels easily in a raster. It then creates the code for Jann to import into the game. A very simple editor, basically. But we are evaluating if we want to add an editor to the game, that the users can use to build their own levels and send them to each other. But it is a low priority, because we have other projects. Such an editor would not only be used for Grooh but for other games that are based on a raster.

Jann Sigrist: We had the idea to create modules that can be reused. I already have a few. For instance a connector to Game Center, where the high scores are sent. Or for audio, we created modules that can be used in further games, that allows us to make games easier.

Nicolas Zanotti: The next games build on top of those modules?

Jann Sigrist: Yes. I have my own framework. I know how to change scenes or how to load sounds. I have a module that saves the hight score to Apple’s Game Center.

Nicolas Zanotti: At what point did you address bugs and performance? Constantly or only towards the end?

Jann Sigrist: As we went along. The larger ones. Some of the smaller ones we left because we knew they are not critical. Like a positioning error or such. But when the game did not run correctly, or a game element was moved to much, we always fixed the bug immediately.

Nicolas Zanotti: Did you use any bug tracking software?

Jann Sigrist: It would be great to use such software. But because of time constraints we just made a list in Excel.

Jörg Sigrist: We tried to play doing everything wrong that can be done wrong. We still did not find everything. There are still users that do stuff we did not think about. I believe we had one bug after the first release, that affected the logic. Compared to other games, things we have read, then I think we had a stable product.

Nicolas Zanotti: So you got feedback from users that found bugs?

Jörg Sigrist: Yes. One receives such feedback, when they fill out the contact form and say what the problem is. Apple makes it mandatory to have a contact form so the user can contact you.

Nicolas Zanotti: Speaking of post-release: What were your marketing efforts? I saw you have a website and a Facebook fan-page, do you have anything else?

Jann Sigrist: Yes. We have a Twitter account as well. We have promo codes and we wrote to all the review sites. We wrote to many sites. We also made the experience that it takes a while for those sites to respond with feedback. Some responded after three or four weeks and said they would have a look. Then a review would be made. We contact the newspaper, for instance “20 Minuten” [National Swiss Newspaper] made an article. But we were also contacted often. It was not only that we needed to get attention. Some asked us if they could write an article. What we also needed to do, what we did not mention before, is support. There are players that cannot proceed in the level and we gave them tips. Or they would ask why something does not work and we would also reply. So we had a lot of work supporting our customers.

Nicolas Zanotti: So it was constant work?

Jann Sigrist: Yes. It is not to be underestimated. Not to think that one can release a game and that is it. The one or two weeks after a release are the important ones. Because then many enquires come in. If we do not answer them immediately it is too late. So many games are released daily and we have to keep at it to get the reviews. I would say, a week before the release and the two weeks after is the most important time.

Nicolas Zanotti: Did you decide to invest in advertisement? Did you have a budget?

Jann Sigrist: We used other channels. We did not invest any money in advertisement. But we will possibly change that. We were asked by an international publisher. They wish to work with us, given that we release our next game and they like it. We would use their network and they would do advertising for us. For instance writing to newspapers and such. We would hopefully be featured again by Apple. That way we can concentrate on development and the marketing would be taken care of.

Nicolas Zanotti: How long will you continue to support the game? Will you wait until it dies out?

Jann Sigrist: It depends what kind of enquiries we get. Questions on how a level can be solved, we will have to answer. But providing further levels and so forth, that will stop. The life-cycle is short. It is a strategic decision. Maybe we will provide an update with another ten levels, because we want to promote the next product. If the game is not a total hit and we do not get many new users, then it will not be worth it.

Nicolas Zanotti: Do you already have the next game in development? During the support phase of the current game, did you already start a new one?

Jörg Sigrist: Even before we released to the App Store, we were already planning the next game.

Jann Sigrist: Before we were even finished we needed to know what to do next. Otherwise we would have lost too much time. We are planning to have the next game out at the end of May, plus an application. In July we would like to start the next.

Nicolas Zanotti: Did you submit you game for any awards?

Jann Sigrist: No. We saw that there is a Swiss game award. In hindsight we should have participated. When we saw who won. Maybe next year we will submit a game. Or with Game Culture, they support Swiss developers, that we could get in that program. On a few sites, we were the game of the week. That is also kind of an award.

Nicolas Zanotti: The ranking was super. I thank that is what the people see.

Jann Sigrist: Yes, in general it is great that the game got such good ratings. We got 4 and a half stars out of five. I think that is pretty good for a first game.

Nicolas Zanotti: Looking back at the entire process, is there anything that you would have done differently? Did something not work at all?

Jann Sigrist: I would not say that something did not work at all, considering it was the first round. But there were some points that we need to do differently. We always presumed the game would be easily understandable for a wider audience. But if one wanted to be commercially successful it has to be something simpler. A point and touch interface or so. Or the entire process, how does one market the game. Or what reviewers are important. Who needs to get sent a promo code and such things. Getting the interface right from the start. Knowing from the start what is intuitive and what makes sense. Not to redo everything in the middle of the process. The other thing was with Grooh, where we redid the graphics. Those were 2, 3 weeks we missed at the end. That we put more effort into that planning, not just starting.

Jörg Sigrist: Something that was critical. If one publishes to the App store, it only takes 6 to 12 hours and there are people that downloaded the app and publish demos to YouTube. Making a very bad quality video and put it on the net. They get very many views. If that video is of bad quality, it is bad publicity for the game. The wrong movie was being linked and everybody thought that the game looked like that. The aspect ratio was twice as long as high and there were boring scenes recorded. Even somebody playing that does not understand the controls. Just playing the tutorial that is not very exciting. For our next release we must have a demo video ready.

Jann Sigrist: We released the video later and that was disastrous. Maybe not disastrous, but the wrong thing went viral. We wanted our video to be it. Ours was done better and we tried to give it a better title. Even "20 Minutes" [National Swiss Newspaper] had to change the video. That was too bad.

Nicolas Zanotti: So you will have the marketing material ready before release next time?

Jann Sigrist: Exactly. In case we start working with the publisher, then we have prerelease events. Like a sweepstakes, to get the game out there.

Nicolas Zanotti: To start hyping the game beforehand?

Jann Sigrist: Yes. There we just do not have enough experience. We are looking forward how it will turn out next time.

Nicolas Zanotti: Great! This brings us to the end. Thank you very much for this interview.

Jörg Sigrist: Thank you.

Jann Sigrist: Thank you.

Interview with Jomoho

The interview with Moritz Laass and Simon Siegenthaler of Jomoho can be found at: http://nicolas.zanotti.me/sae/interview_jomoho.mp3

The interview was held in English and transcribed by Nicolas Zanotti:

Nicolas Zanotti: OK. My name is Nicolas Zanotti and I am here with Moritz Laass, did I say that correct?

Moritz Laass: Laass, yes that is correct.

Nicolas Zanotti: Here in a startup incubator. Moritz, why don't you introduce yourself.

Moritz Laass: Yes, you already said my name. So this is probably going to be about game development. I started programming around the age of fifteen I guess. I started with C++. Back then it was on a Windows/DOS environment and I could start with the Allegro game development library. I built my way up from there and I made quite a few games. Usually engaged in what is now quite popular, it wasn't back then, it's called Ludum Dare. 84 hours of game development. You start off with a topic, 48 hours, sorry. You have 48 hours to develop your game and make everything like sound effects and graphics and code everything from scratch. So that is kind of my start into game development.

Nicolas Zanotti: Great, so I saw you made some more projects like Gridflower and JomohoJS.

Moritz Laass: It was for my Bachelors thesis. I decided to go for something game related since that always the most fun for me. So what I did, I looked at how you could use HTML5 to develop games and I dug myself into JavaScript and I stumbled upon this cool project called NodeJS which is basically a JavaScript server. So you write your server completely in JavaScript and it's pretty fast because it's running on V8 that runs in Chrome. That is highly optimized JavaScript. So, what I did then was to try and build an interactive networked environment in the browser. Which is probably not what the browser is supposed to do. Definitely one can do better, but it's a specialty in itself.

Nicolas Zanotti: It is socket based?

Moritz Laass: Yes, it's web sockets. In NodeJS you have this great library called Socket IO and it is basically an API for web sockets. It falls back onto other modes of transport if the browser does not support it. It even does Flash sockets. It is pretty easy to use but still very powerful in the background. That is what I used. Basically, what I did in the thesis was a level editor that you could use corroboratively. An unlimited amount of people editing the same level at the same time.

Nicolas Zanotti: Now you are working on a game yourself?

Moritz Laass: Yes, I have been working together with a graphic artist. He drew the characters and the background images and all that stuff. I am working on the game design and coding everything. We have worked on it in our spare time. We both work here in the incubator. We said we will have one day a week where we will work on the game.

Nicolas Zanotti: You have a second job and you are doing this in parallel?

Moritz Laass: Yes.

Nicolas Zanotti: It is a two person team?

Moritz Laass: Yes.

Nicolas Zanotti: Your goal is to sell the game?

Moritz Laass: Exactly.

Nicolas Zanotti: To then make money for the company?

Moritz Laass: Yes. We will just look where it takes us. We are new to this whole process of publishing a game and marketing a it. Just finding out how that works will be very interesting. We will probably be making more games. It is definitely a goal.

Nicolas Zanotti: I am going to jump right in to the game design process. Did you start out by creating a game design document? How did you do the very first step?

Moritz Laass: We started with a basic idea for this game. There is no such thing as a process , for me at least. It's always a bit different. You start off with an idea. What I like to do is to prove it really quick. There is a core mechanic for the game that we'd like to see if it feels good. There is a small prototype that kind of gets the feel across. Then we start to build around it.

Nicolas Zanotti: Who decides if the game is fun? Do you decide if it is fun to play, or do you show it to other people?

Moritz Laass: Usually I try it and play around with it a little. You get into it and start to identify with it. You later start showing it to other people. Also, for myself, I try to be really critical about it. To say: "What could I do more?". The whole balancing process, I do that by myself. I see what feels different if I change a variable. Kind of both. Show it to people, but first I try to get to a point where I visualize what the game could be.

Nicolas Zanotti: Do you have a proof of concept? Multiple versions first, or do you jump in to the main game?

Moritz Laass: I build a prototype. That's not the best thing I did with this project. I iterated on from the prototype.

Nicolas Zanotti: So you start small with a prototype then proceed?

Moritz Laass: There is still code from the very first day in the game right now. It is bad because the code starts to get messy. You start to build things that you did not think of before. We don't have that much time for refactoring.

Nicolas Zanotti: Do you have a fixed time frame? Where you say that the game has to be done within a certain amount of time?

Moritz Laass: Yes, you have those milestones but then you keep missing them. You have to get it done. That is the most important thing I guess. I do not want to ship anything that is not finished. I do not need to publish it because of money, because I have a job. I think it is more valuable to invest a little more time to make it complete.

Nicolas Zanotti: Make sure it's your work of art before you send it in?

Moritz Laass: Yes. The thing is, there is also the possibility, with mobile platforms, to update after the release. What we do now, is we limit the amount of stuff we put in the game. Bonus items and stuff like that. We really limited ourselves. That's a possibility to.

Nicolas Zanotti: To grow after the first release?

Moritz Laass: Yes.

Nicolas Zanotti: Looking back, at the previous games you made. Like, for Ludum Dare, how much did your design concept change between your first idea and the actual product? I guess, for Ludum Dare, it is a little special, since you only have two days to finish it. For a longer game, do you think, does it change during the duration of production?

Moritz Laass: Not really. You add details that you didn't think of before. But, the main game is basically what you projected it to be. I have this very complete idea in the beginning and then I try and reach it through the development process.

Nicolas Zanotti: How involved is the graphic artist in this idea? Do you talk to each other before you start? Or is the game design your part and you just ask the graphic designer for certain elements?

Moritz Laass: No. We work together. He doesn't shut up about it. Sometimes I would love to just do it on my own, but it's good to get this input. We are not always on the same page about things, so there is a discussion going on.

Nicolas Zanotti: Do you have a story telling element behind your game? Or is it more about playing and a story might comes out of it.

Moritz Laass: We have, actually, a story. I'm not so sure about how good it is. But it is his character. I try not to kill the character. We have a story, but I think it is more like an introduction, an intro-story, for the character. A back story. It's not a complicated game, a very simple game we are developing right now, so there is not much of a story. It's basically a time killer.

Nicolas Zanotti: So not like a Final Fantasy type story line.

Moritz Laass: No, nothing like that. I have things like that on my mind, but developing that kind of game takes a lot of time and effort.

Nicolas Zanotti: At what point did you decide what the target audience should be? And with that, what the target platform should be?

Moritz Laass: We decided early on to do a really simple, broad range target audience. It’s a very cute character, so it’s not a typical game audience. We wanted to go for mobile so we made it a one button mechanic. You just have to press, or not press, to play the game.

Nicolas Zanotti: Moving on to the development process, but sticking to your last answer, are you building for a small device or a tablet device, or both at the same time?

Moritz Laass: The graphics, we have made them all in very high resolution, so we can target even the desktop. Play with the mouse. With Unity there is the possibility to go for Flash. That is also something we have on our minds. It was one of the ideas in the beginning, not to limit ourselves to one platform. Maybe a target market, but not limit ourselves too much. The other ways are still possible. I only have an iPad, so I am testing on the iPad.

Nicolas Zanotti: The prototypes, you did them on a phone or on the pad? Both at the same time, or did you test on the desktop first?

Moritz Laass: We did development on our desktop machine and tested there. Then we go to the pad and test it there. What we need to do now, is to optimize it to make it run much smoother there. That is one of the difficulties we are struggling with right now.

Nicolas Zanotti: Performance enhancements, at what point do you do that? Fairly early, with performance in mind, or do you already have a version set up before you start optimizing.

Moritz Laass: We already have a version and I’m pretty sure there is a way to make it work smoothly, but it’s fighting against the engine basically.

Nicolas Zanotti: So basically setting the parameters correctly.

Moritz Laass: That is one of the things when you take an engine like Unity. You have a certain way to do things and you have to stick to that way and you don’t have access to the inner parts of the engine. I always was used to writing my own engines, so that you can optimize specific parts. Now you have to do this from the outside. I’m reading a lot about that. Trying to figure out how to do it.

Nicolas Zanotti: I guess that is one of the special things about the Unity editor, where you have to set background and distances and such things, rather than low level optimizations.

Moritz Laass: Yes. If you have lots of different sprites, then you can use your own draw calls. You have to compact them into one mesh. You have to do it in a certain order, so the ones in the background don't appear in the foreground.

Nicolas Zanotti: It's a 2d game?

Moritz Laass: It's a 2d-3d game. The gameplay is 2d. The graphics, they are in space for parallax scrolling.

Nicolas Zanotti: 3d planes layered on top of each other?

Moritz Laass: Exactly.

Nicolas Zanotti: Working with the graphic artist, did you first settle on mock-up graphics to test the game? How much in detail did you go already?

Moritz Laass: I did my own graphics to pump out the prototype. Just really quick, threw something together. I did some Google Image searches and cut out stuff, just to have something there. Then we went through everything an discussed it, how it should look. The color palette and everything, so it fits nicely. He does the stuff and I do the importing in the engine. There is always some kind of tweaking at that stage again. Like tweak the colors and sizes and proportions.

Nicolas Zanotti: Was it necessary to write your own little tools and scripts? Just to build a level or import data?

Moritz Laass: No. The level gets built dynamically. It gets generated. So for this kind of game, it was not necessary to build any tools. We bought something on the Asset Store. A 2d library with animations and asset management.

Nicolas Zanotti: So you already had that from the start and didn't have to make it yourself?

Moritz Laass: We had to buy it for 50 euros or something.

Nicolas Zanotti: When you find a bug, do you put it in a bug list or do you directly try and go and fix it?

Moritz Laass: I always write to-do lists, I put it on my to-do list. But, I might fix it immediately if I know what it is. Depends on the bug really.

Nicolas Zanotti: If you consider all the games you made previously, what do you think made you develop faster – like from game to game. I presume the first game took a certain amount of time and the second game was easier to make. The third game yet easier than that.

Moritz Laass: It's never really that bad. In the beginning I started with really basic stuff. Pong and Breakout. What really changed the way I developed was that I was learning programming. In the beginning, I was lucky they even ran. Programming changed the most for me. With Unity, it’s different because I’m really used to doing everything in code. Now I have to do so much in the editor. You have to drag stuff around there and go back to the code. I have found the workflow now, but for me, it’s not the ideal solution. It’s interesting just to explore that. I started to think about games in a different way. To take them apart. To think about game mechanics as single parts. When you’re there and play a game, you just see the full game. You don’t pick apart the mechanics. I’m seeing it more like Lego, there is a mechanic for this part of the game, and that for that.

Nicolas Zanotti: So it’s a more modular approach now, so there are pieces and you put them together?

Moritz Laass: Yes. I think more about the different parts of the game now. Like: how could I change this, or what would happen if I changed this variable in this part of the game. I think that changed a lot. To be able to differentiate between those parts. To try to find a way to make them work together.

Nicolas Zanotti: At what point did you decide to use Unity as a toolset? Did you have the idea first, of the game, and then you decided Unity would be the best to go with that? Or was it the other way around?

Moritz Laass: I had Unity, because I did another project before I did with another colleague. We did a visualization for a hotel. So we used Unity for that, because we had to run it on Windows and iPad. I was already used to Unity a little bit. I wanted to make a game, so we sat together and said we wanted to make a game. It was kind of an obvious choice at the moment. I had the license and also the experience. There is not much else out there that is comparable.

Nicolas Zanotti: I saw your previous games were all web based?

Moritz Laass: Lot’s of them, yes. The older ones were all C++.

Nicolas Zanotti: All Windows games?

Moritz Laass: Yes.

Nicolas Zanotti: So you decided to move away from the web platform? Generally for game development and make cross-compiled games?

Moritz Laass: You mean the HTML5 games?

Nicolas Zanotti: Yes. Did you make them more as tests? To see how far the web platform, as a development platform, can go?

Moritz Laass: It was my bachelors thesis. So it was, if you want to call it that, a scientific experiment.

Nicolas Zanotti: To see how far you could get on that platform?

Moritz Laass: Yes. I think it’s probably one of the most interesting. But, I think that it is still too limited for serious game development. It’s still interesting. It’s still there in the background. If you make an HTML5 game, you can sell it on the Chrome Web Store, that’s it. There is not much money in it. Or you go and develop for a company that does Facebook games. I wanted to do my own stuff and I didn’t want to do Facebook games or social games.

Nicolas Zanotti: So the choice of the development platforms is dictated by what you can release to and what the market actually wants?

Moritz Laass: Yes. Definitely. It has to be. I’m thinking right now that the desktop and the PC is still an interesting market. The gamers are still on that platform. The people that have played games all of their life and are still interested in it. It might be one of the choices in the future, to go back there.

Nicolas Zanotti: So you might consider releasing to Steam?

Moritz Laass: Yes, something like that. But to release to Steam you have a different audience. The cool thing about it is that you can make much more serious games, like grown-up games. You can also do much more complicated mechanics. That’s one of the things in the back of my head.

Nicolas Zanotti: Projecting into the future, how do you think you are going to market the game? Are you going to make some banner-commercials or are you going to put it into the App-Store and hope it does well? How are you going to spread the word that the game exists?

Moritz Laass: I think there is pretty much a standard. You have this one minute video. A Youtube video. We are working on that right now. I think that is one major thing you need to have. You need to have the graphics in the App-Store and Google Play. They have to look good. You need to have a good description. But then I am working on the company website right now. On the website for the game. We are planning on including something like a high score. Something you could submit to Twitter and Facebook. Something to make it viral. Probably we’ll write to a bunch of blogs, that blog about stuff like that, and give them a free version or something. I’m not really sure about that right now.

Nicolas Zanotti: You think at this point you are going to do all the marketing yourself? Or are you going to ask a third person that specializes in marketing?

Moritz Laass: That is a good idea, actually. What I’ve done until now is think about it myself. That might actually be an option. If I can find someone that can do a good job. Yes, why not. Pretty good idea actually.

Nicolas Zanotti: Do you have your own budget set up for such things? I mean for marketing purposes. Or is it just part of the total development cost?

Moritz Laass: I don’t have this planned out. It’s part of the development cost, I guess.

Nicolas Zanotti: How long do you think you are going to support the actual game? You release it, then I presume there is going to be feedback from customers? How long are you going to continue updates? At what point would you just say: OK, we are going to move on and let the game be at it is.

Moritz Laass: That is a good question. I have not yet thought about that. I guess it also depends on how well it gets, how people react to it. If there are many people that like the game and want more. It's an obvious choice to invest in it, because you don't want to let those people down. Also, it's good if something grows. It's not really decided yet. I always try to see how things develop and make informed decisions. It's the first game we are publishing, so it's kind of new territory.

Nicolas Zanotti: So it's still unknown what is going to happen with the marketing.

Moritz Laass: Yes.

Nicolas Zanotti: OK, so that's actually all of the questions from me already. Thank you very much for this interview.

Moritz Laass: It was interesting.

Interview with Nothing Interactive

The interview with Bastiaan van Rooden and Mark Gruber of Nothing Agency can be found at:
http://nicolas.zanotti.me/sae/interview_nothing_interactive.mp3

The interview was held in Swiss-German and translated to English by Laureen Zanotti and Nicolas Zanotti:

Nicolas Zanotti: My name is Nicolas Zanotti. I am at Nothing agency, and with me are Bastiaan van Rooden and Marc Gruber. My first question to you is: please introduce yourselves and tell me what roles you play in this agency.

Bastiaan van Rooden: I am the founder of the company. The company was founded in 1999 and for the first couple of years I was responsible for the design of our products – the visual design. However, for the last 2-3 years, give or take, I became part of the managing board as a project leader and general manager. So now, everything that falls into the visual aspects I delegate to my colleagues. My job consists of creating concepts and being a project manager and a general manager. And that's enough work for me! (laughs)

Mark Gruber: Originally, I worked in the print industry as a Print Designer. I recently got my Bachelor's degree in Game Design. I learned programming, especially with Unity. I realized a variety of projects. After I graduated, my school commissioned me to create a Serious Game which I am currently working on here at Nothing.

Nicolas Zanotti: There is this term "Indie Game Developer". What is your understanding of this term?

Bastiaan van Rooden: At the moment, this term is misused to describe all sorts of things. Initially, it is linked to the growth of Triple-A companies that bought up many small companies. Normally, if a startup becomes a major corporation with developers and publishers under one roof, there's a gap that can be closed by new companies. It is actually a normal market movement. Initially, one could run a one-man show and produce games. But it lead to the instance that creativity suffered because work was concentrated on "safe" ventures. And that was when the indie scene evolved. It is a normal market development that can be observed in many different fields. The term indie game developer is somewhat woolly because all sorts of people call themselves indie game developers. Actually, it's a term to describe a small company, i.e. a one-man company, which does not have to meet the standards that large companies have to meet. And because of the many publishing channels that are to ones disposal, one can even operate effectively. The tag "indie games" will perish because only a small percentage of indie game developers can make a living with indie game development. Some people just enjoy creating games until everyday life creeps up behind. These people then realize that they just have to make a living. Therefore, many indies will disappear.

Mark Gruber: I agree. And to me, an indie distinguishes him/herself by not depending on a publisher, that he/she carries the development costs. This allows him/her to being creatively open. Although in the end, indies often look for a publisher or the other way around: publishers approach indie game developers.

Bastiaan van Rooden: The problem is, that the indie market is growing so rapidly and that there is so much competition that one must really fight ones way to stay at the top. Being talented is not enough: One has to spread the news about one's work, talk about it, write about it, tweet it, etc. The same marketing rules apply here as in any other field. The indie scene was around in the 80s, 90s – actually, every 10 years there is a new scene.

Nicolas Zanotti: I saw on your website that you have several types of games that you sell on your website. Advergames, edutainment. So, are all these games indie games or would you call yourself an ad agency that produces games for these purposes?

Bastiaan van Rooden: We are neither indie game developers nor are we an ad agency. We are a service provider that makes games. We offer the service that is: design, concept and development. We make ad games, serious games, edutainment games, e-learning. We have no ready-made products. Our games are the outcome of the client's wishes. What we do, is, we find the appropriate channels for the respective games, but we would definitely not call ourselves an advertising agency. However, we rely on PR agencies and advertising agencies for communication purposes. Also, for us to sell indie games, we would have to be a product firm, which we are not. We merely produce games on demand. We are a service company.

Nicolas Zanotti: Do you collaborate with graphic artists and musicians when you produce a game?

Bastiaan van Rooden: We do our own designs but we commission other people to compose the sound. We have a lot of our own licensed material. It is often the case, that our own designers invest half a day to create sound effects and music loops. However, if the client wants something handcrafted, composed by a musician, then this naturally raises the price for the game, since license issues and Suisa [Cooperative Society of Music Authors and Publishers in Switzerland] come into play. Most of our clients find the second option to be too expensive.

Nicolas Zanotti: How big are your teams when you begin a new project?

Bastiaan van Rooden: There are a maximum of two developers that work on a project. The normal set up is: one designer, one developer and someone that does the concept. Then, during the test phase, we commission someone to test the game so we do not have to do it ourselves. So, four to five people, maximum, work on a game.

Nicolas Zanotti: I will now proceed to the game design process. At what instance do you determine your target audience and target platform, i.e. children using the iphone.

Bastiaan van Rooden: Take the example of the Login [Swiss Training Alliance of Transportation] process: Here, marketing already defines the target audience. The client never approaches us and asks if we could define a target audience, our job is to create a game that –we find– will be understood for the respective target audience.

Mark Gruber: As a developer, you are always surprised at how many hidden flaws you still find after the target audience tests your game. Flaws you do not anticipate during the conceptual phase.

Nicolas Zanotti: To what extent does the design change in the phase between concept and release? Do you stick to the initial design concept or are there cases where everything changes?

Bastiaan van Rooden: Well, if you look at the scribbles behind me, you will notice that they are a static concept scheme. But the thing is: a game is interaction, and you cannot sketch interaction. Which is why we operate on an iterative project management. Early on in the process, we create a prototype. Because only the prototype shows if something works and if the game can be played. Before that, you can only assume how a game could work. Therefore, a prototype is a vital element of game development. The idea for the game does not really change, but how the game is operated and played changes greatly. It is impossible to make decisions regarding game operation beforehand. Of course, you gain knowledge and can predict if an audience will understand your game. But one has to accept that game development is an iterative process. A brilliant idea does not determine if the game is going to be fun to play. And in the end, the game has to be fun.

Nicolas Zanotti: Who decides what fun is?

Bastiaan van Rooden: (Laughs) There is no predefined decision making process. There are opinions and exchange of ideas. The process, again, is iterative. This is also the challenge with which the developer is faced. He does not program fun into the game. People that work here have to understand what could create emotional attachment to the product. This is the great difference between a classic business software developer and what we do. We offer a user experience package. We do not merely develop functions, but also incorporate visual design and architecture. There is no decision making process. What matters is that, in the end, everyone agrees that the product works. It is a group decision. Then the product goes to the developer and he needs to decide if it is fun and throws the ball back at us. It is a fluent process.

Mark Gruber: To me, fun is when the consumer reaches that attention span you were hoping to achieve from your game. Also, when you do not have to explain too much how it works. I greatly appreciate it, when the user can figure it out by him/herself.

Bastiaan van Rooden: This is what we call added value. You do not need to read the manual, you immediately understand how it works and the product grabs your attention.

Nicolas Zanotti: Let’s talk about documentation. Do you have a game design document that several people are working on?

Bastiaan van Rooden: Because game design is a collaborative process, it is vital that we have the right tools. We do not work according to the waterfall technique where everything is predefined and we meet up after 3 months with our work. We have a wiki, we have tickets, we swap ideas, we have sprints which are self-contained feature releases, prototypes that we test. Then we have a basic concept. We have a scribble, we have formulated what the vision is. In some instances, we even create a small animation in Photoshop where we simulate what could happen. These things help us simulate interaction design. Then during the process there are some adaptations that the client has to check. And in the end, the documentation is almost finished. Everyone in the team gets together on a daily basis and at the end of the week we evaluate our work. This is called the Scrum method. This type of process is more complex than the waterfall process.

Nicolas Zanotti: Do you decide ad hoc what kind of tools and frameworks you will implement or do you work according to a technology?

Bastiaan van Rooden: We are not technology agnostic. Meaning internally we have Unity and AIR which is Flash. We know these two technologies very well. Currently, we are gaining our HTML5 know how. We would love to do more HTML5 projects but that is pricey and not every client has the budget for such a game. And if I tell the client that their intranet still runs on IE 6 [Internet Explorer] and that they could not view the game, this immediately becomes unattractive for the client. Now, HTML5 is hip and people think that Flash will disappear but we do not agree. Flash will have a long life span. There are many areas of application. And with AIR, Flash shows that it can improve itself. Nevertheless, we are very intrigued by HTML5 and would love to do more projects using this framework. But first all browsers need to meet the latest standards! (laughs)

Nicolas Zanotti: So, after you decided on the technology, do you then create a prototype of the design concept? Or does it flow directly into the process?

Bastiaan van Rooden: Again it is the same issue as it is with the prototype. You can make a scribble but then you cannot know if it will work – if it is going to be fun. If the prototype is not fun, then you can forget about the project.

Nicolas Zanotti: So the prototype tests the technology?

Mark Gruber: During my Bachelor studies, we would create a simple prototype that was made for the sole purpose of interaction. This would allow you to test the fun factor. In parallel, we created the visual components. This way, you can optimize both components and bring them together.

Nicolas Zanotti: Do you begin with a mock-up and wire frames? And how early on is the graphic artist involved during this process?

Bastiaan van Rooden: First, we determine the concept art sequence, since we have to be able to picture what the game could look like. Therefore, I ask the graphic artist, without mentioning any technical details, to do a scribble and color it. Basically, I ask him/her to act as if the game were already finished. This is the fastest process because a scribble can be done by hand or in Photoshop. It has to depict a vision, a story that we can sell to the client. Then we stop and create a visual prototype design and a functional prototype. And when those two prototypes match and are to the client's liking, then we move on and create a set-up for the production.

Nicolas Zanotti: So do you produce in-house tools? For instance, a level-editor or some other tool that converts data?

Bastiaan van Rooden: We do not create one per game, but we have many tools at our disposal when we create our games. This may be a small script. With each new project, we try to put aside some money from our budget in order to improve our tool chain. We see every new game as an opportunity to improve our tool chain. This is the tricky part, since all games run on different software. However, creating a level editor just for one game that only has e.g. 5 levels is not worth the effort. And if we know that we will create 8 games, we will, of course create one. But we do create one if we know that we can use it for several games in the future. We are very pragmatic in that respect. We put more emphasis on collaboration. A tool is still a tool. One could produce the best level editor, but if it is not used across games, there was much effort invested for nothing. Generalizing tasks for the tool chain is very time-consuming. It could take more time than creating the game itself. The bonus of creating a game, it is in itself closed code – its own world. As soon as abstraction is necessary, if the tool needs to work for the next game, good luck! It is very hard. On the other hand, we create frameworks. We have a code base, where, I would presume, 70% is the same. 30% is a layer on top is specific for the game. We want to reuse code as much as possible, because there are many functions, such as saving high scores, that we keep needing. We also have web services that the games connect to. Using them for statistics for example. We can use them as completely separate as tools.

Mark Gruber: Editors only make sense if the scope of the game justifies it and when one wants to outsource the tasks. Meaning, if you give a tool to a designer so he can build together the levels, then it makes sense. But with engines like Unity, which is already an editor, one has to ask oneself if it should be implement.

Nicolas Zanotti: Simple question: At what point are bugs corrected? Do you correct them ad hoc or do you create a "to do list" and take care of them at the very end?

Bastiaan van Rooden: Depends on the level of difficulty. If we have sprints of a one-week duration, then one can even leave them there for weeks, especially if one knows they are not relevant at the moment. If it is relevant for the production process, then we, of course, take care of them immediately. If it is a visual mistake, so be it; that does not matter at the moment. Prototypes are a mere collection of bugs. The challenge presents itself at the very end of a project, where it shows how well the software was written. When the architecture is right, and the iterations are right then there are not many bugs to fix. Because they were taken care of along the process. But there is software where many things go wrong because there were many conceptual changes made that could not be prevented. Then there is a massive hodgepodge of bugs. That is a terrible process in the end and it means that the project did not go well and funny enough, it is not the developers fault, as one might think. Just the whole process went wrong.

Nicolas Zanotti: Which means too many opinions have been changed during the project?

Bastiaan van Rooden: Correct, that is the worst that can happen, because you have to change the architecture and then you might have some legacy code in there that was left behind – despite agile development, that allows changes being made. But sometimes there are changes of direction where you have to decide if you want to go back two steps and start fresh. Of course we try to avoid this, based on our experiences and on the basis of the budget that we are asking for, when we realize a project needs too many sprints and we might run into time issues in the end. But we really need to keep a close eye on the bug fixing process, since it is no standard software, and each time we are dealing with a new a product and new bugs. Then we hope that in the testing process, where the client has to be involved, they can report bugs well.

Nicolas Zanotti: Did it ever happen that the game is not fun, but you knew what would make it fun? Did you ever think you would like to restart? An exit strategy of sorts?

Bastiaan van Rooden: Not really. With these processes it was never necessary. However, in all these years there was one single project where I thought to the very end that conceptually it was not the right move but I could not talk the client out of it. This was because the client, in terms of strategy, was not sure what he wanted to communicate. Such a product is the mirror of insecurity. The more you know what you want to do and what you want to communicate, the more bug-free, the more stabile and fun your product will turn out in the end. But like I said: This was a total exception. It is very rare that you have to deal with such issues.

Nicolas Zanotti: Was it ever the case that a client had a completely different vision and that you were forced to change your design accordingly?

Bastiaan van Rooden: No, that was never the case. Our agile process prevents this from happening.

Mark Gruber: What can happen is that clients with no notion of games, who never played a game before, request that the game is fun to play. But then their view of fun differs from what you know from experience, actually works. This can lead to conflicts and it is difficult to find common ground. For these reasons, I would, from the very beginning, try to get a clear message about what it is the client wants to convey with his/her game.

Bastiaan van Rooden: And when those messages add up and when, in the end, you can say: "this is the best match for that message", then there is no reason to quarrel. Because fun is personal – just like humor or the type of comedy films people like. But when we are clear on what the common denominator is, what the message is, and the best way to communicate it, then there is no reason to debate whether or not it is fun for one or the other person.

Nicolas Zanotti: Am I right in presuming that, in the Scrum process, you hold presentations to the stakeholders on a regular basis?

Bastiaan van Rooden: Correct. Sometimes that depends. There are clients that just give us free reign and only want to see the vision. But I want to get clients involved in the process, because I do not want to hear in the end that the client had a completely different idea about the project.

Nicolas Zanotti: At what point do you optimize performance on the devices?

Bastiaan van Rooden: At the very end. Again, it depends on the architecture. And as a project manager, I need to trust the developer with creating a solid architecture that is capable to perform in the end. However, we have framework code, on which we build upon. We know that it has a certain performance. And, when you have already created countless products, then you already have a basic performance that is completely sufficient. It is like the Pareto principle that stops you from optimizing performance till the cows come home. It just has to run OK on that device. And if it does, then that is enough. In that sense you have to regulate the optimizations and then decide when to make the cut. That is very important. It can be the case that during a sprint, the game does not run at all.

Nicolas Zanotti: But then you know that you can correct that in the end?

Bastiaan van Rooden: Yes, of course. That happens all the time. At first you do a dirty code because you know you can fix it in the end.

Nicolas Zanotti: Did you ever release on many different platforms on the release phase. Meaning on day X there is a version on the tablet, on the web, etc. Or did you first concentrate on one platform and do variations on a later date?

Bastiaan van Rooden: We prefer to operate on one platform for the following reason: Usually there is the version we call 1.0 where there is always bug fixing involved. There are thousands of gamers on the game, then there's feedback, then there is a bug no one discovered before because it is so exotic that only 1 % found it. This, we correct and implement it onto the software which we use for another platform. However, if you launch all at once, you have so many different bugs to fix. And sometimes you have a different custom code for each platform and you have to fix the bugs many times over which takes up too much time. For this reason, we prefer a staggered release on the platforms.

Mark Gruber: At most, I do a stand-alone version and a flash version. But it is usually the case that one device is pre-defined.

Nicolas Zanotti: Was it ever the case that you had to go back and rethink the UI concept to adapt to smartphones and tablets, for instance?

Bastiaan van Rooden: That varies. I would say that with products there are also certain needs. For instance, if you have launched a mobile game, the client might come along and tell you that he would like the game to run on the iPad or on the web. Or, to use another example, the client might need the game for a trade show. Of course, we like the idea of follow-up orders and we optimized for the predefined target devices, but that does not automatically mean that the game will function on every device. Take the trade show example: You can transport it out on an iPad screen, but it is not optimal. We can play it on the iPad but it is not really suited for that kind of screen. It is a bit of a shame, really, but at the same time, we are glad to have a follow-up order. But then we have to show the client how to operate the device. But if he is at a trade show, he might not have a device with a touchscreen or a device that is connected to a video projector. And, at these shows there is only a small attention span, people are watching, you only want to demonstrate something for a couple of seconds. In short: You have to optimize the devices according to the situation. And in this sense, optimizing means optional in certain aspects. However, you have to ask yourself if you really want to launch something on specific platforms, because it can go wrong. For instance, we are assigned to do an iPad version, and then you get the feedback that it is horrible, that it looks like a small mobile screen. And this is not too well received by the market because they expect an optimized version for a different device.

Nicolas Zanotti: So in this instance you can advise the client that this will not work on the video projector, for instance?

Bastiaan van Rooden: Correct. This is where the aspect of user experience comes into play. It is not just about exporting onto a different screen. You have to reconsider the surroundings: Mobile is used on the train, iPad is at home in the evening. Consider this, because the user has more time to look at it closely and maybe show it to others. Whereas, in the train, you just seek distraction from all the commuters. So there you just want to take a quick glance, flip through it, etc. This is a holistic approach where user experience comes into play and we ask all these relevant questions. We are not just the producers and go with it.

Mark Gruber: I think it is vital to know for which device you want to optimize your product, since it has a huge impact on the game play.

Nicolas Zanotti: In that case, it is optimal to know in the very beginning that the game might also have to run on an iPad for instance.

Bastiaan van Rooden: Yes, definitely. I also ask this question early on. "Is it even possible that there is a job to port to another device?" If the answer is "No" then we might even reduce costs during development, where we agree that we do not have to prepare anything. Otherwise, we leave some doors open in the development process, in the architecture, so that there could be a different resolution. For the assets, we prepare a higher resolution: a rendering of 3D elements and 2D elements, we prepare a high resolution and then downscale the elements. But when it is clear that there's only going to be one screen, nothing else, then we spare the trouble. After all, we need to stay pragmatic. We don't just work for the heck of it.

Nicolas Zanotti: Now I would like to talk about post-release. I assume that, after you've finished a game, you promote it. Do you let someone else do that part for you?

Bastiaan van Rooden: Correct. We are no marketers. Some clients would like us to be; to offer tips and tricks on how get a product high up in the app store. We clearly distance ourselves from this process for the simple reason that we don't want to do it. We want to stay focused on our area of expertise. And we have a broad spectrum in that area already. But there are dedicated agencies who take care of that business. Their set-up differs from ours, they have sales people on the market, etc. There are serious and less serious offerings on the net, and we do not want to be involved because, on those platforms, the success rate of the product is being blurred. In the sense that, if you can't market it right, then the product is bad. We want to concentrate on the product. It is stable and it stands by itself, it has the potential to be marketed. But we don't want to be involved in the actual marketing process.

Nicolas Zanotti: Do you leave a support channel open for players to leave comments?

Bastiaan van Rooden: That differs and depends on the size of the product. Marc is currently working on a serious game for a school. There, we have to have a support channel open in order to do maintenance work. It is a very small target audience and there may be some iterations. In other cases, the client wants to appear in the credits and does not want people to know that we produced the game. In that case, he has to deliver adequate support and give us his bug reports. However, considering the size of our company, this is difficult, since we cannot offer 24 hour support. Nobody could afford this. Therefore, we just request for the bug report and we'll see how we can solve it. There is always the release channel, there is a new version, it needs to be published. Sometimes very slow procedures. It is rarely a quick fix on the same day.

Nicolas Zanotti: How long is such a game supported? When the requests for support stop? Or do you say: "The game is out for a year now, we're done with support."

Bastiaan van Rooden: Well, we have different products. We once did a knife game that only runs on the weekend. So after two days it's over (laughs). Or a web game that runs on Facebook for one year and that's repeated every year. And there, we're self-motivated to insert bug-fixes for the next release cycle. It all differs and one can say that we decide according to the product nature on how much attention it will receive. And of course there was never a game in history that didn't contain any bugs after it was launched. That just doesn't exist. It is even the case that bugs are left unfixed because fixing them would be too costly. Period. But in the beginning of a product launch, hell breaks loose for the next couple of weeks.

Nicolas Zanotti: So you consider this in your planning?

Bastiaan van Rooden: Definitely, in the project planning you cannot just begin a new project since you start again with a scribble, a prototype and again, you're in the conceptual phase. You have to schedule a minimum of one week extra after a product launch where somebody has to be available at all times. He/she is not allowed to take a vacation during that time because otherwise we run into serious issues. That does not say anything about the quality of the software; it's just normal. And the more players there are, the higher the chance of something more serious turning up.

Nicolas Zanotti: Did you ever have to deal with cheaters? For instance, in a sweepstakes?

Bastiaan van Rooden: (Laughs) There's a whole wiki page that deals with measures to protect a game. For instance you can attack via memory ram. There are highly developed hacking tools for e.g. Flash games which we protect with our various, top-secret methods (laughs). Again, we first check the relevancy, meaning we raise the bar according to the situation – for instance if cash prizes are included. We have a basic measures catalog that we raise according to the situation. We did Flash games where you could win one million Swiss francs. That was in 2004, and you can imagine how high the standards were. Then you negotiate with a bank as to which safety level you want to fulfill. They wanted 100% in the contract which just does not exist. You can go as far as 99.9%. Everything else is non-negotiable. And they wanted 100% and that is inexistent in the software industry. You could be dealing with a dated server component, or a hole in the firewall, then this or that doesn't function anymore, it can be the software. There are so many reasons. Moreover, IT progresses every day and there are new gaps, new attack vectors. We just have to stay up to date. But, again, we decide depending on the situation. But there, we had good experiences. We even have a monitoring tool that informs us of a possible attack, which we invalidate manually if we think it's nothing to worry about. And with a collection of log files we are able to discover certain behavior patterns, which enables us to decide if we want to allow something or if we have to blacklist it.

Nicolas Zanotti: As in data mining concept? For instance to check if a new player was created that only looses against another player.

Bastiaan van Rooden: Precisely, there's not just game security as such, but one could build a logic: It is just not humanly possible that one person can reach one million points in one day. To see that a high score peaks, that's jus not humanly possible. Nobody can accomplish that. But rather it is a linear progression, meaning that you get better and better, you can reach more points. And this we check. That is, the time factor versus the behavior of the gamer over this time. This is relevant for checking if someone's cheating.

Mark Gruber: Or there's a mistake in the game (laughs)

Bastiaan van Rooden: (Laughs) Or if there's a mistake in the game –exactly! That, we have to examine as well.

Nicolas Zanotti: The Wii version of Zelda comes to mind, where you merely have to walk backwards at the right moment to hack the device.

Bastiaan van Rooden: Okay, well, with the increasing complexity of these games; the production as such, there's endless amounts of code behind it. Which is why I am never surprised if there's a weird bug hiding out somewhere. It is just not possible to test everything.

Nicolas Zanotti: And now, the final question: In your experience over the years, was there an instance where the entire process was optimized – where you can say that this one thing really worked?

Mark Gruber: Unity. (laughs)

Bastiaan van Rooden: OK, in terms of technology, there was Unity. That changed the market in regards to the licenses for big engines, etc. That was market changing. Apart from that, funny enough, this agile approach. This wasn't decided, it was always the company culture. One just has to adhere to it. We have a linear progression and no disruptive elements where we said: "yes, that was it". It was always small, safe, conscious steps. And maybe I, as a person, am culture forming. I can, indeed, push and say: "we want to innovate, and whatnot", however, I want to be on the safe side, I want to test something multiple times and see if it works and then move on. That may not be the fastest progress but it's the most stable one. Of course, if a certain technology comes along. Well, I am on constant technology watch. But I am also interested in improving collaborative elements. For instance, in our company, we have a time tracking system that we developed and implemented. This may come as a surprise, but this time tracking system may have had a great influence on our collaborative work, our exchange of information in the team. It lead to the fact that we now can estimate the amount of time we need for a sprint. We can resort to numbers for instance during the last sprint "you worked 20 hours, you 30 hours, and the visual design took about so long, why did it sum to more?" Actually, we do self data mining, and this had a massive influence on the quality of our products. Not only on the quality but also on the behavior of exchange and offering, i.e. to make offers, where we can say "Sorry, dear client, but for under ten's of thousands you won't get this product, then we'll prefer not to do it. Before that, one took on assignments and thought that it would pay off somehow (laughs). But now we make way more conscious decisions before we say "yes, we can do that".

Nicolas Zanotti: That's a great insight.

Bastiaan van Rooden: I agree, we must evaluate our own work. And we developed a system of reflection. You get your own data, but you can also have insight into time tracks of others and it's associated with an information stream, which leads to the instance that we have teams that are well informed. Therefore, I would say that such things were altering for us –definitely. One can say that it is a surprising time tracking system for such a firm. But it depends how one goes about it. (laughs) That's about the same as when a firm would implement Scrum when before that, they did waterfall.

Nicolas Zanotti: Yes, that would take years.

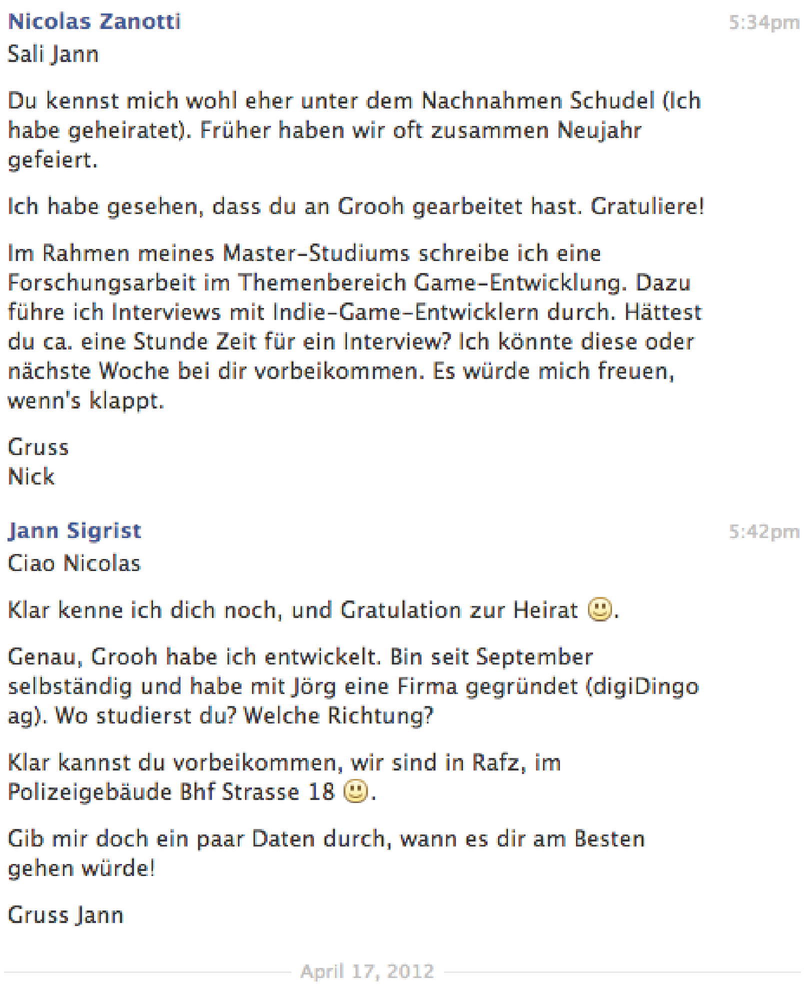
Bastiaan van Rooden: Correct. And we implemented that 2.5 years ago. And slowly but surely, the effects become visible, it took considerably long, but, alas, it's a process. You decide upon it. But it has about the same impact as a company switching from waterfall to Scrum. And it definitely had that influence on us. So, you have a technological input and the internal collaboration aspect. (laughs)

Nicolas Zanotti: Very good. Thanks to both of you for your time.

Bastiaan van Rooden: You're welcome.

G. Written Correspondence

Facebook correspondence with Digidingo



Nicolas Zanotti 5:34pm

Sali Jann

Du kennst mich wohl eher unter dem Nachnahmen Schudel (Ich habe geheiratet). Früher haben wir oft zusammen Neujahr gefeiert.

Ich habe gesehen, dass du an Grooh gearbeitet hast. Gratuliere!

Im Rahmen meines Master-Studiums schreibe ich eine Forschungsarbeit im Themenbereich Game-Entwicklung. Dazu führe ich Interviews mit Indie-Game-Entwicklern durch. Hättest du ca. eine Stunde Zeit für ein Interview? Ich könnte diese oder nächste Woche bei dir vorbeikommen. Es würde mich freuen, wenn's klappt.

Gruss
Nick

Jann Sigrist 5:42pm

Ciao Nicolas

Klar kenne ich dich noch, und Gratulation zur Heirat 🙂.

Genau, Grooh habe ich entwickelt. Bin seit September selbständig und habe mit Jörg eine Firma gegründet (digiDingo ag). Wo studierst du? Welche Richtung?

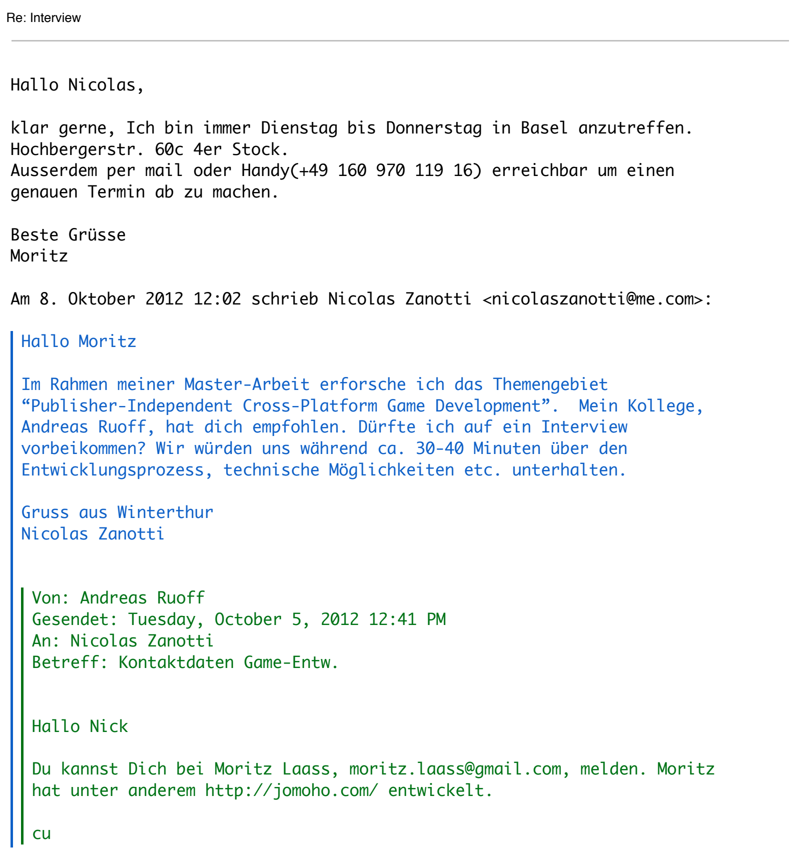
Klar kannst du vorbeikommen, wir sind in Rafz, im Polizeigebäude Bhf Strasse 18 🙂.

Gib mir doch ein paar Daten durch, wann es dir am Besten gehen würde!

Gruss Jann

April 17, 2012

E-Mail correspondence with Jomoho.



Re: Interview

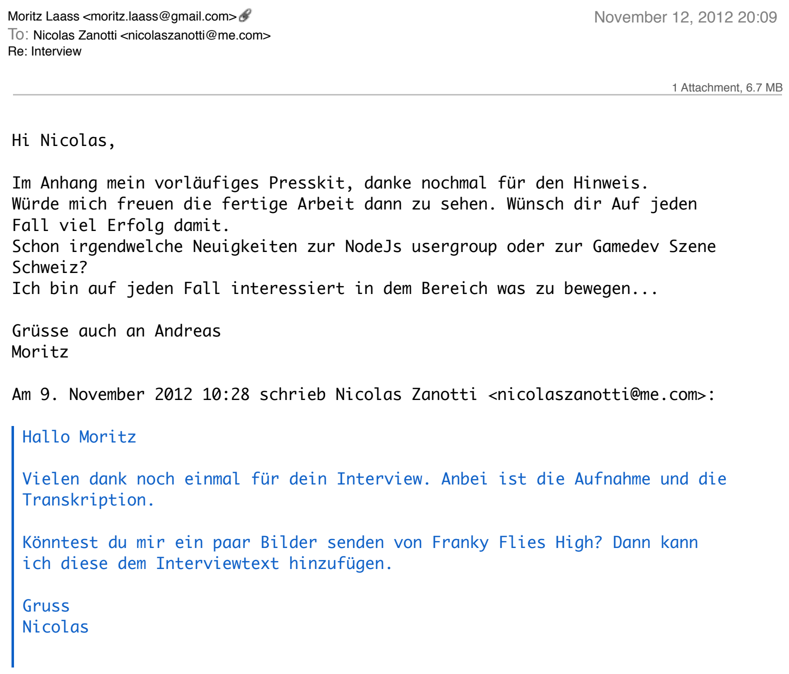
Hallo Nicolas,

klar gerne, Ich bin immer Dienstag bis Donnerstag in Basel anzutreffen.
Hochbergerstr. 60c 4er Stock.
Ausserdem per mail oder Handy(+49 160 970 119 16) erreichbar um einen
genauen Termin ab zu machen.

Beste Grüsse
Moritz

Am 8. Oktober 2012 12:02 schrieb Nicolas Zanotti <nicolaszanotti@me.com>:

> Hallo Moritz
>
> Im Rahmen meiner Master-Arbeit erforsche ich das Themengebiet
> “Publisher-Independent Cross-Platform Game Development”. Mein Kollege,
> Andreas Ruoff, hat dich empfohlen. Dürfte ich auf ein Interview
> vorbeikommen? Wir würden uns während ca. 30-40 Minuten über den
> Entwicklungsprozess, technische Möglichkeiten etc. unterhalten.
>
> Gruss aus Winterthur
> Nicolas Zanotti
>
> > Von: Andreas Ruoff
> > Gesendet: Tuesday, October 5, 2012 12:41 PM
> > An: Nicolas Zanotti
> > Betreff: Kontaktdaten Game-Entw.
> >
> > Hallo Nick
> >
> > Du kannst Dich bei Moritz Laass, moritz.laass@gmail.com, melden. Moritz
> > hat unter anderem http://jomoho.com/ entwickelt.
> >
> > cu



Moritz Laass <moritz.laass@gmail.com>
To: Nicolas Zanotti <nicolaszanotti@me.com>
Re: Interview

November 12, 2012 20:09

1 Attachment, 6.7 MB

Hi Nicolas,

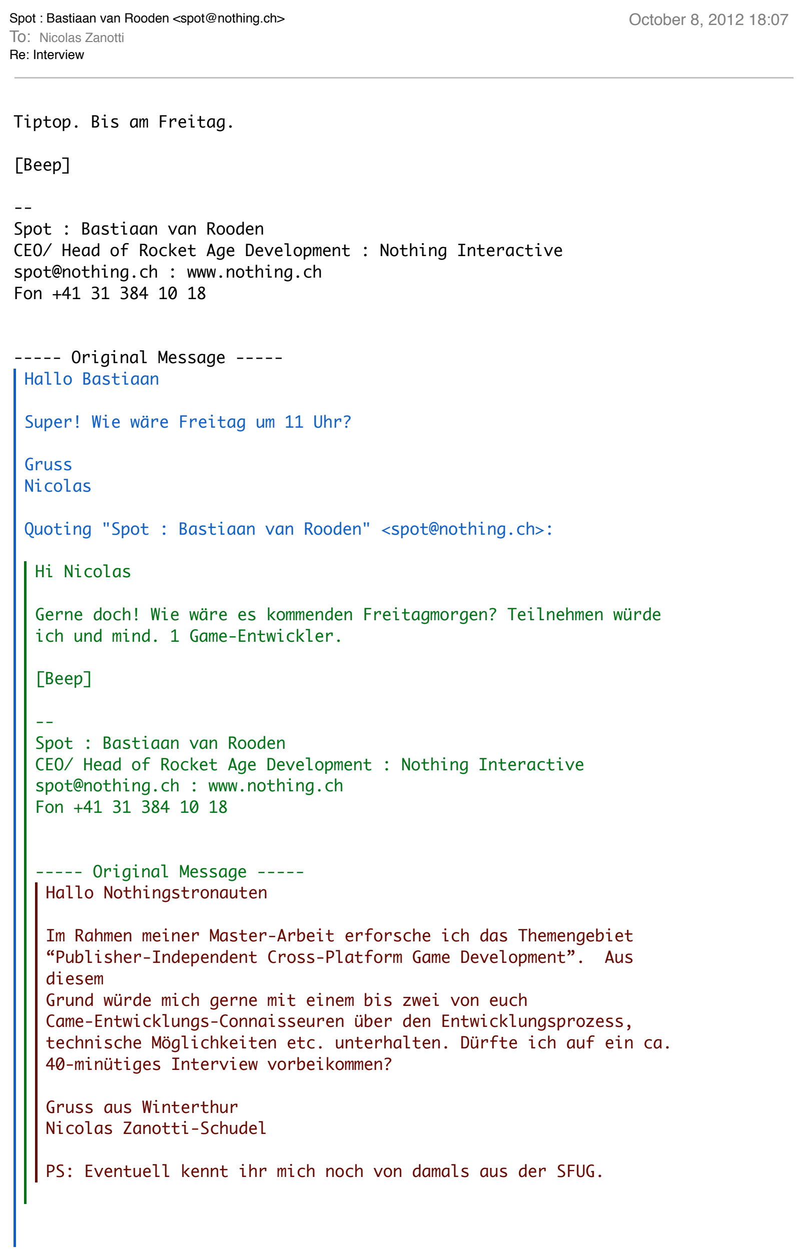
Im Anhang mein vorläufiges Presskit, danke nochmal für den Hinweis.
Würde mich freuen die fertige Arbeit dann zu sehen. Wünsch dir Auf jeden
Fall viel Erfolg damit.
Schon irgendwelche Neuigkeiten zur NodeJs usergroup oder zur Gamedev Szene
Schweiz?
Ich bin auf jeden Fall interessiert in dem Bereich was zu bewegen...

Grüsse auch an Andreas
Moritz

Am 9. November 2012 10:28 schrieb Nicolas Zanotti <nicolaszanotti@me.com>:

> Hallo Moritz
>
> Vielen dank noch einmal für dein Interview. Anbei ist die Aufnahme und die
> Transkription.
>
> Könntest du mir ein paar Bilder senden von Franky Flies High? Dann kann
> ich diese dem Interviewtext hinzufügen.
>
> Gruss
> Nicolas

E-Mail Correspondence with Nothing Interactive



Spot : Bastiaan van Rooden <spot@nothing.ch>
To: Nicolas Zanotti
Re: Interview

October 8, 2012 18:07

Tiptop. Bis am Freitag.

[Beep]

--
Spot : Bastiaan van Rooden
CEO/ Head of Rocket Age Development : Nothing Interactive
spot@nothing.ch : www.nothing.ch
Fon +41 31 384 10 18

----- Original Message -----

> Hallo Bastiaan
>
> Super! Wie wäre Freitag um 11 Uhr?
>
> Gruss
> Nicolas
>
> Quoting "Spot : Bastiaan van Rooden" <spot@nothing.ch>:
>
>> Hi Nicolas
>>
>> Gerne doch! Wie wäre es kommenden Freitagmorgen? Teilnehmen würde ich und mind. 1 Game-Entwickler.
>>
>> [Beep]
>>
>> --
>> Spot : Bastiaan van Rooden
>> CEO/ Head of Rocket Age Development : Nothing Interactive
>> spot@nothing.ch : www.nothing.ch
>> Fon +41 31 384 10 18
>>
>> ----- Original Message -----
>>
>>> Hallo Nothingstronauten
>>>
>>> Im Rahmen meiner Master-Arbeit erforsche ich das Themengebiet “Publisher-Independent Cross-Platform Game Development”. Aus diesem
>>> Grund würde mich gerne mit einem bis zwei von euch Came-Entwicklungs-Connaisseuren über den Entwicklungsprozess, technische Möglichkeiten etc. unterhalten. Dürfte ich auf ein ca. 40-minütiges Interview vorbeikommen?
>>>
>>> Gruss aus Winterthur
>>> Nicolas Zanotti-Schudel
>>>
>>> PS: Eventuell kennt ihr mich noch von damals aus der SFUG.