The impacts of a modular design strategy on modern video game development: An explorative study

Thesis MSc Business Administration

Innovation and Entrepreneurship

Enschede, *July 23, 2015*

University of Twente
Postbox 217
7500 AE Enschede
www.utwente.nl

ing. P.S. Boekhoud (s1248405)
Student Business Administration
University of Twente
Faculty of Behavioural, Management and Social sciences
p.s.boekhoud@student.utwente.nl

Supervisor:
Dr. ir. E. Hofman
Assistant professor
University of Twente
Department of Organization Studies and Innovation
e.hofman@utwente.nl

Co-reader:
Dr. ir. K. Visscher
Assistant professor
University of Twente
Department of Science, Technology, and Policy Studies
k.visscher@utwente.nl

UNIVERSITY OF TWENTE.

Acknowledgements

This study is a master thesis for the Master of Science degree Business Administration at the University of Twente, the Netherlands. At this point I would like to thank a number of people who helped me finishing this project.

First, and most of all, I would like to thank my supervisor Erwin Hofman for his support during the entire thesis period. From the moment I decided to graduate on the topic of modularity, I knew that Erwin would be the best supervisor for the job. During the project, we have had intensive and productive brainstorm sessions which helped to clarify difficult issues. I apologize for the times I accidently kept your coffee card in my possession.

Second, I would like to thank Klaasjan Visscher for his guidance and extensive knowledge on doing research. I would also like to thank Mohammadreza Khelghati for his work on crawling the database used for this study.

Next to that, I would like to thank my friend and fellow BA-student Koen Kuijpers for the numerous hours we spent in the library which boosted motivation and inspiration. I am sure that our complain sessions in the coffee corner eventually contributed to the completion of this thesis. I wish you all the best during your PhD career.

And finally, I cannot thank my parents enough for the support I received during my entire study period, especially in difficult times.

Enschede, July 2015

Peter Sebastiaan Boekhoud

Contents

The impacts of a modular design strategy on modern video game development: An explorative study

Peter Sebastiaan Boekhoud

Abstract

Over the last decades, modularity has received a lot of attention across numerous management scholars. Modularity has shown to be a frequently used technique to overcome complexity in all kinds of systems. The development of a video game has shown to be increasingly complex to manage. However, little academic attention is given to study the possibilities and consequences of following a modular design strategy in the video game industry. This explorative study seeks to clarify how a modular based design strategy influences some critical performance indicators for a video game development project. Next to that, we develop a measurement model and use it empirically assess the influence of synergistic specificity on a modular design strategy. Study results are based on a survey in which 16 worldwide operating game development studios participated. The initial study results indicate that modern video games are often based on a modular design strategy which tents to have a positive influence on the cost efficiency of a development project. Results indicate that an important precondition of implementing a successful modular design strategy is the support of clearly defined design rules. Next to that, this study contributes to the current modularity stream by operationalizing the concept of synergistic specificity and by empirically testing its influence on modularity. From the current findings, it is argued that synergistic specificity complicates the implementation of a modular design strategy and decreases modular benefits. The newly developed measurement items in this study can be used by researchers as a guideline for further research on the concepts of modularity, especially influenced by synergistic specificity.

Keywords: Modularity, design rules, synergistic specificity, video games

1 Introduction

Over the past decades the software industry has gone through substantial technological advances in for example process speed and integrated circuits capabilities, affecting complexity in many consumer products. An industry in which the complexity of system development has dramatically increased over the past two decades is the video game industry (Blow, 2004). Developing a new game is a multi-disciplinary undertaking that has become increasingly difficult to manage (Kanode & Haddad, 2009). Games have progressed from some simple moving figures (e.g. PacMan) up to three dimensional environments in which characters move and behave as realistically as possible. Through the development and implementation of techniques in the field of computer-human interaction, the gaming industry has shown to be an interesting field to study for other software industries, especially for virtual reality. A video game engine – which is at the heart of every game – is a promising tool to create realistic virtual worlds in which collision situations can be simulated, which traditional virtual environment software is not able to do (Trenholme & Smith, 2008).

The growing importance and magnitude of the gaming industry is reflected in worldwide game revenues, developing from 78.8 billion U.S. dollars in 2012, up to 93.2 billion in 2013 and 102.2 billion in 2014[1]. In some

[1] Gartner – Statista 2015

game development teams, more than three hundred people participate. This results in ever increasing game content which almost reaches movie-like reality. Every new game sequel is an upgrade in quality, speed and creativity. However, this increased complexity often comes at a price. Selling prices have remained around the same level for years while investments in new software technology increased, resolving in a lack of profitability across a lot of games (Polygon, 2012).

The difficulty in game development is to strike a balance between exploring and applying new technologies such as improved graphics to the game, while at the same time making sure the game is launched prior to a competing game (Petrillo et al., 2008). This competitive pressure especially comes to light on key moments in the year such as Christmas, as recently reported in the media.[2] The ever increasing development budgets and system complexity in combination with the challenge to develop an innovative and impressive game creates a pressing need for managers to understand how various development strategies influence game performance. However, little academic attention has been given to studying the gaming industry. Apart from 'serious gaming', game development is an underexposed topic among business and engineering scholars.
Confidential part is intentionally removed from thesis (1)

The foundations of the study's conceptual framework are based on a literature study on the concepts of modularity and synergistic specificity, which is described in the first section. Based on these concepts, the research model with the corresponding hypotheses is given in chapter three. The data collection and analysis method are discussed in chapter four. Successful game development is assessed via three performance measures as quantified in section 4.2. Study results are presented in chapter five. Implications and conclusions are discussed in the remainder of the thesis.

2 Conceptual framework

2.1 Product modularity

Modularity is a major research topic among business management and industrial engineering literature. Studied from different perspectives, modularity has become the general term for decomposable and reconfigurable systems. The current literature stream discusses three main dimensions: modularity *in use*, modularity *in production* and modularity *in design* (Baldwin & Clark, 1997).

Modularity *in use* discusses product offerings, often referred to as mass customization (Duray et al., 2000; Schilling, 2000). This dimension discusses modularity from a customer perspective and advocates how organizations benefit from an increase in product flexibility. By modularizing a system, individual system components can be easily interchanged resulting in a wide range of product variants and serving the increased heterogeneous demands of customers. This is a well-known product strategy in automotive industry in which a vast amount of product combinations can be created. By being able to combine different engines, interiors and other details, customers can create a personalized car without the high costs of a unique product.

[2] "**New games often full of mistakes"** Game publishers have to release their game in November due to large revenues of Thanksgiving and Christmas. However, games are getting more and more complex and due to time pressure, big development mistakes arise – *http://www.nos.nl/ November 28th, 2014*

Modularity *in production* describes how the production of complex systems can be organized into separate tasks (Sanchez & Mahoney, 1996).

Modularity *in design* covers the architectural characteristics of a modular product. A system is associated with a high degree of modularity when its architecture is decomposed into subsystems by standardizing the interfaces between those subsystems (Baldwin & Clark, 1997; Lau et al., 2011; Sanchez & Mahoney, 1996; Schilling, 2000). Because each component is assigned a particular function, they can be developed individually yet work together efficiently through standardized interfaces. This type of product design demands a specific design course with visible and hidden design parameters (Baldwin & Clark, 1997). This study focuses on modularity in product design. More specifically, we aim to get a deeper understanding how the modular design principles affect firm performance in the video game industry.

Modularity and complex systems

The major argument to follow a modular product strategy is to overcome complexity within product design (Baldwin & Clark, 2000). Complexity is a somewhat elusive concept. As one of the leading authors explaining complexity in systems, (Simon, 1962, p.468) defines a complex system as "*one made up of a large number of parts that interact in a nonsimple way. In such systems, the whole is more than the sum of the parts."* A system is a hierarchy of several subsystems. Subsequently, a subsystem can be compromised out of several smaller subsystems, which in turn are built up from single components. Notice that in many articles the terms 'system', 'component' and 'module' are used interchangeably. The main tendency in modularity research is that a component is the lowest system attribute. In this paper, a system is the overarching term for a complete set of components and subsystems which together form the product. A system is built up from subsystem – or modules – which themselves are built up from components. A component is a distinct portion of a product which has a function (Clark, 1985). Hence, the process of how components arise depends on how functionality is allocated and will be different of every product type.

Interaction exists between components *within* a subsystem. A step higher in the hierarchy is the interaction *among* subsystems. One could think of several components which together form a car engine. A car engine is a subsystem of the total system – the car. The components within in the car engine have to work together in a specific way. The output of the engine is then transferred to the drive shaft – another subsystem. The interaction *within* a subsystem, called cohesion, is essentially different than interaction *among* subsystems, called coupling. In his comprehensive literature review and reconceptualization of the concept of product modularity, Salvador (2007) clearly explains the difference between these two constructs. Cohesion is the strength of the links between the constituents of a system. A subsystem arises when this cohesion applies to particular group of constituents. These so called clusters of elements – in the current research referred to as subsystems or modules – interact with other groups of cohesive elements. This interaction *between* multiple subsystems is known as coupling. Two subsystems are coupled if a change made to one subsystem requires a change to the other in order for the overall product to work correctly (Ulrich, 1995). The characteristic of a modular product architecture is to stimulate cohesion *within* a subsystem and prevent complex coupling *between* subsystems so that these subsystems can be developed independently without knowing the internal workings of other subsystem.

Modular product architecture

A product architecture is defined as 1) the arrangement of *functional elements; 2)* the mapping from *functional elements to physical components* and 3) the specification of the *interfaces* among interacting physical components (Ulrich, 1995, p. 420). A modular product architecture is one in which the functions of a product are attributed to a particular component – whereas a complex (integral) product has its functions distributed across several components (Ulrich, 1995).

The type of interface between interacting components form the essential element which distinguishes modular from integral product architectures. Interfaces exist between two components that are dependent on each other's functionality (Sosa et al., 2004). They describe in detail how product components and modules interact, including how they fit together, connect and communicate (Baldwin & Clark, 1997; Mikkola, 2006). In modular product architectures, interfaces are standardized so that a large range of components can be developed according to these standards, allowing easy mixing and matching (Baldwin and Clark, 1997; Lau et al., 2011; Garud and Kumaraswamy, 1995; Mikkola, 2006). A modular architecture is essentially different from an integrated architecture, in which the successful operation of any given part is likely to depend on the characteristics of many other parts throughout the system (Langlois, 2002, p.21). Standardized interfaces do not change during the development phase of a system. As a result, a modification of a subsystem does not influence the overall design of the architecture (Sanchez, 1995).

According to the concept of a 'nearly decomposable system' two product elements will always have some coupling (Simon, 1962). In practical terms, coupling is only relevant when one component influences the internal workings of another. Because of this dependency, individual components cannot be designed and produced without knowing the workings of another component. Through modularizing components, specialists can focus on developing an individual component, which increases the innovative possibilities of that element.

However, the functionality of a system is not only dependent on the performance of individual components, but on the extent to which they work *with* each other (Henderson & Clark, 1990). This refers to the notion of Simon (1962), stating that complexity of a system is more than the sum of individual parts. Therefore, loose coupling might negatively influence overall system performance. As the current research seeks to clarify how modularity influences game performance, we further examine what factors influence the decision for both loose or tight coupling – and thus the degree of modularity – within a product architecture. As we will discuss in the remainder of this chapter, synergistic specificity is the central factor is this discussion.

Confidential part is intentionally removed from thesis (2)

Standardization

In modularity literature, 'standards' are described in different ways. Some authors describe standards and interfaces as the same phenomena. In this research we follow the classification used in Baldwin & Clark (1997), which clearly separates these concepts. Modularity is achieved through *design rules*, which cover three elements: *architecture, interfaces* and *standards.* The architecture describes the elements and function of these elements in

a system. Interfaces describe how the elements communicate and standards guide the conformity of the elements (Hofman et al., 2009).

From these three elements, standardized interfaces receive the most attention in modularity research (Mikkola, 2006; Sanchez, 1995; Tiwana, 2008; Ulrich, 1995). Through standardized interfaces, producers know how to design components in such a way that it will successfully interact with other components, supporting overall component compatibility. Instead of developing new components for a new product, standards enable components to be used interchangeably over several product variations, also known as design reuse (Ettlie & Kubarek, 2008). In the example of the car, the engine might be used for a range of car models. Some models of Audi, Volkswagen and Seat all use the same engine since they are built with the same design standards. This results in so called product families (Hofer & Halman, 2005). A product family uses the same platform, which forms the basis of the product architecture. For example, the Volkswagen Group uses the famous MQB platform. Obvious advantages are the cost savings which can be obtained by not having to design and test newly developed product parts, but instead use a part which is already known to work. Next to that, development cycle times of a new product will decrease. However, there are several constrains which have to be dealt with when relying on design reuse.

An important aspect in reuse choices concerns the type of component or module which is to be reused. Depending on the place in the systems' hierarchy, some modules might be easier to reuse than others. Certain modules have a central place in the hierarchy. Such modules have a large span of control (Simon, 1962): they have a large number of other modules to which they are coupled and have to interact. Therefore, they can be seen as *critical* modules (Mikkola, 2006). A module high up the in the hierarchy influences the underlying modules and is therefore harder to reuse than lower up the hierarchy modules. One can imagine that a window wiper is easier to reuse than a car engine. The car engine will be coupled to a lot more underlying components than a window wiper. In software literature, the ease of reuse of a certain component is defined as reusability: the degree to which a thing can be reused (Frakes & Terry, 1996). For example, reusing the engine of a video game will have a major impact on all the other functional elements of the game, as well as on how it looks like (Gregory, 2009). As a result, different games which have been developed with the same engine might be too similar to each other (Kanode & Haddad, 2009). This issue brings up an important downside of relying on design reuse.

The use of standard components opposed to newly developed parts can restrain overall product innovation (Ettlie & Kubarek, 2008). It sounds reasonable to believe that a newly developed game with newly developed elements (i.e. figures, levels and other game content items) has got a stronger innovative capacity than a game which relies on standard content which has been used in previous games. Consequently, developers need to strike a balance between modular benefits while retaining a certain degree of innovativeness. Newly developed elements might bring new technologies into the game, but since they might not match with established standards, compatibility issues may arise since these elements need to be tested and reevaluated (Mikkola, 2006).

Confidential part is intentionally removed from thesis (3)

2.3 Modularity in the software and gaming environment

As with other (tangible) systems, software is increasingly being developed in modular forms, also known as component-based software development (Sametinger, 1997). Software components are encapsulated single entities and can communicate with other components through clearly defined interfaces (Szyperski et al., 1999). According to Sametinger (1997, p. 4) "*using software components to build software systems almost automatically leads to software reuse.*" Reuse is a major research topic among a large stream of software literature, defined as "*The process whereby software systems are created from an existing software rather than building them from scratch"* (Ajila & Wu, 2007; Krueger, 1992; Mohagheghi & Conradi, 2008; Sametinger, 1997). By including the word 'systematic' in the definition, the concept gets a strategic connotation, a deliberate choice of software developers. Given this qualification, reuse is not only a software development practice, but is a business strategy in which an organization is completely devoted to reuse. Modularity in software is expressed via reuse and therefore this section examines how reuse influences product performance.

Software architectures

As discussed in the modularity research section, the place of a component in a system's hierarchy is an important factor determining its ease of modularization. This issue is also applicable in software systems. A software artifact is a broad term for all kinds of software objects and configurations. Reuse can be applied at the level of a few lines of code or by reusing a class, method or a whole system (Ajila and Wu, 2007). In the wide stream of software literature 'components', 'objects' and 'classes' are used interchangeably. This research uses the wide spread definition of (Szyperski et al., 1999): "*A software component is a unit of composition with contractually specified interfaces and explicit context dependencies only. A software component can be deployed independently and is subject to composition by third parties.*" This definition has two major characteristics that are important in the context of modularity.

First, components communicate through specified interfaces, which is an essential pre-condition for a system to have a modular form. Second, components can be developed individually. This means they are isolated from its environment and other components. This is comparable with the loose coupling theory in modularity research. Sametinger (1997) simplifies this by stating that components are individual entities which can be reused rather easily. In recent years, software engineers have been able to encapsulate a wide range of functions into a component. In the gaming industry for example, one component might be responsible for the artificial intelligence among game characters

Frameworks are the biggest software entities in a software system. A framework is an architectural design which describes how components interact (Campbell et al., 1993). Without such a framework, individual components cannot communicate. Due to stable interfaces, the framework offers reusability and extensibility possibilities (Fayad & Schmidt, 1997). A framework is not a component on its own, but a group of many components which can be modified and extended individually (Sametinger, 1997). In the gaming industry, comparable frameworks exist in the form of game engines. A game engine includes all elements which define the architecture and workings of a game, and is separated from the actual game content (Bishop et al., 1998; Gregory, 2009). A game engine is crafted for a particular game genre such as a First Person Shooter game. The engines *Unity3D, Doom, Unreal* and *Quake* are examples of worldwide known game engines. Engines were among the first reusable software artifacts in the gaming industry (Folmer, 2007). Developing a game engine from scratch is expensive and very difficult and therefore game developers often license existing game engines

(Blow, 2004). Because of these new reuse possibilities, the speed of game development dramatically increased. Over the years, more specific game components such as physics and artificial intelligence were developed for reuse by specialized teams.

Software development strategies

An important factor influencing the impact of a development strategy on firm performance is the magnitude of a reuse project. One can choose to approach each new development project as an individual case, or follow a specific design course across a range of projects. Developing components so that they can be used in future projects is called developing *for* reuse, while developing system of reusable components is known as development *with* reuse (Karlsson, 1995). An organization that develops software can stress to make the modules and components reusable for future projects, enabling future projects to be built from reusable components developed in the past. In this case, an organization first applies developing *for* reuse and later develops *with* reuse. Developing with reuse does not require an organization to develop for reuse since reused software entities can also be extracted from outside the own organization.

Confidential part is intentionally removed from thesis (4)

3 Theory and hypotheses

The present study's conceptual framework seeks to explain the relationship among the different concepts discussed in the previous sections and their influence on business performance in the video game industry. Increasing pressure on development time and development budgets are currently two big issues in game production. Therefore, the hypotheses focus on the effect of the research concepts on developed costs, profitability and game quality.

Confidential part is intentionally removed from thesis (5)

3.4 Control factors

Whereas the research model takes into account most important influencing factors, we control for a number of variables.

Development budget. It sounds reasonable to believe that a game which has been developed with a large development budget can do things which a game with a lower development budget cannot do. Therefore, the model controls for *development budget*.

Experience level. Games require deep technical knowledge and it can be difficult to integrate (external) reused components and there is a lot of specific programming knowledge required (Blow, 2004). In order for a game to be innovative, it requires technical skills for programming and design (Zackariasson et al., 2006).

Number of people working on the game. The number of people working in the game by itself does not make a game more innovative than others. However, it seems reasonable to believe that a game which is built by a hundred people can be developed faster than a team with twenty people.

Game size. An extremely large game might be more complex and automatically be more difficult to be managed. Therefore we control whether there is a difference in game size in relation to development costs and development time.

Product platform. There might be a difference in game titles created for the very first time or a game sequel such as 'Spiderman 1' to 'Spiderman 2'. This difference is two sided. On the one hand, the sequel 'Spiderman 2' version might be created faster and with lower costs due to platform and experience advantages (Chai et al., 2012). On the other hand, game studios might invest an extra amount of budget and time in developing the first of the sequel 'Spiderman' to ensure platform advantages in the future.

Marketing budget. Profitability is, among other things, determined by development costs and by sales numbers. It is expected that sales numbers can be affected by the size of the marketing budget of a game developer.

4 Research methodology

4.1 Sample and data collection

The video game industry is a complex network of different actors that each have a specific function in delivering the game from manufacturer to customer. In this network, publishers, developers and distributors can be recognized as the main parties (Johns, 2006). Game development studios are responsible for the engineering, programming and content of the game. Hence, they are responsible to finish the project on time and within the set budget. In this study we investigate the effect of different development strategies on such performance measures and therefore, our sample is a group of worldwide operating game development studios. An online survey-based study was conducted to empirically validate the hypotheses. An advantage of an online questionnaire is that it reaches a lot of people in a short time. A risk is the low response rate. A key priority was to keep the questionnaire as simple and short as possible in order to enlarge response rate. We sent personal emails to the respondents with an explanatory text of the goal of the research and asking them to fill out the questionnaire for a particular game on which they participated, which we explicitly named in the invitation email.

A draft of the questionnaire was pre-tested by a panel group consisting of seven practitioners and academics to check for ease of use and correct interpretation of the measurements. After the feedback of one panel member was processed, the questionnaire was presented to the next member of the panel and so forth. This helped fine tuning the format of the questions and layout of the questionnaire. Prior to the large-scale survey, a final pretest was conducted with a game developer to check for understandability and ease of use.
Confidential part is intentionally removed from thesis (6)

The questionnaire covers a range of aspects on game development including technical questions, such as software reuse and development effort, and non-technical questions such as financial performance. In order to make sure the respondents were able to answer all these question types, this study relies on a key informant strategy (Kumar et al., 1993). In combination with the classification of a game development team by Gregory (2009), we interviewed two practitioners to gain insight what function profile within a game development studio

is capable of answering all the questions. We stressed to contact the head of development within a game studio. Similar titles used in development studios are 'program/technical director', '(production) manager', 'producer', 'lead programmer' or other related function titles. For some bigger development studios, multiple people were titled with these functions. In these cases, we sent out the survey to two or more people within the same studio. In case we were not able to find a personal email address, we used an '*info@company'* address and asked to have the email forwarded to a specific person which we named in the email. The sample size listed 210 people of which 183 email were successfully delivered. These 183 people represented 123 unique games, of which 13 email addresses where in the format of '*info@company.*' Eleven days after the initial email, a reminder was sent. Finally, a second reminder was sent ten days later. Of the initial sample of 123 games, 16 questionnaires were returned completely, representing 16 different game titles, resulting in a responds rate of 13%. Table 4.1 specifies the characteristics of the respondents. Due to confidentially reasons, game titles are not displayed. Respondents are fairly evenly distributed across release date and platform type. Games are often developed for both PlayStation 3 and Xbox 360 at the same time.

Table 4.1 Respondents overview

Function	**Platform**	**Release date**
Project/technical director	Playstation 3	2010
Producer	PS3 and Xbox 360	2010
Level designer	PS3 and Xbox 360	2011
Lead programmer	PS3 and Xbox 360	2011
Build engineer	PS3 and Xbox 360	2011
Project/technical director	PS3 and Xbox 360	2013
Lead designer	PS3 and Xbox 360	2011
Head of production	Playstation 3	2012
Lead programmer	Playstation 3	2013
Producer	Wii	2012
Lead system engineer	PS3 and Xbox 360	2012
Creative director	Xbox 360	2013
Creative director	PS3 and Xbox 360	2011
Project/technical director	Playstation 3	2011
Project/technical director	PS3 and Xbox 360	2010
Lead programmer	Xbox 360	2010

4.2 Measures

The questionnaire uses existing measure scales where possible to ensure validity and reliability. For constructs which did not have an existing measure or the existing measure does not fit the model, the construct measurement model of MacKenzie et al. (2011) is used as a guide to form new measurements items. We found the scale development procedure to be complete, clear and helpful. By emphasizing the conceptualization of a construct before generating measure items, this procedure secures construct validity.

After assessing the theoretical concept of the construct to be measured, a list of measure items was generated by a panel group consisting of experts in the fields of software systems, innovation processes and modularity. Each member individually generated a list of items which best represented the loading of the concept. These items were combined and listed into one complete pool of items. First, items which were similar to a great extent were merged into one item. Second, all items were assessed by each member of the group and rated for content

validity. Items that scored high with all members of the panel were added to the final item list. Items which scored low with all members of the group were removed from the list. Items that had varying scores throughout the panel were internally discussed and reconsidered. After all the items were individually assessed on content validity, the final list of items was combined and controlled for completeness, ease of understanding and content validity. Unless explicitly reported, for all measures a 7-point Likert scale is used. For a complete overview of the measurement items, see Appendix I.

The concepts of modularity, synergistic specificity and tweaking overlap to some extent. Except for modularity, these concepts have not been measured in existing literature. The most essential element which distinguishes the variables is the conceptual dimension. In this research, product modularity is measured at a development strategy level. Synergistic specificity is defined as a system's characteristic and is measured at a systems level by looking closer at the interaction between components. Tweaking is about modifying software and therefore is measured at an implementation level. The coming sections will elaborate on the specific formation of the construct measurements.

Independent variable

The research model consists out of one independent variable. *Modular design strategy* [MDS] captures to what degree the game was designed on a modular basis. In the current modularity literature stream no consensus can be found on the exact measurement model for modularity, resulting in an immense list of modularity definitions (Campagnolo & Camuffo, 2010; Salvador, 2007). Despite this inconsistency for the exact measurement model, some important generalizations about the concept of modularity can be recognized. One of these main elements is the decomposition of a product into separate components which are grouped into modules (Gershenson et al., 2004). This creation of modules (or subsystems) is also integrated in the following wide accepted definition which forms the basis for the item generation process in this study: "*Modularity is the degree to which a system can be decomposed into subsystem by standardizing the interfaces between the subsystems*" (Baldwin & Clark, 1997; Schilling, 2000; Sanchez and Mahoney 1996). Modularity is a relative concept, and consequently, measurement items should be on a relative scale.

Deriving from the definition above, a modular product uses separate modules, which is captured by including measure item of Worren et al. (2002) and standardized interfaces, captured by including the measure of (Tiwana, 2008). The measurement model is specified for video games by including measure items from software modularity literature. In the software environment, modularity is heavily related with software reuse. Furthermore, existing measures include reuse as a measure item for product modularity (Worren et al., 2002). Hence, the degree of reuse is included in the measure items.

In a systematic reuse program, organizations use a common architecture across several projects to maximize modular benefits, which is captured by including the measures of (Rothenberger et al., 2003).

Among other things, (Mikkola & Gassmann, 2003) measure the degree of modularity in a product architecture by measuring the among of so called 'New To The Firm' (NTF) components. A modular product design is said to use as many standardized components as possible to maximize cost efficiency and development speed. The measurement item of (Rothenberger et al., 2003) is included to capture the degree to which the development project relied on standardized components.

Dependent variables

Video game performance can be interpreted in a number of ways. In this research we measure game performance of game development studios. Over the last decades, it has become clear that many game developers have difficulties in scheduling the project and to remain within a planned budget (Kanode & Haddad, 2009; Petrillo et al., 2008). Therefore, we measure to what extent developers have been able to remain in the planned development budget. This study uses relative dependent measures to test the performance of a game development project so that scores on these items can be compared to other games which might differ in size and complexity. The research model includes three variables to test firm performance.

Cost efficiency [COE] represents to what degree the development project remained within the set budget. On a 7-point Likert scale, a high score represents the project to be developed as costly as expected. *Profitability* [PRF] represents to what degree the game has been profitable as expected. *Game quality* [GAQ] can be measured at a variety of ways. Comparing the quality of two games is complex do to the fact that no two games are the same and will differ in game play, customer group, genre and theme. In order to control for these differences, we measure the perceived quality of customers. In that manner, we measure the degree to which the game lived up to the expected quality level of the customer.

Table 4.2 Dependent variables

Variable	**Item source**
Cost efficiency [COE]	*COE01 and COE02 – Akgün et al. (2007)* *COE03 – own development*
Profitability [PRF]	*Song and Parry (1997)*
Game quality [GAQ]	*Atuahene-Gima (2003)* and *Sahay and Riley (2003)*

Moderating variables

The main research model includes three moderating variables. Whereas a modular design strategy is expected to have a strong influence on the performance measures as defined in the previous section, this relationship is bounded to the scores on the following moderating variables.

Design rules [DRL] represents to what degree a product design has clear set of rules about what function each part in the overall system has and how components interact and communicate, as well as integration protocols. For this measure we used the measure of Hofman (2010), who bases the measures on Baldwin and Clark (2000). Confidential part is intentionally removed from thesis (8)

Control variables

The overall research model controls for a number of variables. Table 4.2 gives an overview of the measurements. In order to control for the size of the development team, *Team size* [TSZ] is measured by the number of full time employees working on the game in the development studio. This excludes any external parties such as publishers or marketers. The *Experience level* [EXL] of the development team is assessed on a 7-point Likert scale ranging from ‘beginner’ to ‘expert’, based on Ajila and Wu (2007). To control for the *Game size* [GSZ], we asked respondents to indicate the total amount of lines of code (LOC). To check for *Development budget* [DVB] used for the game, we asked respondents to fill in the development budget in US dollars. To control whether

excessive marketing influences sales and profitability, we used Hofman (2010) for the control variable *Marketing budget* [MKB]. Since *Profitability* [PRF] is a relative measure, we used a relative measure for MKB (see table 4.3).

In this research, 'platform' is defined as a family of products derived from one central product design (Tatikonda, 1999). Due to a platform strategy, sequel games might have been developed with a smaller budget and less development effort. Existing measures to control for *Product Platform* [PPL] did not fully capture the loading of the construct in this research. Chai et al. (2012) focus on platform extensibility and success based on a formalized process. In this research, we seek to control for return on investments made for a platform. This way, profitability and development costs can be explained by platform dependency. A new measure was developed (α = 0.937) on a three item 7-point Likert scale (see Appendix I).

Table 4.3 Control variables

Variable	**Measure item**
Team size [TSZ]	Number of full time employees participated in developing the game
Experience [EXL]	*How would you indicate the level of experience of this development team?* (1 = Beginner, 4 = Moderate, 7 = Expert)
Game size [GSZ]	# Lines of Code
Development budget [DVB]	In US dollars
Marketing budget [MKB]	*Relative to other projects, marketing budget was high* (1 = Strongly disagree, 7 = Strongly agree)
Product platform [PPL]	Newly developed *(see Appendix I)*

4.3 Data analysis method

Data is analyzed via correlation and regression. Correlation by itself is not enough to draw conclusions upon the casual relationship between two variables (Babbie, 2010). However, it does give a meaningful insight whether the constructs in the research model behave as expected. Before performing the analyses, scale reliability and normality is assessed.

Scale reliability and validity

For all multiple item constructs, Cronbach's Alpha is used to determine internal measurement reliability. Some of the constructs have been newly developed for this research, but also existing scales were tested to determine whether they fit in the current research model. In this process, items which did not correlate substantially with overall construct were deleted (Field, 2009). This was done by looking at the inter-item correlation matrix. Consequently, a substantial increase of the Cronbach's Alpha in case of excluding the item resulted in the item to be deleted.

Whereas Cronbach's Alpha helps determine the reliability of a construct, it does not determine a constructs' validity, meaning whether the items combined correctly measure the intended. Content validity is tested via the method described in section 4.2. Construct validity is tested by analyzing correlations among other variables. Results are discussed in the next chapter.

A part of the hypothesis testing is performed via regression analysis, which assumes normality (Field, 2009). Normality is tested via the Shapiro-Wilk test. In this test, a non-significant value ($p > 0.05$) indicates that the data is normally distributed (Field, 2009). Chapter 5 discusses the results.

Hypothesis testing method

For the hypothesis testing, several tests will be used for the different hypothesis models. Both tests are run in IBM SPSS 22©. In the H1(a-c) model, one predictor variable (MDS) influences a single outcome variable for which we an ordinary least square (OLS) regression analysis. In the H2-4 models, a moderating effect is tested. In order to do so, an interaction effect between the main variable and moderating variable has to be found. A simple OLS regression is not able to show this effect since it estimates the beta value off all single predicting variables, while remaining neutral over all other variables in the model (Hayes & Matthes, 2009). In the moderating formula, we seek to find whether the change of one focal variable influences the regression value of another variable. Therefore, a multiple regression analysis has to be performed. In this case, we test whether the regression coefficient for *Modular design strategy* (MDS) changes as a function, for example *Synergistic specificity* (SYN). A widely adapted method to test this effect is done via the following equation (Hayes and Matthes, 2009):

$$\hat{Y} = a + b_1 F + b_2 M + b_3 (F \times M) + \sum_{i=4}^{k+1} b_i W_i.$$

The interaction effect is determined by looking at the beta value belonging to the product of the independent variable with the moderating variable (F x M in this equation). ϒ is the outcome variable, α is the constant factor, *b*1 is the beta effect of F, which is the main effect variable and is expected to change as a function of *M*, the moderating variable. The beta effect of M is given by *b*2. An essential value in this equation is *b*3, which is the beta effect how strongly *b*1 changes as a function of a change in M. The last part of the equation *W* represents all control factors. The following equation gives more insight in what happens with this interaction effect:

$$\hat{Y} = a + \left(b_1 + b_3 M\right) F + b_2 M + \sum_{i=4}^{k+1} b_i W_i.$$

In this equation, it becomes clear how the interaction effect influences the regression value of *F*. If the effect of *b*3 is significant, it influences the strength of the regression coefficient of *F*, which is the main independent variable. An initial test simulating the interaction effects of the moderators showed unsatisfying results on the requirement for *b*3 to be significant, since none of the interaction effects as described above were significant. In order to study possible moderating effects, another data analysis method is used.

The overall research model is a comprehensive set of variables which consists out of one independent variable, three moderating variables, three dependent variables and four controls. The current sample size ($n = 16$) forces us to work with a limited number of statistical tests. In order to perform a multiple regression analysis, a minimal sample size of $50 + 8k$ is advised, in which k represents the number of predictors (Field, 2009, p.222). Performing a multiple regression analysis to test the interaction effects in the H2-4 models is therefore not possible. Based on the *Framework for identifying moderator variables* by (Sharma et al., 1981), this study uses a subgroup analysis. The next section will go deeper into the steps which are made in order to perform this analysis.

Subgroup analysis

Sharma et al. (1981) describe how in case of insignificant results from a multiple regression analysis, subgroup analysis can be used in order to find an interaction effect. The goal of the subgroup analysis is to divide a variable which is expected to be a moderating variable into two or three subgroups. Suppose the relationship between *income* (Y) and *age* (X) is moderated by *sex* (M). One might expect that the older an employee is (X), the higher the income (Y). But perhaps this relationship is moderated by sex (M). In the subgroup analysis 'sex' is dichotomized in the subgroups men and female. If there is a substantial difference between the two subgroups, a moderating effect is assumed.

In this research, the moderating variables are dichotomized into a 'low group' and 'high group'. The low group consists out of the 50% lowest values of the moderator; the highest group consists out of the 50% highest values. As an addition, we display some extra tables and figures to gain (visual) insight in the effects of the moderators.

The procedure in the subgroup analysis is to test whether the regression equation of the research model differs across the subgroups (Sharma et al., 1981). The simple regression equation between the independent and dependent variable, excluding control variables, is given via:

$$\hat{Y} = a + b_1 F$$

In all analyses, *Modular design strategy* (MDS) is the independent variable (F). The dependent variable Y is, for example, *Cost efficiency* (COE). In the subgroup method, the regression analysis is performed over the low and high group of each moderator. Recall that these numbers are the regression coefficients of the independent variable MDS on the dependent variables and do not display an interaction effect.

To test whether the regression coefficients between two subgroups are significantly different from each other, the Chow (1960) test is used. In this test, the null hypothesis assumes the two groups to have the same regression coefficient. The method tests to what degree the sum of square residuals of the subgroups is significantly different from each other via this equation:

$$\frac{[\mathrm{RSS} - \mathrm{RSS}(1) - \mathrm{RSS}(2)]/k}{\frac{[\mathrm{RSS}(1) + \mathrm{RSS}(2)]}{(\mathrm{n} - 2k)}}$$

RSS = Sum of squared residual overall regression analysis
RSS(1) = Sum of squared residual subgroup 1
RSS(2) = Sum of squared residual subgroup 2
k = number of parameters in the equation
n = total sample size

In this explorative research, we set a 90% confidence interval. The associated critical F-value belonging to this confidence interval with F(2,14) is 3.18. When the Chow-test between two subgroups has a value of ≥ 3.18, we can assume the moderator to have a significant effect.

5 Results

5.1 Descriptive statistics

Internal construct reliability analysis is presented in table 5.1. Appendix I provides a complete overview of the multiple item constructs. According to Kline (2013) a measure is said to be reliable at a cut-off point above Cronbach's Alpha 0.70. Looking at table 5.1, all the constructs clearly surpass this criterion,
Confidential part is intentionally removed from thesis (9)

The general tendency that modularity has got a positive influence on cost efficiency is supported by the findings in the current research. In this research, a modular design strategy has been the only explanatory factor which can be linked to a positive score on cost efficiency. Design rules, which support a modular strategy, have shown to strengthen this positive relationship and therefore tends to be an important factor for successful implementation of a modular design strategy, as suggests in the current literature (Baldwin & Clark, 1997).

Findings on the effect of modularity on profitability are less clear and require deeper analyses. From the 16 cases in this sample, only five games scored higher than the cutoff point of 4 based on the 7-point Likert scale, indicating that only five games in this sample have been a financial success. Modularity by itself is not able to explain the variance in profitability. It seems that the profitability of a game is explained through a combination of factors, which is not surprising. It is notable that the data shows a negative relationship between *cost efficiency* and profitability (r = -0.27). Hence, other factors than development efficiency seem to clarify whether a game is profitable or not. An important issue here is the selling price of a game. Whereas complexity and investments have increased dramatically over the past years, selling prices have remained around the same level (Polygon, 2012). Obviously, this development cannot continue forever. Whereas modularity contributes to developing a game on a cost efficient way, new investments in for example console specific requirements bring up ever increasing costs which cannot be returned just by working efficiently. However, raising consumer prices is a problematic issue since other parties than development studios (i.e. publishers and retailers) will protest.

From these sample results, team size is strongly positively related with profitability. In this study, we transformed the original data on team size to *log* values due to the strong skewness of the data. This resulted in *Team size* to be positively related with *Profitability* (r = 0.48; $p < 0.1$). If the analysis is run over the original data for *Team size*, correlation is even stronger ($r = 0.54$; $p < 0.05$). From the current data *Marketing budget* seems to explain the positive relationship between team size and profitability since team size and marketing budget are positively correlated ($r = 0.54$; $p < 0.05$). Subsequently, marketing budget is positively related to profitability ($r = 0.68$; $p < 0.01$). Due to the limited sample size, we have not been able to perform a regression analysis for modularity and profitability, controlled for team size.

From the three performance measures, the effect of modularity on game quality is most unclear. In this study we argued that modularity will positively influence game quality since modularity helps to overcome complex interactions which may result in development mistakes. Next to that, through modularization system elements have been extensively tested which positively influences overall system quality. However, results from this study do not support this hypothesis. This is not surprising since the current literature discussing the effect of modularity on product quality and innovativeness presents diverse conclusions. Jacobs et al. (2007) found a positive relationship between modularity and product quality, whereas Antonio et al. (2007) found a negative

relationship. Modularity, especially in the software industry relies on standardized components, resolving in reuse. A higher percentage of reuse can lower product novelty (Ettlie and Kubarek, 2008). Whereas novelty and innovation are not the same concepts as game quality, they can be expected to have influence on how customers perceive quality.
Confidential part is intentionally removed from thesis (8).

6 Discussion & limitations

Managing development parameters in the video game industry such as scheduling has shown to be a challenging undertaking (Petrillo et al., 2008). Modularity is a comprehensive product design strategy which received a lot of scientific attention over the last decades. One major positive contribution of modularity on firm performance is the ability to manage product and process complexity by decoupling a system into modules and components which each have an explicit function. This study contributes to the existing modularity research by empirically assessing the concept of synergistic specificity and its impact on a modular design strategy.

In this study, we measured how the performance of the game matched with customer expectations. This performance might not only be dependent on how well the game works, but also to what degree the game is able to enthrall a customer (Kanode & Haddad, 2009). Hence, the relationship between product modularity and product performance is not a clearly defined relationship and will be dependent on the degree of modularity, in combination with other product features. These features will be different for each product type. A theory explaining the enjoyment of gamers is the so called Core Elements of the Gaming Experience (CEGE) by Calvillo-Gámez et al. (2010) (see Appendix III).The model consists out of two main factors influencing the enjoyment of users. One factor concerns the internal game-play elements and the quality of technical subsystems of the game such as sound and graphics. Whereas this part of the CEGE model might be influenced by the degree of modularization throughout the system, many other items in the model do not seem to be influenced by the degree of modularization. Even though the model is focused on user enjoyment and it is not directly comparable with the current research measure for assessing game quality, we can use this model to explain that perceived game quality of users is expected to be influenced by many other factors than *Modular design strategy*.

Limitations

There are some limitations to this study. First, due to the limited sample size results are sensitive for extreme scores and might be influenced by some specific score patterns. In this sample we studied games which are developed for the PlayStation 3 and Xbox 360 hardware. Video games developed for other consoles can have different architectures and different development implications.

Second, we have not been able to include firm and game characteristics such as development budget and game size in the research model. The included controls marketing budget, team size, marketing budget and platform investments are describe via correlation analysis but have not been included as a control in the regression analysis. Some of these variables have an influence on the performance measures, especially on profitability. Therefore results have to be carefully interpreted in terms of generalizability.

Third, the study used a key informant strategy for data collection. All participants have been carefully selected for their role in the project development and are instructed to fill in the questionnaire based on 'how things are' instead of 'how they are ought to be.' However, results should be interpreted with this possible respondent bias. Especially scores on experience level and game quality could be subjected to some degree of bias.

References

Ajila, S. A., & Wu, D. (2007). Empirical study of the effects of open source adoption on software development economics. *Journal of Systems and Software, 80*(9), 1517-1529.

Akgün, A. E., Keskin, H., Byrne, J., & Imamoglu, S. Z. (2007). Antecedents and consequences of team potency in software development projects. *Information & Management, 44*(7), 646-656.

Antonio, K. L., Yam, R. C., & Tang, E. (2007). The impacts of product modularity on competitive capabilities and performance: An empirical study. *International Journal of Production Economics, 105*(1), 1-20.

Atuahene-Gima, K. (2003). The effects of centrifugal and centripetal forces on product development speed and quality: how does problem solving matter? *Academy of Management Journal, 46*(3), 359-373.

Babbie, E. (2010). The Practice of Social Research.

Baldwin, C. Y., & Clark, K. B. (1997). Managing in an age of modularity. *Harvard Business Review, 75(5)*, 84-&.

Baldwin, C. Y., & Clark, K. B. (2000). *Design rules: The power of modularity* (Vol. 1): MIT press.

Baldwin, C. Y., & Clark, K. B. (2006). *Modularity in the design of complex engineering systems*: Springer.

Basili, V. R., Briand, L. C., & Melo, W. L. (1996). How reuse influences productivity in object-oriented systems. *Communications of the ACM, 39*(10), 104-116.

Bishop, L., Eberly, D., Whitted, T., Finch, M., & Shantz, M. (1998). Designing a PC game engine. *IEEE Computer Graphics and Applications*(1), 46-53.

Blow, J. (2004). Game development: Harder than you think. *Queue, 1*(10), 28.

Calvillo-Gámez, E. H., Cairns, P., & Cox, A. L. (2010). Assessing the core elements of the gaming experience *Evaluating user experience in games* (pp. 47-71): Springer.

Campagnolo, D., & Camuffo, A. (2010). The concept of modularity in management studies: a literature review. *International Journal of Management Reviews, 12*(3), 259-283.

Campbell, R. H., Islam, N., Raila, D., & Madany, P. (1993). Designing and implementing Choices: An object-oriented system in C++. *Communications of the ACM, 36*(9), 117-126.

Cebon, P., Hauptman, O., & Shekhar, C. (2008). Product modularity and the product life cycle: new dynamics in the interactions of product and process technologies. *International Journal of Technology Management, 42*(4), 365-386.

Chai, K. H., Wang, Q., Song, M., Halman, J. I., & Brombacher, A. C. (2012). Understanding Competencies in Platform-Based Product Development: Antecedents and Outcomes. *Journal of product innovation management, 29*(3), 452-472.

Clark, K. B. (1985). The interaction of design hierarchies and market concepts in technological evolution. *Research policy, 14*(5), 235-251.

Clark, K. B., & Fujimoto, T. (1989). The power of product integrity. *Harvard business review, 68*(6), 107-118.

de Waard, E. J., & Kramer, E.-H. (2008). Tailored task forces: Temporary organizations and modularity. *International journal of project management, 26*(5), 537-546.

Ding, F., & Jie, L. (2008). *An empirical study of flexible business process based on modularity system theory.* Paper presented at the Computing in the Global Information Technology, 2008. ICCGI'08. The Third International Multi-Conference on.

Duray, R., Ward, P. T., Milligan, G. W., & Berry, W. L. (2000). Approaches to mass customization: configurations and empirical validation. *Journal of Operations Management, 18*(6), 605-625.

Ettlie, J. E., & Kubarek, M. (2008). Design Reuse in Manufacturing and Services*. *Journal of Product Innovation Management, 25*(5), 457-472.

Fayad, M., & Schmidt, D. C. (1997). Object-oriented application frameworks. *Communications of the ACM, 40*(10), 32-38.

Fichman, R. G., & Kemerer, C. F. (2001). Incentive compatibility and systematic software reuse. *Journal of Systems and Software, 57*(1), 45-60.

Field, A. (2009). *Discovering statistics using SPSS*: Sage publications.

Folmer, E. (2007). Component Based Game Development–A Solution to Escalating Costs and Expanding Deadlines? *Component-Based Software Engineering* (pp. 66-73): Springer.

Frakes, W., & Terry, C. (1996). Software reuse: metrics and models. *ACM Computing Surveys (CSUR), 28*(2), 415-435.
Frakes, W. B., & Succi, G. (2001). An industrial study of reuse, quality, and productivity. *Journal of Systems and Software, 57*(2), 99-106.
Garud, R., & Kumaraswamy, A. (1995). Technological and organizational designs for realizing economies of substitution. *Strategic Management Journal, 16*(S1), 93-109.
Gershenson, J. K., Prasad, G. J., & Zhang, Y. (2004). Product modularity: measures and design methods. *Journal of Engineering Design, 15*(1), 33-51.
Gregory, J. (2009). *Game engine architecture*: CRC Press.
Gulley, N. (2004). In praise of tweaking: a wiki-like programming contest. *interactions, 11*(3), 18-23.
Hayes, A. F., & Matthes, J. (2009). Computational procedures for probing interactions in OLS and logistic regression: SPSS and SAS implementations. *Behavior research methods, 41*(3), 924-936.
Henderson, R. M., & Clark, K. B. (1990). Architectural innovation: The reconfiguration of existing product technologies and the failure of established firms. *Administrative science quarterly*, 9-30.
Hofer, A. P., & Halman, J. I. (2005). The potential of layout platforms for modular complex products and systems. *Journal of Engineering Design, 16*(2), 237-255.
Hofman, E. (2010). Modular and Architectural Innovation in Loosely Coupled Networks. *Unpublished doctoral dissertation). University of Twente*.
Hofman, E., Voordijk, H., & Halman, J. (2009). Matching supply networks to a modular product architecture in the house-building industry. *Building Research & Information, 37*(1), 31-42.
Jacobs, M., Vickery, S. K., & Droge, C. (2007). The effects of product modularity on competitive performance: do integration strategies mediate the relationship? *International Journal of Operations & Production Management, 27*(10), 1046-1068.
Johns, J. (2006). Video games production networks: value capture, power relations and embeddedness. *Journal of Economic Geography, 6*(2), 151-180.
Kanode, C. M., & Haddad, H. M. (2009). *Software engineering challenges in game development.* Paper presented at the Information Technology: New Generations, 2009. ITNG'09. Sixth International Conference on.
Kline, P. (2013). *Handbook of psychological testing*: Routledge.
Krueger, C. W. (1992). Software reuse. *ACM Computing Surveys (CSUR), 24*(2), 131-183.
Kude, T., & Dibbern, J. (2009). Tight versus loose organizational coupling within inter-firm networks in the enterprise software industry-the perspective of complementors. *AMCIS 2009 Proceedings*, 666.
Kumar, N., Stern, L. W., & Anderson, J. C. (1993). Conducting interorganizational research using key informants. *Academy of management journal, 36*(6), 1633-1651.
Langlois, R. N. (2002). Modularity in technology and organization. *Journal of economic behavior & organization, 49*(1), 19-37.
Langlois, R. N., & Robertson, P. L. (1992). Networks and innovation in a modular system: Lessons from the microcomputer and stereo component industries. *Research Policy, 21*(4), 297-313.
Lau, A. K., Yam, R., & Tang, E. (2011). The impact of product modularity on new product performance: Mediation by product innovativeness. *Journal of Product Innovation Management, 28*(2), 270-284.
Lau, A. K., Yam, R. C., & Tang, E. P. (2007). Supply chain product co-development, product modularity and product performance: empirical evidence from Hong Kong manufacturers. *Industrial Management & Data Systems, 107*(7), 1036-1065.
Lim, W. C. (1994). Effects of reuse on quality, productivity, and economics. *Software, IEEE, 11*(5), 23-30.
MacKenzie, S. B., Podsakoff, P. M., & Podsakoff, N. P. (2011). Construct measurement and validation procedures in MIS and behavioral research: Integrating new and existing techniques. *MIS quarterly, 35*(2), 293-334.
Mikkola, J. H., & Gassmann, O. (2003). Managing modularity of product architectures: toward an integrated theory. *Engineering Management, IEEE Transactions on, 50*(2), 204-218.
Mohagheghi, P., & Conradi, R. (2008). An empirical investigation of software reuse benefits in a large telecom product. *ACM Transactions on Software Engineering and Methodology (TOSEM), 17*(3), 13.
Mohagheghi, P., Conradi, R., Killi, O. M., & Schwarz, H. (2004). *An empirical study of software reuse vs. defect-density and stability.* Paper presented at the Software Engineering, 2004. ICSE 2004. Proceedings. 26th International Conference on.
Petrillo, F., Pimenta, M., Trindade, F., & Dietrich, C. (2008). *Houston, we have a problem...: a survey of actual problems in computer games development.* Paper presented at the Proceedings of the 2008 ACM symposium on Applied computing.
Polygon. (2012). The state of games: State of AAA. from http://www.polygon.com/2012/10/1/3439738/the-state-of-games-state-of-aaa

Ravichandran, T., & Rothenberger, M. A. (2003). Software reuse strategies and component markets. *Communications of the ACM, 46*(8), 109-114.
Rothenberger, M., Dooley, K. J., Kulkarni, U. R., & Nada, N. (2003). Strategies for software reuse: A principal component analysis of reuse practices. *Software Engineering, IEEE Transactions on, 29*(9), 825-837.
Sahay, A., & Riley, D. (2003). The role of resource access, market considerations, and the nature of innovation in pursuit of standards in the new product development process. *Journal of Product Innovation Management, 20*(5), 338-355.
Salvador, F. (2007). Toward a product system modularity construct: literature review and reconceptualization. *Engineering Management, IEEE Transactions on, 54*(2), 219-240.
Sametinger, J. (1997). *Software engineering with reusable components*: Springer Science & Business Media.
Sanchez, R., & Mahoney, J. T. (1996). Modularity, flexibility, and knowledge management in product and organization design. *Strategic management journal, 17*(S2), 63-76.
Schach, S. R. (2002). *Object-oriented and classical software engineering* (Vol. 6): McGraw-Hill New York.
Schilling, M. A. (2000). Toward a general modular systems theory and its application to interfirm product modularity. *Academy of management review, 25*(2), 312-334.
Schilling, M. A., & Steensma, H. K. (2001). The use of modular organizational forms: an industry-level analysis. *Academy of Management Journal, 44*(6), 1149-1168.
Sharma, S., Durand, R. M., & Gur-Arie, O. (1981). Identification and analysis of moderator variables. *Journal of marketing research*, 291-300.
Simon, H. A. (1962). The Architecture of Complexity. *Proceedings of the American Philosophical Society, 106*(6), 467-482. doi: 10.2307/985254
Song, X. M., & Parry, M. E. (1997). A cross-national comparative study of new product development processes: Japan and the United States. *The Journal of Marketing*, 1-18.
Sosa, M. E., Eppinger, S. D., & Rowles, C. M. (2004). The misalignment of product architecture and organizational structure in complex product development. *Management science, 50*(12), 1674-1689.
Staudenmayer, N., Tripsas, M., & Tucci, C. L. (2005). Interfirm Modularity and Its Implications for Product Development*. *Journal of Product Innovation Management, 22*(4), 303-321.
Susarla, A., Barua, A., & Whinston, A. B. (2010). Multitask agency, modular architecture, and task disaggregation in SaaS. *Journal of Management Information Systems, 26*(4), 87-118.
Szyperski, C., Bosch, J., & Weck, W. (1999). *Component-oriented programming.* Paper presented at the Object-oriented technology ecoop'99 workshop reader.
Tatikonda, M. V. (1999). An empirical study of platform and derivative product development projects. *Journal of Product Innovation Management, 16*(1), 3-26.
Tiwana, A. (2008). Does technological modularity substitute for control? A study of alliance performance in software outsourcing. *Strategic Management Journal, 29*(7), 769-780.
Tomer, A., Goldin, L., Kuflik, T., Kimchi, E., & Schach, S. R. (2004). Evaluating software reuse alternatives: a model and its application to an industrial case study. *Software Engineering, IEEE Transactions on, 30*(9), 601-612.
Trenholme, D., & Smith, S. P. (2008). Computer game engines for developing first-person virtual environments. *Virtual reality, 12*(3), 181-187.
Ulrich, K. (1995). The role of product architecture in the manufacturing firm. *Research policy, 24*(3), 419-440.
Van Assche, A. (2008). Modularity and the organization of international production. *Japan and the World Economy, 20*(3), 353-368.
Wäljas, M., Segerståhl, K., Väänänen-Vainio-Mattila, K., & Oinas-Kukkonen, H. (2010). *Cross-platform service user experience: a field study and an initial framework.* Paper presented at the Proceedings of the 12th international conference on Human computer interaction with mobile devices and services.
Worren, N., Moore, K., & Cardona, P. (2002). Modularity, strategic flexibility, and firm performance: a study of the home appliance industry. *Strategic management journal, 23*(12), 1123-1140.
Yoo, Y., Boland Jr, R. J., Lyytinen, K., & Majchrzak, A. (2012). Organizing for innovation in the digitized world. *Organization Science, 23*(5), 1398-1408.
Zackariasson, P., Styhre, A., & Wilson, T. L. (2006). Phronesis and creativity: Knowledge work in video game development. *Creativity and Innovation Management, 15*(4), 419-429.

Appendix I - Variable construct

Confidential part is intentionally removed from thesis (10)

The coming figures support the findings as described in section 5.3. The figures are clustered as follows:

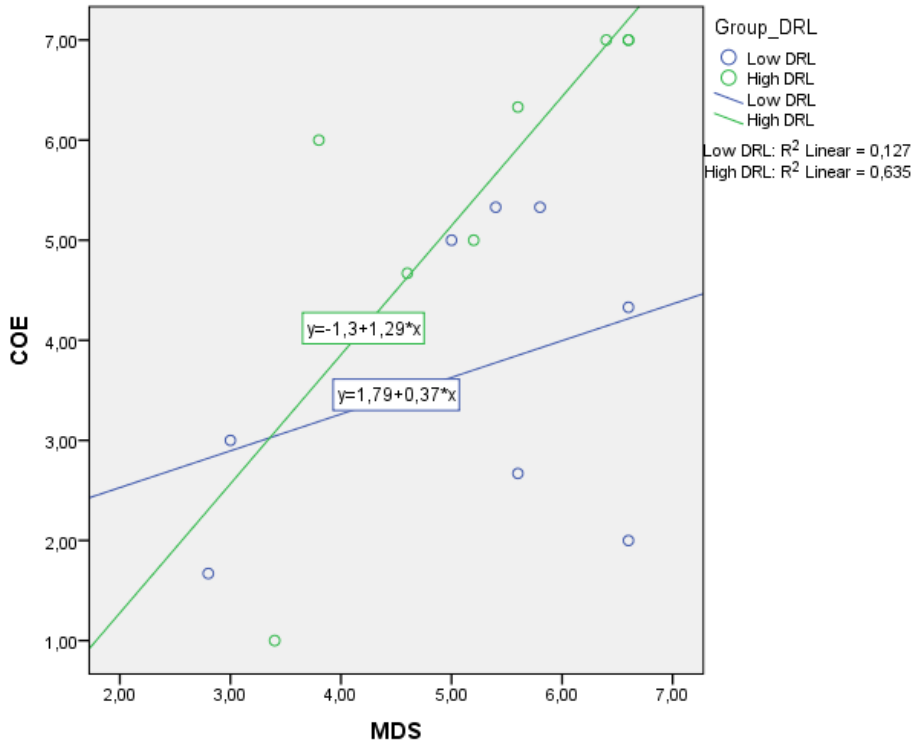
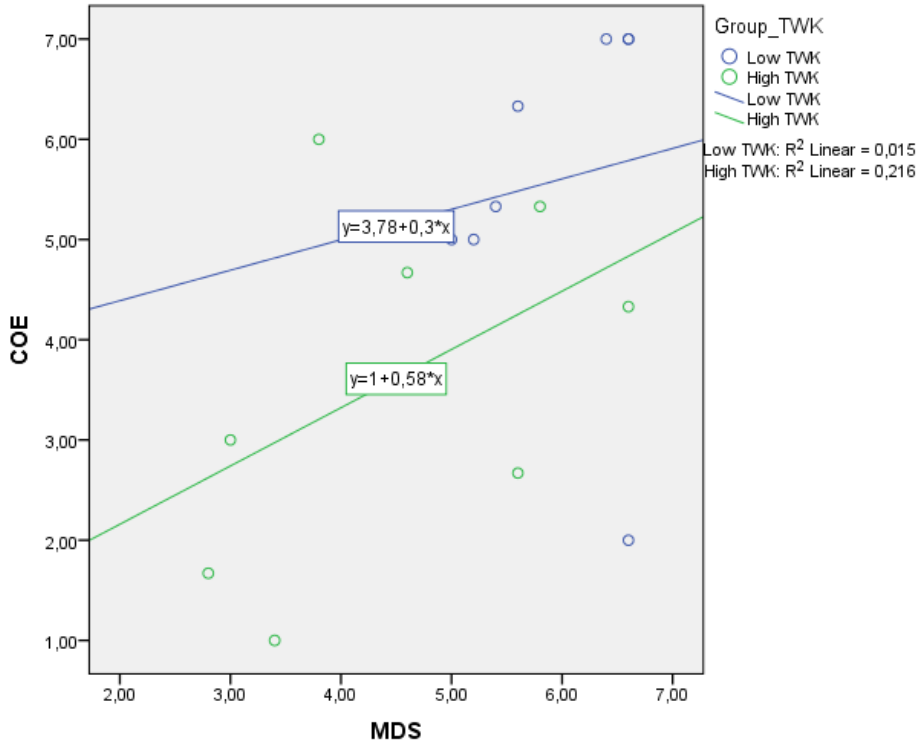
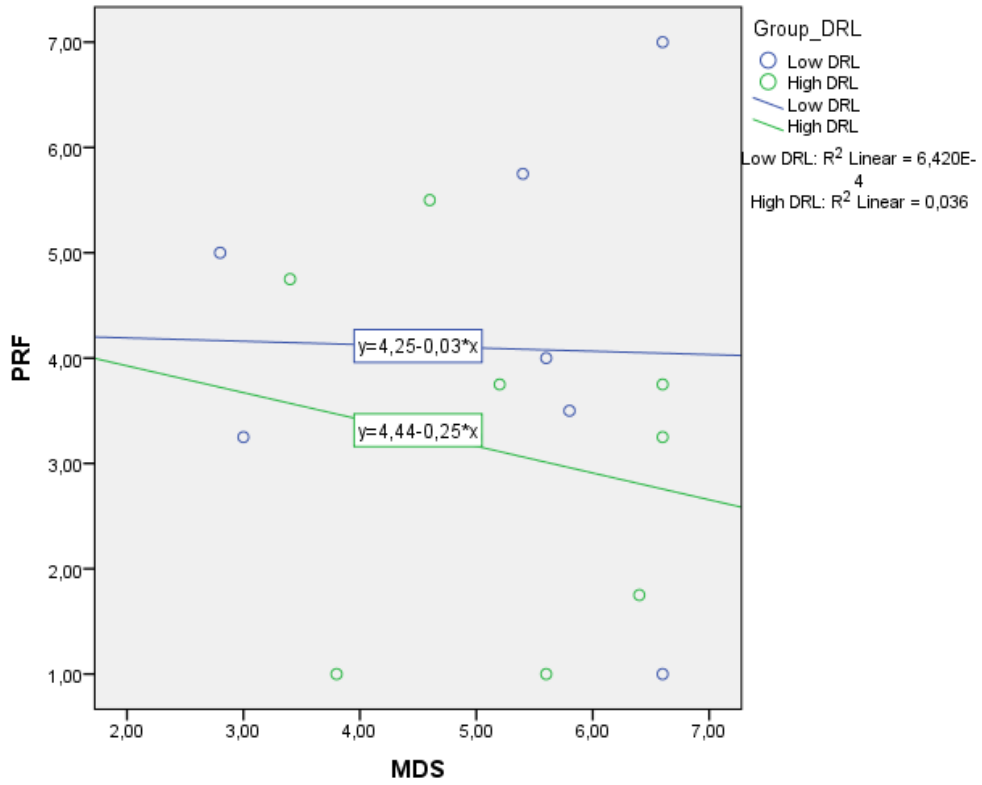
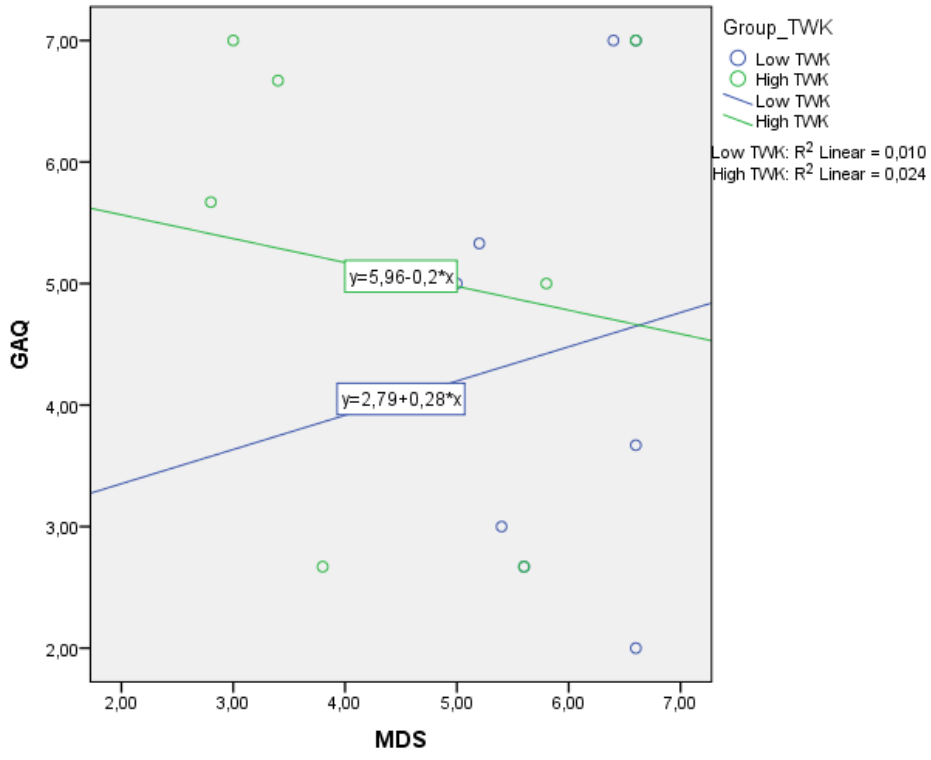


Figure A1: *Design rules*



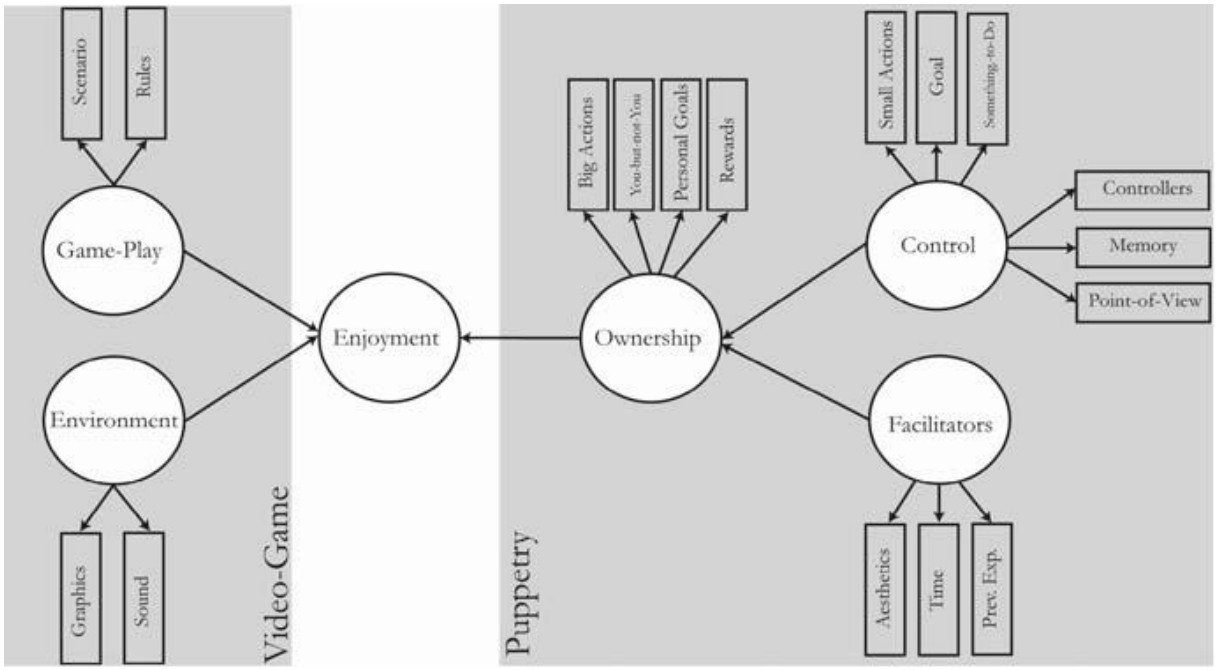




Appendix III – The CEGE model

The model of Gamez et al. (2010) can help explain how gamers evaluate video games and hence, game success. This model is one of probably many factors clarifying the limited results in the current study to test the effect of a design strategy on game quality.

.


Scenario
Rules
Game-Play
Environment
Graphics
Sound
Video-Game
Enjoyment
Ownership
Big Actions
You-but-not-You
Personal Goals
Rewards
Small Actions
Goal
Something-to-Do
Control
Controllers
Memory
Point-of-View
Facilitators
Aesthetics
Time
Prev. Exp.
Puppetry