HOW A SIMPLE CARD GAME INFLUENCES DESIGN REASONING: A REFLECTIVE METHOD

Master Research Thesis

Key words
Software Design, Design Discourse, Design Reasoning, Design Card Game, Decision Making, Reflective Practice, Reflective Thinking

Courtney Schriek
C.J.Schriek@student.uu.nl
29-6-2016

ADDITIONAL INFORMATION

Author	Courtney J. Schriek
E-mail	C.J.Schriek@students.uu.nl
Student ID	3590569
Project Supervisor	Dr. ir. Jan Martijn van der Werf J.M.E.M.vanderWerf@uu.nl Department of Information and Computing Sciences Utrecht University, Utrecht, The Netherlands
Second supervisor	Dr. Floris J. Bex F.J.Bex@uu.nl Department of Information and Computing Sciences Utrecht University, Utrecht, The Netherlands
External Supervisor	Dr. Antony Tang atang@swin.edu.au Department of Computer Science and Software Engineering Swinburne University of Technology, Melbourne, Australia

A Master Thesis Project submitted in fulfilment for the degree of
Master Science in Business Informatics

Faculty of Science
Department of Information and Computing Sciences
Princetonplein 5, De Uithof
3584 CC Utrecht

ABSTRACT

In this Master thesis a reflective card game is developed to help designers improve their design reasoning. Software design is a highly complex activity in which many factors have to be taken into consideration; stakeholders, the teams abilities, and available technologies are just a few. The decisions designers make during software design are also important for the software development as a whole. If decisions made at this stage lead to problems later on development work will need to be fixed, costing time and money. The quality of design decisions is based on the reasoning that underlies the decision making process, and so in order to improve design decisions more thorough reasoning needs to be applied.

For the development of a reflective method to help designers during their design discourse several experiments are run. The experiments are to test if the reflective method has a significant influence on design reasoning. We focus on professional and student designers to represent experience and inexperience, and have them fulfil a software architecture assignment. The student and professional groups are then divided into test and control groups, with the test groups using the cards in order to see its effect on design reasoning. The differences between the groups are measured using qualitative analysis, with the student results also including quantitative analysis to find any significant differences.

The card game is based on rational decision making and reflective thinking. Rational decision making is careful evaluation of the situation before making a decision, and this is different from naturalistic decision making, which is intuitive. In many cases naturalistic decision making works well, but in an unstructured and complex situation such as software design this kind of decision making is not sufficient. Designers need to use rational decision making which means thorough reasoning, this involves deliberate thinking and discussing of alternative solutions and finding the optimal one. Designers can be encouraged to use rational decision making by use of reflective questions in the form of reasoning techniques like assumption or risk analysis. With reflective thinking designers are prompted to re-evaluate their decisions and think critically whether the right choice has been made. These reasoning techniques have been represented in a card form and are used to trigger the designers into reflection, and ultimately rational decision making.

Our results show that for the student groups the card game has a significant influence when it comes to improving thorough reasoning, making the reflective method a success. But there are also some discrepancies regarding the identification of design problems and constraints which are interesting subjects for further research. For the professional groups, our results are inconclusive but show many of the same patterns as the student groups, though to a lesser degree.

ACKNOWLEDGEMENTS

Writing this thesis and conducting all the necessary experiments has been a long process, involving such tasks as typing out the audio files of the design sessions (do not recommend), a three month trip to Australia, Melbourne (do recommend), and learning how to use Nvivo (I'm pretty good now).

In 8 months this research has grown from an idea created by overlooking the word 'metaphor', into a full-fledged card system with a lot of potential for further use and research in the design reasoning field. It was hard work at every stage and I couldn't have done it alone.

My supervisors, Jan Martijn van der Werf, Floris Bex and Antony Tang have been of great help to me. From developing my idea and the card game, to providing excellent feedback, and keeping me on track to realistically finish this thesis. Your enthusiasm for my card game idea and its application has been an inspiration.

Thanks also go to my fellow Master student Rizkiyanto, who together with me has had the arduous task of transcribing, and then coding the student transcripts in order to have reliability in coding. Syncing our work while in different countries and time zones has been no easy task, but we got through.

And last but not least, all thanks go to my family and friends who, even when they didn't really understand what I was doing, were at least willing to sit still and listen to my ramblings.

Have fun reading!

Courtney

CONTENT

1. INTRODUCTION

The success of software systems and IT in the world has become an indisputable fact. Though software engineering is a relatively young field it is in high demand as no industry is unaffected by its progress. Increasingly, the world is becoming digitized and the importance of quality software systems increases with it, to the point that when important software systems fail it becomes world news.

Failure of software systems can have many causes; incomplete requirements, ineffective testing, or design flaws. As software engineering is such a new field, the software development process is not a standard process and development approaches can differ due to factors such as domain, product complexities, experience level of developers and the mind-set of the company. But all development processes share the same core activities; eliciting requirements, designing the architecture, building the software system, and eventually deploying and maintaining the system. In this thesis the focus lies on one of these core activities; software design.

Software design is a highly complex activity as the business requirements and available technologies that need to be considered can change rapidly. It also concerns 'wicked problems' where the problem and solution are so intertwined that understanding the problem depends on how the designer wants to solve it. It is not surprising then that the design decisions made at this stage in the development process can have great influence on the eventual product (Rittel & Webber, 1973).

How these design decisions are made depends on the reasoning of the designers, i.e. their argumentation. In software design this reasoning process is important because it ultimately influences the quality of the design decisions and consequently the design. It has been shown that improving the design reasoning process of designers will produce a higher quality design, especially when the designers are inexperienced (Tang, Tran, Han, & Van Vliet, 2008). But people are not by nature logical thinkers and tend to be naturally more subjective than objective, making decisions based on instinct with what is known as naturalistic decision making (Klein, 2008). This can cause flawed reasoning to be applied as the basis for decisions, which can be biased due to past experiences and preferences in design solutions. These flawed designs will need to be changed when problems occur in later stages of the software development, taking up money and time to do so.

This is not an unknown problem in software design and there has been extensive research done in the field of design rationale (DR), which looks at argumentation-based design rationale and capturing design decisions. Many tools have been developed for this purpose such as IBIS (Kunz & Rittel, 1970), DRL (Lee, 1991), SOAD (Zimmermann, Koehler, Leymann, Polley, & Schuster, 2009), AREL (Tang, Jin, & Han, 2007) and others. Design rationale is the "body of information that explicitly records the design activity and the reasons for making choices (and reasons for not making some choices)" (Chandrasekaran, Goel, & Iwasaki, 1993, p.48). The tools designed to capture design rationale focus on documenting the design options, the choices that were made and the reasons behind these choices (Chandrasekaran et al., 1993). But while the software industry is aware of the need for DR in their software design, and many tools and methodologies are available, their use does not guarantee a quality design. The problem with these tools is that they provide designers with a way to justify their design decisions, and communicate these to other designers and/or stakeholders, but does not focus on the internal thought process which centres on the ability to reason (Tang, 2011). In order to understand these problems, designers need to move from naturalistic decision making to rational decision making, which incorporates logical reasoning.

Design reasoning is the process of reasoning which occurs within the software design process during the design discourse about design decisions. It has been shown that simply stating the design options and issues explicitly during design discourse can already have a positive effect on design by improving the reasoning process (Tang et al., 2008). And further experiments which aim at prompting designers with reasoning techniques have had similar results (Razavian, Tang, Capilla, & Lago, 2016; Tang & van Vliet, 2009).

The reasoning techniques work through encouraging reflective thinking, meaning that the person consciously evaluates their ideas and decisions (Schön, 1983). This reflection helps designers to reconsider their previous design decisions and arguments, and improve their logical reasoning. The reflection is based on critical thinking which comes from a place of uncertainty, and involves questioning whether the decisions taken were right (Larrivee, 2000).

The aim of this research is to support designers to move their decision making process from naturalistic decision making to rational decision making. Currently available tools do not explicitly focus on improving the reasoning process, but rather document the rationale behind designs for better communication between the initial designers and others which were not part of the initial design. As such, this thesis will aim to develop a system using physical cards, which has as its primary focus to support the reasoning process itself through reflective thinking and guide designers' internal thinking to avoid design reasoning failures.

The research question which this research thesis centres around is:

MRQ: *How can the use of a reflective method, prompting reasoning techniques during the decision process with the use of reflection, influence and support design reasoning in software architecture.*

In order to answer this question there are several sub-questions which need to be researched.

SQ 1: *What is the design discourse of designers within software architecture?*

As this research looks at influencing software design, we will first look at how software architecture is designed by designers. This focusses mainly on the discussion between designers as they come to design decisions, and why this process remains a complicated one.

SQ 2: *How are design decisions made with naturalistic decision making?*

In software design the end result, software architecture, is made up of the design decisions the designers discuss. In order to understand how to improve design reasoning we must first look into what the normal process of decision making is for designers.

SQ 3: *How and when do people use logical reasoning for making design decisions?*

One of the main elements involved in making design decisions is the reasoning to justify those decisions. For this question we look at what the logical reasoning process is and what makes it different from the process people naturally use.

SQ 4: *How can inferior design reasoning be improved using reflection?*

When the process of decision making, and differences between naturalistic and logical reasoning are understood, research will focus on improving design reasoning in designers. We look at ways that the design reasoning process can be improved with the use of reflective thinking.

In the following chapters the theoretical framework behind the research question will be addressed in the literature study. A conceptual model will then be derived from the theory hypothesizing how the reflective method, in this case a card system, will influence design reasoning. The card system will then be tested focussing on both a student and professional case study involving experiments set up to see if the cards have the intended effect. The results of these two groups will be analysed and then compared to come to a greater understanding of the influence of reasoning on experienced and inexperienced designers. The thesis ends with a discussion, concluding all results and commenting on further research.

2. Research Method

As mentioned in the introduction, the card system that will be developed using both student experiments and professional experiments in a case study to evaluate the card system. This research will use a combination of quantitative and qualitative methods focusing on differences between the test and control groups.

There are several instances in which a research method is employed:

- Literature Study
- Student Experiments
- Professional Experiments

These will be explained in detail below. The design case setup used in the experiment in this research is based on the Irvine experiment. The design case used a traffic simulation assignment for designers to design (Appendix A). The subsequent experiments use the same assignment.

The research is designed according to the principles of design science and employs a design research cycle (Figure 1). The research cycle begins with a problem, which is analysed in order to come to a solution. This solution is then instantiated and the results are evaluated to conclude if the problem is solved. If not, the cycle is repeated in order to improve the solution.

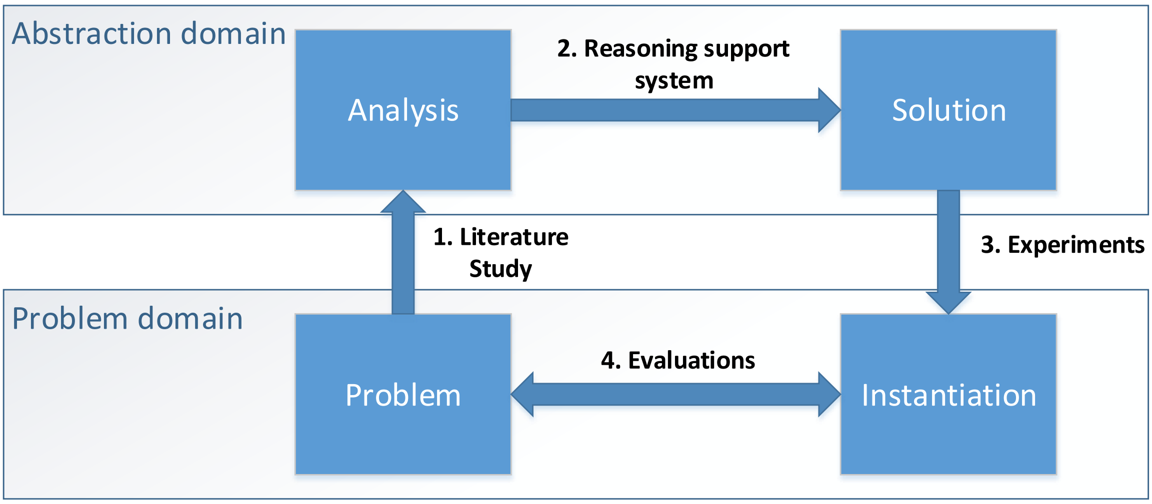


Figure 1 - Design research cycle

The research is meant to solve a problem, in this case the prevalence of design reasoning failures in software architecture, when encountering a problem which the designer is unfamiliar with. An example is anchoring; committing to the first solution that comes to mind, without regard of whether this argumentation comes to a sound conclusion.

The research cycle starts with this problem, for which the research question is formulated. Before the problem can be solved it must first be analysed and understood in its context. This is the first step, the literature study, which focuses on answering SQ 1 and 2.

A plan is then made, which is to answer SQ 3, which is grounded in the literature study, but is elaborated upon through experiments in which the solution is instantiated and tested. This is also to answer SQ 4.

This is performed in several iterations in which the solution is tested in both student experiments and professional experiments. Evaluations are made to ensure that the outcome of the experiments solves the problem, and adjustments are made accordingly.

2. 1. Literature Study

The literature research method used for this research is mainly that of Snowballing, in which articles related to the topic of research are found by reading articles and relevant citations. For this literature research Google Scholar has been a prevalent tool. The literature study's main focus is on answering the sub-questions needed to answer the main research question.

The topic researched for this thesis is that of design reasoning and reflective thinking. A central experiment throughout the thesis and prevalent in the literature is the Irvine experiment which was performed at the University of California, Irvine. In the form of an NSF Sponsored Internal Workshop (UCI, 2010), three design teams were asked to design a traffic simulator and the development was recorded and transcribed. Participants of the workshop then analysed and submitted papers on the subject which was published in a book (Petre & van der Hoek, 2013). This book was a main source of articles on which snowballing was applied. Other key articles were "Software designers, are you biased?" (Tang, 2011) and "Design reasoning improves software design quality" (Tang et al., 2008). These articles served as starting points in finding articles which focused on design reasoning as a subject, and the underlying issues on which it builds. A central experiment on which much of the current experiments are based is that of "In Two Minds: How Reflections Influence Software Design Thinking" (Razavian et al., 2016). This article added reflective thinking as a central component to making design reasoning work, and was a source of inspiration for the reflective card game developed in this thesis. There are also several sources which are used to represent key concepts on which the theory presented in this thesis is build. These are "Thinking, fast and slow" (Kahneman, 2011), which explains the thought processes used in decision making, and "Distinguishing the reflective, algorithmic, and autonomous minds: Is it time for a tri-process theory" (Stanovich, 2009), which takes a deeper look into what makes rationality possible.

The inclusion criteria for key articles were that they focused on design reasoning or reflective thinking and how to support and/or improve this process. Articles related to purely documenting design rationale or other aspects of design were seen as outside of the scope of the research and excluded. Articles were also found searching with several key words on Google Scholar; software design, design discourse, reflective thinking and design reasoning.

2. 2. Case Studies

The theory described in this thesis is that reasoning techniques, through the means of reflection, have a positive influence on design reasoning with software designers. This theory is tested using multiple case studies; one focusing on inexperienced designers, i.e. students, and one on experienced designers, i.e. professionals. These both involve experiments where test and control groups are compared to one another. And finally, the results from both the inexperienced and experienced experiments will be analysed and compared. The experiments use quantitative and qualitative measures in their analysis.

In the context of the thesis the experiments are meant to develop answers for SQ 4, which will ultimately lead to answering the main research question. The experiment structure can be seen in Figure 2.

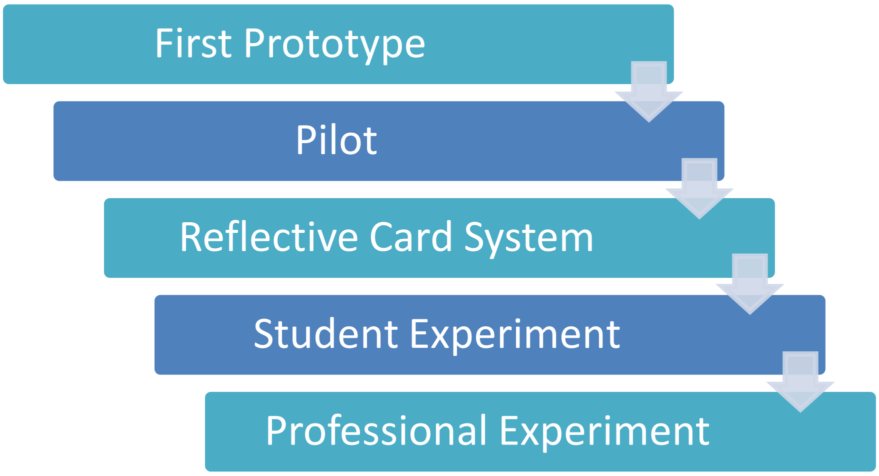


Figure 2 – Experiment Structure

2.2.1 *Subjects*

The subjects for the case study involving inexperienced designers are 12 teams of Master students from the University of Utrecht, following a Software Architecture course. These were split into 6 control and 6 test teams, with most having three designers working together, two teams with two designers, and one team with four designers. Teams were then ranked according to test scores for a previous assignment from which they were randomly selected for the test or control groups to ensure an equal amount of skill.

The subjects for the case study involving experienced designers from various companies, working in groups of two. The 3 control groups were taken from the Irvine experiment (UCI, 2010), and 2 test groups were selected via the Swinburne University of Technology. These involved 2 IT companies situated in Australia.

2.2.2 *Pilot Procedures and Improvements*

Before the student experiment began, a pilot study was run to test the card game and refine it by finding any major flaws or misunderstandings. The pilot study was performed by two Master students whom had already completed the Software Architecture course. The results of the pilot resulted in several important changes being made to the card game. Firstly, the cards were simplified and reduced in number from 15 to 7, to simplify card play as the number of cards made it difficult to choose from them. Card play also tapered off towards the end of the design session, to enforce card play three reflective periods were added and evenly spread throughout the session. In these periods it was mandatory to play cards. Lastly, the card rules were simplified to remove restrictions on fluid discussion.

2.2.3 *Experiments Setup*

The assignment used for both experiments is the same as used in the Irvine experiment (Appendix A). This assignment is well known in the field of design reasoning as participants to the workshop analysed the transcripts made and submitted papers on the subject (Petre & van der Hoek, 2013). The assignment is to design a traffic simulator and designers are provided a problem description, requirements, and a description of the desired outcomes. The design session was two hours long. For the student groups the assignment was slightly adjusted to include several viewpoints as end products in order to conform to the course material (Bass, Clements, & Kazman, 2012), of which there are two different version; the pilot instructions (Appendix B), and the final student instructions (Appendix C). The pilot and student groups were also asked to document the cards that they played (Appendix D). After the design session both student and professional test groups were asked to answer a short questionnaire on their experience with the cards (Appendix E). The student sessions were recorded with audio only and transcribed by two researchers. The professional sessions used the three original transcripts from the Irvine experiment as control groups, and video recorded two other sessions involving professional teams. These were transcribed by the author.

A deductive analysis approach is used on the transcripts using a predetermined framework based on the conceptual model. The control and test group are coded with the same framework, after which the groups are compared for their use of the design reasoning techniques.

2. 3. Quantitative and Qualitative Analysis Procedures

The goal of this research is to build on the theory that applying reasoning techniques can influence the design reasoning process of designers. Previous studies have been made on this topic (Razavian et al., 2016; Tang et al., 2008; Tang & van Vliet, 2009; Van Heesch, Avgeriou, & Tang, 2013) and in this research the techniques are applied in a different context, that of a reflective card game. The chosen research method is a combination of quantitative and qualitative measures.

2.3.1 Quantitative Analysis

For the quantitative measures we look at significant differences in use of reasoning techniques by test and control groups. To test for significance both an independent T-test and the Mann Whitney test are used (Field, 2013). The independent T-test compares the means of the test and control groups for significant differences. With a significance level below 0.05 there is a significant difference. The groups are independent as the control and test groups do not share the same participants. There are 6 assumptions which the data must adhere to before the independent T-test can be used (Laerd Statistics, 2016).

Assumption 1: The dependent variable is measured on a continuous scale (either interval or ratio level).

Assumption 2: The independent variable consists of two categorical, independent groups

Assumption 3: There is independence of observations, meaning that there is no relation between the participants of one group and another.

Assumption 4: No significant outliers are present. Any outlier can reduce the validity of the independent T-test and should be reported. Tests should be performed with the outlier and without the outlier to see if the outlier chances the significance of the results. Outliers need not necessarily be removed as these can be natural to the data. If the outlier is a result of erroneous data it should be removed. Boxplots can be used to find outliers in the data.

Assumption 5: The dependent variable is normally distributed for each group of the independent variable. To test for normality the Shapiro-Wilk test of normality can be used, the significance level should be above 0.05.

Assumption 6: There is homogeneity of variances. To test for equality of variances we look at the Levene's test, with a significance level above 0.05 the result can be used.

The bigger the samples are, the less a violation of an assumption matters, but as we are working with small samples confirming these assumptions is an important step in assuring the results are accurate. When the assumption of normal distribution is violated there are still other tests which could be performed, non-parametric tests. In our case we would use the Mann Whitney test.

Non-parametric tests can be used when the assumption of normal distribution is used because it ranks the data, making a normal distribution unnecessary. It is not as powerful as the independent T-test and should therefor only be used when assumptions have been violated. With a significance level below 0.05 differences between groups are significant.

2.3.2 Qualitative Analysis

The qualitative measures are in the form of discourse analysis. The discourse in this context refers to the communication between designers in a design session where ideas are exchanged. The discourse analysis is performed on the transcripts made of various design sessions.

Discourse analysis is an approach to interpreting a text rather than a specific analysis. It has as its underlying assumption that the meaning of the language being used is assigned by the participants in the communication and the situation where they speak. It is therefore necessary to understand the intertwined relation between text and context in order to interpret the text (Cheek, 2004).

Context in this research is in the form of the architectural assignment, the designers' skill level, and the use of the supportive system. The transcripts are coded for analyses with the use of a predetermined coding scheme based on the theory regarding influencing the logical reasoning process. Identifying the various techniques used by the designers, and their reasoning process, is a difficult process. It requires a context-based interpretation of the text as the reasoning techniques used in this research are natural to the reasoning process and not distinct activities in the mind of designers.

There are several limitations due to the nature of the research and the qualitative measures used. Discourse analysis is itself subjective and reliant on the view of the researcher, and so the process of the research must be transparent (Horsburgh, 2003). The steps taken during the experiments will be explained here and in their respective chapters.

The coding scheme is based on theory and the results show the credibility of this theory in another context, the supportive system. The interpretation of the text is checked against those of another researcher using the same coding scheme for inter-reliability, using Cohen's kappa (Cohen, 1968).

The generalizability of the research is also of a qualitative nature. The participants are selected for their situational representativeness, i.e. the information they can provide in the area of software architecture. The results are generalizable to the experiences of others in similar situations of designing (Horsburgh, 2003).

3. Literature Research

The aim of this literature research is to answer the research questions about design reasoning and to develop a reflective method to support designers.

In the following sections we first delve into software design itself, what it is and why it is a difficult stage in software development, and what considerations go into making design decisions. We then go further into how design decisions are usually made, and what drives naturalistic decision making. This is followed by an explanation of design reasoning, what are the causes of its failure, and how can it be triggered using reflection. Based on this research the reflective card system and conceptual model are developed.

3. 1. Software Design

In order to develop a software product there are several different methods which can be used, but all focus on the core activities of eliciting requirements, designing the software, constructing the software, deploying it and eventually maintaining the software product (Figure 3). The part which this research will focus on is the design phase of which the end-product is a software architecture design from which a software product can be constructed.

Figure 3 - Software Development Process

Software architecture is defined as the "fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution" (ISO/IEC, 2014). In this phase the requirements of the software product are translated into a feasible architecture structure, on the basis of which the code for the product will later be written. This process is most often an iterative one in which the architecture is continuously refined to satisfy the requirements and the stakeholders.

The process of software design is based on making design decisions. An architectural design decision is "a description of the set of architectural additions, subtractions and modifications to the software architecture, the rationale, and the design rules, design constraints and additional requirements that (partially) realize one or more requirements on a given architecture" (Jansen & Bosch, 2005). A collective of these design decisions makes up the software architecture.

The design decisions are the outcome of a design discourse where designers collaborate and discuss the design. Problems or issues are discovered about a certain topic and these are disputed as designers take on opposing positions. To defend their own positions and convince the opposing designer arguments are made, and questions are asked concerning the requirements. Answers can be questioned in turn or turned into issues. Through this discourse the problem and solution become more structured and design decisions are made (Kunz & Rittel, 1970).

As becomes clear, design is mainly a problem-solving activity. It is also highly complex because requirements can change or characteristics of the software remain unknown. These are known as ‘wicked’ problems, meaning that they are inherently ill-defined and have no standard solutions. Solving these problems is difficult because understanding the problem depends on how the designer chooses to solve it. The process of formulating the problem and solution are intertwined. This ‘wicked’ problem stands in opposition to ‘tame’ problems such as mathematical ones where all information necessary for understanding and solving the problem can be given, and it is clear when the problem has been solved (Rittel & Webber, 1973). With software design there is no standard solution and each can be equally valid, it depends entirely on the designer and the strategies they use.

Designers can employ different design strategies, which can be separated into problem or solution focused design. The difference between these two strategies is how it influences the information seeking behaviour of designers. A solution-oriented approach is shown to be limited in how much information is sought out, while the problem-oriented approach makes designers more willing to seek and process information. Their results show more creativity in terms of the designs they develop (Restrepo & Christiaans, 2004).

This corresponds with the problem-solution co-evolution model of software design. Problem-solving here is a co-evolution of the problem space and the solution space with information being exchanged between the two which influences each other. This co-evolution involves such actions as a solution idea being defined, which then redefines the problem, and this in turn means modifying the new solution and so on. The iterative synergy between these two spaces entails the act of problem-solving (Dorst & Cross, 2001). Exploring the problem space involves analysing problems and formulating the design problems, with the focus on finding all problems. Exploring the solution space involves generating solution options, evaluating solutions and finally selecting a solution (Tang, Aleti, Burge, & Van Vliet, 2010). Comparable to this is the Twin Peaks model which shows the relation between requirements and design issues. These too are interconnected and any change made in one influences the other. The specification of the design moves through these two stages in an iterative manner, going from more general to detailed design. This process represents how designers realistically work in the software industry (Nuseibeh, 2001).

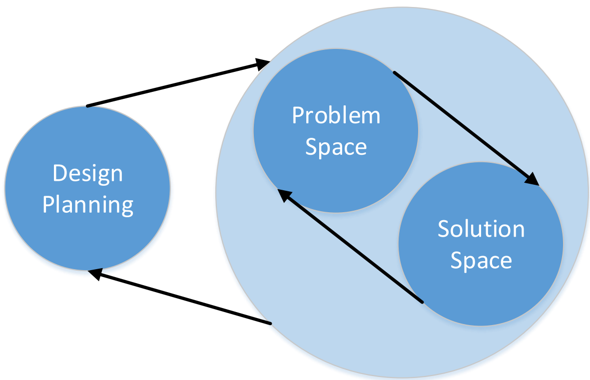


Figure 4 - Design Decision Making Model

But for the process of rational design decision making there is still another important activity that designers do, the design planning. It involves scoping the requirements, identifying key design issues and prioritizing design problems to develop a high-level design approach (Tang et al., 2010). When the design problem is well structured and understood designers will use a more rational approach to decision making then they would otherwise. These activities come together in the design decision making model (Figure 4).

To ensure a good design there are several methods and processes for reviewing or assisting on software architecture which are focused on improving the design outcome, such as ATAM (Bass et al., 2012), SAAM (Kazman, Abowd, Bass, & Clements, 1996) and TARA (Woods, 2012). These methods help to improve the eventual design but do not resolve the cause of flawed designs, which is that during the design discourse the designers apply cognitive reasoning to support their design decisions, but do not explicitly acknowledge them as part of the process. Such problems can be caused by flawed reasoning principles being applied as the basis for decisions, which can be biased due to past experiences and preferences in design solutions. Such biases are not as detrimental when the designer is experienced, but the more complex and/or new a problem is, combined with the designers lack of expertise, the more likely flawed reasoning will be used to create a flawed design (Tang & van Vliet, 2009). This leads to flawed designs which need to be changed when problems occur in later stages of the software development, taking up money and time to do so.

In order to understand how flawed reasoning can occur we need to look into how design decisions are usually made, using naturalistic decision making and rational decision making as lenses.

3. 2. Naturalistic Decision Making

Design reasoning and careful consideration of design options is not as easy and prevalent as one might think. People are not by nature logical thinkers and tend to naturally make decisions based on instinct. This kind of decision making process is known as naturalistic decision making. It stands in opposition to rational decision making which is a systematic approach using logical reasoning to look at the issues, generate options, and after careful consideration make a decision. The naturalistic decision making process used by most people is made with intuition based on experience (Zannier, Chiasson, & Maurer, 2007). Their experience provides them with patterns of what decisions are valid for particular situations. Their decision making process is then essentially recognizing a situation as being one of the patterns and responding accordingly. This allows for making fast decisions with a minimum amount of cognitive effort (Klein, 2008).

This cognitive effort is further minimized by rarely considering different options in order to make a decision. This is shown with the concept of bounded rationality, meaning that the decisions people make uses limited rationality. When the designer needs to seek alternative solution options for a design, the decision to use one of the options is usually made before all the alternatives have been found. The activity of searching for a solution has costs for the designer, in the form of time or cognitive ability. Because of this most people tend toward satisficing behaviour. They go with a solution which is satisfying, instead of one which would be optimal to solve the problem (Simon, 1978; Tang & van Vliet, 2015).

Naturalistic decision making is often the first choice for people, and the most natural to use. And though in most cases it performs well enough for people to make quick decisions, in other cases it can lead to reasoning failures. These different approaches, the naturalistic decision making, and rational decision making, can be seen as a dual system working in the same mind which underlie reasoning.

Here System 1 is fast and intuitive with unconscious processes, naturalistic decision making. And System 2 is slow and deliberate with controlled processes, rational decision making (Figure 5). Typical uses of System 1 would be identifying if a person is angry based on their facial expressions. This happens automatically, without any real consideration put into the effort, i.e. you just know. System 2 is used when solving a mathematical equation, the thought process slows down as you consider the steps you have to take, and there is a conscious effort involved.

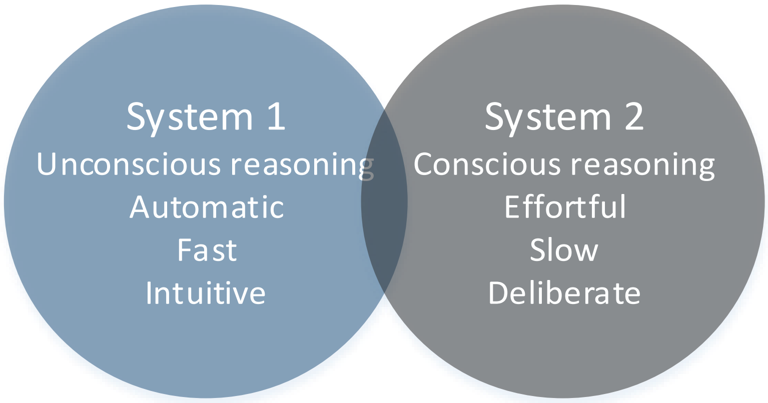


Figure 5 - Dual Systems

These two systems work together in daily life, with System 1 running automatically and calling on System 2's active involvement when there is a difficulty involved that System 1 alone cannot solve. In order for System 1 to work well there are two conditions; First, to have an environment which is regular enough to be predictable, and secondly, the opportunity to learn these regularities through practice. With the case of software design it involves 'wicked' problems, and as a result the environment can have many irregularities. These irregularities will cause System 1 to fail in some of its decisions, and so expert intuition in this case cannot be fully trusted (Kahneman, 2011). The trick then is to know when System 2 should be engaged, when to move from naturalistic decision making to the rational decision making.

Though the 'wicked' problem of designing means that even an expert's intuition can never be fully trusted, that doesn't mean that experts and novices are alike in their reasoning. Experience in designing does make a difference in approaching the design problem.

Experts in general have a more well-rounded approach. Familiar with the effort involved in solving problem they are more proactive; identifying missing information, important context information and inferring more requirements. Their expertise allows them to relate the problem context to previous or similar projects, to recognize the challenges which come with a design decision, and are more aware of specialists outside their field whose input would be useful for the design. Novices on the other hand are not as focused, and can spend more time on task-irrelevant activities. Their use of previous knowledge is limited and they tend to expect ready answers from clients, being less active in information gathering (Björklund, 2013). Inexperienced designers rely on what personal experience and expertise they have and go through a long process of deliberation before they can reason successfully about their design decisions. Problems especially occur in the matter of evaluation of design decisions. Often design decisions are not validated afterwards, which means previous decisions are rarely rejected (Van Heesch & Avgeriou, 2010).

The problems which novices experience are not limited to them, but in experienced designers they are lessened. This is why improving design reasoning will be particularly beneficial to inexperienced designers (Tang et al., 2008).

3. 3. Design Reasoning

Design reasoning is the process of reasoning about design decisions, which involves the rational decision making process. As design is a 'wicked' problem, in which not all factors entailing the design can be known, there is no ideal process to follow, and no structured process to come to a logical reasoning about design. Despite this, there is still value in attempting to follow an idealized process. Such a process would provide structure and guidance to the designer, taking away confusion about how to proceed. This structure is also more likely to help designers make a more rational design than when they are left to make their own procedure (Parnas & Clements, 1986). To improve the design reasoning process of the designers to a more rational one therefor has merit.

The reasoning process consists of identifying the relevant context and requirements, formulating and structuring design problems, creating solution options and deciding on a solution (Figure 6). These activities are not followed in any fixed order while designing but designers go back-and-forth between activities. This is typical of a 'wicked' problem, design problems might lead to identifying more requirements and a design decision might open up more solution options. It is an incremental and iterative process (Razavian et al., 2016).

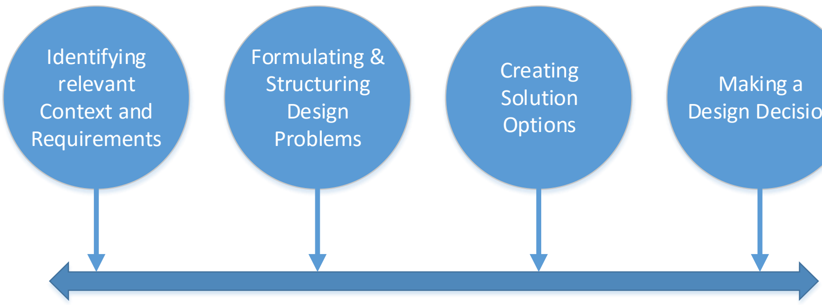


Figure 6 - Design Reasoning Process

To improve decision making the design reasoning process is one which has been identified as a measure of improving this process in the research field, design rationale (DR). This field focuses on making the design activities and the reasoning behind choices explicit by documenting them. For this purpose several tools have been designed to capture design rationale (Chandrasekaran et al., 1993).

But while the software industry is aware of the need for DR in their software design, and many tools and methodologies such as IBIS (Kunz & Rittel, 1970), DRL (Lee, 1991), SOAD (Zimmermann et al., 2009) and AREL (Tang et al., 2007) are available, their use does not guarantee a quality design. The problem with these tools is that they provide designers with a way to justify their design decisions, and communicate these to other designers and/or stakeholders, but does not focus on the internal thought process which centres on the ability to reason, i.e. design reasoning (Tang, 2011). This leads to design reasoning failures which negatively influence design decisions. Some of the main causes of design reasoning failures can be divided into three categories: cognitive bias, illogical reasoning and low quality premises.

Cognitive bias occurs when judgments are distorted because the probability of something occurring is not inferred correctly or there is an intuitive bias. This can be seen with representativeness bias and availability bias, where the probability of an event is mistaken because it either looks more typical, or representative of that event, or because it is more easily envisioned. An example is anchoring, where software designers choose solutions for a design problem based on familiarity, even when it is ill-suited to solve the problem (Stacy & MacMillan, 1995; Tang, 2011).

Illogical reasoning we have discussed in part already. This is when the design reasoning process isn't followed and problems occur with identifying the relevant requirements. The basis premises and arguments being used in the design discussion aren't based on facts.

Low quality premises is also known as hidden assumptions. When an assumption hasn't been presented truly they can be either a personal believe and not accurate, or the premise is incomplete or missing, making it inadequate. Much of reasoning depends on the quality of the premises themselves, if these are not explicitly stated or questioned software designers are more likely to make incorrect decisions (Tang, 2011).

Improving design reasoning by supporting rational decision making can guide designers to follow the logical reasoning and thereby influence the design discourse in a positive manner. As mentioned previously the decision making which is most natural for people to use, naturalistic decision making, is not without its flaws. In order to fully understand how this reasoning process can go wrong several examples will be used to illustrate the issue, taken from Tversky & Kahneman's well-known article "Judgement under uncertainty: heuristics and biases" (1974).

Consider first a question of probability:

> *Steve is very shy and withdrawn, invariably helpful, but with little interest in people, or in the world of reality. A meek and tidy soul, he has a need for order and structure, and a passion for detail.* (Tversky & Kahneman, 1974).

The question here is which occupation is Steve more likely to have; farmer or librarian? Invariably people will judge Steve to more likely be a librarian. In fact what is happening here is that prior probabilities are not considered. When looking at this question from a logical reasoning perspective, considerations of there being more farmers than librarians would have to be taken into account. But what happens in fact is that the description of Steve matches that of the librarian stereotype, and instead of using probability people look at representativeness.

Consider another question, this one about chance: *When tossing a coin it comes up either heads or tails, what is the most likely sequence of outcomes; H-T-H-T-T-H or H-H-H-T-T-T?*

The first sequence will appear the most random and this is what most people would judge to be the most likely sequence of outcomes. This is a misconception of chance, as people expect that the true randomness of chance is represented even in a short sequence. In truth the sequences are too short to tell if it is truly random or not, both sequences are equally likely.

These examples illustrate that the naturalistic decision making process is natural to everyone, and that everyone is also susceptible to its fallacies, even when they feel to be perfectly logical at the time. To follow a logical reasoning process requires more effort and consideration than one might suspect, and knowing when unconscious biases are involved in decision making is not always clear.

One method of moving deliberately from System 1 to System 2 is to influence the reasoning process by mitigating some of the main causes of design reasoning failures. To mitigate these reasoning failures there are several design reasoning techniques which can be implemented during design, such as risk analysis (Poort & Van Vliet, 2011), trade-offs (Bass et al., 2012), assumption analysis (Lago & van Vliet, 2005), and problem structuring (Restrepo & Christiaans, 2004). These support logical reasoning and work against the cognitive bias. The assumption analysis in particular helps define low quality premises. The reasoning techniques are mostly question and awareness based; the designer should ask questions surrounding a particular topic within software design where problems most often occur.

Problem structuring is the process of constructing the problem space by decomposing the design into smaller problems. These then lead to reasoning about requirements and the unknown aspects of the design (Simon, 1973). It focuses on identifying design problems and how these can be resolved when the situation is one the designer is unfamiliar with. It is used to identify the problem space and the key issues in design by asking questions related to the problem, such as what are the key issues. It occurs initially in the first stage, design planning, but should be used throughout the design process in all stages. Its aim is to prevent the designer from overlooking key issues because of unfamiliarity with the problem. The more time spend on problem structuring the more rational an approach the designer uses.

Option generation is a technique specifically directed at the problem of anchoring by designers, in which the first solution which comes to mind is implemented without considering other options. Designers rely more heavily on prior beliefs and intuition than a reasoning process. With option analysis the designer looks at each decision point at what options are available to solve a design problem.

Constraint analysis looks at the constraints exerted by the requirements, context and earlier design decisions and how they impact the design. These constraints are often tacit and should be explicitly expressed in order to take them into account. Trade-offs can come from conflicting constraints.

Risk analysis is a technique to identify any risks or unknowns which might adversely affect the design. Risks can come from the designer not being aware if the design would satisfy the requirements, in which case the design needs to be detailed in order to understand these risks and mitigate them. Or the design might not be implementable because designers are unaware of the business domain, technology being used and the skill set of the team. These risks should be explicated and estimated.

Trade-off analysis is a technique to help assess and make compromises when requirements or design issues conflict. It can be used for prioritization of problems and to weigh the pros and cons of a design which can be applied to all key decisions in a design (Tang & Lago, 2010).

Assumption analysis is a technique to question the validity and accuracy or the premise of an argument or the requirements. It focusses mainly on finding hidden assumptions. It is a general technique which can be used in combination with the other reasoning techniques (Roeller, Lago, & Van Vliet, 2006).

What all these reasoning techniques have in common is that they enable the designer to question their reasoning process by explicitly stating the assumptions, constraints, risks and trade-offs which can occur during software design.

3. 4. Reflective Practice

How these reasoning techniques prompt designers to use more logical reasoning is through reflective thinking, which in the simplest sense means that the person consciously evaluates their ideas and decisions. Using reflection to increase reasoning is not a new technique and has been applied in health and educational professions. It was first championed by John Dewey who differentiated reflective thinking from normal thinking. This is generally called reflective practice or experiential learning and is used in situations where people learn from their own experiences by using deliberate reflection. The reflective process is a conscious one and involves critical thinking about decisions in a continual loop of reflection on experience, which results in a more effective problem-solving and learning (Boud, Keogh, & Walker, 1985; Dewey, 1933). Critical reflection comes from a position of uncertainty in which beliefs, assumptions and hypotheses have to be continuously evaluated against the existing data and other plausible interpretations. Without this feeling of uncertainty deeper understanding and different ways of thinking won't occur. To insert this feeling of uncertainty people need to question a particular action or decision as to whether it is the right one. This questioning behaviour can then lead to changes and re-evaluation (Larrivee, 2000).

Several different, but similar, reflective practice models have been made showing learning cycles including reflective thinking. Some of the most well-known are by Borton (Borton, 1970), Kolb and Fry (Kolb & Fry, 1975), and Gibbs (Gibbs, 1988), but these focus primarily on questioning the experience after the fact, i.e. reflection-on-action. For this research, what we are exploring is reflection-in-action for which Schön's reflective model suits best (Figure 7). Reflection is used by designers in two ways; reflection-in-action and reflection-on-action. Reflection-in-action is to think about the actions that you are taking, as you are doing them, reflecting on your understandings while criticizing and restructuring them. Reflection-on-action is retrospective in nature, thinking back to a previous event and exploring their understanding of the event (Schön, 1983).

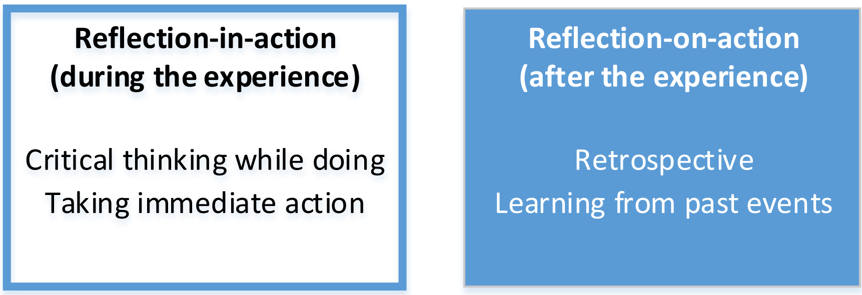


Figure 7 - Schön's Reflective Model

There have been several qualitative experiments which use the reasoning techniques to influence and improve design reasoning (Tang et al., 2008; Tang & van Vliet, 2009). In the latest experiment "In Two Minds: How Reflections Influence Software Design Thinking" (Razavian et al., 2016) the reasoning techniques discussed were tested to see how and if they influences the design reasoning process. An experiment was set up involving student groups to determine if questions based on the reasoning techniques would enhance the reasoning process. The results were that the questions did trigger reflective thinking and that this enhanced the reasoning process. These were evaluated by looking at the design discourse quality of the participants.

The model used to test this hypothesis was based on two interacting minds. Mind 1 is the logical reasoning process which is based on rationale, while Mind 2 is the conscious reflective thinking on Mind 1 (Figure 8). Here Mind 1 can be seen as System 2 thinking, the logical reasoning process, and to facilitate this Mind 2 is necessary as a reflection of that logical reasoning process. Most of the research focus in the field has been on Mind 1, the reasoning process, but the hypothesis of the authors was that stimulating reflective thinking on this process would improve design reasoning and rational decision making (Razavian et al., 2016).

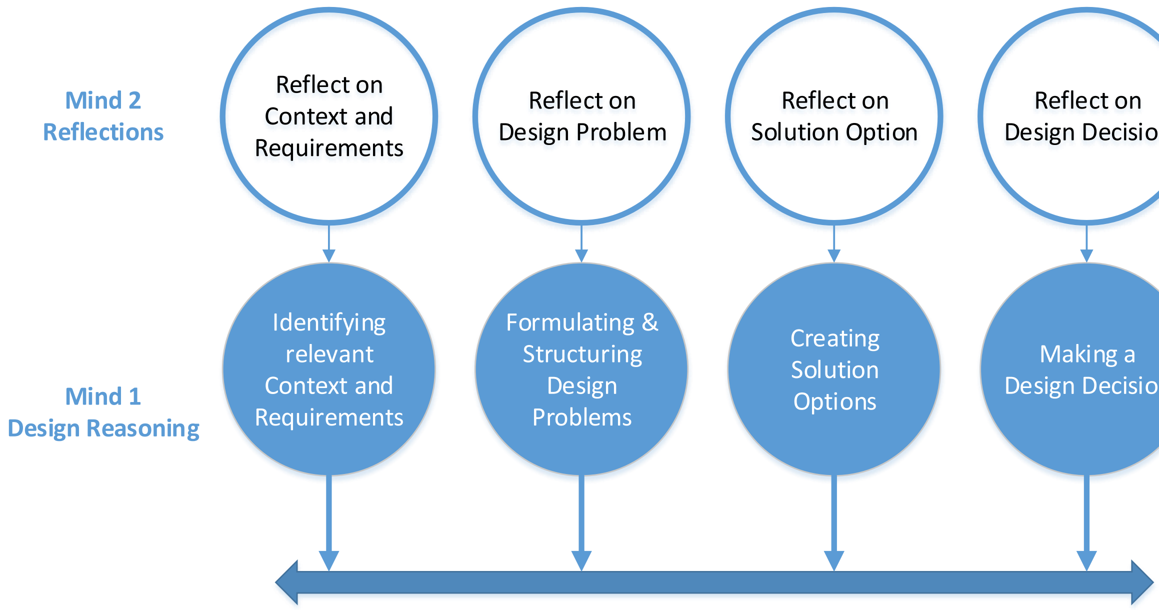


Figure 8 - Reflection on the Design Reasoning Process (Razavian et al., 2016)

These processes of System 2 thinking are further supported by recent theories in cognitive science which argues that System 2 has two levels of processing; the algorithmic level and the reflective level. It proposes that the algorithmic level or design reasoning in this case, is where individual differences in intelligence come out. This is because with critical thinking tests results showed that intelligence was not a significant factor for rationality. It is instead the reflective mind where critical thinking skills for avoiding biases lies. Rationality then is more than just intelligence, or design reasoning, it needs both Mind 1 and Mind 2 in order to work sufficiently (Stanovich, 2009).

In the experiment the design reasoning techniques chosen were; assumption analysis, risk analysis, constraint analysis, problem analysis and trade-off analysis. These activities and techniques were combined to make up general questions which lecturers, whom joined the student groups, could ask the students during designing to prompt reflection (Table 1). These questions were external reflections as they came from external sources, in this case the lecturer. Their use was to trigger the participants into reflecting on their reasoning process and thinking further about the design. The conclusion of the study was that these reflective questions did have an influence on the reasoning process (Razavian et al., 2016). These reflective questions serve as the inspiration from which the reflective card system is derived.

	Design Contexts and Requirements	Design Problems	Design Solutions	Decision Making
Assumption Analysis	What assumptions are made?	Do the assumptions affect the design problem?	Do the assumptions affect the solution option?	Is an assumption acceptable in a decision?
Risk Analysis	What are the risks that certain events would happen?	How do the risks cause design problems?	How do the risks affect the viability of a solution?	Is the risk of a decision acceptable? What can be done to mitigate the risks?
Constraint Analysis	What are the constraints imposed by the contexts?	How do the constraints cause design problems?	How do the constraints limit the solution options?	Can any constraints be relaxed when making a decision?
Problem Analysis	What are the contexts and the requirements of this system? What does this context mean?	What are the design problems? Which are the important problems that need to be solved? What does this problem mean?	What potential solutions can solve this problem?	Are there other problems to follow up in this decision?
Trade-off Analysis	What contexts can be compromised?	Can a problem be framed differently?	What are the solution options? Can a solution option be compromised?	Are the pros and cons of each solution treated fairly? What is an optimal solution after trade-off?

Table 1 - Reflective questions (Razavian et al., 2016, p. 16)

3. 5. Conclusion

An important process within software design is design reasoning, as the design decisions which make up a software design can influence the quality of the eventual product. Software design is also a 'wicked' problem, meaning that a problem will not have a standard solution so that the solution given depends on the designers' approach. The problem here is that designers often use naturalistic decision making, and these can lead to common reasoning failures such as cognitive bias, illogical reasoning and low quality premises. Especially inexperienced designers are more likely to show reasoning failures, but experienced designers are not exempt. To improve design reasoning a rational decision making process needs to be used by the designers. These can be triggered with the use of reasoning techniques such as; problem structuring, option generation, risk analysis, constraint analysis, trade-off analysis and assumption analysis. These reasoning techniques work through the use of reflective thinking, evaluating previous decisions and revising them when needed.

For this research the reasoning techniques are given form as a simple card game, which prompts designers to reflect on their design decisions. The effectiveness of this card game in improving the design reasoning of designers is then tested in a series of experiments involving experienced and inexperienced designers.

4. Card Game Development

As explained in 2. Research Method, the card game is used in several experiments. Both experienced and inexperienced designers are tested with the cards to see the influence they have on design reasoning. Before these experiments a simple run-through of the initial prototype is made in order to determine if there are any obvious problems with the game. This is then used to improve the card game to the final prototype which is used in the experiments. The experiments consist of a test group using the cards, and a control group without. Results of these initial prototype experiment is shown in this section.

The reflective experiment in the "In Two Minds" article made use of a lecturer to influence the designers and have them reflect on their decisions (Razavian et al., 2016). As the lecturer is a designer themselves with some experience, they can choose their own timing and the relevance of their questions to the designer groups. This could have an influence on the performance of these groups. For this experiment the lecturer is replaced with cards, removing this potential influence.

The choice for cards came about because with DR many tools have been developed but they are not prevalently used in the design world. The most common reason for this is that the adoption and use of these design rationale systems take too much time and are too costly to be effectively used (Tang, Babar, Gorton, & Han, 2006). The cost-effectiveness of such a system is in fact the most important characteristic for consideration by software companies (Lee, 1997).

As a result the system being developed in this research should be easily adoptable by the software industry, and so cost-effectiveness and timeliness become important requirements. Many of the previous DR tools are software systems which designers would have to learn to use, and which have collaborative options meaning that people would also have to learn to use the tool together. Designers are already overrun with software tools and checklists to improve software design. The aim is to create something new which is simple to learn and intuitive to use.

As a result, the system developed in this thesis will be a card system which designers can easily learn to use, can be used at the moment of discussion, and are much cheaper to produce. Developing a card system is an obvious choice as this is a common tool within software development on several fronts. It is used for stimulating creativity with IDEO method cards which focus on inspiring design and exploring new approaches, while keeping designers in the centre of the design process (IDEO, 2015). Cards are also used for learning about designing an architecture with Smart Decisions playing cards, which follow an architecture design process (Smart Decisions, 2016). And more practically used with planning poker, a card game which is used during release planning to estimate the time needed to implement a requirement (Grenning, 2002).

An advantage of cards is that they also have a visual element to them and can visually prompt designers when they look at them to consider what they represent. Playing cards also structures the mind and thinking as it creates a focus point for everyone to think about. It provides an interactive checklist for designers to become more aware of the design reasoning process.

4. 1. Card Game Prototype Development

4.1.1 *Pilot Prototype*

Based on the literature and particularly the reflective experiment, an initial design of the cards was made. For the prototype the questions table in the reflective experiment was simplified and the problem analysis and design decision removed from what the cards would represent (Table 2). As the main activities of context, problem and solution will already be represented by cards this should be sufficient for problem analysis, which questions designers to ask what the context is. The design decision was also not made into a card as this is the end-result of the three main design activities. A decision is choosing a solution and is the aim of software design.

The cards will be based on these main activities and the design reasoning techniques used in the reflective experiment; assumption, risk, constraint and trade-off.

	Design Contexts and Requirements	Design Problems	Design Solutions
Assumption	What assumptions are made?	Do the assumptions affect the design problem?	Do the assumptions affect the solution option?
Risk	What are the risks that certain events would happen?	How do the risks cause design problems?	How do the risks affect the viability of a solution?
Constraint	What are the constraints imposed by the contexts?	How do the constraints cause design problems?	How do the constraints limit the solution options?
Trade-off	What contexts can be compromised?	Can a problem be framed differently?	What are the solution options? Can a solution option be compromised?

Table 2 - Card System Prompts

These reasoning techniques are primarily implemented by asking questions to trigger the designer into reflective thinking on the reasoning process.

The card system is mainly based on the previous reflective experiment, and provides the starting point for the prototype as these questions have already proven to be effective (Table 1). A card system was chosen to pose reflective questions because the aim is not necessarily to document rationale, and a software tool is expensive and rarely adopted by the software field. Card games are familiar to designers and are currently being used in software development.

The design of the cards focuses on their appearance, the types and number of cards represented and the gameplay rules.

Three main cards were designed (Figure 9) representing the design activities. They are simple in design to avoid confusion about their meanings. The cards are represented by an image which invokes their meaning. The context card holds a blue cloud to represent that all options are still open at this stage. The problem card holds an orange circle to represent that the problem can be looked at from all sides. The solution card holds a green square which represents a clearly defined solution.

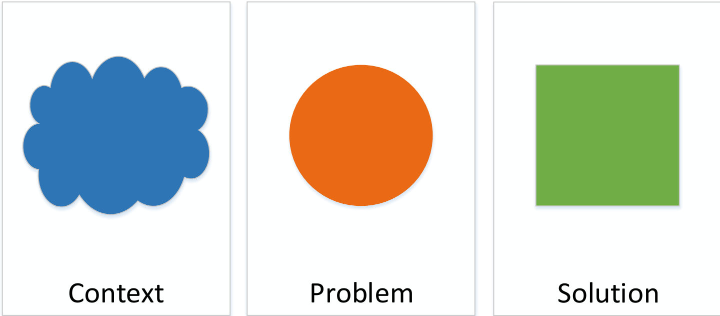


Figure 9 - Main Cards

The remaining four reasoning techniques are represented as letters combined with the main activity cards (Figure 10). This ensures that the images do not become too cluttered, and that the shapes are familiar to the designers.

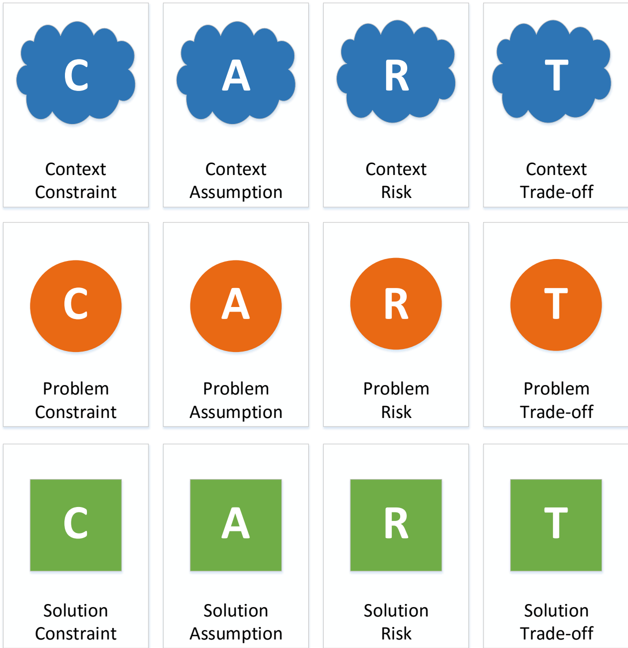


Figure 10 - Reasoning Technique Cards

For the card game each participant would be given a deck of the fifteen cards, and the questions table (Table 2) to help them with question examples.

The gameplay rules were kept simple and focused on using the cards during the discussion but did not describe how the cards should be used. Gameplay rules were:

1. The game is played in rounds, in which a round represent a discussion;
2. A round starts when a card is played in order to start a discussion;
3. During discussion other cards can be played and reused;
4. When a decision is made the discussion ends;
5. Cards related to the discussion are removed.

When discussing these cards players can agree with the statements being made, or disagree and explain why. They also ask questions about the statement when something is unclear and must be clarified.

The assignment to be designed was based on the Irvine experiment with a few differences (Appendix B). These related to the course which the students were following in which several architecture viewpoints needed to be designed and their rationale for their designed needed to be documented after the design session. During the design session students are required to maintain a log of the cards being played (Appendix D). After the design session a questionnaire about the cards is filled out (Appendix E).

4.1.2 Pilot Results

To test the first prototype a dry run was performed involving two Master students who had completed the course in which the student experiments will be taken. They were familiar with the viewpoints which needed to be designed and had a technical background.

The dry run was important in that it provided the opportunity to see the card game in action, and how designers react to their reflective nature. Based on the feedback provided the cards were improved to be used in the student experiments. The feedback is based on observations during the dry run and the questionnaire.

The main findings of the dry run session are that the cards were hardly played, and use tapered off over time. Related to this issue is that the timing to play the cards was unclear. Despite this the cards did have an influence on the design discussion as the designers immediately started using the terms used on the cards, such as assumption and constraint, to refer to points in their discussion.

Timing to Play Cards

The first finding is that the timing to play the cards was unclear. The discussion didn't occur in clearly defined rounds and it was not obvious that the cards could be connected to each other to represent the discussion process. Instead the designers discussed one card at a time, even when they brought in different topics which are represented by cards.

Cards Not Played

As it was not clear when the cards should be played the gameplay became confusing and physically playing cards was hardly seen. Instead the designers would mentally lay down cards, and only state them in their discussion. There were fifteen cards to choose from for the discussion and this quickly turned out to be too much. The combination of design activities with techniques also became a problem. At one point a discussion arose over if the designers were discussing a problem constraint or a solution constraint. The cards were too specific to fit the flow of discussion.

The risk card especially was not used during the discussion as the assignment was not seen as technical enough to have any real risks. Despite this risks were discussed when mentioning trade-offs, but these were not consciously seen as risks. Another reason for why the cards were played less is that the designers felt that it interrupted their discussion, especially as they had to note down the cards played.

Influence On Design Discourse

Despite these issues simply reading the cards and the question table was enough to make the designers use terms such as constraint and assumption to refer to their discussion. When they identified these as aspects of their discussion they would write these down in the log without playing a physical card.

It was mentioned that the cards did remind them of issues, and helped them think in such terms as constraints and assumptions. It helped them with issues they might otherwise have forgotten but they claimed it didn't have any real influence. The designers claimed that their usual process was much the same as they have some experience with designing and architecture. As there was no control group these is no evidence whether this was truly the case

Their initial discussion followed the reasoning process in that they refined the context, made a list of features and then focused on the problem and eventual solutions. Though they discussed the problems there was no prioritization of these, about which was more important or problematic. During the design session they came to new possibilities about the problems and how to solve these, but did not look thoroughly at the solution options. They found what would work and further designed this and when any other problem came up they did not reconsider the solution.

When the designers made up a documentation explaining their rationale for the models they designed, they did make use of the cards and their topics to explain why they chose to design the model a certain way.

Questionnaire Findings

The designers were asked to fill out a questionnaire after the assignment which focused on the experience the designers had working with the cards. The conclusions were that the symbols on the cards were not intuitive in their meaning, but on the other hand they as they rarely physically played the cards they had no time in which to become familiar with them. After the initial read-through of the assignment the questions table was not looked at further. The designers found the cards to be nice as a checklist, but distracting in use. Especially logging the cards was seen as distracting, but the expectation is if this was not necessary they probably wouldn't have used the cards, even mentally, and their use even as a checklist would have diminished.

The designers were not convinced that the cards helped them identify design issues and solutions they would not otherwise have identified. But their favourite card to play was assumption as it helped them in defining the problem and also was important in finding hidden assumptions.

4.1.3 Improved Reflective Cards

Despite the problems concerning playing the cards and the timing of when the cards should prompt players to consider their decisions, the card game did have an influence on the design discussions. Merely reading the questions table and being aware of the cards topics made the participants use the terms provided naturally in their discussion. During their discussion they spoke naturally of constraints and trade-offs and identified whether they were talking about a problem, solution or both. Many times during the discussion the designers would realize that what they were discussing fit into one of the card categories. But as there was no control group, for now we can only state that the language used during discussion was influenced by the cards.

For the second prototype several changes were made to the card system to ensure that the problems concerning card play and timing issues would be addressed. These include having fewer cards by not combining the design reasoning activities with the techniques. This means a deck now has only seven cards; context, problem, solution, assumption, constraint, trade-off and risk. These can be played in combination with each other (Figure 11).

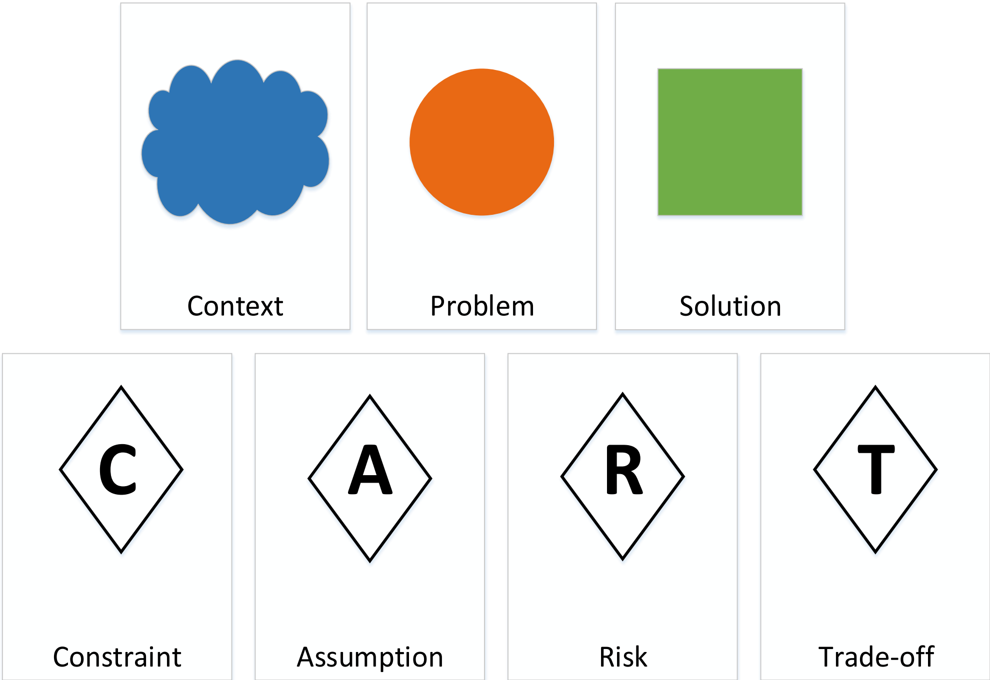


Figure 11 - Design Activity and Technique Cards

The gameplay rules were also simplified, and during the presentation an example was given of how the cards can be connected to each other. The rules focus more on the discussion and discard the idea of playing in rounds as discussion is too fluid to be constraint in this way. The gameplay rules are:

1. The game is played in terms of a discussion;
2. The discussion starts when you play a card;
3. Others contribute to the discussion by playing their own cards;
4. When a decision is made the cards related to that topic can be removed;

Finally, three reflective periods were added to the design session to help with the timing issues and ensure that the cards are played for at least 5 minutes. This ensures that the designers become more familiar with the cards and how to play them, without restricting the discussion to only the cards. Three periods were chosen equally spread out across the two hours to ensure that if the use of re-flection tapers off, the reflection periods will trigger these again. These reflective periods are:

1. At 15 minutes into the design session;
2. At 45 minutes into the design session;
3. At 1 hour and 45 minutes into the design session.

Another improvement which was discussed but eventually not added to the final prototype was the addition of an option card, to enforce that design alternatives are stated. This addition was thought of to be too early to add based on the dry run only. Better use of the cards themselves might prompt the designers into stating more alternatives on their own. The necessity cannot yet be asserted.

The questionnaire and card log were kept to be used in the student experiment. The improved assignment can be found in Appendix C.

5. CONCEPTUAL MODEL

The research question for this thesis is; *How can the use of a reflective method, prompting reasoning techniques during the decision process with the use of reflection, influence and support design reasoning in software architecture.* We answer this question with the use of cards that represent design reasoning techniques and activities as explained in the section before. The theory here is that using the card game will influence designers to move from naturalistic decision making to rational decision making by enforcing more thorough reasoning through reflection-in-action.

There are several hypotheses on how the card system will influence the design reasoning process of the designers, both experienced and inexperienced. As mentioned before, experienced designers can rely more on their intuition than inexperienced designers, and as a result the card game is more likely to have a greater effect on inexperienced designers. The hypothesis explained here are there for focussed on how the inexperienced designers are affected by the cards.

In order to mitigate the causes of design reasoning failures; cognitive bias, illogical reasoning and low quality premises, designers should follow a rational reasoning process. This can be achieved by providing reflection using several analyses techniques. The conceptual model is based in part on the design reasoning process and the reflective process described in the “In Two Minds” experiment (Razavian et al., 2016) (Figure 8).In the conceptual model we see how the cards are intended to influence this process (Figure 12). Three of the cards are based on the design activities of finding design problems, options and making decisions. These cards are intended to increase the use of the reasoning techniques; problem structuring and option generation. These in turn encourage reflection on the design reasoning activities in helping identify problems and generate options.

The other four cards are focused on representing certain analysis techniques; constraint, assumption, risk and trade-off. These cards are intended to increase the use of the corresponding reasoning techniques; constraint analysis, assumptions analysis, risk analysis and trade-off analysis. The effect of these cards through reflection is to make more considered decisions, using thorough reasoning instead of satisficing behaviour.

The following 6 hypotheses are aimed at the significant influence which the cards have on the reasoning behaviour and use of reasoning techniques. These are represented as null and alternative hypotheses.

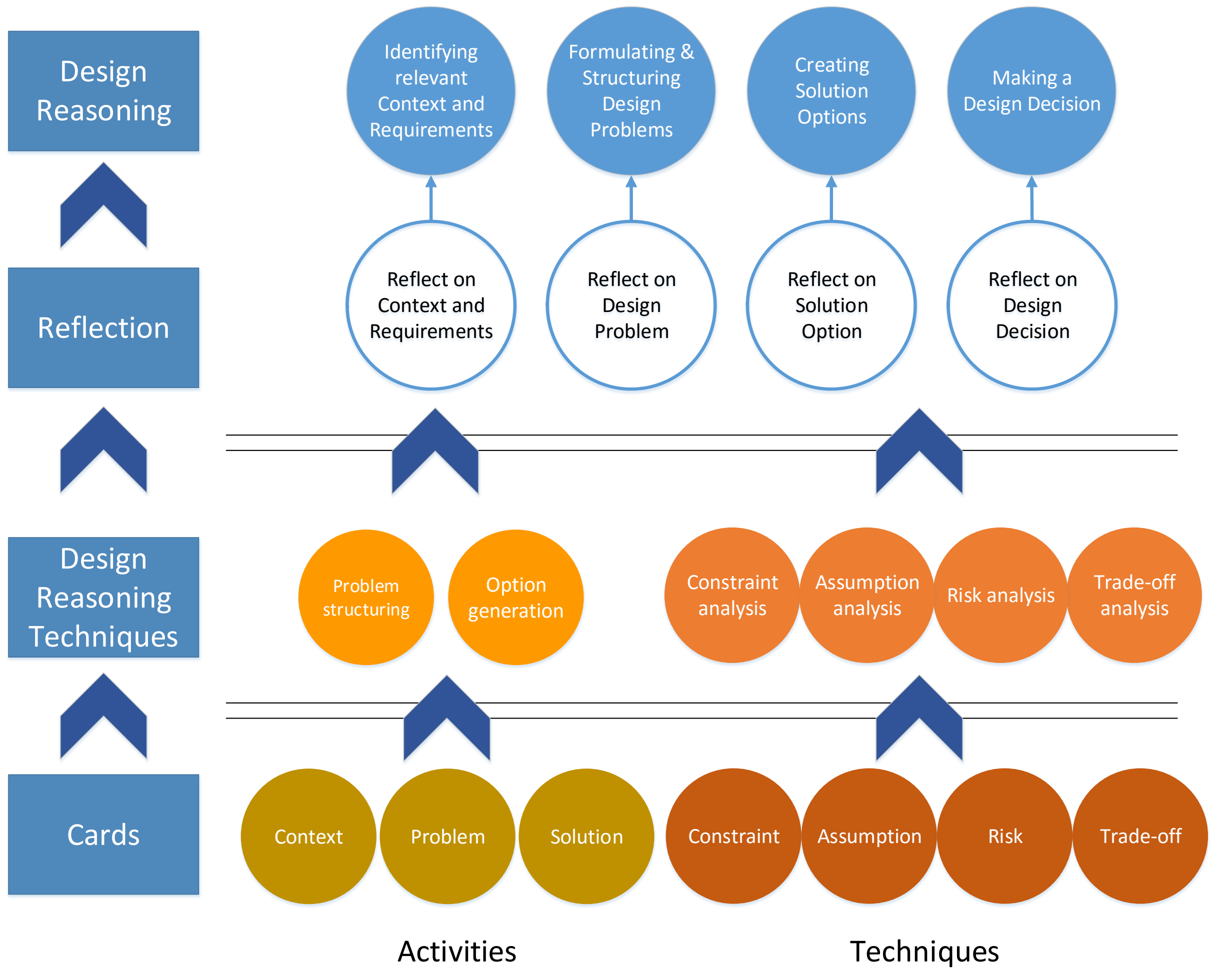


Figure 12 - Conceptual Model

5. 1. Problem structuring

H0: There is no significant difference for problem structuring use between test and control groups.

H1: There is a significant difference for problem structuring use between test and control groups.

The three activity cards influence designers to use more problems structuring. This is focused on structuring the problem space and specifically looking at key issues which need to be resolved. The cards will make the designers spend more time on exploring the problem and looking at key issues, which will lead to a more rational approach and more problems identified by the test group.

5. 2. Option generation

H0: There is no significant difference for option generation use between test and control groups.

H1: There is a significant difference for option generation use between test and control groups.

The three activity cards will influence designers to generate several options before coming to a decision. The cards will also prompt designers to discuss the different options for their design. This will lead to the test group using more option generation and identifying more options than the control group.

5. 3. Constraint analysis

H0: There is no significant difference for constraint analysis between the test and control groups.

H1: There is a significant difference for constraint analysis between the test and control groups.

The hypothesis for constraint analysis is that the constraint card will make the designers more aware of constraints in the requirements and make them explicit by stating them aloud. The card system will also make the designer more aware of how constraints can influence the system being designed. As a result the test group is expected to use more constraint analysis than the control group.

5. 4. Assumption analysis

H0: There is no significant difference for assumption analysis between the test and control groups.

H1: There is a significant difference for assumption analysis between the test and control groups.

The hypothesis for assumption analysis is that the assumption card will serve to substantiate the arguments used during the design discussion. The cards will direct the designers to question both the premises of the requirements and context to ensure that they are well understood, but also each other's arguments and potential assumptions. The test group is expected to question more assumption than the control group.

5. 5. Risk analysis

H0: There is no significant difference for risk analysis between the test and control groups.

H1: There is a significant difference for risk analysis between the test and control groups.

The hypothesis regarding risk analysis is that the risk card will influence designers to find risks in the design and think more critically about the design. When risks are found designers can simplify the design solution, compromise the requirements and design concerns to reduce its complexity. The test group is expected to use more risk analysis than the control group.

5. 6. Trade-off analysis

H0: There is no significant difference for trade-off analysis between the test and control groups.

H1: There is a significant difference for trade-off analysis between the test and control groups.

The hypothesis for the trade-off analysis is that the trade-off card will ensure that trade-offs are recognized as such and to prompt designers to discuss pros and cons of a design option and prioritize what's important. The cards will also help designers recognize situations in which the requirements and constraints cannot both be satisfied. The test group is expected to use more trade-off analysis than the control group.

5. 7. Conclusion

The overall hypothesis is that all the techniques will have a noticeable and significant influence on the design discussion in the ways described in the hypothesis, more with inexperienced designers than with experienced designers. To test the hypothesis, the card system will be tested by students and professionals. This will involve a control group and a test group. The aim of the experiment is to analyse if the test group conforms more to the conceptual model than the control group, and to see if there is a change in effectiveness with students and professionals.

At the hand of these results the main research question, if the card system supports the design reasoning process and influences design discourse, can be answered.

6. Student Experiments

The student experiment involved 12 groups of students, with most having three designers, two groups with two designers, and one group with four designers. Of these groups, 6 were selected to use the improved card game, the other 6 serving as the control group (Table 3). To select the groups they were first divided according to the test scores of a previous assignment, and then randomly selected by a person unfamiliar with the groups.

Control Group	Test Group
C1, C2, C3, C4, C5, C6	T1, T2, T3, T4, T5, T6

Table 3 - Control and Test Group Distributions

Each groups design session was recorded and transcribed after which it was coded and analysed. For the coding of the transcript two different coding schemes were used. One for the use of the physical cards, and the other for the reasoning process.

The coding scheme for the physical cards is only used on the test group (Appendix F). Combined with the logs that the participants had to fill in it shows where a card was played. This disregards other mentions of the card topics and looks only at what the participants register was played.

For the reasoning process another coding scheme was made to be used on all the groups (Appendix F). The codes enveloped the design reasoning process and the design reasoning techniques described in the literature and conceptual model. There are two interlocking categories; the design reasoning techniques are the overarching category and the design reasoning elements are singular elements which make up the reasoning techniques (Figure 13).

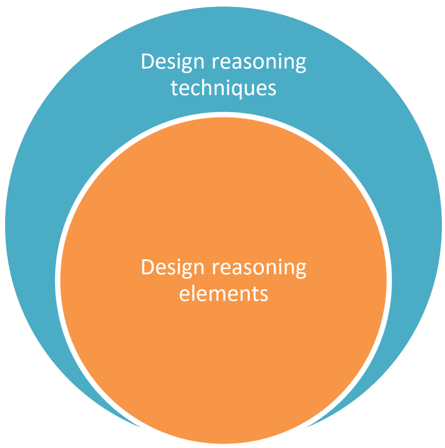


Figure 13 - Reasoning Categories Venn Diagram

The reasoning techniques are those of constraint analysis, option generation etc., and these consist in their turn of reasoning elements such as a singular constraint or option.

Coding the transcripts according to these coding schemes was also done at the hand of a few rules to maintain consistency across the groups (Appendix F). For further interpretation of the results two of these rules in particular are important to keep in mind;

- Option generation needs at least two options;
- A trade-off analysis needs at least one pro and con.

This means that for a discussion to be marked as a trade-off analysis, the designers need to discuss at least one pro *and* one con, only pros or cons will not be considered a trade-off.

6. 1. RESULTS

The main purpose of the experiments is to find out if the card game has any obvious influence on the reasoning process of designers. In this case it involves inexperienced designers who could most benefit from a logical reasoning decision making process. In the following section first the results of the transcript coding are given in terms of frequencies, after which these are tested for significance according to the conceptual model.

6.1.1 DESIGN SESSION LENGTH

The first and most obvious difference is the time taken for the design session between the control and test groups. Both were given two hours to complete their design session, but half of the control groups finished their design earlier, while almost all the test groups made full use of their time (Table 4). The control groups in particular had less groups making use of the full two hours, only two out of six approached the two hours. In the test group only one group did not reach two hours because of a misunderstanding in how much time they had left. One other group went over the two hours with a half hour.

Control Group	Time	Test Group	Time
C1	1:43:51	T1	2:01:23
C2	1:57:15	T2	1:23:00
C3	1:22:47	T3	1:59:56
C4	1:13:39	T4	2:24:42
C5	1:17:20	T5	1:54:48
C6	2:05:33	T6	1:51:34

Table 4 - Design Session Times

6.1.2 CARD GAME INFLUENCE

To establish if there are any noticeable differences in the use of the reasoning techniques, the frequencies in which they were used are measured and compared (Table 5).

The results show that there is a noticeable difference in the frequencies in which the reasoning techniques are used. From the techniques directly influenced by the cards especially assumption and risk analysis are consistently more used by the test groups. Option generation and problem structuring which are indirectly influenced by the cards are also more prevalent with the test groups. The percentage change between the test and control groups overall is great, a 75% difference. These results correspond closely with the hypothesis that the cards would influence design reasoning.

There are only two analysis techniques which have questionable results. Trade-off analysis has the lowest occurrences for both groups, which is peculiar as trade-offs usually occur when options are generated, but the results here do not show a similarity. And though there is a difference in their use between the test and control group, this is not very large. With constraint analysis the differences between results are even smaller.

Analysis techniques	T1	T2	T3	T4	T5	T6	Total
Assumption analysis	14	6	9	5	8	7	**49**
Constraint analysis	7	9	2	7	8	10	**43**
Risk analysis	6	7	6	5	5	5	**34**
Trade-off analysis	5	2	2	4	2	1	**16**
Option generation	19	2	11	6	6	8	**52**
Problem structuring	33	19	24	25	20	25	**146**
Total	**84**	**45**	**54**	**52**	**49**	**56**	**340**

Analysis techniques	C1	C2	C3	C4	C5	C6	Total
Assumption analysis	2	0	2	3	1	3	**11**
Constraint analysis	4	6	10	7	7	5	**39**
Risk analysis	2	2	3	4	1	3	**15**
Trade-off analysis	1	1	0	3	0	1	**6**
Option generation	1	3	4	6	9	7	**30**
Problem structuring	15	20	18	12	15	13	**93**
Total	**25**	**39**	**37**	**35**	**33**	**32**	**194**

Table 5 - Design Reasoning Technique Frequencies

To better understand the results we need to look at the unique values of the individual reasoning elements. The analysis techniques are overarching and can contain several elements, which can be repeats of unique elements, e.g. one constraint analysis addressing two constraints. The unique values are given in Table 6, shown in the dark grey columns. The percentage difference shows 77% more identification of reasoning elements by the test groups.

Design reasoning elements	T1		T2		T3		T4		T5		T6		Total Unique
Assumption	15	15	6	6	10	6	5	5	11	8	7	7	**47**
Constraint	9	8	17	8	5	4	9	7	13	8	18	7	**42**
Risk	6	6	7	6	7	5	7	7	7	7	8	6	**37**

Design reasoning elements	C1		C2		C3		C4		C5		C6		Total Unique
Assumption	2	2	1	1	2	2	5	5	1	1	3	3	**14**
Constraint	4	4	12	8	17	9	9	5	22	8	16	8	**42**
Risk	2	2	2	2	3	3	4	4	1	1	3	3	**15**

Table 6 - Design Reasoning Elements Frequencies

The results show that assumptions and risks occur with a similar frequency as with their reasoning techniques. The constraints are shown to have an even more similar frequency across the test and control groups, there is hardly any difference at all.

To investigate the trade-off analysis results we need to look at the elements which make up a trade-off, pros and cons (Table 7). With this closer look towards the results the differences between the test and control group becomes more obvious. The frequencies of pros and cons more closely match that of option generation. It shows that more pros and cons for various options are given, only the combination of pro *and* con is scarce. As a trade-off is required by the coding scheme to have both a pro and a con for an option, this explains why the trade-off analysis has had such low frequencies. What is of interest here is that compared to the control groups, the test groups use both more pro with 53% more, but also far more cons to argue about their options, tripling the amount with 269% compared to the control group.

	T1	T2	T3	T4	T5	T6	Total
Trade-off analysis	5	2	2	4	2	1	**16**
Pros	17	4	10	8	4	3	**46**
Cons	10	2	8	13	9	6	**48**

	C1	C2	C3	C4	C5	C6	Total
Trade-off analysis	1	1	0	3	0	1	**6**
Pros	2	4	5	12	3	4	**30**
Cons	1	2	0	4	2	4	**13**

Table 7 - Trade-off analysis, pros and cons

When looking at the he identified design problems, options and solutions we find that mostly the design options have increased in the test groups compared to the control group, with a percentage difference of 56% (Table 8). This corroborates with the increase in option generation established before. The identified design problems and solutions have increased with the test groups, but not by much, with a percentage difference of 34% in design problems, and 24% with design solutions.

	T1	T2	T3	T4	T5	T6	Total
Design Issues	29	10	17	17	8	13	**94**
Design Options	42	9	33	28	18	18	**148**
Design Decisions	29	10	17	17	8	11	**92**

	C1	C2	C3	C4	C5	C6	Total
Design Issues	3	8	13	19	16	11	**70**
Design Options	5	10	14	18	25	23	**95**
Design Decisions	4	9	13	20	17	11	**74**

Table 8 - Design issues, options and decisions

6.1.3 Card Influence on Language

To see exactly how the cards influence the design reasoning process of the test groups, we look into more detail on how they directly influence the language used in the design sessions. During the initial dry run the participants were very quick to use the terms on the cards to incorporate the analysis techniques into their design session. The inclusion of such language seemed very natural and as it was used less during the end of the session, so too did the use of such techniques. This suggested that the language use could be a hint to how card play directly influences design reasoning.

As a result, in this experiment we also look at the use of the specific card terms throughout the session, and how this corresponds with both the reflective periods and first use of the cards, and the use of analysis techniques. The activity cards; context, problem and solution, are excluded from this process because their use is too obvious in a problem-solving activity such as software design. In Table 9 the frequencies for the use of assumption, constraint, risk and trade-off are counted for both the control and test groups.

Term used	T1	T2	T3	T4	T5	T6
Assumption	8	12	14	5	13	23
Constraint	6	3	1	18	12	21
Risk	8	19	12	0	5	11
Trade-off	6	1	9	4	11	20
Term used	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**
Assumption	0	0	0	3	0	0
Constraint	0	1	4	5	10	0
Risk	1	0	0	0	0	0
Trade-off	0	0	0	0	0	0

Table 9 - Analysis Technique Term Mention

As is instantly apparent, the test group makes more use of the terms than the control group. This is of course natural as the cards directly mention these terms to discuss, but on the other hand these terms and their analysis techniques are not unfamiliar words in software design. While assumption, risk and trade-off are barely mentioned in this form, they are also rarely discussed with any kind of synonym. Only constraints are mentioned as is, and several groups otherwise use the synonym rules to refer to the constraints instead. This corresponds with the earlier results in that constraint analysis use does not change across test and control groups.

6. 2. Analysis

The results of the experiment show significant differences between control and test groups which corresponds with previous research results using the same reasoning techniques. The cards overall trigger more design reasoning in the test groups, as more assumptions and risks are identified, and more key issues are defined with problem structuring. In this section we go further into detail about the results and what they mean, focusing on both quantitative and qualitative analysis.

6.2.1 Quantitative Significance Testing

The results of the student experiments show several noticeable differences between test and control groups, in this section we test these differences for statistical significance.

To test the results two different tests are used. The first being the independent T-test, and the second being the Mann Whitney test. In most cases the independent T-test will be used, except in those cases where the assumptions needed to perform an accurate independent T-test cannot be met. In these cases the Mann Whitney test will be used as it is more robust and does not need to have the assumptions of the independent T-test met in order to come to reliable results. These tests and the assumptions have been further explained in the section 2.3.1 Quantitative Analysis. Statistics were performed with the PASW Statistics program. In general, to be considered significant the *p*-value should be below 0.05 (Field, 2013). All statistical results can be found in Appendix J.

The first three assumptions needed for the independent T-test have to do with the set-up of the experiment. There needs to be a dependent variable measured at the ratio level. In this case these are the frequencies in which for example the assumption analysis have been counted per group. The independent variable is dichotomous category of test and control groups. As the experiment has been set-up so that participants only contribute to one group, and were not allowed to interact during the design session, there is also independence of observation. And lastly, there also needs to be homogeneity of variances which is tested using Levene's test for equality of variances. When the *p*-value is greater than 0.05 the variances are considered equal. In all cases the variance *p*-value is above 0.05 (Appendix J). With all of the previous assumptions there have been no problems which violate the assumptions, but with the following assumptions there are some problems with the values which result in the Mann Whitney test being used.

In order to use the T-test there should be no significant outliers in the data. With the student groups there are some outliers which can be seen in the boxplots (Appendix I); T1 for design decisions, design issues, assumption, problem structuring, option generation and assumption analysis. C4 for pros and trade-off analysis. T3 for constraint and constraint analysis. In order to determine if the outliers were significant the T-test was performed both with and without the outliers. Results were that in most cases the significance value did not change, results which were not significant with all samples, did not become significant without outliers and vice versa.

The exception is the trade-off analysis with C4. In this case the *p*-value with all groups was 0.053, not significant but very close to it, and with C4 removed the *p*-value was 0.018, meaning that the outlier does make a significant difference in the results. C4 has more trade-off analysis than any other control group, and also finds many more pros than the comparable groups. Their focus on finding so many pros means that their combination of pros and cons is higher as well, resulting in more trade-off analysis used. As the outlier is not a result of erroneous data it was not removed from the samples. But as the assumption is violated because there is a significant outlier the Mann Whitney test is used instead.

For the T-test the dependent variables must be normally distributed for each group of the independent variable. In this case we use the Shapiro-Wilk test of normality; if the *p*-value is above 0.05 then the distribution is normal. With three groups concerning design reasoning elements there is a problem concerning normality where the results are below 0.05. This is with the assumption test group, constraint test group and pros control group. In these cases distribution is not normal. The reason for this is an outlier, with groups T1, T3 and C4 providing extreme values in comparison with the other teams. For these three design reasoning elements a nonparametric test must be applied to find the significance, the Mann-Whitney test, for which a normal distribution is not necessary.

Design reasoning techniques & elements	T-test *p*-value	Mann Whitney test *p*-value
Assumption analysis	**.001**	
Constraint analysis	.648	
Risk analysis	**.001**	
Trade-off analysis		**.040**
Option generation	.199	
Problem structuring	**.004**	
Assumption		**.005**
Constraint		.451
Risk	**.001**	
Pros		.369
Cons	**.006**	
Design issues	.322	
Design options	.159	
Design decisions	.460	

Table 10 - Design Reasoning Techniques & Elements Significance Values

The results of the tests can be found in Table 10, the difference is significant if the *p*-value is below 0.05. The significant results are given in white on grey.

As we now have the results of the significance tests we can look to see if the hypotheses made in 5. Conceptual Model can be accepted or rejected. We address the analysis techniques which the three activity cards (context, problem, solution) and the four technique cards (assumption, constraint, risk, trade-off), represent.

Problem Structuring

H0: There is no significant difference for problem structuring use between test and control groups.

H1: There is a significant difference for problem structuring use between test and control groups.

The hypothesis concerning the problem structuring reasoning technique was that the three activity cards would prompt designers to explore problems and find key issues, prompting designers to use a rational approach and identify more problems. The three activity cards do prompt the inexperienced designers to use more problem structuring, and between the test and control groups this is a significant difference. Results are not entirely as expected as there is no significant difference between the number of problems identified by the test and control groups. This means that problem structuring does help with having a more rational approach and thorough reasoning, but not with identifying more problems.

As the results are significant we can reject the null hypothesis and accept the alternative hypothesis.

Option Generation

H0: There is no significant difference for option generation use between test and control groups.

H1: There is a significant difference for option generation use between test and control groups.

The hypothesis concerning the option generation reasoning technique was that the three activity cards would prompt designers to generate several options and discuss the alternatives before making a decision. While there is a noticeable difference in frequencies of both option generation used and options identified, between the student test and control groups this is not a significant difference. This means that the difference found could be due to chance as well.

As the results are not significant the null hypothesis cannot be rejected, and the alternative hypothesis cannot be accepted.

Constraint Analysis

H0: There is no significant difference for constraint analysis between the test and control groups.

H1: There is a significant difference for constraint analysis between the test and control groups.

The hypothesis concerning constraint analysis was that the constraint card would influence designers to become more aware of constraints. The results show that the card has no significant influence, and frequencies of constraint analysis used and constraints identified are nearly equal across both professional and student test and control group.

As the results are not significant the null hypothesis cannot be rejected, and the alternative hypothesis cannot be accepted.

Assumption Analysis

H0: There is no significant difference for assumption analysis between the test and control groups.

H1: There is a significant difference for assumption analysis between the test and control groups.

The hypothesis concerning assumption analysis was that the assumption card would influence designers to question premises and arguments during the design discussion. Results show that there is a significant difference in assumption analysis use and assumption identification by the test group.

As the results are significant we can reject the null hypothesis and accept the alternative hypothesis.

Risk Analysis

H0: There is no significant difference for risk analysis between the test and control groups.

H1: There is a significant difference for risk analysis between the test and control groups.

The hypothesis concerning risk analysis was that the risk card would influence designers to think more critically about their design and identify risks. Results show that there is a significant difference in the use of risk analysis and the amount of risks identified by the test and control groups.

As the results are significant we can reject the null hypothesis and accept the alternative hypothesis.

Trade-off Analysis

H0: There is no significant difference for trade-off analysis between the test and control groups.

H1: There is a significant difference for trade-off analysis between the test and control groups.

The hypothesis concerning trade-off analysis was that the trade-off card would influence designers to recognize trade-offs and discuss the pros and cons of design options. Results show that the trade-off analysis does have a significant difference between the test and control groups. The pros identified were not significantly different, but the cons identified were, showing that the test group focused more on critical thinking.

As the results are significant we can reject the null hypothesis and accept the alternative hypothesis.

Conclusion

The end results of the hypotheses show that many of the cards have been accepted as making a significant difference on the analysis techniques. These are problem structuring, assumption analysis, risk analysis and trade-off analysis. With only two hypothesis could the null hypothesis not be rejected, being option generation and constraint analysis. With this we can say that the card game does make a significant influence on the design reasoning of students.

In the next section we go further into analysing this data from a qualitative perspective to better understand what the groups are doing, and how the cards use reflection to influence design reasoning.

6.2.2 *Thorough Reasoning vs Satisficing Behaviour*

One of the first results is the difference in length of time that the groups took for their design session, with the control groups taking less time than the test groups. This is a marked difference that explains overall why the test group uses more reasoning techniques. The control group is more partial to satisficing behaviour. This is a common phenomenon where designers do not look exhaustively for every potential solution to a problem but go with the first solution that is satisfying, which is typical of natural decision making (Simon, 1978; Tang & van Vliet, 2015). This suggests that for the test groups, due to the cards, they were reminded to reason with the reasoning topics, and were encouraged to explore more about the design. The test groups were less easily satisfied with their decisions. We can see this difference in attitude in the transcripts themselves.

The test groups often mention how they have run out of time before they are completely satisfied with their design. As can be seen in the extract of T5 (Figure 14), a new design issue is mentioned, but there is no time to solve it.

With the control groups on the other hand, especially those which did not reach the two hour mark, they simply run out of issues to resolve. In the extract of C5 (Figure 14) they were touching on design issues that they needed to solve, but they convinced themselves that what they had was good enough (satisficing). They do not go further into detail to explore more about that decision but rather end the discussion.

The control groups are easier satisfied with their decisions and design, even when they have not reached the full two hours given. For the test groups the card game stimulates the designers to keep refining their design and consider their decisions, and often the time given is too short for these groups to fully explore the design.

6.2.3 *Reasoning with Risk, Assumption and Trade-off*

A main purpose of the reasoning card game is to prompt the students to consider design elements. The results of the experiment have shown that especially risks and assumptions were considered more by the test groups. Trade-off analysis did not show much difference, and constraint remained the same.

In many cases for the test groups the design scope appeared to be clear at first glance, but when they started using the cards and thought more about the design topics, they found that it became more complicated than first realized. Designers reflect on their previous ideas, discuss and redefine them, which clearly shows that the cards trigger reasoning in designers.

For the control groups considering assumptions and risks when making a decision about the design is clearly not at the forefront of their minds, as seen by their low unique element frequencies. With the test groups the cards remind the designers to take these considerations into account, as again can be seen in T3 where person 2 lists the cards and this prompts him to identify a risk (Figure 14).

For the trade-off analysis the combination of pro *and* con is rare, contributing to the low number of trade-offs. But comparing test and control groups does show that the test groups generate more pros, and especially more cons to argue about options. This suggests that the control groups are more concerned with validating their options, or arguing why they are good, instead of looking at potential problems. The test groups are more critical of their choices and look at options from different viewpoints. This is further supported by the fact that only the cons results are significant, meaning that the cards influence designers to be more critical. The extract of T4 shows part of a larger trade-off analysis in which several options are heavily discussed; mostly to have either a standalone program, or one which is cloud or web-based (Figure 14). In this part, person 1 mentions that a pro for a cloud based program would be that you can update every hour, but a con is that a strong server is necessary which would be costly. The person then proceeds to suggest another option, for the user to pay for the usage of the server. This is not well-received and person 1 admits that this option would still be a very expensive one and gives a pro to their first option, to use a local standalone version to which the others agree.

Even though the group eventually went with their first option, they took the time to explore multiple options and critically assess them by providing both pros and cons. The control groups do this at a much lower rate.

6.2.4 Reasoning with Design Context, Problems and Solutions

These results show that the effect of the card game was to combat satisficing behaviour and lack of design reasoning by stimulating the designers to reconsider their options and decisions, ultimately taking more time to delve into the issues. And yet, when we look at the problems and solutions identified, the percentage difference is much lower than that of the other elements and not significant. The options and problems structuring have a larger increase, and problem structuring shows a significant difference, so the cards do prompt designers to consider their problems and ways to solve it more thoroughly. But this does not make designers identify more problems in the same way that it does for the other techniques.

The amount of design problems being identified seems to be influenced by another factor besides the cards, the design strategy that the designers use. Design strategies such as problem-oriented, or solution-oriented can influence the information seeking behaviour of designers (Restrepo & Christiaans, 2004). The approach used for problem-solving, whether to focus on finding problems or solutions first, seems to have more of an impact on the design problems being identified. When comparing groups with similar strategies the cards influence becomes clearer.

As an example we have groups T2 and C1, whom both use a satisficing strategy, where they actively avoid going into more detail, rather choosing to view a problem as being outside of their scope. Their option generation and trade-off analysis results are very similar. But the problem structuring, risk, assumption and constraint analysis of T2 is at least double that of C1. Despite their adherence to a minimum satisficing strategy, the cards prompted T2 to recognize problems which often resulted from identified risks and constraints, for which they made assumptions to simplify the problem and solution.

It seems that the design strategy used by the groups could be a clearer indicator for how many and what kind of design problems are identified, while the cards influence how the designers solve these problems. The cards improve thorough reasoning, but this does not lead to more problem identification.

6.2.5 *Constraint Identification*

But what then about the one reasoning technique which the cards seem to have no influence on, either in frequencies or significance, the constraint analysis. The individual constraints are the exact same number in the results, showing no difference and the results are not surprisingly not significant. This result in itself is interesting considering the effect of the cards on the other reasoning techniques. The question here is why constraint analysis is different. One explanation for this is that the very nature of constraints, limitations on the design, is intrinsically bound to the requirements. When thinking about design and what the system must accomplish, it is natural to also think of what the limitations are, what is not required. As a result both test and control groups identify constraints as things that are not allowed or rules the system must follow. What is interesting here is that both groups identify much of the same constraints, with some coming directly from the requirements in the assignment, even taking on the same wording.

We find that both test and control groups frequently take over the literal requirements presented in the text as constraints. To give a more detailed representation of this, for the test groups there are 11 identified constraints which are shared in various degrees amongst the groups (Appendix G). There are 11 other constraints which they do not share and had to be inferred from the assignment, with 5 of these being identified by only one group. The control groups share 12 constraints from the text, and only 5 are other.

This then goes to explain the similar results when it comes to constraint analysis, many of them are found literally in the text of the assignment and require minimal effort to find. It is easy to see why these constraints would be in the text as requirements, as it is to the clients benefit to give clear instructions on what the program should do, and what they do not want the program to do. This means that constraints are easier to identify, causing the cards to have no influence, because these are given as requirements. The other techniques, such as assumptions and risks, must all be inferred from the text and are not clearly given. The effect of the cards is more obviously shown there.

T5 (1:52:06-1:52:15)
PERSON 2: So we have we got everything. I think maybe only the traffic light is not taken into account and that's connected to intersection.
PERSON 1: Yeah. Definitely need to be there just make it here. And do we also model dependencies.
PERSON 2: Okay I think we don't have the time to put in. Maybe we can sketch it.

C5 (1:16:38 – 1:17:19)
PERSON 2: Oh ok. Do we have to say something more? Are we done actually? Or do they actually also wanna know how we include the notation and such, because-
PERSON 1: No they also get the documents, so they can see
PERSON 2: Yeah ok, but maybe how we come up with the- I don't know. No? isn't necessary?
PERSON 3: Mm
PERSON 1: It's just use UML notation, for all
PERSON 2: For all?
PERSON 1: No, and lifecycle model, and petri net. No, no petri net
PERSON 2: Perhaps petri net. Ok, shall we- shall I just?
PERSON 1: Yeah
PERSON 2: Ok

T3 (0:20:31-0:21:10)
PERSON 2: HTML 5 yeah? Information would of course [inaudible] *constraints or risk or trade-offs,* we have to make- a risk might be of course that- of course there is a [inaudible] so while you are travelling. For example, when you have an older device that could be a problem of course. So then you couldn't use the navigation maybe, the- well, [inaudible] right?
PERSON 1: What do you mean exactly? For example.
PERSON 2: Yeah well, for example, if you are travelling and you want to use the application. You want to use the traffic simulator, then of course that might be the case that your device is not suitable for it. For example. So, on the other hand-

T2 (0:28:14-0:28:28)
PERSON 1: So this was a problem
PERSON 3: This was a problem
PERSON 1: Yeah
PERSON 2: Yeah. Because [inaudible]
PERSON 1: And a risk right
PERSON 2: A constraint? Yeah but it was also like an assumption that you have a minimum length. That is our assumption right or-
PERSON 3: Yeah we created that now, and that's ok because it's our own system

T4 (1:25:13-1:26:05)
PERSON 1: So that's the trade-off. The other side is good to have in the cloud because you can *easily push a new update every* hour if you want but you *need really really strong server for all this simulations. Now professor did not say how much money she has.* So it can be also. There can be also an option to pay for usage of this server for every simulation or for every hour of simulation.
PERSON 2: I don't think so.
PERSON 1: There can be an option. But it *can be also very expensive* so when I think about everything I think that is *cheaper and easier to have local stand-alone version.*
PERSON 2: Yeah.
PERSON 3: Yeah.

Figure 14 – Students Transcript Extracts

6.2.6 How Cards Influence Design Discourse

There are 3 reflective periods that are mandated in the test groups. They are at 15, 45 and 1 hour and 45 min into the design session. During these reflective periods the designers must actively play the card game and lay down cards. In this section we examine how the test group members carry out design discourse and how playing the cards influences the design discourse that follows. We focus specifically on language shift and discourse structure.

Reflection Periods – Language Shift

With the test groups we see that especially the first reflection period corresponds with a strong language shift in the design session. Though three reflective periods were prescribed by the assignment, not everyone held to this. Group T2 and T3 have only two reflective periods, while group T4 has four.

There are two groups who use the terms throughout the design session, corresponding with playing or mentioning the card game early in the session. These are groups T6 and T1 and as a result their language use and use of reasoning techniques does not show a clear shift around the reflective period mark. Their mention of techniques and usage is more evenly spread out throughout the design session.

The other four groups do show a dramatic language shift when they start their first reflective period, which results in the use of reasoning techniques which had not been mentioned beforehand. Especially risks and assumptions are used more after the card game, which corresponds with the test groups using more of these techniques.

Group T2. Before the first reflective period group T2 does not mention any of the analysis terms, though all analysis techniques are used at least once before the card game. Before the card game rules is mentioned as a synonym for constraint. As a result most of the constraint analysis occurs before the first card game (Table 11).

Analysis occurrence	Before first card game	After first card game
Constraint analysis	6	3
Risk analysis	2	5
Assumption analysis	1	5
Trade-off analysis	1	1

Table 11 - Analysis Occurrence - Group T2

Group T3. With group T3 only assumption is mentioned before the first card game, the rest occur afterwards. Corresponding with this, before the card game the only analysis technique used is the assumption analysis. All other analysis techniques are used after the first card game, showing a clear change brought on by the card game (Table 12).

Analysis occurrence	Before first card game	After first card game
Constraint analysis	0	2
Risk analysis	0	6
Assumption analysis	4	5
Trade-off analysis	0	2

Table 12 - Analysis Occurrence - Group T3

Group T4. None of the terms are mentioned before the first reflection period, though rules are used as constraint synonyms. Before the card game only assumption is not used (Table 13). There is no mention of risk throughout the session, though these are identified mainly after the card game. This might be because one of the designers has experience in creating a traffic simulator and therefore more easily identifies problematic issues and sees no need to play a specific card for them.

Analysis occurrence	Before first card game	After first card game
Constraint analysis	2	5
Risk analysis	1	4
Assumption analysis	0	5
Trade-off analysis	1	3

Table 13 - Analysis Occurrence - Group T4

Group T5. Before the first reflective period the only term mentioned is constraint, and a few of these are identified. Some assumptions are also considered, these are mainly connected with a discussion about the user goals (Table 14). Risk and trade-off analysis only occur after the first card game.

Analysis occurrence	Before first card game	After first card game
Constraint analysis	2	6
Risk analysis	0	5
Assumption analysis	3	5
Trade-off analysis	0	2

Table 14 - Analysis Occurrence - Group T5

This shows that for inexperienced designers using the analysis techniques is not a natural part of their reasoning. Only with the use of the reflection periods, where they must work with the cards and the reasoning techniques, do we see that their reasoning starts to include the analysis techniques. The cards encourage reasoning through reflection. In order to find the exact type of reflection being used we take a closer look at the structure of the discourse triggered by the reflection periods.

Reflection Periods – Discourse Structure

The card game clearly influences the reasoning of designers, and it does this through the use of reflection. We have discussed earlier that there are two types of reflection; reflection-in-action and reflection-on-action (Schön, 1983). Reflection-in-action is to think about your actions as you are taking them, and what to do next in a project. Reflection-on-action is retrospectively looking back on a finished project to learn what could have been done better. The student experiment in this case falls under reflection-in-action, but we find that these same reflection strategies are also used within the experiment, prompted by the cards themselves.

There are two ways in which the cards directly influence design discourse. First, the cards provide inspiration for students to investigate a certain topic, e.g. reminding them to consider assumptions. This is similar to the concept of reflection-in-action as the designers consider what to do next. Secondly, the students used the cards to reassess their previous discussion by classifying it in card terms, e.g. a system rule is later identified as having been a constraint. This is similar to reflection-on-action and retrospective in nature, but the term reflection-on-action is used specifically for reflection on a finished project. Therefore these two reflection strategies used within the experiment will be called inspiration and classification.

With the extract from T3 we see how these cards are used for inspiration (Figure 14). Person 2 is looking over the cards looking for issues to discuss and comes up with a risk, which needs to be clarified for the other person. This risk makes the designers reconsider an earlier assumption that they had made, that the program is a web-based application, which later turns into a nearly 5 minute long trade-off discussion.

With the extract from T2 we can see the cards being used for classification (Figure 14). Here they have just discussed a problem and found a solution for it. But when reassessing the discussion as a problem, they realize that in order to solve the problem, they also identified risks and used assumptions.

Especially this second use of the cards as classifying tools is interesting as it show the influence of reflection. In this way the cards are being used for their intended purpose, to help designers evaluate on their decisions, and thereby improve that reasoning.

Reflection periods	**T1**	**T2**	**T3**	**T4**	**T5**	**T6**
Period 1	Inspiration	Inspiration, classification	Classification	Classification, Inspiration	Classification, inspiration	Classification, inspiration
Period 2	Inspiration, classification	Classification	Classification	Inspiration, classification	Classification, inspiration	Classification
Period 3	Classification			Classification	Classification, inspiration	Classification
Period 4				Classification, inspiration		

Table 15 - Student Reflection Card Uses

The effect of the cards is most clear during the reflection period. The usual pattern seen by the use of reflection periods is that card play spikes during these times and then tapers off, to spike again with the next reflection period. Reflection periods can be seen as distinct periods in the card play timeline.

In Table 15 the different card uses are shown per test group. We see here that classifying is the most common use of the cards. This classifying can occur as a short summary at the beginning of a reflection period, or be stated shortly after a statement has been made, e.g. what you just said is a constraint. This shows that the cards have their intended effect, to increase reasoning through reflection.

6.2.7 Comparison Reflective Experiments

The card game is based on a previous experiment involving reflection in the form of reasoning techniques to increase the quality of the design discourse (Razavian et al., 2016). In this experiment a lecturer was placed with student designers to prompt them with reflective questions. In the experiment laid out in this thesis the lecturer has been replaced by the card game. The reasoning behind this was that the lecturer as an external factor could unintentionally influence the group of designers by the timing of when they ask their questions.

In the previous experiment the data was similarly qualitatively analysed at the hand of several transcript coding's, but a different coding scheme was used. These involved the design reasoning process (Mind 1) which was composed of design issues, options, decisions and context, and reflection (Mind 2) which was categorized as external, an outside source, and internal, internal awareness. The quality of the design discourse was measured into 5 levels based on the presence of completeness, accuracy and relevance. The different coding for Mind 1 and 2 would be graded according to these levels. They did not look at the influence of the separate reasoning techniques.

Because the experiments were judged in a different manner we cannot easily compare the two as the previous experiment looked at the quality of the categories given, instead of their frequencies and overall influence on reasoning. But we can look at the quality levels as signs of more thorough reasoning. The previous reflective experiments showed that the lecturer and reflective questions did have an effect on the quality of the discourse. Reflection helped students to be clearer and more precise with their design discourse, the questions given made students reconsider their previous decisions. This same pattern of behaviour we have also found in our recent experiments, this suggests that the lecturer can easily be replaced by cards to get the same effect.

6. 3. Key Findings

To summarize the analysis results of the student experiments the key findings are listed shortly.

1. **Most cards significantly influence design reasoning**

Using quantitative tests we find that most of the cards have a significant influence on the design reasoning of designers (Table 10). The three activity cards (context, problem, solution) have a significant influence on problem structuring, while the assumption, risk and trade-off cards influence their respective analysis techniques. The three activity cards did not influence option generation, and the constraint card also showed no significant difference on constraint analysis. With this we can state that most of the hypotheses regarding the influence of the cards on design reasoning are accepted.

2. **Cards lead to more use of reasoning techniques**

Results of the student experiments corroborate with the hypothesis and shows that test groups made 75% more use of reasoning techniques than the control groups (Table 5). Test groups also showed a clear language shift when they started using the cards as they became more familiar with the reasoning concepts.

3. **Cards combat satisficing behaviour**

Test groups were less easily satisfied with their decisions as they were reminded to reason with the reasoning topics, and were encouraged to explore more about the design. The control groups showed the more natural pattern of satisficing behaviour which can be seen by lesser use of the reasoning techniques, and that some groups did not take advantage of the full 2 hours they were provided with (Table 4).

4. **Problem identification dependent on design strategy**

The test group identifies more design problems than the control group, but the percentage difference is much lower than found with the reasoning techniques and not significant. The reason for this can lie in the strategy the groups use. Depending on the kind of strategy more problems will be identified, which in combination with the cards can be enhanced.

5. No noticeable difference in constraint identification

While the reasoning techniques are used more by the test groups, there is one odd one out. The identification of constraints by both test and control groups is equal and does not seem influenced by use of the cards. There are two possible explanation for this. First is that constraints by their very nature are intrinsically bound to requirements. As requirements are what should be in the design, it is natural to also consider what should not be in the design. A second explanation is that the Irvine assignment has an influence as it explicitly states constraints in the form of requirements. To support this, most of the constraints identified by the groups are those which can be found in the assignment text itself (Appendix G).

6. Cards trigger reflection

During the reflection periods the cards are played by the test groups in a distinct pattern, with card play spiking and then tapering off. During these periods clear reflective thinking is used in the form of reflective strategies; inspiration and categorization (Table 15).

7. PROFESSIONAL EXPERIMENTS

The professional experiment involved 5 groups of 2 professionals each. Of these groups 3 were used as control groups, these groups were transcripts from the Irvine experiment (UCI, 2010). For the test groups, 2 groups were selected from different IT companies in Australia to use the card game during a design session (Table 16).

Professional Control Group	Professional Test Group
PC1, PC2, PC3	PT1, PT2

Table 16 - Professional Control and Test Group Distributions

The design session of the professional test groups were recorded and transcribed, in these sessions the designers did not have to log their own use of the cards, but a third person observing the session noted the cards being used.

Coding and analyses was applied to all 5 groups in the same manner as was done with the student groups (Appendix F).

7. 1. RESULTS

As with the student experiment the main purpose of the experiments is to find out if the card game has any obvious influence on the reasoning process of designers. In this case it involves experienced designers who it has been theorized would derive less benefit of using the card game as they have more experience and are therefore more familiar with the concepts the cards represent.

Unlike the student groups, the professional groups are not tested for significance as several assumptions are violated. With some reasoning techniques the Levene's test for equality of variances is not passed, but a greater obstacle is that the sample size of two for the test group is not large enough to meaningfully test for normality with Shapiro Wilk (Appendix K). With the small sample size the possibility of bias and inaccuracy in the results is great; especially considering the test group results are very diverse (Field, 2013).

As a result, for the analysis of the professional results we focus on the qualitative perspective only. No conclusions can be made about the significant difference between the test and control groups. In order to compare the test and control groups averages are used because there is an unequal number of groups, i.e. 3 control groups and 2 test groups.

7.1.1 DESIGN SESSION LENGTH

Looking at the amount of time the groups took for the design session we see that almost all groups took advantage of the 2 hours they were given. Only one of the control groups did not use the full 2 hours and instead used only 1 hour (Table 17).

Control Group	Time	Test Group	Time
PC1	0:59:20	PT1	1:49:45
PC2	1:53:34	PT2	2:00:31
PC3	1:52:46		

Table 17 - Professional Design Session Times

7.1.2 CARD GAME INFLUENCE

In order to find any noticeable differences between the test and control groups in the use of the reasoning techniques, first the frequencies are measured. And because we are dealing with an uneven amount of groups these are then turned into averages which will be used for comparisons (Table 18).

The results here show that there is a noticeable difference in the use of most reasoning techniques. Especially assumption analysis and trade-off analysis are used more by the test groups, followed by option generation, risk analysis and constraint analysis. Problem structuring shows no noticeable difference.

The total average difference shows a 57% increase of reasoning technique use by the test groups. Most notable here though is that while the control group's performance is mostly equal in frequency, those of the test groups are varied. PT1 makes much more use of the reasoning techniques than PT2, which has a result more similar to that of the control groups.

Analysis techniques	**PT1**	**PT2**		**Total**	**Average**
Assumption analysis	10	14		**24**	**12**
Constraint analysis	12	7		**19**	**9.5**
Risk analysis	16	6		**22**	**11**
Trade-off analysis	6	2		**8**	**4**
Option generation	14	6		**20**	**10**
Problem structuring	22	19		**41**	**20.5**
Total	**80**	**54**		**134**	**67**

Analysis techniques	**PC1**	**PC2**	**PC3**	**Total**	**Average**
Assumption analysis	5	6	2	**13**	**4.3**
Constraint analysis	5	6	9	**20**	**6.7**
Risk analysis	3	12	3	**18**	**6**
Trade-off analysis	1	3	0	**4**	**1.3**
Option generation	4	6	6	**16**	**5.3**
Problem structuring	18	19	20	**57**	**19**
Total	**36**	**52**	**40**	**128**	**42.7**

Table 18 - Professional Design Reasoning Technique Frequencies and Averages

Especially assumption and risk analysis show great differences between the control and test groups, while constraint analysis has a lower degree of difference. But in the analysis techniques the same element can be discussed multiple times, so we must look at the amount of unique design reasoning elements that the groups found (Table 19). The unique values of the elements can be seen in the dark blue columns. The total average of the unique elements shows a 56% increase by the test groups.

When we look at the individual elements we see that in the case of constraints, that unlike what the constraint analysis showed, there is actually very little difference between the test and control groups. While for the assumptions and risks the increase in use by the test groups remains.

Design reasoning elements	**PT1**		**PT2**				**Total Unique**	**Unique Average**
Assumption	11	10	16	9			**19**	**9.5**
Constraint	15	9	10	6			**15**	**7.5**
Risk	21	13	6	5			**18**	**9**

Design reasoning elements	**PC1**		**PC2**		**PC3**		**Total Unique**	**Unique Average**
Assumption	6	6	8	6	2	2	**14**	**4.7**
Constraint	7	7	7	6	14	10	**23**	**7.7**
Risk	3	2	12	8	3	3	**13**	**4.3**

Table 19 - Professional Design Reasoning Elements Frequencies and Averages

The trade-off analysis shows a noticeable increase in use by the test groups. To better understand these results we need to look at the elements which make up a trade-off, the pros and cons (Table 20).

Here we see that not all pros and cons were used in combination to make a trade-off analysis. To argue about their cons the test group used more pros with a difference of 96%, nearly double that of the control group, and more cons with 179%. The biggest difference though remains the trade-off analysis with 208% more use by the test groups.

	PT1	**PT2**		**Total**	**Average**
Trade-off analysis	6	2		**8**	**4**
Pros	9	4		**13**	**6.5**
Cons	19	5		**24**	**12**

	PC1	**PC2**	**PC3**	**Total**	**Average**
Trade-off analysis	1	3	0	**4**	**1.3**
Pros	3	7	0	**10**	**3.3**
Cons	1	5	7	**13**	**4.3**

Table 20 - Professional Trade-off analysis, pros and cons

With the design issues, options and decisions we see that more design options are generated by the test group, with 55% more (Table 21). The design issues and decisions are also used identified more by the test group, but in a lesser degree, with decisions increasing with 15%, and issues increasing with 38%.

	PT1	**PT2**		**Total**	**Average**
Design issues	21	13		**34**	**17**
Design options	43	18		**61**	**30.5**
Design decisions	15	12		**27**	**13.5**

	PC1	**PC2**	**PC3**	**Total**	**Average**
Design issues	9	13	15	**37**	**12.3**
Design options	11	23	25	**59**	**19.7**
Design decisions	8	12	15	**35**	**11.7**

Table 21 - Professional Design issues, options and decisions

7.1.3 *Card Influence on Language*

To determine the influence of the cards on the design reasoning process we also look at their influence on the language used in the design sessions (Table 22). As is immediately apparent the test groups make more use of the card terms, though with PT2 this is with a lesser amount for risk and trade-off. The control groups shows almost no use of the card terms, though in the case of PC2 a synonym for assumption is used, this being 'assume'.

Term used	**PT1**	**PT2**	
Assumption	36	42	
Constraint	15	17	
Risk	27	4	
Trade-off	22	5	
Term used	**PC1**	**PC2**	**PC3**
Assumption	6	0	1
Constraint	0	0	8
Risk	0	0	0
Trade-off	0	0	0
Assume	3	16	1

Table 22 - Professional Analysis Technique Term Mention

7. 2. Analysis

The results of the analysis of the professional groups are very similar to those of the student groups in terms of the increase in use of reasoning techniques. Though in most cases this is of a lesser degree, supporting the idea that experienced designers would be less influenced by the cards then inexperienced designers. In this section we go further into the details about the results and what they mean.

7.2.1 *Thorough Reasoning vs Satisficing*

With regards to use of the 2 hour design session we see, as with the student groups, that one of the control groups did not take full advantage of this, only using 1 hour. The test groups also used more reasoning techniques than the control groups and this suggests that the cards encourage more thorough reasoning.

As I mentioned before, there is a marked difference between the two test groups. One used far more cards than the other group, PT1 using 48% more reasoning techniques than PT2. The results of PT2 were also more similar to those of the control groups, except in the case of assumption analysis. It is therefore of interest to look at what could explain these differences, as they could also explain the results of the control groups.

First of all, there is a clear difference in the frequency of card play by the two groups (Table 23). Group PT1 used the cards the most with 67 cards played. Group PT2 used the cards less with 20 cards played.

	PT1	**PT2**
Cards Played	67	20

Table 23 - Professional Frequency Cards Played

Both groups used the cards mostly around the reflection periods, but PT1 played many cards in quick succession and for a longer time than PT2. As an example we see in extract 1 of Figure 15 that PT1 is being reminded that it's time for the reflection period and within one minute they play 5 cards, which have been bolded. In the 7 minutes following they play an additional 14 cards, making that 19 cards in total. PT2 on the other hand needs some reminding to play cards in the reflection period as can be seen in extract 2 (Figure 15). In the following 3 minutes they play 7 cards before stopping.

The amount of reasoning techniques used seems to have a connection to the amount of cards played. A large amount of cards needs to be played before the cards show their effect and more thorough reasoning is applied, combating satisficing behaviour. And this in turn is connected to how much the designers adopt the card game into their own method and its fit to their usual style of designing.

The reason for this easy adoption of the cards might lie in how the two companies approach design habitually, i.e. their usual way of designing. As can be seen in Table 24, the company profiles of both test groups are given. PT1 is a larger company than PT2 and operates worldwide to make products commissioned by various companies, their largest client being in the UK. PT2 is a smaller company which develops products for the domestic market and focusses of municipalities. The experience of the designers is comparably equal.

	PT1	**PT2**
Company No. Employees	60 employees	5 employees
Experience	7 to 10 years	7 to 8 years
Business Focus	Build IT systems for financial clients, manages software	Develop applications for municipalities and building industry
Markets	International, UK	Domestic market

Table 24 - Professional Company Profiles

With PT1 the international structure and larger company might mean that designers are asked to work with different methods in rotating teams, making them more flexible in adjusting to a new method such as the cards.

In the case of experienced designers we propose that the more the cards are used during the design session, and the more easily they are adopted as part of the discussion, the more reasoning techniques are used and satisficing behaviour is discouraged. If there is only a limited use of the cards, the effect will not be as pronounced in encouraging thorough reasoning.

7.2.2 Reasoning with Risk, Assessment and Trade-off

Throughout the design session the cards especially triggered the use of risk, assessment and trade-off analysis as each were used at least a 100% more by the test groups than the control groups.

Both of the test groups have larger frequencies of assumptions, while PT1 identified far more risks, and PT2 is slightly higher than the control groups. Only PC2 has more risks identified than the other control groups, which arises from their concern about their lack of domain knowledge and understanding and is a major focus of their session as can be seen in extract 3 (Figure 15).

The fact that PT2 shows a larger increase in assumptions as well could be because this group played the assumption card the most (Table 25). All other cards were of much lower frequency. With PT1 we see that they played most cards at least 10 times, with the exception of context and solution. This supports the theory that experienced designers benefit the most when they play a lot of cards.

With trade-off analysis there is a definite increase in use overall, but again this is mostly due to the contribution of PT1. The individual pros and cons are used more often than the trade-off combination of pro *and* con. We see here that PT2 has similar results to the control groups, while PT1 shows double their use, especially in using cons. This could also be attributed to the amount of cards played by each group as PT1 plays 10 trade-off cards, while PT2 only plays 1.

Frequencies cards played	**PT1**	**PT2**
Context	1	3
Problem	11	3
Solution	4	1
Assumption	17	8
Constraint	10	2
Risk	14	2
Trade-off	10	1
Total	**67**	**20**

Table 25 - Professional Test Group Card Play Frequencies

7.2.3 Reasoning with Design Context, Problems and Solutions

When we look at the results of the design issues, options and decisions we find that the individual options have increased by more than 50%, which corresponds with the increase in option analysis, but the design issues and decisions do not show the same difference with identified issues only differing 38%. These results are similar to the student experiment, and the possible explanation can also be the same. Though the cards have some effect on identifying design issues and decisions, they are also affected by the design strategy the group uses.

Unlike with the student experiments, we cannot compare two test and control groups using the same strategies to identify the difference a design strategy makes, but there are groups using opposing strategies. In the case of PT1 which finds the most problems, they focus on a problem-oriented approach finding problems which need to be solved for the design in the first hour, and then focus on solutions in the second hour as can be seen in extract 1 (Figure 15). This is a strategy that identifies many problems, but as they are not immediately solved, the solution frequency is lower. PC1 has the lowest number of identified issues and decisions. This can be explained by the fact that they have a solution-oriented approach and make an assumption about the problem in the first 10 minutes of the design session. As can be seen in extract 4 they state that it is a classic problem for which a controller type pattern can be used to solve the issue (Figure 15). PC1 is also the only group not to make full use of the 2 hour design session.

A solution-oriented approach limits designers in how much information they seek out, as they start working with the preferred solution early on, while a problem-oriented approach makes designers more willing to go deeper into issues and find information (Restrepo & Christiaans, 2004). This difference in strategy the can explain the decreased difference in finding issues and decisions between the test and control groups.

7.2.4 Constraints and Problem structuring Identification

With constraint identification we see similar results as with the student experiment, the constraint card shows no influence on reasoning. Again there is no noticeable difference between the constraint analysis, and the unique constraints even show a slight decrease in constraints by the test groups. The same possible explanation for why the card has no influence can be given here as well. Constraint may be an intrinsic part of identifying requirements. And both control and test groups identify most of the same constraints, which have been literally taken from the assignment text (Appendix H). The control groups share 9 of the same constraints, with only 3 unique ones which do not explicitly come from the assignment. The test groups share 7 of the same constraints and 5 unique others. As constraints can be found with minimal effort, the card does not have an effect on reasoning.

Further similar results we see is that there is no noticeable difference between the test and control groups concerning problem structuring analysis, all groups use around 20 instances of problem structuring. There is no specific card that represents problem structuring, meaning that the cards as a whole do not influence more reasoning in this case. Problem structuring is when designers explore the problem space by asking questions related to the problem or to resolve uncertainty and confusion about the requirements. The reason for the similar results between groups is not as obvious as with the constraints as the questions discussed during problem structuring are different with each group. A probable explanation for the similarities could be because the experiment involves experienced designers. As experienced designers they have more knowledge about the process of problem-solving that makes up software design, and more general knowledge about the possibilities for solving a problem. They might also be more practiced in knowing which questions should be asked and what remains unclear for the design, and to focus on these points of uncertainty in order to resolve them.

Extract 1 - PT1 (0:45:10-0:46:49)
Instructor: It's now 45 minutes
PERSON 1: Ok, what does it mean, more cards. [inaudible] design decision, how many decisions have we actually made or have we just been talking about the problem.
PERSON 2: Talking a lot. We need to start trying to chase into where we actually want to go for- so I guess we've been talking a lot about the **problem** working out all the different bits and pieces of it and stuff. And we found a lot of different parts of it and there's a lot of **risks** as to, you know, that go of the **trade-off** as to how complex we want to make this and how difficult and stuff like that. Obviously need to make sure we don't run out of time as well. And obviously a lot of **assumptions**, where we've got things to do with how we're going to represent them and what distances mean and the data structure and so forth.
PERSON 1: Ok. We've focused mostly on data model. Would you put that as a problem or solution, I'd be- it's somewhere in between, I'd be more **solution** than
PERSON 2: We're verging on a solution so yeah, so

Extract 2 - PT2 (0:45:02-0:45:15)
Instructor: It's now 45 minutes
PERSON 1: Yep
PERSON 2: Ok
PERSON 1: So are we supposed to do anything? At 45 minutes?
Instructor: It's the reflection period, so you're supposed to use the cards.
PERSON 1: Ok

Extract 3 - PC2 (0:22:46- 0:23:43)
Male: Yeah, I think we're going to have to rely on professor E for creating the details about the theory of how that works. I mean, we might have to code whatever her understanding is of traffic light theory into it, right? But we don't have enough information here. There might be packages out there, I don't know. Yeah, I feel like this stuff over here I kind of eventually we'll want to build it in to our diagram to say how they're all related. We could do that.

Extract 4 - PC1 (0:07:10-0:08:02)
Male 1: Cool, cool. Sounds good. Kind of teasing out some of these desired outcomes. It actually seems like a classic, a pretty classic software design issue where we want to separate the UI off and we want the underlying data structures and supporting things. So I think we can make that first clean division, you know I think about this as a software application. Looks totally like you want to pull out some kind of patterns and [inaudible] controller type of thing. And so if we extract the UI piece first, and then let's focus on kind of the underlying stuff in order to support you know kind of the traffic flow.

Figure 15 - Professional Transcript Extracts (bold is card play)

7.2.5 *How Cards Influence Design Discourse*

There are 3 reflective periods that are mandated in the test groups. They are at 15, 45 and 1 hour and 45 min into the design session. During these reflective periods the designers must actively play the card game and lay down cards. In this section we examine how the test group members carry out design discourse and how the playing of cards influence the design discourse that follows. We focus specifically on language shift and discourse structure.

Reflective Periods – Language Shift

We see that the influence of the cards on the language used during the discussion is a great one. Even though these experiments involve experienced designers who are aware of these analysis techniques and use them in their projects, the control group shows that the terms themselves are of part of their normal vocabulary.

The test groups on the other hand show a great influence, especially with the term 'assumption', while 'risk' and 'trade-off' are used far less by PT2. The term 'constraint' has a similar frequency. This corresponds with the results of the reasoning techniques. PT2 has lower values than PT1, except with assumptions, which is the card the group plays the most as well. While constraints show no influence by the cards.

PT1 uses cards already at the beginning of the session so there is no clear language shift as they mention assumption, risk and trade-off, and also use 9 reasoning analysis techniques. During the reflection period more cards are used, but before the first reflection period 5 cards have already been played and there are 9 occurrences of reasoning techniques. This is the effect of their easy adoption of the card game method into their discussion.

With PT2 we see a more clear language shift as before the first card game only assumption is mentioned as a term and 2 cards are played; problem and assumption. As a result there are only 3 occurrences of the reasoning techniques being used before the card game (Table 26).

Analysis occurrence	Before first card game	After first card game
Constraint analysis	0	7
Risk analysis	1	5
Assumption analysis	1	13
Trade-off analysis	1	1

Table 26 - Analysis Occurrence - Group PT2

What we see here again reaffirms the previous results that thorough reasoning for experienced designers depends on in how far they can adopt the card game into their discussion, and how many cards they play. There is a clear language shift for PT2 as the cards encourage more reflective thinking, but most effective is to use the cards more naturally within the discussion.

Reflective Periods – Discourse Structure

There is a clear difference in card use and influence through reflection with the two test groups. We look further into the specific structure of the discussions prompted by the reflection periods here. We see the same reflection techniques, inspiration and classification, being used with the cards as we saw with the student groups.

During the reflection periods both groups make use of the classifying techniques, and also use this to make a small summary of what they were doing (Table 27). PT1 makes more use of the cards as inspiration for finding more issues. During the design session both groups show spikes in card play around the reflection periods. The difference is that the duration in which the cards are then used is longer with PT1 than with PT2. Around the 1 hour and 45 min mark PT1 plays 19 cards in 8 min, while PT2 plays 7 cards in 4 min before they stop.

Reflection periods	PT1	PT2
Period 1	Classification, inspiration	Classification
Period 2	Classification, inspiration	Classification, inspiration
Period 3	Classification, inspiration	Classification

Table 27 - Professional Reflection Card Uses

The reflection periods both show that reflection is being applied through the cards, but that the effectiveness of this is not equal as PT1 simply uses reflection for a longer period of time.

7. 3. KEY FINDINGS

To summarize the analysis results of the professional experiments the key findings are listed shortly.

1. **A minimum amount of cards lead to more reasoning techniques**

There is a clear difference in the performance of the two test groups where PT1 has the greatest scores while PT2 has scores similar to that of the control groups (Table 18). PT1 also played 67 cards while PT2 played 20 (Table 23). For experienced designers there seems to be a minimum amount of cards that need to be played before greater effects are shown.

2. **Card game adoption combats satisficing behaviour**

Both of the test groups made full use of the 2 hour design session, while one of the control groups only used 1 hour (Table 17). But PT1 used the cards more during their discussion than PT2, resulting in more thorough reasoning. A probable explanation for this is that PT1 more easily adopted the new card method as they are a larger and more global oriented company (Table 24). Their experience with working with different methods could have made the adoption of the card game simpler.

3. **Problem identification dependent on design strategy**

Similar to the student experiments the test groups found more design issues, but at a slighter decrease than the other reasoning techniques (Table 21). The possible explanation for this is again that the design strategy used by the group is more telling than the cards played for how many problems are found. This is seen with PT1 who found the most problems using a problem-oriented approach, while PC1, who found the least problems used a solution-oriented approach.

4. **No noticeable difference in constraint and problem structuring identification**

Similar to the student experiments the constraints again show no influence by the cards. This is because the constraints are intertwined with the requirements and are easily found in the assignment text (Appendix H). Unique to the professional experiment is that there is no noticeable difference between problem structuring frequencies either (Table 18). A possible explanation could be that as experienced designers they are more prepared in knowing what questions to ask, and when to focus on solving the problem.

5. **Cards trigger reflection**

As with the student experiments we see the same pattern regarding the use of cards during the reflection period. There is a definite spike of card play in the form of two reflective strategies; inspiration and categorization (Table 27).

8. Comparison Analysis Results

Here we discuss the analysis results of the student and professional groups and compare them. Before this comparison it is necessary to repeat the differences in methods used for the student and professional experiments. Both the student and professional groups made use of the Irvine assignment, but in the case of the students this assignment was adapted to fit the Software Architecture course they participated in. This means that the desired outcomes focused on three viewpoints; context, functional and information, and documentation of the architecture needed to be handed in. In addition the student groups had to maintain their own log of the cards they played.

The professional groups used the original Irvine assignment and were given the same card game instructions as the student group. As part of the original assignment they needed to explain their architecture to the developers, instead of handing over documentation. The cards played were logged by an observer.

As with the other analysis, the experiments are based on a limited amount of tests and no conclusive results can be given, the following discussion presents several hypotheses to explain the results. As the professional results could not be tested for significance, the purpose of this comparison is to qualitatively explain the differences and similarities in approach that both student and professional groups have.

8. 1. Thorough Reasoning vs Satisficing Behaviour

When it comes to thorough reasoning, we see that the cards have a noticeable effect with both professional and student groups. This effect is larger with the student groups with a difference of 75% (Table 5), while the professional groups have 57% (Table 18). We find that the control groups of both experiments are more prone to satisficing behaviour with some groups not taking full advantage of the time given to complete the design. But professionals are less likely to satisficing behaviour than the students as well, as we can see with the results of the professional and student groups (Table 28). For the student groups the averages were added, and the standard deviation of the professional and student groups was also given in order to compare the results. When we look at the average results there is an obvious ranking where the test groups perform better than the control groups.

1. Professional test group
2. Student test group
3. Professional control group
4. Student control group

These results are also visualized in boxplots to better see the difference between the groups and their variance (Appendix I). When it comes to thorough reasoning and the use of reasoning techniques the student control group has the lowest results. The professional control group has results which are closer to that of the student test group in some cases, showing that satisficing behaviour is something that they use less, most likely because of their experience.

Comparing the professional and student test groups shows that the professional group has higher results, except for problem structuring which students perform more. As problem structuring aims to identify what problems need solving it makes sense that more experienced designers may have better insight than this and would not need as much of this as students. Interesting here is that risk analysis is used 93% more by the professional group then the student group. The other techniques do not show this amount of increase. The risk analysis of the professional control group is also very similar to the student test group. This suggests that one of the main benefits of experience is that identifying risks becomes easier. Though it should also be noted that the variance here is much greater than with the student groups, meaning that the professional groups have a wider range of frequencies, this is mostly caused by PC2 and PT1 who both identify more risks than the other groups. When we look at the boxplots of the reasoning techniques we see that especially with assumption analysis there is a marked difference between control and test groups (Appendix I - AA). There is only a slight overlap between the professional control group and the student test group. This could mean that especially with assumption analysis the cards have a noticeable effect on the designers. With other boxplots like risk analysis (RA), trade-off analysis (TA), assumption (A), risk (R) and cons we see the same ranking occurring as with the average results. Exceptions are constraint analysis (CA), option generation (OG), problem structuring (PROB), constraint (C), pros, design issues (DI), design options (DO) and design decision (DD).

When it comes to satisficing behaviour, the student control groups perform this more than any other group. This might be the reason why the cards have a more pronounced effect on the student group than seen with the professional group. As a result, to move students to more thorough reasoning is easier, because they start from a lower amount of reasoning. While with professionals this is harder as their experience already prompt them to use reasoning. To increase this reasoning a minimum amount of cards needs to be played as shown by PT1, and the cards have to be adopted to be naturally used during discussion.

Analysis techniques	**Professional Test Total**	**Professional Test Average**	**Std.Dev**	**Student Test Total**	**Student Test Average**	**Std.Dev**
Assumption analysis	24	**12**	2.828	49	**8.2**	3.189
Constraint analysis	19	**9.5**	3.536	43	**7.2**	2.787
Risk analysis	22	**11**	7.071	34	**5.7**	.816
Trade-off analysis	8	**4**	2.828	16	**2.7**	1.506
Option generation	20	**10**	5.657	52	**8.7**	5.854
Problem structuring	41	**20.5**	2.121	146	**24.3**	4.967
Total	134	**67**		340	**56.7**	

Analysis techniques	**Professional Control Total**	**Professional Control Average**	**Std.Dev**	**Student Control Total**	**Student Control Average**	**Std.Dev**
Assumption analysis	13	**4.3**	2.082	11	**1.8**	1.169
Constraint analysis	20	**6.7**	2.082	39	**6.5**	2.074
Risk analysis	18	**6**	5.196	15	**2.5**	1.049
Trade-off analysis	4	**1.3**	1.528	6	**1**	1.095
Option generation	16	**5.3**	1.155	30	**5**	2.898
Problem structuring	57	**19**	1	93	**15.5**	3.017
Total	128	**42.7**		194	**32.3**	

Table 28 - Professional and Student Analysis Techniques Frequencies

8. 2. Reasoning with Risk, Assumption, Constraint and Trade-off

When we look at the unique values of risk and assumption we see similar results as with the reasoning techniques (Table 29). The effect of the cards is that more risks and assumptions are identified by the test groups. The professional control group has higher values than the student control group, showing that experience influences more thorough reasoning. Interesting here is that the risks identified by the professional groups is on average of a higher frequency than the difference in assumption identification by the professional and student groups.

The professional control group identifies 72% more risks than the student control group, and the professional test group identified 45% more risks than the student test group. This reaffirms that with experience comes more risk identification. The constraints on the other hand have almost exactly the same results across the different groups, around 7 on average. The similar results of the constraints by both the professional and student groups here confirm earlier speculations. The constraint card has no influence on the group results because these constraints can already easily be found in the assignment itself.

Design reasoning elements	Professional Total Unique	Professional Test Unique Average	Std.Dev	Student Total Unique	Student Test Unique Average	Std.Dev
Assumption	19	**9.5**	.707	47	**7.8**	3.656
Constraint	15	**7.5**	2.121	42	**7**	1.472
Risk	18	**9**	5.657	37	**6.2**	.753

Design reasoning elements	Professional Total Unique	Professional Control Unique Average	Std.Dev	Student Total Unique	Student Control Unique Average	Std.Dev
Assumption	14	**4.7**	2.309	14	**2.3**	1.506
Constraint	23	**7.7**	2.082	42	**7**	2
Risk	13	**4.3**	3.215	15	**2.5**	1.049

Table 29 - Professional and Student Unique Design Reasoning Elements

With trade-off analysis we see that the professional test and control groups focus more on identifying cons, while the student test and control groups have higher values of pros (Table 30). This shows that experienced designers are more likely to think critically, which is reaffirmed by the professional control group using more thorough reasoning.

	Professional Test Total	Professional Test Average	Std.Dev	Student Test Total	Student Test Average	Std.Dev
Trade-off analysis	8	**4**	2.828	16	**2.7**	1.506
Pros	13	**6.5**	3.536	46	**7.7**	5.317
Cons	24	**12**	9.899	48	**8**	3.742
	Professional Control Total	**Professional Control Average**	**Std.Dev**	**Student Control Total**	**Student Control Average**	**Std.Dev**
Trade-off analysis	4	**1.3**	1.528	6	**1**	1.095
Pros	10	**3.3**	3.512	30	**5**	3.578
Cons	13	**4.3**	3.055	13	**2.2**	1.602

Table 30 - Professional and Student Trade-off Analysis, Pros and Cons

8. 3. Reasoning with Design Context, Problems and Solutions

As with the previous analysis on the student and professional groups we find that the context, problem and solution cards do not have the same impact on identifying design issues and decisions as the other cards have on the reasoning techniques (Table 31). The design options have increased similarly as with the option analysis, but design issues are more similar when comparing the professional and student groups.

We have already seen that the design issues do not increase with the same ratio as the other reasoning techniques, and a probable explanation for this is that besides the cards, the design strategy also has some measure of influence. With these results comparing the professional and student groups we see this confirmed as the difference between the groups is not with the same ratio as the other reasoning techniques. Meaning that design strategy remains an influence, but also that experience does not show a great impact into the number of design issues identified either.

	Professional Test Total	Professional Test Average	Std.Dev	Student Test Total	Student Test Average	Std.Dev
Design issues	34	**17**	5.657	94	**15.7**	7.474
Design options	61	**30.5**	17.678	148	**24.7**	11.961
Design decisions	27	**13.5**	2.121	92	**15.3**	7.659
	Professional Control Total	**Professional Control Average**	**Std.Dev**	**Student Control Total**	**Student Control Average**	**Std.Dev**
Design issues	37	**12.3**	3.055	70	**11.7**	5.715
Design options	59	**19.7**	7.572	95	**15.8**	7.679
Design decisions	35	**11.7**	3.512	74	**12.3**	5.715

Table 31 - Professional and Student Design issues, options and decisions

8.4. Card Play Differences

With how many cards are played during the design session there are some differences between the professional and student groups. In the professional analysis we already remarked on the difference between the professional groups and how many cards they played, 67 by PT1 and 20 by PT2. This seems to be caused by how far the group can adopt the cards into their discussion. But looking at the differences in card play between student and professional groups we see that there might be another reason as well (Table 32).

The first difference seen is that the professional groups have used all cards, while only two of the student groups, T5 and T6, did the same. Another is that the professional groups play more cards than the student groups. As mentioned before, unlike the student groups the professional groups did not have to log the cards they played. This might have made it easier for the professional groups to adopt and use the cards fluently in their discussion. On the other hand, the results of PT2 are very similar to the student groups. The highest frequency of cards played by the student groups is 16, while PT2 has 20, which does not show a great difference. This again suggests that even if the student groups did not have to log their cards, how many cards they played would have depended more on their willingness to use the cards throughout their discussion.

Frequencies cards played	PT1	PT2	T1	T2	T3	T4	T5	T6
Context	1	3	1		1		1	2
Problem	11	3	2	2		6	3	5
Solution	4	1	2	2	1	7	3	2
Assumption	17	8	3	1	2	1	3	2
Constraint	10	2	1	3	1		2	1
Risk	14	2			1	1	2	4
Trade-off	10	1		1		1	1	1
Total	**67**	**20**	**8**	**9**	**6**	**16**	**15**	**16**

Table 32 - Professional and Student Frequencies Cards Played

This is further supported when we look at the difference between the terms used by both groups (Table 33). Here we see that the professional groups make mention the card terms more than the student groups. Especially the comparison between PT1, PT2, T5 and T6 is interesting here as these groups used all the cards at least once. These student groups show the highest frequency of terms mentioned and also use some of the most cards. These frequencies of terms used are similar to that of PT2 which only uses 4 more cards than T6. PT1 uses both more terms and plays more cards, while PT2 has similar results to some of the student groups. If logging the cards has any effect on card play, it will probably be small. In how far the design group is willing to adopt the card game is more important to increase card play.

Term used	PT1	PT2	T1	T2	T3	T4	T5	T6
Assumption	36	42	8	12	14	5	13	23
Constraint	15	17	6	3	1	18	12	21
Risk	27	4	8	19	12	0	5	11
Trade-off	22	5	6	1	9	4	11	20
Total	**100**	**68**	**28**	**35**	**36**	**27**	**41**	**75**

Table 33 - Professional and Student Frequencies Term Used

8. 5. QUESTIONNAIRES

After the experiments, student and professional groups were asked to provide feedback in the form of a questionnaire (Appendix E). Here again we find similar results.

The cards and their symbols were mostly intuitive for the students and professionals and perceived to be useful. The questions table which provided example questions for the cards were mostly used at the beginning of the session, but not constantly used for help.

The helpfulness of the cards during the discussion is debatable for the participants. The professionals are mostly positive about the cards and perceive them as being helpful during discussion as reminders of what to look out for, which is reflection. Opinions are mixed on whether the cards helped to find more design issues. One the on hand participants of both teams state that the cards prompted different parts of the discussion which otherwise might not have been looked at. But on the other hand they also state that the issues found were things they always looked at even without the cards.

By the students the cards were perceived as being especially helpful in the beginning, for structuring the discussion and thinking more deeply about the problems. But the game also interrupted the natural flow of discussion and felt repetitive and took too long. Notably as with the professionals, participants stated that the cards themselves didn’t help them think of new issues, but it did help elaborate on concepts and as a discussion starter.

The results of the questionnaires show that the cards helped the participants reflect on their decisions, and that this was also perceived as being useful. But when it came to identifying design issues they didn’t think the cards had much influence, which corroborates with the theory that the design strategy holds some influence here as well.

Overall the participants also wanted more structure and rules in the game itself, such as a mandatory rule to use all the cards at least once. The current game rules are very open and especially by the students this was seen as a hindrance as the participants then first have to find out what to do with the cards.

One other question the participants were asked was which card was their favourite (Table 34). Participants could recommend several cards. We see here an interesting connection with the cards most played. With both groups assumption is the favourite card of most participants, and it is played frequently among the groups. Especially with the students we see that the cards most played and favourite cards are a similar top three of problem, solution and assumption, but problem and solution are mostly used by T4 while everyone uses assumption. With the professional groups both use assumption more than any other card.

An interesting difference between the groups is that the risk card is a favourite with the professional group PT1, while none of the 6 student groups mention the risk card as a favourite, even though it is used by almost all groups. Something of note is that the context cards was only ever played in the first reflective period, except with PT2. This could have a natural explanation in that the context is often what designers begin with during their problem-solving.

Favourite cards	Professional	Student
Context	1	1
Problem	0	4
Solution	0	4
Assumption	2	5
Risk	2	0
Trade-off	0	2
Constraint	0	1

Table 34 - Questionnaire Favourite Cards

Overall the participants found the card game to be useful, but difficult in use and often disruptive. As the aim of the cards is to make participants look back on their decisions it is perhaps natural that it would interrupt the discussion flow. But loose rules and a general confusion about the card game made adoption, especially by the student groups, difficult and is a point which should be improved.

8. 6. Key Findings

To summarize the analysis results of the professional and student experiments comparison the key findings are listed shortly. In most cases the professional results support the student results.

1. **Cards lead to more use of reasoning techniques**

Comparing the professional and student test groups with the control groups there remains a noticeable difference between use of reasoning between the groups (Table 28). The boxplots also show marked differences between control and test groups (Appendix I). The pattern that mostly emerged is that the student control group has the lowest values followed by the professional control group which overlaps slightly with the student test group. The professional test group has the highest values.

2. **Professionals are more critical thinkers**

When comparing results of the unique values of assumption, risk and trade-offs we find that the professional control groups especially identify more risks and more cons than the student group control groups (Table 29; Table 30). This shows that their experience has made them more critical thinkers, more likely to think of possible problems involved in a solution.

3. **Problem identification dependent on design strategy**

For both the student and professional groups differences between identified problems and decisions are smaller than with other reasoning techniques. The design issues identified are also of similar values when comparing the professional and student test and control groups with each other (Table 31). This reaffirms the theory that the cards themselves do not have much influence on finding problems, but that the design strategy chosen by the groups is more important.

4. **No difference in constraint identification**

When comparing all the professional and student groups it becomes clear that the constraint identified are equal, as on average they all centre around 7 (Table 29). There is no difference between the test and control groups. This reaffirms the theory that the constraint card has no influence because the constraints are easy to find in the assignment itself, being listed as part of the requirements (Appendix G; Appendix H).

5. Card game adoption increases term use

When the professional and student test groups use more cards, they use more of the terms in their discussion (Table 32; Table 33). What seems important for the adoption of the card game is that all of the cards are used at least once.

9. CONCLUSION

For this thesis we have looked into the research question; *How can the use of a reflective method, prompting reasoning techniques during the decision process with the use of reflection, influence and support design reasoning in software architecture.* The reflective method described here is a card game consisting of 7 cards, each representing a design reasoning technique or a step in the design process. There are 3 activity cards representing context, problem, solution, and 4 technique cards representing constraint, assumption, risk and trade-off. The expected effect of the cards was that it would influence designers to use rational decision making by reminding them to carry out reasoning through using reflection (Figure 12). The cards influence designers to use more reasoning techniques which in turn lead to more reflection on the design reasoning process. The cards were tested in several experiments involving students and professionals. Our hypothesis was that the reasoning cards would influence designers, but that especially inexperienced designers would have more noticeable effects compared to experienced designers.

In the case of the professional results, differences between the test and control groups could not be tested for significance as several assumptions were violated, and the samples were too small to ignore these. For the student groups quantitative analysis could be used and the results were mixed as overall the cards show a significant difference between test and control group, influencing designers to use more thorough reasoning, but other reasoning techniques do not show significant influences. These results can be seen in detail in 6.2.1 Quantitative Significance Testing (Table 10). Results showed that the hypotheses regarding problem structuring, assumption analysis, risk analysis and trade-off analysis could be accepted to have a significant influence on design reasoning. With the hypotheses regarding option generation and constraints analysis the null hypothesis could not be rejected.

We find that most of the cards have a significant positive influence on the design reasoning of inexperienced designers as seen in 6. Student Experiments. Most of the noticeable differences seen with the experienced designers, though not significant, do mirror the results seen with the students as seen in 7. Professional Experiments and 8. Comparison Analysis Results. From these results there are three main conclusions we can make.

9.1.1 THOROUGH REASONING AND REFLECTION

The first conclusion we can make is about the main research question which we can answer positively. A reflective method can be made which influences design reasoning, using reasoning techniques and reflection.

The card game prompts designers to use more thorough reasoning instead of satisficing behaviour, i.e. being more easily satisfied with the decisions made instead of re-evaluating if the decision is really the right one and everything has been taken into consideration. For the students there were significant differences between the test and control groups with the cards representing assumption analysis, risk analysis and trade-off analysis. The three activity cards, context, problem and solution, also prompted more use of problem structuring.

Results also showed that when it came to trade-off analysis the student test groups identified significantly more cons than the control group, showing that the cards encouraged critical thinking. Card play with both the professional and student test groups were centred mostly around the reflection periods, a period when designers were asked to use the cards. The pattern was that when they decided to play the cards the use of cards would spike and then taper off. During these periods two distinct reflection strategies were found where designers used the cards for classifying previous discussions and using them as inspiration for further discussion were observed.

9.1.2 Constraint Analysis

The results regarding the constraint card and the constraint analysis are a unique case within this thesis. Unlike all of the other techniques there is practically no difference in the amount of constraints identified by either test or control group, from the professionals or the students. In all cases they are the same, centring around 7 constraints identified for each of 17 groups used in the experiments (Table 29). We conclude that the constraint card made no difference in the constraints identified.

When we look deeper into the transcripts into what constraints are identified we find that the groups share many of the same constraints (Appendix G; Appendix H). And in their transcription they even use the same wording. Looking into the assignment the groups were given the reason for this becomes clear (Appendix A). In the assignment the designers are given a list of requirements to design their software architecture, and within those requirements several constraints are also added, such as ‘at least 6 intersections’. And these constraints are the ones identified by all the groups.

To identify the constraints takes as little effort as simply reading the assignment, and as there are also plenty of constraints to be found, the constraint card does not give the test groups a sense that they should identify more than are already present. This might chance if the designers are given an assignment where the constraints are not as obvious in the text.

9.1.3 Design Strategy

For inexperienced designers the effect of the cards has been to use more thorough reasoning and critical thinking to make more informed decisions. But identifying design issues, design options and its accompanying reasoning technique option generation, and design decisions are not triggered by the cards and the student designers did not identify significantly more while using the cards. Problem structuring did show a significant difference, but did not have the intended result of identifying more problems, instead helping with more thorough reasoning.

A probable cause for the three activity cards being less effective than the technique cards is that to identify problems and generate options there is another factor which has a contribution, design strategy. The design strategy chosen by the group may be a more contributive factor of how many design issues and options are identified than simply the use of more reasoning techniques.

This is supported by looking at various groups used in the experiment and what their design strategies were. For two of the student groups, one test and one control, is was obvious that they used the same type of design strategy based on satisficing. While the test group had slightly better results in identifying reasoning techniques due to the other cards, both still had some of the lowest identified problems and options compared to their respective groups. With the professional test groups we also see the use of two different strategies; problem-oriented and solution-oriented approaches, by respectively a test group and a control group. A solution-oriented approach limits designers in how much information they seek out, as they start working with the preferred solution early on, while a problem-oriented approach makes designers more willing to go deeper into issues and find information (Restrepo &

Christiaans, 2004). As a result the test group identifies far more problems and options than the control group, while the other test and control groups have more equal results.

We can conclude here that for identifying problems through reasoning alone is not enough, and we theorize that it is the design strategy that makes a difference here (Figure 16).

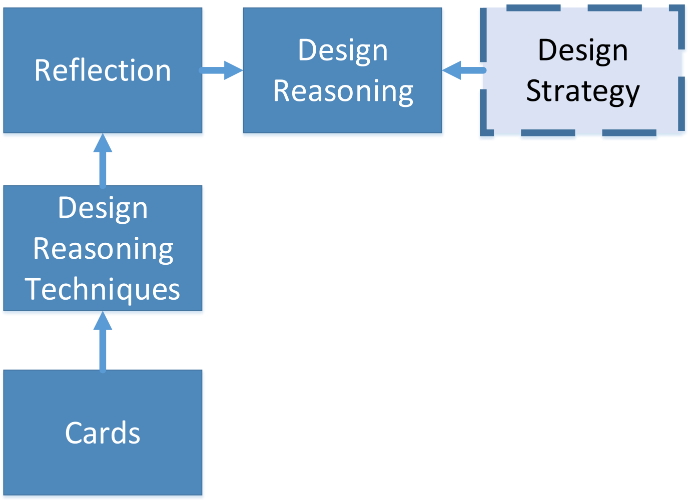


Figure 16 - Design Reasoning Influence - Design Strategy

9. 2. Validity of the Research

The analysis performed in this thesis is a combination of quantitative and qualitative measures, in this section we address any concerns about the validity and reliability of the research. The student results were tested for significant differences using the quantitative independent T-test and Mann Whitney test (Field, 2013). The professional results could not be tested in the same manner as several assumptions were violated. For the transcripts the qualitative discourse analysis was used to interpret the text, which in itself is subjective and reliant on the view of the researcher (Horsburgh, 2003). This thesis is an empirical research in the form of a case study, involving experiments. Empirical research is one of the main research methods within the software architecture research field, and it also creates evidence for examining the validity of the research results. Most common is to address the internal and external validity of the results, and acknowledge any limitations which may apply (Galster & Weyns, 2016).

9.2.1 Internal Validity

Internal validity is about how far a valid conclusion can be made from the experiment and data collected. Firstly, this research makes use of the Irvine assignment which has been used and tested in other research and is well-known in the field of design reasoning (Petre & van der Hoek, 2013). This limits the results of this research to those applicable to this kind of assignment.

Secondly, participants were selected for their situational representativeness. The student experiment uses students of software architecture, and the result of this research is limited to novice designers in the Netherlands. The professional experiment uses experienced designers situated in America and Australia and results of this research are limited to these regions. However there is evidence the cards have a strong effect on reasoning by inexperienced designers.

There is also a difference in how the assignment was performed by the student and professional groups. The students were required to log when a certain card was played and there was no instructor to tell them when to use the card game. For the professional groups an instructor was present who noted down the cards played and gave the times for the reflection periods.

9.2.2 *External Validity*

External validity is about to what extent the results from the case study can be generalized across other published design reasoning articles. The results of this case study are supported by similar experiments (Razavian et al., 2016; Tang et al., 2008; Tang & van Vliet, 2009; Van Heesch et al., 2013), showing that in the case of novice designers, being made aware of reasoning techniques actively counteracts satisficing behaviour and results in more thorough reasoning.

But while most of the reasoning techniques prompted by the cards showed a significant difference between the student test and control group, not all cards had the same effect. Especially option generation, constraint analysis and trade-off analysis did not show a significant difference, together with design issues, options and decisions. There are several explanations for why these techniques were not affected by the cards.

9.2.3 *Reliability*

Reliability is about ensuring that the results found in the study are consistent, and would be the same if the study is conducted again. To ensure that the coding of the student transcripts is reliable it was tested for inter-reliability using Cohen's kappa coefficient (Cohen, 1968) to measure the level of agreement. The transcripts were each coded by two researchers using Nvivo 10. The average kappa coefficient of each of the transcripts was above 0.6 which is considered to show a good level of agreement (Table 35). The average of all transcripts combined was 0.64.

Control Group	Kappa Coefficient	Test Group	Kappa Coefficient
C1	0.76	T1	0.60
C2	0.65	T2	0.66
C3	0.67	T3	0.60
C4	0.67	T4	0.62
C5	0.61	T5	0.63
C6	0.61	T6	0.61

Table 35 - Average Kappa Coefficient Values per Group

For the professional transcripts the coding was not checked for inter-reliability, but as the same coding was used it is not expected that the coding reliability would be different.

For the student groups the results were tested according to the independent T-test, or if assumption were violated, with the non-parametric Mann Whitney test. To ensure the test could be performed the data was tested using a Levene's test for equality of variances and the Shapiro Wilk test for normal distribution. The results of these tests can be found in Appendix J.

9. 3. Further Research

In the process of conducting the experiments and writing this thesis there have been several questions raised in regard to further research. The card system designed in this thesis is a new one and opens many opportunities for research within the field of design reasoning. In this section we focus on a few of the more interesting opportunities for research.

9.3.1 *Further Professional Experimentation*

First of all, it would be interesting to conduct further experiments for the professional groups, adding more test and control groups so that quantitative analysis could be performed. Seeing which results are significant, and if they match the student's results and the qualitative speculation in this thesis would be interesting and help confirm if the card game could have great influence on professional designers as well.

9.3.2 *Incorporate Design Strategy*

Another interesting field of research is into some of the unexpected results of the experiments. Especially the fact that the three strategy cards; context, problem and solution did not have an influence on the identified design issues. Is design strategy truly the missing factor here, and how could this be added to the card game. More research into what the most effective strategy is that would lead to more design reasoning is also of interest.

9.3.3 *Influence of the Constraint Card*

The constraint card is a separate topic of research as it had no influence at all and constraints were all similar. Is this truly because the constraints are so easy to find in the Irvine assignment, and how representative is the assignment to what designers can usually expect. Are requirements and constraints so intertwined that constraints will always be easier to identify. More research with a more vague assignment in terms of requirements could help show the true effectiveness of the constraint card.

9.3.4 *Card Game Tablet*

The card game has now been tested in physical form, but it might also be of interest to see how it performs as a software system. Perhaps a tablet could be used to represent the cards and designers need merely tap the card they want to discuss. Would this influence how many cards are played, and how they are played. A benefit of using a tablet would be that the cards are immediately recorded, this data could be used for design rationale later on. But it might also be seen as a detriment as designers are more reluctant to use the cards in this manner, overlooking the tablet when they couldn't with a physical deck of cards in their hands.

9.3.5 *Student Learning Tool*

As the results with students has shown itself to be significant, it might also be worthwhile to use the card game as a learning tool to become more familiar with what should be considered during a design. The card game is simple to use, and with more development to improve the card game and make it more effective, this tool could help students avoid satisficing behaviour and use more thorough reasoning. With reflection being used as a common teaching tool as well there is already precedent for its viability (Larrivee, 2000).

It is clear that there are many opportunities for further research based on this thesis and the card game in the field of design reasoning. The card game has shown itself to be effective and could be developed further to become a useful tool for improving rationale with reflection as decisions are being made.

REFERENCES

Bass, L., Clements, P., & Kazman, R. (2012). *Software Architecture in Practice: Third Edition*. USA, Boston: Addison-Wesley Professional.

Björklund, T. A. (2013). Initial mental representations of design problems: Differences between experts and novices. *Design Studies*, *34*(2), 135–160. doi:10.1016/j.destud.2012.08.005

Borton, T. (1970). *Reach, touch, and teach: Student concerns and process education*. California, US: McGraw-Hill Paperbacks.

Boud, D., Keogh, R., & Walker, D. (1985). *Reflection: Turning experience into learning*. (D. Boud, R. Keogh, & D. Walker, Eds.). London; New York: Routledge.

Chandrasekaran, B., Goel, a. K., & Iwasaki, Y. (1993). Functional representation as design rationale. *Computer*, *26*(1). doi:10.1109/2.179157

Cheek, J. (2004). At the Margins? Analysis and Qualitative Research, *14*(8), 1140–1150. doi:10.1177/1049732304266820

Cohen, J. (1968). Weighted kappa: nominal scale agreement with provision for scaled disagreement or partial credit. *Psychological Bulletin*, *70*(4), 213–220. doi:10.1037/h0026256

Dewey, J. (1933). *How We Think*. Boston, US: DC Heath.

Dorst, K., & Cross, N. (2001). Creativity in the design process: Co-evolution of problem-solution. *Design Studies*, *22*(5), 425–437. doi:10.1016/S0142-694X(01)00009-6

Field, A. (2013). *Discovering statistics using IBM SPSS statistics* (4th ed.). London, UK: Sage Publications.

Galster, M., & Weyns, D. (2016). Emperical Research in Software Architecture. In *13th Working IEEE/IFIP Conference on Software Architecture, Italy, Venice* (pp. 11–20). doi:10.1109/WICSA.2016.10

Gibbs, G. (1988). *Learning by doing: A guide to teaching and learning methods*. London, UK: Further Education Unit.

Grenning, J. (2002). Planning poker or how to avoid analysis paralysis while release planning. *Hawthorn Woods: Renaissance Software Consulting*, (April), 1–3. Retrieved from http://www.renaissancesoftware.net/files/articles/PlanningPoker-v1.1.pdf

Horsburgh, D. (2003). Evaluation of qualitative research. *Journal of Clinical Nursing*, *12*(2), 307–312.

IDEO. (2015). IDEO Method Cards. Retrieved November 15, 2015, from https://www.ideo.com/by-ideo/method-cards/

ISO/IEC. (2014). Systems and software engineering? Defining architecture. Retrieved November 10, 2015, from http://www.iso-architecture.org/ieee-1471/cm/

Jansen, A. J. A., & Bosch, J. B. J. (2005). Software Architecture as a Set of Architectural Design Decisions. In *WICSA 2005 - 5th Working IEEE/IFIP Conference on Software Architecture, Pittsburgh, PA, USA* (pp. 109–120). IEEE. doi:10.1109/WICSA.2005.61

Kahneman, D. (2011). *Thinking, Fast and Slow*. UK, London: Penguin Books, 2012.

Kazman, R., Abowd, G., Bass, L., & Clements, P. (1996). Scenario-based analysis of software architecture. *IEEE Software*, *13*(6), 47–55. doi:10.1109/52.542294

Klein, G. (2008). Naturalistic Decision Making. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *50*(3), 456–460. doi:10.1518/001872008X288385.

Kolb, D., & Fry, R. (1975). Towards a Theory of Applied Experiental Learning. In *Theories of Group Processes* (pp. 33–58). London; New York: John Wiley & Sons Inc.

Kunz, W., & Rittel, H. W. J. (1970). *Issues as elements of information systems*. *Working paper No. 131* (Vol. 131). Heidelberg, Germany (Reprinted May 1979). doi:10.4324/9780203851586

Laerd Statistics. (2016). Independent T-test using SPSS statistics. Retrieved June 13, 2016, from https://statistics.laerd.com/spss-tutorials/independent-t-test-using-spss-statistics.php

Lago, P., & van Vliet, H. (2005). Explicit assumptions enrich architectural models. In *Proceedings of the 27th International Conference on Software Engineering, 2005. ICSE '05.* (pp. 206–214). doi:10.1109/ICSE.2005.1553563

Larrivee, B. (2000). Transforming Teaching Practice: becoming the critically reflective teacher. *Reflective Practice*, *1*(3), 293–307.

Lee, J. (1991). Extending the Potts and Bruns model for recording design rationale. In *ICSE 1991 - 13th International Conference on Software Engineering* (pp. 114–125). Austin, TX: IEEE. doi:10.1109/ICSE.1991.130629

Lee, J. (1997). Design rationale systems: Understanding the issues. *IEEE Expert-Intelligent Systems and Their Applications*, *12*, 78–85. doi:10.1109/64.592267

Nuseibeh, B. (2001). Weaving Together Requirements and Architectures. *Software Management*, (March), 115–117.

Parnas, D. L., & Clements, P. C. (1986). A Rational Design Process: How And Why To Fake It. *Transactions on Software Engineering*.

Petre, M., & van der Hoek, A. (2013). Software Designers in Action: A Human-Centric Look at Design Work. USA, Florida: CRC Press.

Poort, E. R., & Van Vliet, H. (2011). Architecting as a risk- and cost management discipline. In *Proceedings - 9th Working IEEE/IFIP Conference on Software Architecture, WICSA 2011* (pp. 2–11). doi:10.1109/WICSA.2011.11

Razavian, M., Tang, A., Capilla, R., & Lago, P. (2016). In Two Minds: How Reflections Influence Software Design Thinking. *Journal of Software: Evolution and Process*, *28*(6), 394–426.

Restrepo, J., & Christiaans, H. (2004). Problem structuring and information access in design. *Journal of Design Research*, *4*(2), 1551–1569.

Rittel, H. W. J., & Webber, M. M. (1973). Dilemnas in a general theory of planning. *Policy Sciences*, *4*(December 1969), 155–169. doi:10.1007/BF01405730

Roeller, R., Lago, P., & Van Vliet, H. (2006). Recovering architectural assumptions. *Journal of Systems and Software*, *79*(4), 552–573. doi:10.1016/j.jss.2005.10.017

Schön, D. A. (1983). *The Reflective Practitioner: How Professionals Think in Action*. USA, New York: BasicBooks.

Simon, H. A. (1973). The structure of ill structured problems. *Artificial Intelligence*, *4*(3-4), 181–201. doi:10.1016/0004-3702(73)90011-8

Simon, H. A. (1978). Rationality as Process and as Product of a Thought. *American Economic Review*, *68*(2), 1–16. doi:10.2307/1816653

Smart Decisions. (2016). Smart Decisions: A Software Architecture Design Game. Retrieved February 26, 2016, from http://smartdecisionsgame.com/#

Stacy, W., & MacMillan, J. (1995). Cognitive bias in software engineering. *Commun. ACM*, *38*(6), 57–63. doi:10.1145/203241.203256

Stanovich, K. E. (2009). Distinguishing the reflective, algorithmic, and autonomous minds: Is it time for a tri-process theory? In *In Two Minds: Dual Processes and Beyond* (pp. 55–88). Oxford University Press. doi:10.1093/acprof:oso/9780199230167.003.0003

Tang, A. (2011). Software Designers, Are You Biased? In *Proceedings of the 6th International Workshop on SHAring and Reusing Architectural Knowledge* (pp. 1–8). USA, New York: ACM. doi:10.1145/1988676.1988678

Tang, A., Aleti, A., Burge, J., & Van Vliet, H. (2010). What makes software design effective? *Design Studies*, *31*(6), 614–640. doi:10.1016/j.destud.2010.09.004

Tang, A., Babar, M. A., Gorton, I., & Han, J. (2006). A survey of architecture design rationale. *Journal of Systems and Software*, *79*(12), 1792–1804. doi:10.1016/j.jss.2006.04.029

Tang, A., Jin, Y., & Han, J. (2007). A rationale-based architecture model for design traceability and reasoning. *Journal of Systems and Software*, *80*, 918–934. doi:10.1016/j.jss.2006.08.040

Tang, A., & Lago, P. (2010). *Notes on Design Reasoning Techniques*. Retrieved from http://hdl.handle.net/1871/52438

Tang, A., Tran, M. H., Han, J., & Van Vliet, H. (2008). Design reasoning improves software design quality. In S. Becker, F. Plasil, & R. Reussner (Eds.), *Quality of Software Architectures. Models and Arhcitectures. 4th International Conference on the Quality of Software-Architectures, QoSA 2008, Germany, Karlsruhe* (Vol. 5281 LNCS, pp. 28–42). Berlin, Heidelberg: Springer Berlin Heidelberg. doi:10.1007/978-3-540-87879-7-2

Tang, A., & van Vliet, H. (2009). Software Architecture Design Reasoning. In *Software Architecture Knowledge Management* (pp. 155–174). doi:10.1109/ASWEC.2008.4483186

Tang, A., & van Vliet, H. (2015). Software Designers Satisfice. In D. Weyns, R. Mirandola, & I. Crnkovic (Eds.), *Software Architecture: 9th European Conference, ECSA 2015, Dubrovnik/Cavtat, Croatia, September 7-11, 2015. Proceedings* (Vol. 1, pp. 105–120). Switzerland,Cham: Springer International Publishing. doi:10.1007/978-3-319-23727-5_9

Tversky, A., & Kahneman, D. (1974). Judgment under Uncertainty: Heuristics and Biases. *Science*, *185*(4157), 1124–1131.

UCI. (2010). Studying Professional Software Design. Retrieved November 12, 2015, from http://www.ics.uci.edu/design-workshop/

Van Heesch, U., & Avgeriou, P. (2010). Naive Architecting - Understanding the Reasoning Process of Students. *Software Architecture*, *6285*(Chapter 5), 24–37.

Van Heesch, U., Avgeriou, P., & Tang, A. (2013). Does decision documentation help junior designers rationalize their decisions? A comparative multiple-case study. *Journal of Systems and Software*, *86*(6), 1545–1565. doi:10.1016/j.jss.2013.01.057

Woods, E. (2012). Industrial architectural assessment using TARA. *Journal of Systems and Software*, *85*(9), 2034–2047. doi:10.1016/j.jss.2012.04.055

Zannier, C., Chiasson, M., & Maurer, F. (2007). A model of design decision making based on empirical results of interviews with software designers. *Information and Software Technology*, *49*(6), 637–653. doi:10.1016/j.infsof.2007.02.010

Zimmermann, O., Koehler, J., Leymann, F., Polley, R., & Schuster, N. (2009). Managing architectural decision models with dependency relations, integrity constraints, and production rules. *Journal of Systems and Software*, *82*(8), 1249–1267. doi:10.1016/j.jss.2009.01.039

APPENDICES

APPENDIX A

UCI Design Workshop Prompt

Design Prompt: Traffic Signal Simulator

Problem Description

For the next two hours, you will be tasked with designing a traffic flow simulation program. Your client for this project is Professor E, who teaches civil engineering at UCI. One of the courses she teaches has a section on traffic signal timing, and according to her, this is a particularly challenging subject for her students. In short, traffic signal timing involves determining the amount of time that each of an intersection's traffic lights spend being green, yellow, and red, in order to allow cars in to flow through the intersection from each direction in a fluid manner. In the ideal case, the amount of time that people spend waiting is minimized by the chosen settings for a given intersection's traffic lights. This can be a very subtle matter: changing the timing at a single intersection by a couple of seconds can have far-reaching effects on the traffic in the surrounding areas. There is a great deal of theory on this subject, but Professor E. has found that her students find the topic quite abstract. She wants to provide them with some software that they can use to "play" with different traffic signal timing schemes, in different scenarios. She anticipates that this will allow her students to learn from practice, by seeing first-hand some of the patterns that govern the subject.

Requirements

The following broad requirements should be followed when designing this system:

1. Students must be able to create a visual map of an area, laying out roads in a pattern of their choosing. The resulting map need not be complex, but should allow for roads of varying length to be placed, and different arrangements of intersections to be created. Your approach should readily accommodate at least six intersections, if not more.

2. Students must be able to describe the behavior of the traffic lights at each of the intersections. It is up to you to determine what the exact interaction will be, but a variety of sequences and timing schemes should be allowed. Your approach should also be able to accommodate left-hand turns protected by left-hand green arrow lights. In addition:

   a. Combinations of individual signals that would result in crashes should not be allowed.

   b. Every intersection on the map must have traffic lights (there are not any stop signs, overpasses, or other variations). All intersections will be 4-way: there are no "T" intersections, nor one-way roads.

   c. Students must be able to design each intersection with or without the option to have sensors that detect whether any cars are present in a given lane. The intersection's lights' behavior should be able to change based on the input from these sensors, though the exact behavior of this feature is up to you.

3. Based on the map created, and the intersection timing schemes, the students must be able to simulate traffic flows on the map. The traffic levels should be conveyed visually to the user in a real-time manner, as they emerge in the simulation. The current state of the intersections' traffic lights should also be depicted visually, and updated when they change. It is up to you how to present this information to the students using your program. For example, you may choose to depict individual cars, or to use a more abstract representation.

4. Students should be able to change the traffic density that enters the map on a given road. For example, it should be possible to create a busy road, or a seldom used one, and any variation in between. How exactly this is declared by the user and depicted by the system is up to you. Broadly, the tool should be easy to use, and should encourage students to explore multiple alternative approaches. Students should be able to observe any problems with their map's timing scheme, alter it, and see the results of their changes on the traffic patterns. This program is not meant to be an exact, scientific simulation, but aims to simply illustrate the basic effect that traffic signal timing has on traffic. If you wish, you may assume that you will be able to reuse an existing software package that provides relevant mathematical functionality such as statistical distributions, random number generators, and queuing theory.

You may add additional features and details to the simulation, if you think that they would support these goals.

Your design will primarily be evaluated based on its elegance and clarity – both in its overall solution and envisioned implementation structure.

Desired Outcomes

Your work on this design should focus on two main issues:

1. You must design the interaction that the students will have with the system. You should design the basic appearance of the program, as well as the means by which the user creates a map, sets traffic timing schemes, and views traffic simulations.

2. You must design the basic structure of the code that will be used to implement this system. You should focus on the important design decisions that form the foundation of the implementation, and work those out to the depth you believe is needed.

The result of this session should be: *the ability to present your design to a team of software developers who will be tasked with actually implementing it*. The level of competency you can expect is that of students who just completed a basic computer science or software engineering undergraduate degree. You do not need to create a complete, final diagram to be handed off to an implementation team. But you should have an understanding that is sufficient to explain how to implement the system to competent developers, without requiring them to make many high-level design decisions on their own.

To simulate this hand-off, you will be asked to briefly explain the above two aspects of your design after the design session is over.

Timeline

• 1 hour and 50 minutes: Design session

- 10 minutes: Break / collect thoughts
- 10 minutes: Explanation of your design
- 10 minutes: Exit questionnaire

APPENDIX B

Pilot Card Game Instructions

Software Architecture course at Utrecht University

Assignment B.1: Design an architecture

Develop together an architectural design for the problem using the design method assigned to you. Please track all your design decisions and rationale so that we can analyze why you have come to the design you delivered. Use the prescribed format.

The cards and their meanings

For designing an architecture you are asked to use a card game to help with your discussion. These cards are meant to prompt discussion and support collaboration by providing questions to consider, which can be seen in Table 1. The columns represent the activities and the rows are the question types.

	Design Contexts and Requirements	Design Problems	Design Solutions
Assumption	What assumptions are made?	Are there any assumptions that affect the design problem?	Are there any assumptions affect the solution option?
Risk	Are there any risks that certain events would happen?	Are there any risks which cause design problems?	Are there any risks that affect the viability of a solution?
Constraint	What are the constraints imposed by the contexts?	Which constraints cause design problems?	Which constraints limit the solution options?
Trade-off	What contexts can be compromised?	Can a problem be framed differently?	What are the solution options? Can a solution option be compromised?

Table 1 - Design questions [1]

Each player gains a deck of fifteen cards to help make design decisions. The cards are split into three categories, these are the design activities;

- **Context**, these constitute the requirements and context of the design.
- **Problems**, the important issues and problems that need to be resolved in order to come to a design.
- **Solution**, the different ways that the problems can be resolved.

These three categories each have four different cards based on design question types; assumptions, constraint, risk and trade-off.

[1] Razavian, M., Tang, A., Capilla, R., & Lago, P. (2015). In Two Minds: How Reflections Influence Software Design Thinking. Technical Report.

C. Constraints are limitations or restrictions which can influence design. These most often come from the requirements and stakeholders, e.g. budget constraints or 'need to process 100 people in a day'.

A. Assumptions refer to the claims being made. These can contain hidden assumptions which should be questioned to ascertain their validity. E.g. 'model-view-controller is the best solution'.

R. Risks can be technical or non-technical issues which can negatively influence the design. These cannot always be resolved but should be acknowledged as risks. E.g. 'the development team is unfamiliar with the needed tool'.

T. Trade-offs are a compromise when the requirements and constraints cannot both be satisfied. In this case a decision based on the pro's and con's and an evaluation of the priorities must be made. E.g. 'the preferred tool addresses a high priority requirement, but means going over budget'.

Gameplay rules

The discussion cards are meant to prompt the players to question their design decisions in order to come to better thought-out designs. Some example questions for the cards can be found in Table 1. In order to keep the design discussion organic the cards do not represent a method structure to follow. They are meant to be used in accordance with the player's discussion and argumentation for their choices.

There are several rules for gameplay:

1. The game is played in rounds, in which a round represent a discussion;
2. A round starts when a card is played in order to start a discussion;
3. During discussion other cards can be played and reused;
4. When a decision is made the discussion ends;
5. Cards related to the discussion are removed.

When discussing these cards players can agree with the statements being made, or disagree and explain why. You can also ask questions about the statement when something is unclear and must be clarified.

Documentation

It is important to log the design decisions being made and the rationale behind them as these are also a part of the documentation. The cards being played during your design discussion can be used to support your reasoning.

For this purpose we ask that you **record** and **document** your session. This means that you need to **speak clearly** during the discussion and think aloud, state what you're doing and what cards are being played or removed.

As you will be referring to the models you're designing you should also document your design progression. You can make pictures or save the designs of your developing model and refer to them during designing as model A, B, C etc., to show the order of your design progression and make it clear on the recording what you are discussing.

During play you can shortly document the cards being played in an excel sheet which will be provided Here you can mark which card is being played and when. You should also note who played the card and when it is removed from play. Please record the minutes and the seconds from the time that you start designing, you can use a stopwatch for this purpose.

This sheet can later be further elaborated upon for your rationale, using the recording for adding why the card was played and what the outcome of the discussion was.

Traffic Simulation Assignment

Design Prompt: Traffic Signal Simulator

Problem Description

For the next two hours, you will be tasked with designing a traffic flow simulation program. Your client for this project is Professor E, who teaches civil engineering at UCI. One of the courses she teaches has a section on traffic signal timing, and according to her, this is a particularly challenging subject for her students. In short, traffic signal timing involves determining the amount of time that each of an intersection's traffic lights spend being green, yellow, and red, in order to allow cars in to flow through the intersection from each direction in a fluid manner. In the ideal case, the amount of time that people spend waiting is minimized by the chosen settings for a given intersection's traffic lights. This can be a very subtle matter: changing the timing at a single intersection by a couple of seconds can have far-reaching effects on the traffic in the surrounding areas. There is a great deal of theory on this subject, but Professor E. has found that her students find the topic quite abstract. She wants to provide them with some software that they can use to "play" with different traffic signal timing schemes, in different scenarios. She anticipates that this will allow her students to learn from practice, by seeing first-hand some of the patterns that govern the subject.

Requirements

The following broad requirements should be followed when designing this system:

1. Students must be able to create a visual map of an area, laying out roads in a pattern of their choosing. The resulting map need not be complex, but should allow for roads of varying length to be placed, and different arrangements of intersections to be created. Your approach should readily accommodate at least six intersections, if not more.

2. Students must be able to describe the behavior of the traffic lights at each of the intersections. It is up to you to determine what the exact interaction will be, but a variety of sequences and timing schemes should be allowed. Your approach should also be able to accommodate left-hand turns protected by left-hand green arrow lights. In addition:

    a. Combinations of individual signals that would result in crashes should not be allowed.

b. Every intersection on the map must have traffic lights (there are not any stop signs, overpasses, or other variations). All intersections will be 4-way: there are no “T” intersections, nor one-way roads.

c. Students must be able to design each intersection with or without the option to have sensors that detect whether any cars are present in a given lane. The intersection’s lights’ behavior should be able to change based on the input from these sensors, though the exact behavior of this feature is up to you.

3. Based on the map created, and the intersection timing schemes, the students must be able to simulate traffic flows on the map. The traffic levels should be conveyed visually to the user in a real-time manner, as they emerge in the simulation. The current state of the intersections’ traffic lights should also be depicted visually, and updated when they change. It is up to you how to present this information to the students using your program. For example, you may choose to depict individual cars, or to use a more abstract representation.

4. Students should be able to change the traffic density that enters the map on a given road. For example, it should be possible to create a busy road, or a seldom used one, and any variation in between. How exactly this is declared by the user and depicted by the system is up to you. Broadly, the tool should be easy to use, and should encourage students to explore multiple alternative approaches. Students should be able to observe any problems with their map’s timing scheme, alter it, and see the results of their changes on the traffic patterns. This program is not meant to be an exact, scientific simulation, but aims to simply illustrate the basic effect that traffic signal timing has on traffic. If you wish, you may assume that you will be able to reuse an existing software package that provides relevant mathematical functionality such as statistical distributions, random number generators, and queuing theory.

You may add additional features and details to the simulation, if you think that they would support these goals.

Your design will primarily be evaluated based on its elegance and clarity – both in its overall solution and envisioned implementation structure.

Desired Outcomes

Your work on this design should focus on two main viewpoints:

1. The Context Viewpoint
   You must design the interaction that the students will have with the system. You should design the means by which the user creates a map, sets traffic timing schemes, and views traffic simulations.

2. The Functional Viewpoint
   You must describe the system’s runtime functional elements and their responsibilities, interfaces and primary interactions. You should focus on the important design decisions that form the foundation of the implementation, and work those out to the depth you believe is needed.

The result of this session should be: *the ability to present your design to a team of software developers who will be tasked with actually implementing it.* The level of competency you can expect is that of students who just completed a basic computer science or software engineering undergraduate degree. You do not need to create a complete, final diagram to be handed off to an implementation team. But you should have an understanding that is sufficient to explain how to implement the system to competent developers, without requiring them to make many high-level design decisions on their own.

To simulate this hand-off, you will be asked to make a documentation of the model you have developed, including a glossary, description and rationale.

Timeline

- 30 min – Assignment Explanation
    - 15 min - Feedback round
- 1 ½ hour Design Session
    - 5 min - Feedback/break
- 45 min - Make documentation

Can choose own 5 min break

APPENDIX C

Final Prototype Card Game Instructions

Software Architecture course at Utrecht University

Assignment B.1: Design an architecture

Develop together an architectural design for the problem using the design method assigned to you. Please track all your design decisions and rationale so that we can analyze why you have come to the design you delivered. Use the prescribed format.

The cards and their meanings

For designing an architecture you are asked to use a card game to help with your discussion. These cards are meant to prompt discussion and support collaboration by providing questions to consider, which can be seen in Table 1. The columns represent the activities and the rows are the question types.

	Design Contexts and Requirements	Design Problems	Design Solutions
Assumption	What assumptions are made?	Are there any assumptions that affect the design problem?	Are there any assumptions affect the solution option?
Risk	Are there any risks that certain events would happen?	Are there any risks which cause design problems?	Are there any risks that affect the viability of a solution?
Constraint	What are the constraints imposed by the contexts?	Which constraints cause design problems?	Which constraints limit the solution options?
Trade-off	What contexts can be compromised?	Can a problem be framed differently?	What are the solution options? Can a solution option be compromised?

Table 1 - Design questions [2]

Each player gains a deck of fifteen cards to help make design decisions. The cards are split into three categories, these are the design activities;

- **Context**, these constitute the requirements and context of the design.
- **Problems**, the important issues and problems that need to be resolved in order to come to a design.
- **Solution**, the different ways that the problems can be resolved.

These three categories each have four different cards based on design question types; assumptions, constraint, risk and trade-off.

[2] Razavian, M., Tang, A., Capilla, R., & Lago, P. (2015). In Two Minds: How Reflections Influence Software Design Thinking. Technical Report.

C. Constraints are limitations or restrictions which can influence design. These most often come from the requirements and stakeholders, e.g. budget constraints or 'need to process 100 people in a day'.

A. Assumptions refer to the claims being made. These can contain hidden assumptions which should be questioned to ascertain their validity. E.g. 'model-view-controller is the best solution' or 'the context means all users are experienced with this tool'.

R. Risks can be technical or non-technical issues which can negatively influence the design. These are very common in design, e.g. 'the development team is unfamiliar with the needed tool' or 'does the design satisfy all the requirements'.

T. Trade-offs are a compromise when the requirements and constraints cannot both be satisfied. In this case a decision based on the pro's and con's and an evaluation of the priorities must be made. E.g. 'the preferred tool addresses a high priority requirement, but means going over budget'.

Gameplay rules

The discussion cards are meant to prompt the players to question their design decisions in order to come to better thought-out designs. Some example questions for the cards can be found in Table 1. In order to keep the design discussion organic the cards do not represent a method structure to follow. They are meant to be used in accordance with the player's discussion and argumentation for their choices.

There are several rules for gameplay:

1. The game is played in terms of a discussion;
2. The discussion starts when you play a card;
3. Others contribute to the discussion by playing their own cards;
4. When a decision is made the cards related to that topic can be removed;

During play there are three timed reflection periods where the discussion can only be held with the use of the cards. These are to make sure you reflect on your design decisions. Cards **must** be played during a period of 5 minutes. These are at:

4. At 15 minutes into the design session;
5. At 45 minutes into the design session;
6. At 1 hour and 45 minutes into the design session.

When discussing these cards players can agree with the statements being made, or disagree and explain why. You can also ask questions about the statements when something is unclear and must be clarified.

Documentation

It is important to log the design decisions being made and the rationale behind them as these are also a part of the documentation. The cards being played during your design discussion can be used to support your reasoning.

For this purpose we ask that you **record** and **document** your session. This means that you need to **speak clearly** during the discussion and think aloud, state what you're doing and what cards are being played or removed.

As you will be referring to the models you're designing you should also document your design progression. You can make pictures or save the designs of your developing model and refer to them during designing as model context, functional, information etc., to show the order of your design progression and make it clear on the recording what you are discussing.

During play you can shortly document the cards being played in a log document which will be provided during the session. Here you can mark which card is being played and when. You should also note who played the card and when it is removed from play. Please record the minutes and the seconds from the time that you start designing, you can use a stopwatch or your recording time for this purpose.

This sheet can later be further elaborated upon for your rationale, using the recording for adding why the card was played and what the outcome of the discussion was.

Traffic Simulation Assignment

Design Prompt: Traffic Signal Simulator

Problem Description

For the next two hours, you will be tasked with designing a traffic flow simulation program. Your client for this project is Professor E, who teaches civil engineering at UCI. One of the courses she teaches has a section on traffic signal timing, and according to her, this is a particularly challenging subject for her students. In short, traffic signal timing involves determining the amount of time that each of an intersection's traffic lights spend being green, yellow, and red, in order to allow cars in to flow through the intersection from each direction in a fluid manner. In the ideal case, the amount of time that people spend waiting is minimized by the chosen settings for a given intersection's traffic lights. This can be a very subtle matter: changing the timing at a single intersection by a couple of seconds can have far-reaching effects on the traffic in the surrounding areas. There is a great deal of theory on this subject, but Professor E. has found that her students find the topic quite abstract. She wants to provide them with some software that they can use to "play" with different traffic signal timing schemes, in different scenarios. She anticipates that this will allow her students to learn from practice, by seeing first-hand some of the patterns that govern the subject.

Requirements

The following broad requirements should be followed when designing this system:

1. Students must be able to create a visual map of an area, laying out roads in a pattern of their choosing. The resulting map need not be complex, but should allow for roads of varying length to be placed, and different arrangements of intersections to be created. Your approach should readily accommodate at least six intersections, if not more.

2. Students must be able to describe the behavior of the traffic lights at each of the intersections. It is up to you to determine what the exact interaction will be, but a variety of sequences and timing

schemes should be allowed. Your approach should also be able to accommodate left-hand turns protected by left-hand green arrow lights. In addition:

a. Combinations of individual signals that would result in crashes should not be allowed.

b. Every intersection on the map must have traffic lights (there are not any stop signs, overpasses, or other variations). All intersections will be 4-way: there are no “T” intersections, nor one-way roads.

c. Students must be able to design each intersection with or without the option to have sensors that detect whether any cars are present in a given lane. The intersection’s lights’ behavior should be able to change based on the input from these sensors, though the exact behavior of this feature is up to you.

3. Based on the map created, and the intersection timing schemes, the students must be able to simulate traffic flows on the map. The traffic levels should be conveyed visually to the user in a real-time manner, as they emerge in the simulation. The current state of the intersections’ traffic lights should also be depicted visually, and updated when they change. It is up to you how to present this information to the students using your program. For example, you may choose to depict individual cars, or to use a more abstract representation.

4. Students should be able to change the traffic density that enters the map on a given road. For example, it should be possible to create a busy road, or a seldom used one, and any variation in between. How exactly this is declared by the user and depicted by the system is up to you. Broadly, the tool should be easy to use, and should encourage students to explore multiple alternative approaches. Students should be able to observe any problems with their map’s timing scheme, alter it, and see the results of their changes on the traffic patterns. This program is not meant to be an exact, scientific simulation, but aims to simply illustrate the basic effect that traffic signal timing has on traffic. If you wish, you may assume that you will be able to reuse an existing software package that provides relevant mathematical functionality such as statistical distributions, random number generators, and queuing theory.

You may add additional features and details to the simulation, if you think that they would support these goals.

Your design will primarily be evaluated based on its elegance and clarity – both in its overall solution and envisioned implementation structure.

Desired Outcomes

Your work on this design should focus on two main issues:

1. You must design the interaction that the students will have with the system. You should design the basic appearance of the program, as well as the means by which the user creates a map, sets traffic timing schemes, and views traffic simulations.

2. You must design the basic structure of the code that will be used to implement this system. You should focus on the important design decisions that form the foundation of the implementation, and work those out to the depth you believe is needed

Deliver an architecture document containing at least the context, functional and information viewpoints, according to the provided architecture document template. Remember that a viewpoint may consist of multiple views.

The result of this session should be: *the ability to present your design to a team of software developers who will be tasked with actually implementing it.* The level of competency you can expect is that of students who just completed a basic computer science or software engineering undergraduate degree. You do not need to create a complete, final diagram to be handed off to an implementation team. But you should have an understanding that is sufficient to explain how to implement the system to competent developers, without requiring them to make many high-level design decisions on their own.

To simulate this hand-off, you will be asked to make a documentation of the model you have developed, including a glossary, description and rationale.

Timeline

- 30 min – Assignment Explanation
  - 15 min - Questions
- 2 hours - Designing
  - 5 min - Break
- 45 min - Make documentation
- 5 min – Fill in the provided questionnaire

Can choose own 5 min break

Appendix D

Card Log Sheet

Timestamp				C	A	R	T	Player	Timestamp of removal
0:00:00									0:00:00

Appendix E

Design Support Game Questionnaire

Thank you for using the Design Support Card game during your design session. To improve the card game I would like to ask you a few short questions concerning your experience using the cards.

Group: ___ Name: _______________

1. Did you intuitively connect the symbols on the cards to their meanings, or did they lead to confusion? If so, please explain what symbols would be more appropriate.

2. Did the questions table help you understand the cards meaning, or did you not use it at all? Please explain why.

3. Were the cards helpful during your discussion and a natural addition, or did it feel like they were getting in the way? Please explain.

4. Did the cards help you think of design issues and solutions you otherwise wouldn't have come up with? Please explain.

5. Which was your favorite card and why?

Optional: Do you have any suggestions to improve the Design Support Card game?

____
____
____

APPENDIX F

Discourse Analysis Coding Schemes

Physical cards coding scheme

Main cards	Technique cards
Context Problem Solution	Constraint Risk Trade-off Assumption

Reasoning process coding scheme

Category	Code	Description
Design reasoning techniques		
Problem structuring	PROB	Identifying and discussing the key issues of the design
Trade-off analysis	TA	Weighing the pros and cons concerning the design to come to a decision
Option generation	OG	Discussing the options available for design solutions
Assumption analysis	AA	Questioning the premises of the requirements and context, the validity of arguments
Constraint analysis	CA	Identifying constraints in the design and how these constraints influence the design
Risk analysis	RA	Identifying risks in the design and how to mitigate those risks
Design reasoning elements		
Pro	PRO	Argument for a proposition
Con	CON	Argument against a proposition
Constraint	C	A restriction on the condition of the design
Assumption	A	A supposition that is taken for granted or questioned to determine the validity
Risk	R	An aspect of the design which is identified to be a threat to achieving the system goals
Design option	DO	A solution option
Design issue	DI	A design problem
Design decision	DD	Evaluating solution options to make a decision

Coding rules

- A design decision follows a design issue
- A design issue is not a requirement, but a stated problem that needs to be solved
- Option generation needs at least two options
- A trade-off analysis needs at least one pro and con
- Assumption is questioning a premise, whether something is right or not, is it valid. There is no confusion about something, but something is assumed

- Problem structuring is a specification, should something be added, does it fit the requirements, is there something they missed, what does it mean, need more knowledge. is a confusion that needs to be resolved.
- Context space, when they go back to the requirements, identify key issues (important things that have to go into the design), prioritization
- Solution space, when talking about the solution to a problem, specifying it
- Problem space, analysing the problems and formulating the design problems.

APPENDIX G

Irvine Assignment Identified Constraints - Student

Irvine assignment requirements	T1	T2	T3	T4	T5	T6
Crashes are not allowed	x	x		x	x	x
Intersection must have traffic lights		x				x
All intersections are four-way		x	x	x	x	x
Intersections have optional sensors		x	x			x
Left hand turns are protected by left hand green arrow lights					x	
Students can change traffic density	x			x		
Accommodate at least six intersections	x	x	x		x	x
Allow for roads of varying lengths		x	x		x	x
students must be able to create a visual map of the area				x		
Students must be able to describe traffic light behaviour				x	x	
Students should see the results of changes made to traffic patterns					x	
Other	5	2	0	2	1	1
	C1	**C2**	**C3**	**C4**	**C5**	**C6**
Crashes are not allowed			x	x	x	x
Intersection must have traffic lights		x	x	x	x	x
All intersections are four-way		x	x	x	x	x
Intersections have optional sensors	x	x	x	x		x
Left hand turns are protected by left hand green arrow lights			x			x
Students can change traffic density	x	x	x		x	x
Accommodate at least six intersections		x	x	x	x	x
Allow for roads of varying lengths			x		x	
students must be able to create a visual map of the area		x				
Students must be able to describe traffic light behaviour		x				
No one way roads			x			
Traffic moves in real-time						x
Other	2	1	0	0	2	0

Appendix H

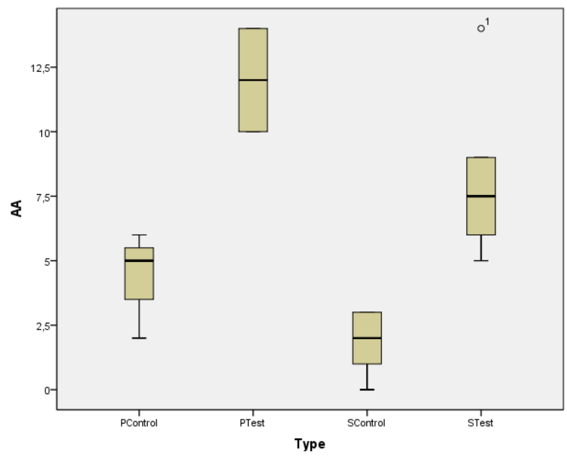
Irvine Assignment Identified Constraints - Professional

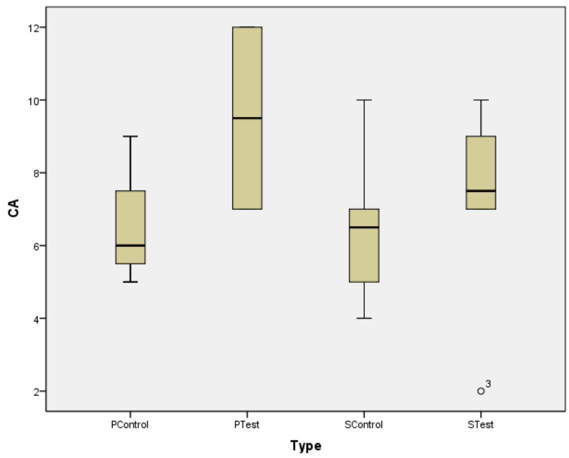
Irvine assignment requirements	PT1	PT2	
Crashes are not allowed	x	x	
All intersections are four-way	x	x	
Intersections have optional sensors		x	
Left hand turns are protected by left hand green arrow lights	x	x	
Students can change traffic density	x		
Accommodate at least six intersections	x		
Allow for roads of varying lengths	x		
Other	3	2	
	PC1	**PC2**	**PC3**
Crashes are not allowed	x	x	x
Intersection must have traffic lights		x	x
All intersections are four-way			x
Intersections have optional sensors	x	x	x
Left hand turns are protected by left hand green arrow lights	x	x	x
Students can change traffic density	x		x
Accommodate at least six intersections	x	x	x
Allow for roads of varying lengths			x
Students must be able to describe traffic light behaviour		x	x
Other	2	0	1

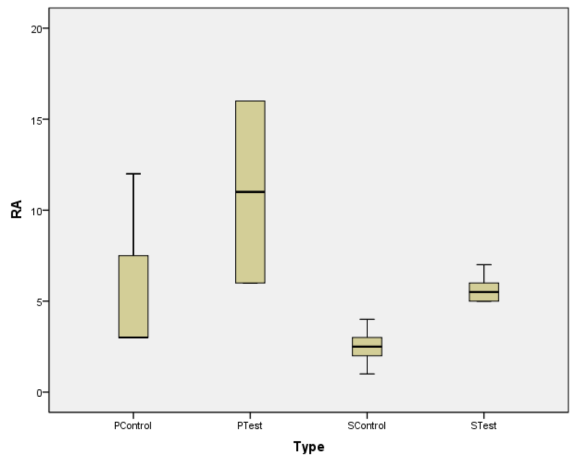
Appendix I

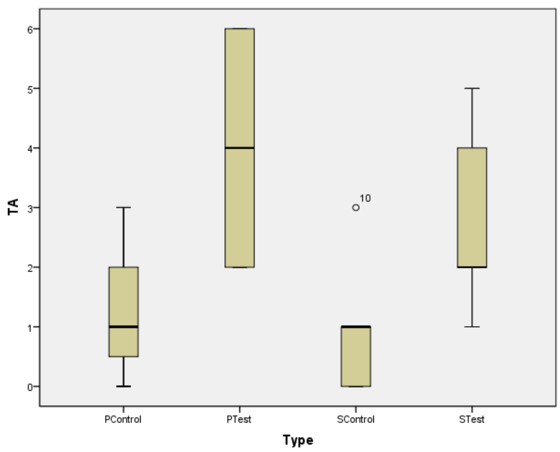
Professional and Student Results Boxplots

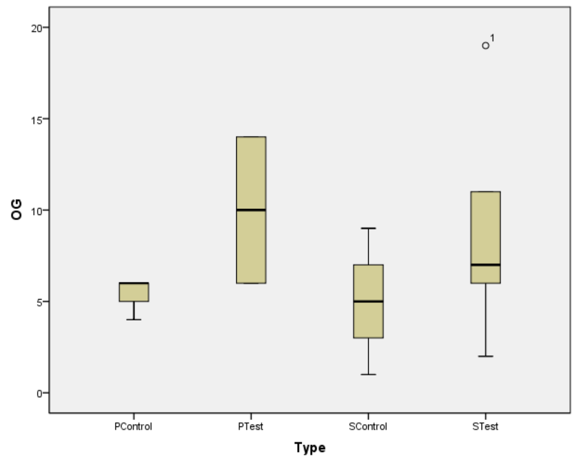
Design Reasoning Techniques

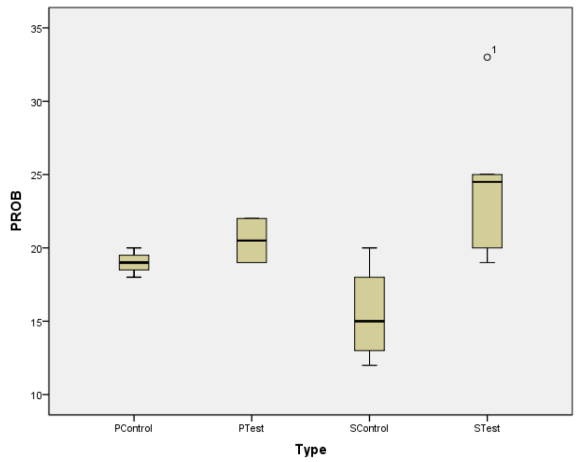




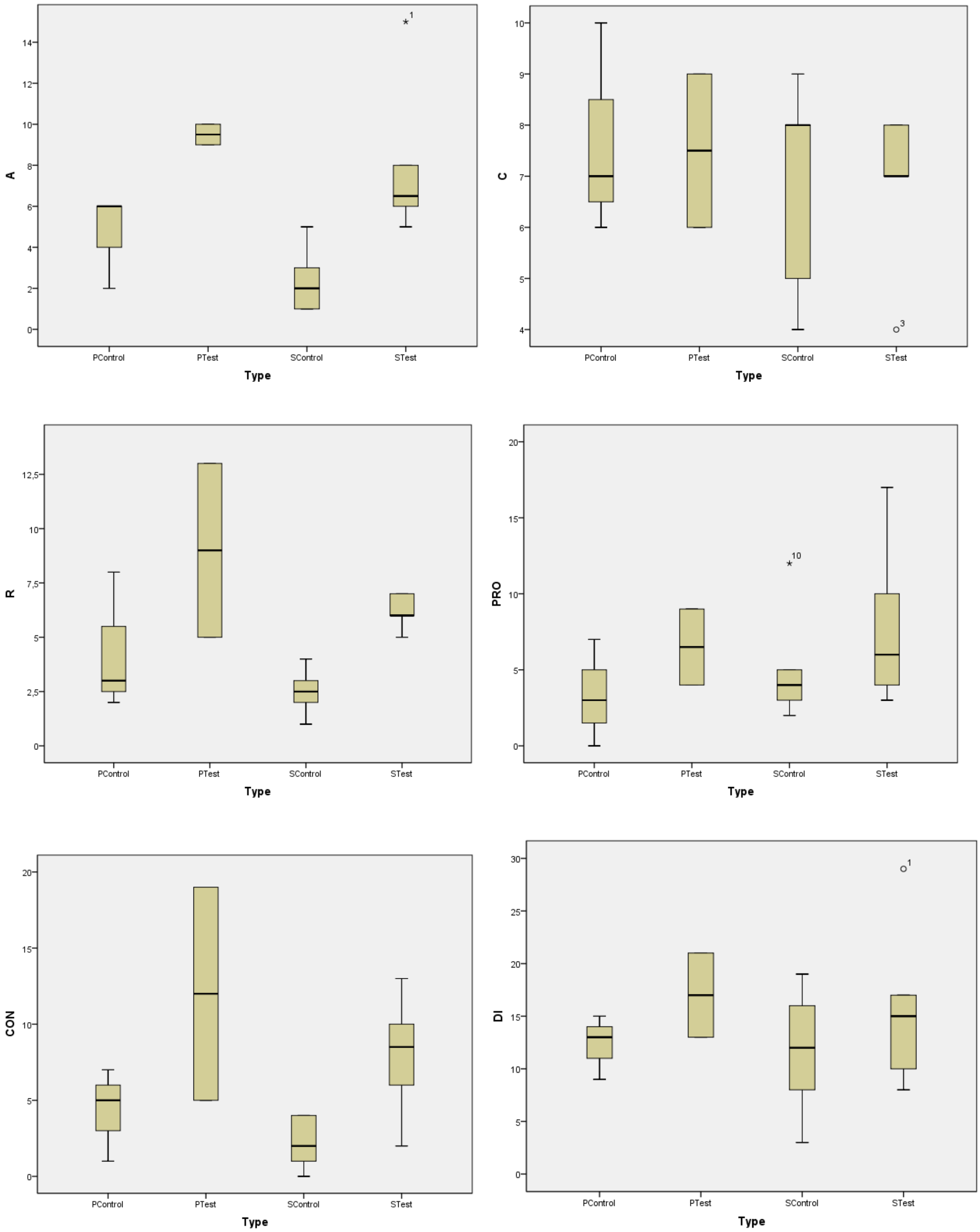


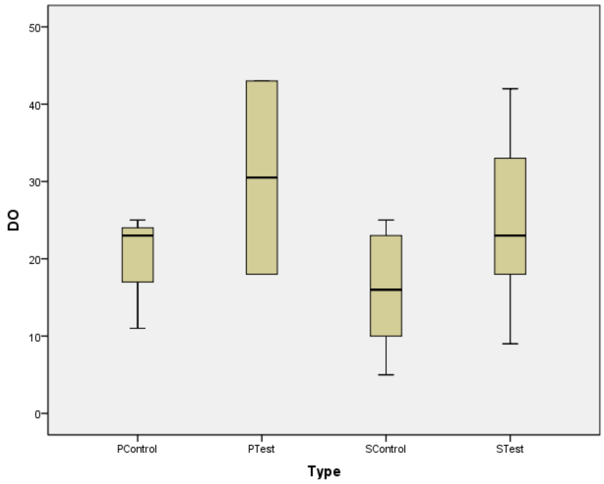


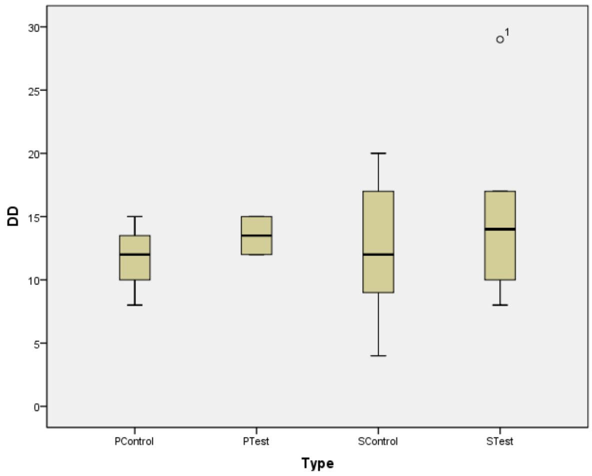




Design Reasoning Elements




50
40
30
20
10
0
DO
PControl
PTest
SControl
STest
Type


30
25
20
15
10
5
0
DD
1
PControl
PTest
SControl
STest
Type

Appendix J

Student Experiment Results: Levene's test and Shapiro Wilk test

Design reasoning techniques & elements	Levene's test	Student Test Shapiro Wilk	Student Control Shapiro Wilk
Assumption analysis	.161	.306	.421
Constraint analysis	.731	.204	.700
Risk analysis	.533	.091	.820
Trade-off analysis	.231	.078	.212
Option generation	.256	.460	.985
Problem structuring	.526	.333	.677
Assumption	.245	.020	.212
Constraint	.212	.020	.094
Risk	.341	.212	.820
Pros	.322	.197	.027
Cons	.187	.953	.425
Design issues	.674	.351	.987
Design options	.227	.877	.852
Design decisions	.566	.236	.987

Student Experiment Results: Independent T-test

Design reasoning techniques & elements	*t-statistic*	df (degrees of freedom)	T-test *p*-value	T-test *p*-value(no outliers)
Assumption analysis	4,568	10	.001	.001
Constraint analysis	0,470	10	.648	.148
Risk analysis	5,836	10	.001	
Trade-off analysis	2,193	10	.053	0.018
Option generation	1,375	10	.199	.413
Problem structuring	3,724	10	.004	.003
Assumption				
Constraint				
Risk	6,957	10	.001	
Pros				
Cons	3,511	10	.006	
Design issues	1,041	10	.322	.673
Design options	1,522	10	.159	
Design decisions	0,769	10	.460	.933

Student Experiment Results: Mann Whitney test

Design reasoning techniques & elements	Mann Whitney U	Mann Whitney *p*-value
Assumption analysis	-	-
Constraint analysis	-	-
Risk analysis	-	-
Trade-off analysis	5,500	.040
Option generation	-	-
Problem structuring	-	-
Assumption	0,500	.005
Constraint	13,500	.451
Risk	-	-
Pros	12,500	.369
Cons	-	-
Design issues	-	-
Design options	-	-
Design decisions	-	-

Student Experiment Results: Mean and Standard Deviation

Design reasoning techniques & elements	Student Test Mean	Student Control Mean	Student Test Std.dev	Student Control Std.dev
Assumption analysis	8.2	1.8	3,189	1,169
Constraint analysis	7.2	6.5	2,787	2,074
Risk analysis	5.7	2.5	0,816	1,049
Trade-off analysis	2.7	1	1,506	1,095
Option generation	8.7	5	5,854	2,898
Problem structuring	24.3	15.5	4,967	3,017
Assumption	7.8	2.3	3,656	1,506
Constraint	6.8	7	1,472	2
Risk	6.2	2.5	0,753	1,049
Pros	7.7	5	5,317	3,578
Cons	8	2.2	3,742	1,602
Design issues	15.7	11.7	7,474	5,715
Design options	24.7	15.8	11,961	7,679
Design decisions	15.3	12.3	7,659	5,715

Appendix K

Professional Experiment Results: Levene's test and Shapiro Wilk

Design reasoning techniques & elements	Levene's test	Student Test Shapiro Wilk	Student Control Shapiro Wilk
Assumption analysis	.528	-	.463
Constraint analysis	.228	-	.463
Risk analysis	.495	-	.001
Trade-off analysis	.184	-	.637
Option generation	.002	-	.001
Problem structuring	.148	-	1
Assumption	.112	-	.001
Constraint	.935	-	.463
Risk	.173	-	.298
Pros	.970	-	.843
Cons	.019	-	.637
Design issues	.184	-	.637
Design options	.044	-	.253
Design decisions	.540	-	.843

Professional Experiment Results: Mean and Standard Deviation

	Professional Test Mean	Professional Control Mean	Professional Test Std.dev	Professional Control Std.dev
Assumption analysis	12	4.3	2,828	2,082
Constraint analysis	9.5	6.7	3,536	2,082
Risk analysis	11	6	7,071	5,196
Trade-off analysis	4	1.3	2,828	1,528
Option generation	10	5.3	5,657	1,155
Problem structuring	20.5	19	2,121	1
Assumption	9.5	4.7	0,707	2,309
Constraint	7.5	7.7	2,121	2,082
Risk	9	4.3	5,657	3,215
Pros	6.5	3.3	3,536	3,512
Cons	12	4.3	9,899	3,055
Design issues	17	12.3	5,657	3,055
Design options	30.5	19.7	17,678	7,572
Design decisions	13.5	11.7	2,121	3,512