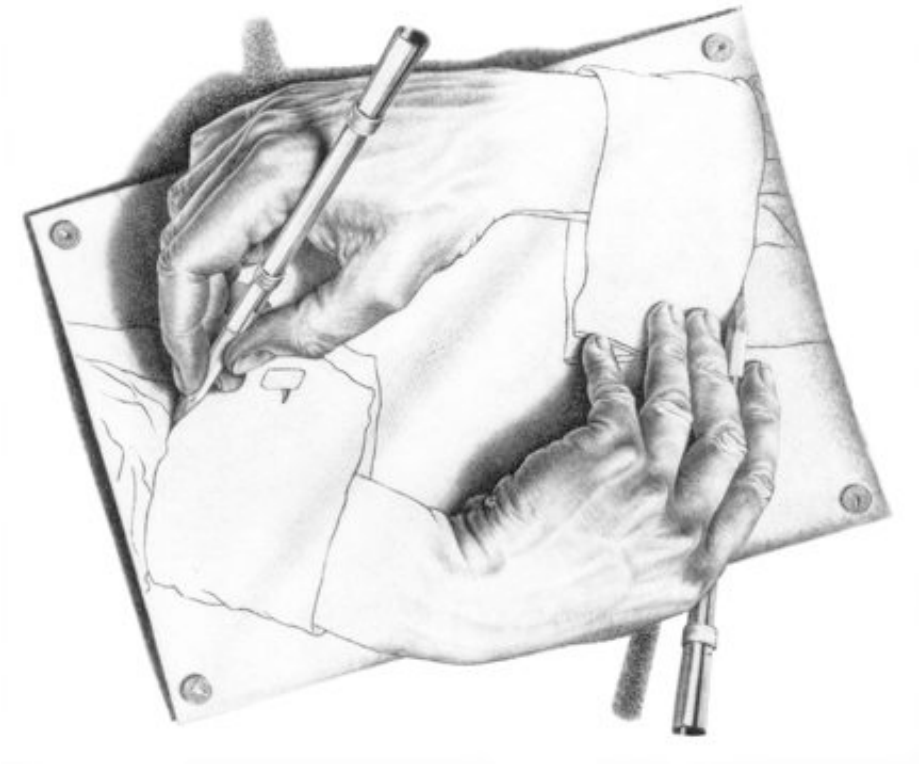
Prototyping

A stepping-stone to more successful Prototyping



Master's Thesis

Arne Timmerman

4 september 2008

Prototyping

A stepping-stone to more successful Prototyping

Thesis

submitted in fulfillment of the
requirements for the degree of

Master of Science in
Software Engineering

by
ing. Arne Timmerman
born in Zierikzee, the Netherlands

Universiteit van Amsterdam

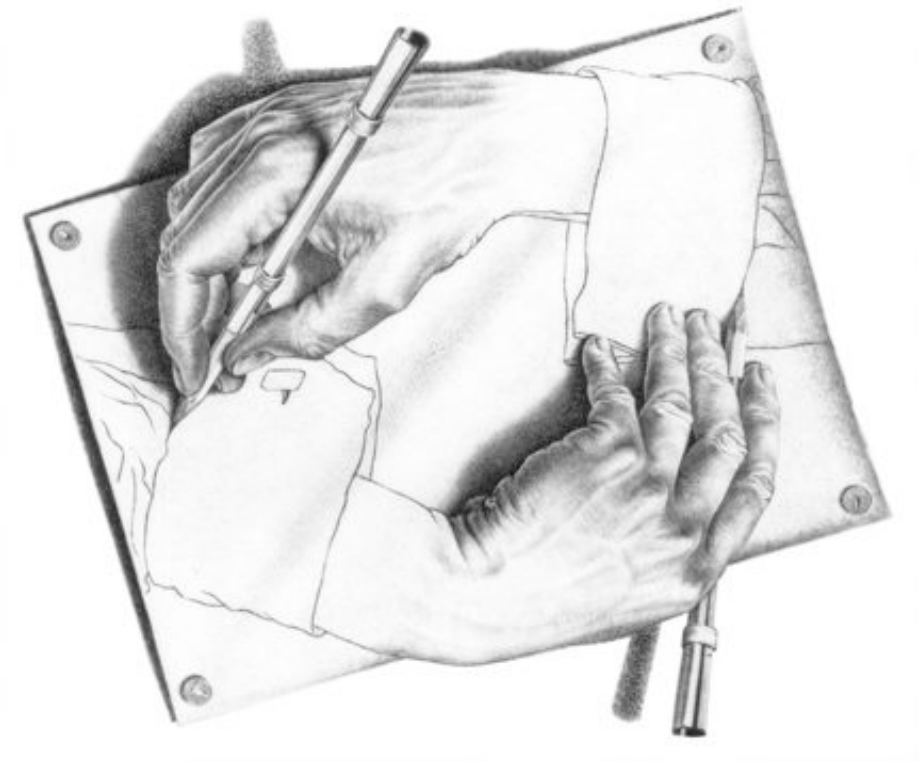
drs. Hans Dekkers
prof. dr. Paul Klint

drs. Bram Vranken
ir. Edwin Essenius

What I hear, I forget. What I see, I remember. What I do, I understand!
(Lao Tse)

Scientific Article

A stepping-stone to more successful Prototyping



A stepping-stone to more successful Prototyping

Arne Timmerman
Logica - Working Tomorrow
University of Amsterdam - Master Software Engineering

Abstract—On average only 29% of software projects are completed on time and within budget. The biggest single cause of failing software projects are incomplete or incorrect requirements. The participation of users in the development of software is seen as an important software success factor.

A well-known method that facilitates user participation is prototyping. It is widely believed that prototyping increases the quality of software requirements. Although the adoption rate of prototyping is high and the number of prototyping methods increases, empirical research shows that the effect of prototyping on software quality is actually moderate. There is a lack of research that addresses the question of how prototyping is best applied in order to increase the success of software.

In this research a theoretical framework is proposed that addresses this question. The completion of a two-day experiment with 18 participants, shows that the cooperation between the developer and users in a prototyping process, compared to the demonstration of a prototype, increases the number of mentioned ideas per person 4 times. The use of a low-fidelity prototype, instead of a high-fidelity prototype, in a cooperative prototyping method increases the number of ideas per person even further, with a factor 1.5, in this particular experiment.

Index Terms—prototyping, requirements engineering, user participation, knowledge sharing, software success.

1 Introduction

On average only 29% of software projects are completed on time and within budget (Standish Group, 2004). The development of software fails because of unexpected complexity, but most of all because the software does not meet the expectations of future users (Ernst & Young, 2008).

The biggest single cause of failing software projects are incomplete or incorrect requirements. It is estimated that requirements errors account for 60% of defects in software (McConnell, 2004). Fixing and removing defects in the software maintenance has a cost increase of 1:100 in comparison to correcting defects in the requirements phase (Boehm et al., 1984).

User participation in the early stages of software development is seen as one of the most important software success factors. The participation of future users helps developers to get a better understanding of desired functionality that results in higher requirements quality (Kujala, 2003) and increased user satisfaction (McKeen et al., 1994; Hartwick & Barki, 1994).

1.1 Prototyping

A well-known method that facilitates user participation is prototyping (Robertson & Robertson, 1999). The International Organization for Standardization defines a prototype as a *representation of all or part of a product or system that, although limited in some way, can be used for evaluation* (ISO 13407). Prototyping helps to validate proposed requirements or a future software solution in early stages of software development. It is widely believed that prototyping increases the quality of requirements (Gordon & Bieman, 1995; Beynon-Davies et al., 1999).

The adoption rate of prototyping in software development is high. The use of prototyping is embedded in the requirements phase of most software projects (Robertson & Robertson, 1999; McConnell, 2004) and classified as a best practice in requirements literature (Young, 2004). Hardgrave indicated, based on a survey, that prototyping is used in 71% of software projects in 1995. Neill & Laplante (2003) confirmed the high adoption of prototyping by a web-survey with 194 participants from different companies, and showed that some form of prototyping is applied in 60% of software projects.

Although the adoption rate of prototyping is high, Jones (1996) estimated that prototyping only reduces requirements creep with 10 to 25%. Empirical research shows that the use of prototyping reduces software defects with a maximum of 27%, a reduction of 35,6 to 26,0 software defects per 1 million lines of code (MacCormack et al., 2003). In contrast to the costs of the application of prototyping, an estimated increase of 5 to 10% in comparison to the total costs of a software development project (Beynon-Davies et al., 1999), it is questionable if increased software quality compensates the effort.

In the past decades a high number of prototyping methods were introduced. The prototyping methods differ from the position in the software development life cycle (Boehm, 1988; Carr & Verner, 1997), the amount of detail of the prototype (Rudd et al., 1996) and the way users participate (Bødker & Grønbæk, 1989; Snyder, 2003). Although the adoption rate of prototyping is high and the number of prototyping methods increases, there is a lack of research that addresses the question of how prototyping is best applied in order to increase the quality of requirements and raise the success of software.

1.2 Related work

It is widely believed that user participation in a prototyping process is a critical factor for the success of software (Lichter et al., 1994; Kimmond, 1995; Gordon & Bieman, 1995; Beynon-Davies et al., 1999, i.e.). It is, on the other hand, remarkable that research is often not grounded by theory (Beynon-Davies et al., 1999), lacks empirical validation (Markus & Mao, 2004; Mattia & Weistroffer, 2008) and shows contradictory results (Kujala, 2003; Heinbokel et al., 1996). Although there is a lot of research on the subject of user participation and prototyping, there are five issues that deserve attention.

First of all, research on the effect of user participation and prototyping on software success often make uses of contradictory or weak definitions of the success of software. The measurement of software success diverges from software quality (Kujala, 2005), user satisfaction (McKeen et al., 1994) to the adoption and use of software (Hartwick & Barki, 1994). An often cited research to the relation between user-developer links and the success of software (Keil & Carmel, 1995) leave the definition of software success to the subjective opinion of the examined companies. The consequence of different definitions of software success is that it is hard to compare and combine the results of these studies.

Second, research on user participation shows contradictory results. Based on a study of a number of case studies Kujala (2003) concludes that the amount of user participation positively correlates with the quality of requirements and software success. On the other hand Heinbokel et al. (1996) state, based on interviews with 200 managers in 29 software projects, that user participation in software development negatively influences the quality of software and results in low flexibility and innovation of software.

Third, there is a lack of research in literature that compares the influence of different types of user participation on software success. A great contribution on this subject is done by Hartwick & Barki. They investigated the influence of different user participation activities in the development of software on the psychological involvement of users and adoption of software (Barki & Hartwick, 1989, 1994; Hartwick & Barki, 2001). Nevertheless, none of these studies examine the influence of different participation activities on software quality. He (2004) proposed a theoretical model that explains the influence of different user participation types on software quality, however this model has not been empirically validated.

Fourth, research on prototyping often does not examine the effect of different dimensions of a prototyping process on the success of software (Gordon & Bieman, 1995, i.e.). Studies that do address the influence of different prototyping methods on software success are limited to a certain dimension of prototyping, for example participation (Grønbæk, 1989) or representation (Engelbrektsson, 2002). Although a combination of research on different prototyping dimensions could possibly show interesting insights, it is not possible to study the influence of different combinations of prototyping dimensions on software success purely based on existing research, because the contradictory use of the definition of software success.

The fifth issue with research on prototyping is the fact that research on the influence of one prototyping dimension on software success is often empirical weak. Engelbrektsson (2002) compared the number of comments of users in a prototype session with a high level and low level of detail. The author concluded that there are no significant differences. The result of the experiment although is questionable. The difference in the level of detail that is used are images of a tram and a real tram. It is questionable if this difference in *level of detail* is actually a difference in level of detail to the participants of the experiment. Besides that, the appearance of a tram is common knowledge to almost everyone, the number and type of comments is not likely to differ from representation.

1.3 Research question

The central research question of this article is stated as follows: *How should prototyping be applied in order to increase the success of software?*

The remainder of this article is organized as follows. To be able to understand how prototyping should be applied in order to increase software success it is necessary to understand the differences between how prototyping *can* be applied. Section 2 points out the different dimensions that are variable in a prototyping process. Based on study of existing theory and empirical research a theoretical framework that describes the influence of the prototyping dimensions on software success is proposed in section 3. Validation of this framework through an executed experiment is described in section 4 and 5. Section 6 wraps up with a discussion on the results of this research and states an answer to the central research question of this article.

2 Prototyping dimensions

In the past decades, many forms of prototyping were introduced. Floyd (1984) was one of the first that categorized prototyping methods. He distinguished prototyping methods based on different goals: exploratory prototyping, experimental prototyping and evolutionary prototyping. This article focuses on exploratory prototyping methods. A definition of prototyping from Boar (1984) describes exactly the type of prototyping that is subject to this research:

> *Prototyping is a specific strategy for performing requirements definitions wherin user needs are extracted, presented, and succesively refined by building a working model of the ultimate system quickly and in its working context.* (Boar, 1984)

The definition of Boar emphasizes two aspects of a prototyping process. First the fact that the purpose of prototyping is to elicit requirements from users. In the second place, that prototyping uses a working model of the future software. These two aspects of prototyping can be viewed as two dimensions of prototyping process: the way users participate in the process in order to elicit requirements and the way that the working model is represented compared to the future system. These dimensions are elaborated on in this section.

2.1 The participation of users

There is a lot of discussion of how and how often users should participate in software development. Proponents of Participatory Design principles promote full user participation in order to create a collaborative environment where users and developers work together on the development of software (Greenbaum & Kyng, 1992). Opponents of these principles state that the involvement of users results in low flexibility of software, low team effectiveness (Heinbokel et al., 1996) and limits the discussion on a software system to one single solution (Davis et al., 2006).

Mattia & Weistroffer (2008) proposed four types of user participation in the development of software, based on four paradigms that are introduced by Hirschheim & Klein (1989). The four types of user participation of Mattia & Weistroffer are used to distinguish between four types of user participation in a prototyping process: prototyping by demonstration, prototyping by testing, cooperative prototyping and user-led prototyping.

Prototyping by demonstration In the most traditional form of prototyping, a prototype is demonstrated to stakeholders of a software system (Boehm et al., 1984; Pomberger et al., 1991). The software developer is responsible for the development of a prototype and gives a demonstration in order to validate his view on the future software.

Prototyping by testing The goal of prototyping by testing is to validate a prototype by letting users perform tasks using the prototype in order to observe problems that users experience with a prototype (Nielsen, 1993). The role of the software developer is less dominant than in the demonstration of a prototype, due to the fact that the user executes the tasks and the developer acts as a facilitator.

Cooperative prototyping In a cooperative prototyping process, the software developer and users jointly work on the development of a prototype (Trigg et al., 1991). It is a prerequisite that the developer and users work in a emancipated relation and are both able to make modifications to the prototype (Grønbæk, 1991).

User-led prototyping The development of a prototype normally belongs partially or fully to a software developer. User-friendly prototyping tools are used in a user-led prototyping process to facilitate full prototype development by the users (Mattia & Weistroffer, 2008). The role of the software developer is to guide the development process and distribute prototyping tools.

2.2 Representation of the prototype

As stated by Boar and the International Organization of Standardization a prototype is defined as *a working model of the ultimate system* or *a representation of all or part of a product or system that is limited in some way*. Budde et al. (1992) distinguish between horizontal and vertical prototyping. A horizontal prototype is a prototype that is a representation of a specific layer of a software system, e.g., the user interface layer of software. A vertical prototype is a representation of a chosen part of the future software that is implemented completely.

Although the terms horizontal and vertical are common in prototyping literature, Snyder (2003) replaces the adjectives by the nouns breadth and depth. They are more applicable, because a prototype often varies between horizontal or vertical. Snyder defines breadth as the percentage of the future functionality that is represented in a prototype and depth as the extent to which this functionality is fleshed out and functional.

The depth of a prototype is often associated with the level of detail that a prototype represents. Literature discerns between two extremes in the detailedness of prototypes, low-fidelity and high-fidelity prototypes (Rudd et al., 1996; Robertson & Robertson, 1999). A low-fidelity prototype is a prototype that does not has the fidelity of a real software application and is often sketched on paper. Unlike a low-fidelity prototype, a high-fidelity prototype provides full fidelity in comparison to the future software (Rudd et al., 1996).

3 Theoretical framework

As stated in section 1.2, the success of software is often defined in a contradictory or weak fashion in research on the subject of prototyping. The definition of software success varies from the quality of software (Jones, 1996; MacCormack et al., 2003), software usability (Gordon & Bieman, 1995) to user satisfaction (Kimmond, 1995).

In order to understand the influence of prototyping dimensions on the success of software, paragraph 3.1 of this section defines software success and emphasizes the impact of knowledge sharing on software quality. In paragraph 3.2 the influence of user participation on software success is elaborated, paragraph 3.3 explains the influence of the representation of a prototype on software success.

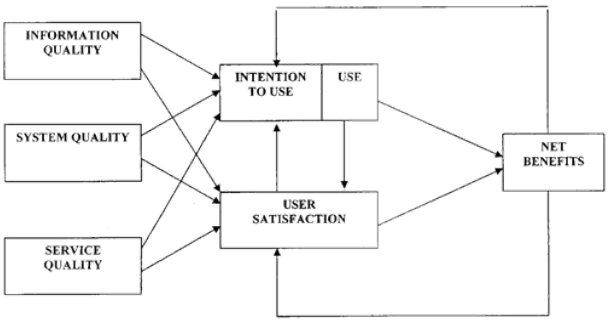


Figure 1: *The Updated Delone & McLean Success Model* (Delone & McLean, 2003).

3.1 Defining software success

The DeLone and McLean Model of Information Systems Success is a widely cited and accepted model to define software success (Delone & McLean, 2003). The model distinguishes between the efficiency and effectiveness of a software solution. Delone & McLean (2003) define the quality of a software solution as the technical quality of software, the quality of the representation of information and service to users (figure 1). The effectiveness of software is defined as the satisfaction of users and use of software. In contrast to the original success model (DeLone & McLean, 1992), the Updated Delone & McLean Success Model distinguishes between the intention to use software and the actual software use (Delone & McLean, 2003). Eventually, the success of software determines the actual net benefits of software.

According to the success model of Delone & McLean, the quality of software is an important determinant of successful software. Seddon & Kiew (1996) showed, based on a survey distributed between 104 users of a recent implemented software system, that the quality of software explains 70% of variation of user satisfaction. Kujala (2003) confirmed, based on analysis of multiple case studies, that user satisfaction and acceptance of software correlates with the quality of software.

3.1.1 Knowledge impacts on software quality

The quality of software is largely determined by the notion of desired functionality of software by developers (Keil & Carmel, 1995; Standish Group, 2004). An important source of knowledge of the software domain and the activities that the future software will support are the users (Béguin, 2003; Pirinen & Pekkola, 2006). The development of software requirements intensive knowledge sharing between software developers and users (He, 2004).

He (2004) pretends that obtaining knowledge from users in software development determines the quality of requirements. The quality of software depends largely on the quality of requirements (Boehm & Basili, 2001). Tiwana (2003) demonstrates, based on a study of 209 software projects, that domain knowledge of software developers through requirements elicitation with users has a large impact on software quality. The study emphasizes that requirements elicitation is most beneficial in the development of new software.

Knowledge sharing is twofold. He (2004) separates knowledge sharing into the acquisition of knowledge and knowledge exploitation. He defines knowledge acquisition as *the process by which a team accesses and applies its members' individual knowledge for a team task* and knowledge exploitation as *the process that brings collective knowledge to bear a problem*. In order to facilitate successful knowledge delegation it is necessary to obtain knowledge from users, but it is equally important that the obtained knowledge is actually exploited by developers (He, 2004).

3.2 The influence of user participation

In order to obtain knowledge from future users of a software system, the participation of users is inevitable. The participation of users in the development of software is necessary to be able to obtain domain knowledge and get a better understanding of the desired functionality of software (Pirinen & Pekkola, 2006). He (2004) states that the amount and nature of user participation determines the acquisition and exploitation of knowledge (figure 2).

McKeen & Guimaraes show, based on a study of 151 software projects, that the type of user participation correlates with the satisfaction of users. McKeen & Guimaraes (1997) notice that user satisfaction, through software quality, mainly benefits from participation activities when the responsibility of a user in the activity is high, e.g. cooperating in the development of a prototype. This leads to the following statement:

> *Statement 1:* The quality of software increases when users have a high degree of responsibility in the development of a prototype.

The result of McKeen & Guimaraes (1997) can be explained by the Zone of Proximal Development. The theory states that there is a difference in *the distance between the actual developmental level as determined by independent problem solving and the level of potential development*

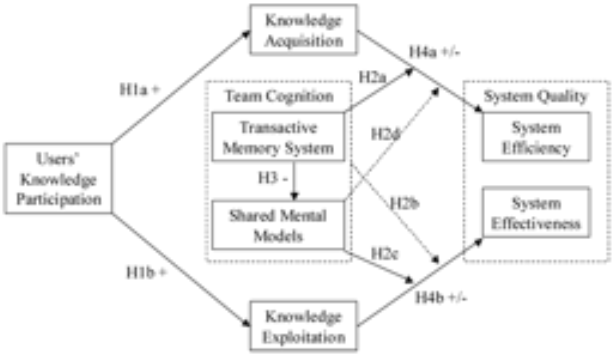


Figure 2: The relationship between knowledge acquisition and exploitation with software quality. (He, 2004).

as determined through problem solving under adult guidance, or in collaboration with more capable peers (Engeström, 1987). In order to increase the acquisition of knowledge and the effectiveness of user participation the developer should be working in cooperation with the user (Bødker & Grønbæk, 1991).

Even if user participation in a development process increases the acquisition of knowledge, the exploitation of knowledge is still a challenging task (He, 2004). Clement & den Besselaar (1993) observed that, although users participated in the development of software, comments and ideas from users are often ignored and criticized by software developers because of their different backgrounds and lack of understanding. A way to stimulate the exploitation of knowledge and mutual understanding is to establish an environment where user participants actively work together with software developers in the development of software (Bannon, 1991; He, 2004).

> *Statement 2:* The quality of acquisition and exploitation of knowledge in a prototyping process increases when the developer works in cooperation with users.

In section 2.1 a categorization of different types of user participation in a prototyping process is proposed. Table 1 shows the relationship between the statements that are mentioned in this paragraph and the prototyping participation types.

3.3 The influence of representation

In contrast to other methods that facilitate user participation in software development prototyping makes intensive use of a conceptual model of the future software system. A prototype makes it possible to discuss the desired functionality of software, by actually seeing a conceptual model of the future software, in early stages of development. A representation of a future software system can be of great help to users in understanding proposed requirements, compared to textual written requirements (Robertson & Robertson, 1999).

3.3.1 The Gradient of Resistance

The development of a prototype, on the other hand, is a time consuming activity. Beynon-Davies et al. (1999) estimate that the investment of designing and evaluating a prototype can take up to 5 to 10% of the total software development costs. A pitfall of the effort a designer puts in the development of a prototype is that he gets blind for other software solutions and resists changes of the prototype. Bowers & Pycock (1994) elaborate this theory as The Gradient of Resistance.

Statement	Demonstration	Test	Cooperative	User-led
Responsibility	–	-	+	++
Cooperation	–	+	++	–

Table 1: The statements applied to the types of user participation in a prototyping process.

The effect of the gradient of resistance in a prototyping process can be that comments and ideas from users are neglected by developers, which might cause that shared knowledge of users is not exploited. The gradient of resistance might even lead to resistance by users to share useful knowledge, because they feel that shared knowledge is not exploited.

> *Statement 3:* The more effort a developer puts in the development of a prototype, the more a prototype limits the acquisition and exploitation of users' knowledge.

3.3.2 The mind-set of users

The fact that a prototype represents a software solution in early stages of development, can cause that this concept is seen as the only solution to the software problem by users and no other solutions are considered (Davis et al., 2006). The unintentional mind-set is a threat to knowledge acquisition by users, because all the knowledge that is shared is biased by the prototype.

The mind-set of users is seen in research of Gordon & Bieman (1995), where the analysis of 39 case studies shows that prototyping barely leads to new functionality in software. Engelbrektsson (2002) confirms, based on an experiment with 23 participants, that demonstrating a prototype mainly results in discussions on details of the prototype. An early prototype apparently limits the knowledge that is shared by users.

> *Statement 4:* The more detail and functionality a prototype represents, the more shared knowledge of users' is biased by the prototype.

The time that the development of a prototype costs and the degree that a prototype looks like a definitive solution towards users, is mainly determined by the level of detail and amount of functionality that a prototype represents. Rettig (1994); Snyder (2003) state that the use of a low-fidelity prototype, on paper, results in more, and more precise requirements than the use of a high-fidelity prototype. It is therefore stated that:

> *Statement 5:* The quality of acquisition and exploitation of knowledge in a prototyping process is higher, when a low-fidelity prototype is used instead of a high-fidelity prototype.

3.4 A theoretical model

In this section literature on the influence of prototyping on software success is studied and clarified. An outline of the study is shown in figure 3. This theoretical model distinguishes between reasonable, empirical validated, connections and statements that are mentioned in this study (dashed lines).

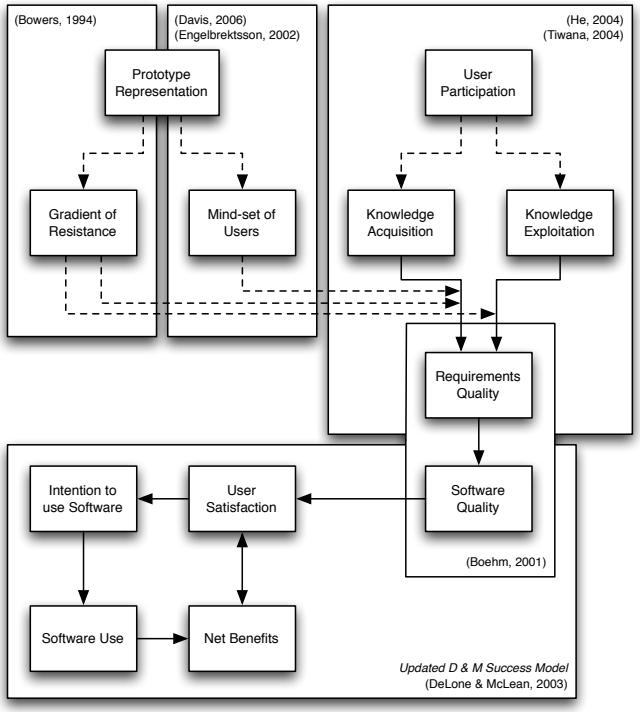


Figure 3: A model of the theoretical framework, based on existing theories and research. The dashed lines represents proposed statements.

4 Executing an experiment

In order to validate the statements in the proposed theoretical model, a two-day experiment was set up. Paragraph 4.1 explains the prototyping dimensions that were chosen as independent variables in the experiment. The dependent variables that were measured in the experiment are described in paragraph 4.2, hypotheses about the measures are stated in paragraph 4.3. The effort that is invested in in order to control threats to validity of the experiment is worked out in paragraph 4.4.

4.1 Comparing prototyping methods

In order to examine the influence of different types of user participation and prototype representation on software success two types were chosen from each independent variable. The types of user participation that were tested in the experiment are prototyping by demonstration and cooperative prototyping (according to table 1), the representation of the prototype differed between a paper prototype and interactive prototype.

The independent variables were interchanged between four different groups of participants. The participants of two groups participated in the demonstration of an interactive prototype. In the third group the same prototype was used, but instead of a prototype demonstration a cooperative prototyping method was applied. In the fourth group the same cooperative user participation method was applied, but instead of the interactive prototype a paper prototype was used.

4.2 Measuring software success

To measure the influence of the independent variables on software success the number and usefulness of ideas was measured. As pointed out in section 3.1 the success of software is largely determined by the quality of software. In order to be able to create high quality software it is crucial that knowledge of future users is obtained successfully. The number of ideas is an often used measure to indicate the quality of knowledge sharing in a software development process (Engelbrektsson, 2002).

An important flaw of using the number of ideas as an indicator for the quality of knowledge sharing, although, is that it does not account for the value of an idea. Therefore, the ideas were judged by the designer of the prototype on a Likert scale from 1 (totally not important) to 5 (very important).

4.3 Hypotheses

Adapted from statements that are proposed in this article, two hypotheses according to the described independent and dependent variables were formulated. The hypotheses are stated as follows:

> *Hypothesis 1:* The application of cooperative prototyping results in more, and more valuable ideas compared to the demonstration of a prototype.
>
> *Hypothesis 2:* The use of a paper prototype in a prototyping session results in more, and more valuable ideas compared to the use of an interactive prototype.

4.4 Controlling threats to validity

In the execution of an experiment it is important that the measured differences in dependent variables can be related as much as possible to the tested independent variables (Norvig, 2007).

4.4.1 Participant selection

Participants of the experiment were approached personally and by e-mail within the organization where this research was accomplished. Of all approached employees, 18 persons were willing to participate. Their average age was 35 years. The 14 men and 4 women comprised 6 graduate students, 2 secretaries and 10 consultants. There were no participants that had extensive experience with prototyping.

A matching procedure was used to assign the participants to the different groups (Goodwin, 2005). The procedure was chosen instead of a repeated-measures, because the chance of a learning effect in this experiment is high due to the fact that a prototype of the same software system is used in all types of sessions.

A matching procedure is normally based on variables, like the age, gender and profession of participants. Research shows that the amount and type of comments that a user gives in a prototype session is strongly influenced

by previous experience that a user has with the type of software that is equal to the prototype (Engelbrektsson, 2002). In order to create equal groups, the previous experience of participants with software equal to the prototype was questioned by a Likert scale from 1 to 5 to the participants, and used in the matching procedure. The matched variables are shown in table 2.

4.4.2 The prototype

A design team within the organization where this research is accomplished provided an existing prototype that could be used in this experiment. The main criteria for the prototype was that, due to the nature of the participants, almost everyone could be a future user of the software. The interactive prototype that eventually was used is a job application website, where a job seeker can create and publish a curriculum vitae and search for jobs.

In order to create a paper prototype from the existing interactive prototype, parts of the prototype were printed and cut out into pieces. The appearance of the prototype actually had a quite a lot of of detail, as it was a print from the original prototype. Seen afterwards, it might have been better to sketch the prototype on paper. The influence of the representation on knowledge sharing is therefore expected to be low.

The number and value of ideas is not only expected to differ slightly between prototyping sessions because of the limitation of the paper prototype, but also due to the low complexity of the prototype. McKeen & Guimaraes (1997) state that the participation of users in the development of software mainly affects software success when the complexity of the software system is high.

The reason that the complexity of the prototype, regarding the participants of the experiment, is considered low is twofold. First, the prototype that is used is a job application website and the participants of the experiment are all working in a consulting company where jobs are acquired via a job application website. Second, each participant was asked after a prototyping session how he or she experienced the session and if the prototype was complex or easy to understand. All the participants indicated that the prototype had a low complexity.

4.4.3 The facilitator

The prototyping sessions in the experiment were led by a facilitator. In order to measure the influence of the gradient of resistance on knowledge sharing, as described in the theoretical model (figure 3), one or more sessions in the experiment should be led by the developer of the prototype. Unfortunately, the developer was not able to facilitate the experiment. The sessions were eventually facilitated by an experienced facilitator.

Due to the fact that all prototyping sessions were led by an independent facilitator, the influence of the gradient of resistance was not measured in the experiment. This ensures, on the other hand, that measured differences between the influence of the prototype representation on software success can be fully allocated to the quality of knowledge acquisition (section 3.3).

After the first prototype demonstration session was executed the facilitator and observant noticed individually that the session suffered from bias. The role of the facilitator in the session was too prevailing, in a way that the facilitator directs the participants to ideas mentioned earlier in cooperative sessions. Due the expected influence of the facilitator on the results of the experiment, another experienced facilitator led the other prototype demonstration session. The ideas of the 5 participants of the biased prototyping sessions were omitted from the results in this article.

4.4.4 The observant

A reasonable threat to validity in a controlled experiment is observer bias (Goodwin, 2005). The observer in an experiment is susceptible to wishful thinking, which means that the observer, unintentionally, writes down whatever corresponds to hypothesized results of the experiment. A method that was used in the experiment in order to reduce observer bias is a formalized form that specifies a checklist of variables to observe (Goodwin, 2005).

4.4.5 Time

Time is an important factor in the experiment, because the number of ideas is likely to increase with the duration of a prototyping session. In order to compare the number of ideas between different sessions in a fair way, the time is supposed to be limited equally. Although it can actually be interesting to examine the progression of the number of ideas in a prototyping session while time elapses, the available time in the experiment was limited to one hour per session because of the limited time of the experiment.

Group	# Persons	Age	Male	Experience
Cooperative - High	3	38	67%	2
Cooperative - Low	4	37	50%	2
Demonstration - High	5	37	80%	2,4
Demonstration - High	6	32	100%	2,3

Table 2: Age, percentage of male and experience with job application websites of participants, per type of session.

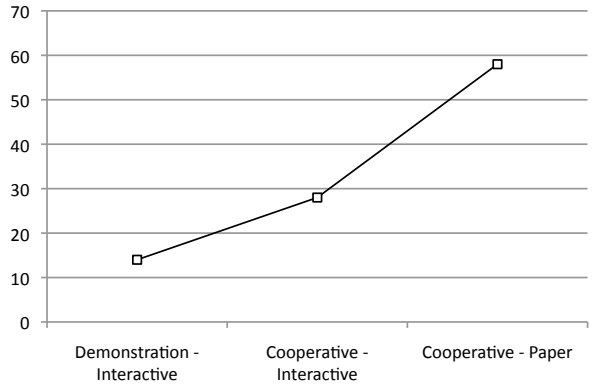


Figure 4: The total number of unique ideas, per prototyping method. The ideas are combined and normalized from the different sessions that were executed of each type. Note that the number of participants differ.

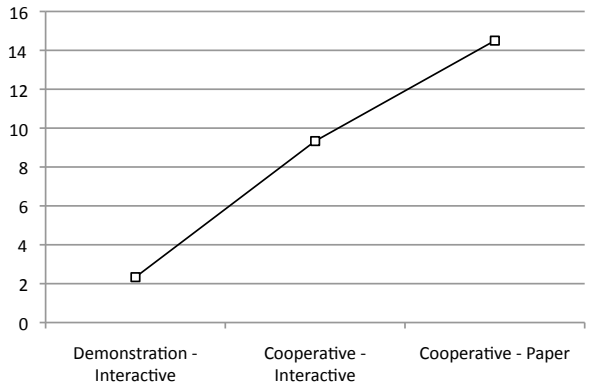


Figure 5: The average number of ideas per participant, per prototyping method.

4.4.6 Procedure

In order to reduce the learning-effect from the facilitator as he gets experienced with the prototype and leading the prototyping sessions, a protocol was defined. The protocol describes the workflow of the sessions and describes example questions that the facilitator can use in the sessions. The protocol was based on a prototyping protocol that was proposed by Beyer & Holtzblatt (1998).

In short, the workflow of the sessions was divided in four phases. In phase one the participants filled in a survey about previous experience with job application websites to encourage people to think about desired functionality of a job application website without seeing the prototype. In phase two the facilitator explained about the purpose of the session and discussed the survey with the participant. The third phase of the sessions was all about using or demonstrating the prototype. Participants were encouraged in all types of sessions to express all kinds of problems and ideas for the prototype. A session ended in the fourth phase with a wrap-up of the session.

5 Results

A total of 94 unique ideas was proposed by the 13 participants in the experiment. The prototype demonstration session with 6 participants resulted in 14 unique ideas, the 3 participants of the cooperative prototyping sessions with an interactive prototype proposed 28 ideas and the cooperative prototyping sessions with a paper prototype produced 58 unique ideas (figure 4).

The average number of ideas per person is 2.3 in the prototype demonstration session, 9.3 in a cooperative prototyping session with an interactive prototype and 14.5 in a same type of session with a paper prototype (figure 5). The result shows that cooperative prototyping has an increase over 4 times of the average unique ideas per person in comparison to a prototype demonstration and the use of a paper prototype produces 55% more unique ideas per person over the use of an interactive prototype. A cooperative paper prototyping session produces 5 times more unique ideas per person in comparison to an interactive prototype demonstration.

In addition to the average number of ideas per participant, figure 6 shows the average value of ideas per participant. The figure shows that a person that participates in an interactive prototype demonstration mentioned ideas with a value of 2.4, participants in a cooperative prototyping session proposed ideas with a value of 3.1 and when a paper prototype is used, the average value increased to 3.3. The results indicate that the application of cooperative prototyping increased the value of ideas with 29%, compared to the prototype demonstration and that the use of a paper prototype increased the average value of ideas per person with 7%.

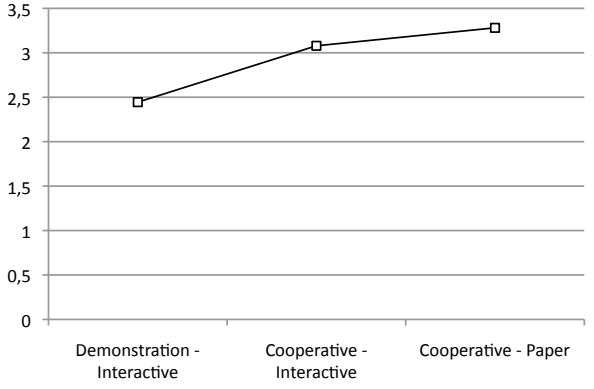


Figure 6: The average value of ideas per participant, per prototyping method.

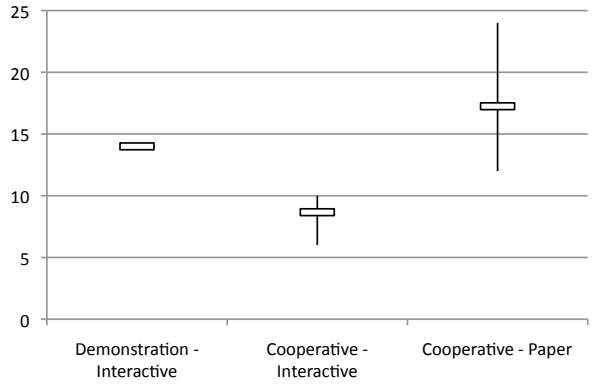


Figure 7: The minimum, mean and maximum of the number of ideas that were mentioned in the different single prototyping sessions. The prototype demonstration was executed in one session with multiple participants, and therefore displays only one value according to the number of ideas.

When the number of ideas per session, independent of the number of participants, is compared it appears that a single cooperative prototyping session results in 38% less ideas than prototyping demonstration session with an interactive. A single paper prototyping session however produced on average 99% more ideas compared to a identical session with an interactive prototype, and still 23% more ideas than a demonstration session with the same prototype (figure 7).

The average value of ideas mentioned per session differs from a score of 2.8 in the prototype demonstration session, to 3.1 in the cooperative interactive prototyping session and 3.3 in the session with a paper prototype (figure 8). The differences in scores correspond with an increase of 7% and 11%.

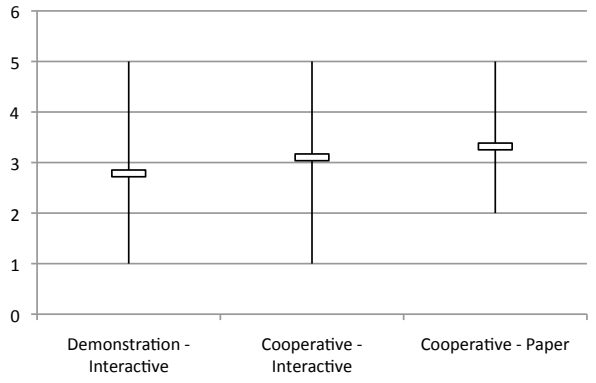


Figure 8: The minimum, mean and maximum of the value of ideas that were mentioned in one single prototyping session.

6 Discussion

The results of the experiment indicate that increasing the responsibility of users (McKeen & Guimaraes, 1997) and cooperation between the developer and users (Bannon, 1991; Bødker & Grønbæk, 1991) in a prototyping session increases the amount and value of ideas mentioned by a participant. Although it is not possible to distinguish between the effect of user responsibility and cooperation on knowledge sharing, the increased knowledge acquisition is expected to have a positive effect on the quality of requirements and software.

Although the results of the experiment indicate that the number of ideas per person is higher in a cooperative session than a demonstration session, it appears that a single cooperative prototyping session produces less ideas than one prototype demonstration. This result can be explained by the fact that the number of participants in these types of sessions differ.

Regarding to the time that is necessary to perform cooperative prototyping sessions, the time to execute this type of sessions increases with the number of participants, in contrast to the demonstration of a prototype. The increase of time, however, mainly affects the developer. The time that users participate is equal, independent of the type of participation. It is thus hard to compare the effect of different prototyping participation types on software success, regarding time, and thus cost.

According to Rettig (1994); Davis et al. (2006), the experiment indicates that the use of a prototype affects knowledge sharing. As Davis et al. stated: *What actually appears to be happening is that prototypes tend to focus the discussion on the artifact (mock-up, prototype). This then prevents information that cannot be reflected (visualized, experienced) in the displayed artifact from being gathered.* The results of the experiment show that knowledge acquisition is particularly limited when an interactive prototype is used. The use of a paper prototype in the experiment increased the number and value of ideas that were mentioned by a participant in a prototyping session, even though the paper prototype that was used still represented a lot of detail. It is however, not possible to assign the influence of the prototype representation completely to the mind-set of users, because knowledge sharing is affected by a lot of other factors, like the proactiveness of a person.

Although the experiment shows that the use of a paper prototype produces more, and more valuable ideas than the use of an interactive prototype, it is not stated that the use of an interactive prototype should be discarded. Rudd et al. (1996) state that *low-fidelity prototypes should be used in early stages to generate ideas and high-fidelity in late phases to find problems*. In this research the difference in the types of ideas that were mentioned was not examined. It is possible that, according to Rudd et al., there is a difference in the type of knowledge that is produced when working with different representations of a prototype.

6.1 External validity

The results of the experiment show a number of significant connections between the type of user participation or representation of the prototype and, i.e. the average number of ideas mentioned by a participant. It is however questionable if these results can be generalized.

First of all, most of the participants in the experiment have great affinity with software which can explain the high number of ideas that were mentioned in some prototyping sessions. The amount of ideas that were proposed in the cooperative prototyping sessions with a paper prototype, for example, suffers from a great variance between participants. This could be caused by the participation of a person in one of these sessions who, due to great affinity with software, mentioned a lot of ideas. However, it must be mentioned that the minimum number of ideas in sessions with a paper prototype is still higher than the same type of sessions with an interactive prototype.

Second, the prototype that was used in the experiment had a low complexity. McKeen & Guimaraes (1997) state that the complexity of software can have great influence on the effect of user participation on software success. The complexity of the prototype therefore possibly influences knowledge sharing.

6.2 Final words

As stated in the introduction, the effect of prototyping on software quality as shown in existing research is moderate. This article shows that, in order to increase the quality of software by prototyping it is necessary to stimulate knowledge sharing. The results of the experiment indicate that the application of cooperative prototyping and the use of a paper prototype does have a positive influence on the amount and value of knowledge that is shared. The moderate results of MacCormack et al. (2003); Jones (1996) can possibly be explained by the results of this article, because the mentioned research applies prototyping by demonstration and probably uses interactive prototypes.

The main purpose of this article was to answer the question *How should prototyping be applied in order to increase the success of software?* In this article it is indicated that, in order to achieve software success through prototyping, both the responsibility of users and the cooperation between the developer and users in a prototyping process should be encouraged. Besides that, the experiment indicates that a paper prototype is likely to contribute more to software success, than an interactive prototype.

7 Future work

This article shows some interesting insights, but to be able to state a well-grounded answer to the research question of this article, future research is required.

First of all, as mentioned in paragraph 6.1, there is a lot to say about the validity of the experiment that was performed. Future research should investigate the hypotheses that were stated in this article in an experiment with more participants, but most of all with participants that have low affinity with software.

Second, the prototype for a future experiment should be selected carefully. The prototype that was used in this experiment had a low complexity and, besides that, the paper prototype had a lot of detail. Future research should put more effort in the investigation of the influence of complexity and representation of a prototype on knowledge acquisition and exploitation. It is especially valuable that research on this particular subject examines differences between the types of ideas mentioned, as proposed by Rudd et al. (1996).

Third, the types of user participation that were examined in the experiment were limited to prototyping by demonstration and cooperative prototyping. The influence of the other two user participation types, prototyping by testing and user-led prototyping, should be studied in future research.

Last of all, section 6 states that it is hard to compare cooperative prototyping sessions with a prototype demonstration, due to the different number of participants. Future research should set up an experiment where multiple prototype demonstrations are executed and compared to cooperative prototyping sessions with the same total number of participants.

8 Acknowledgements

This article is the result of the research project that was carried out in fulfillment of the requirements for the degree of Master of Science in Software Engineering. The research project was performed at Logica in cooperation with the University of Amsterdam.

I would like to thank a lot of people that helped me during this research. First and foremost, I would like to thank my supervisor from the University of Amsterdam, Hans Dekkers, who invested a lot of time in guiding my research. I can truly state that without his guidance, the experiment would not have been accomplished and this article would never have been written.

Second, I am grateful to all people at Logica that helped and supported me, especially my mentors Bram Vranken and Edwin Essenius and all the participants of my experiment.

Third, I would like to thank all the people that I have interviewed at Logica, Belastingdienst and Het Marinebedrijf.

Fourth, I would like to thank the teachers at the Master Software Engineering who taught me so much the past year. I am especially grateful to Paul Griffioen who helped me out in the execution of the experiment.

Last of all, even though I normally skip this part, I am truly grateful to my parents who made it possible for me to accomplish this study in many ways, and my girlfriend who has been very patient and helpful to me the past year.

REFERENCES

Bannon, L. (1991). From human factors to human actors: the role of psychology and human-computer interaction studies in system design. In J. Greenbaum & M. Kyng (Eds.), *Design at Work: Cooperative Design of Computer Systems* (pp. 25–44). Mahwah, NJ, USA: Lawrence Erlbaum Associates, Inc.

Barki, H. & Hartwick, J. (1989). Rethinking the concept of user involvement. *MIS Quarterly*, *13*(1), 53–63.

Barki, H. & Hartwick, J. (1994). Measuring user participation, user involvement and user attitude. *MIS Quarterly*, *18*(1), 59–82.

Béguin, P. (2003). Design as a mutual learning process between users and designers. *Interacting with Computers*, *15*(5), 709–730.

Beyer, H. & Holtzblatt, K. (1998). *Contextual Design: Defining Customer-centered Systems*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Beynon-Davies, P., Tudhope, D., & Mackay, H. (1999). Information systems prototyping in practice. *Journal of Information Technology*, *14*(1), 107–120.

Boar, B. H. (1984). *Application Prototyping: a Requirements Definition Strategy for the 80s*. New York, NY, USA: John Wiley & Sons, Inc.

Bødker, S. & Grønbæk, K. (1989). Cooperative prototyping experiments - users and designers envision a dental case record system. In *Proceedings of the first EC-CSCW '89*, (pp. 343–357)., London, UK.

Bødker, S. & Grønbæk, K. (1991). Cooperative prototyping: Users and designers in mutual activity. *International Journal of Man-Machine Studies*, *34*(3), 453–478.

Boehm, B. & Basili, V. (2001). Software defect reduction top 10 list. *IEEE Computer Society*, *34*, 135–137.

Boehm, B. W. (1988). A spiral model of software development and enhancement. In *IEEE Computer*, volume 21, (pp. 61–72).

Boehm, B. W., Gray, T. E., & Seewaldt, T. (1984). Prototyping vs. specifying: A multi-project experiment. In *ICSE '84: Proceedings of the 7th international conference on Software engineering*, (pp. 473–484)., Piscataway, NJ, USA. IEEE Press.

Bowers, J. & Pycock, J. (1994). Talking through design: Requirements and resistance in cooperative prototyping. In *CHI '94: Proceedings of the SIGCHI conference on Human factors in computing systems*, (pp. 299–305)., New York, NY, USA. ACM.

Budde, R., Kautz, K., Kuhlenkamp, K., & Zullighoven, H. (1992). What is prototyping? *Information Technology & People*, *6*(1 + 2), 89–95.

Carr, M. & Verner, J. (1997). Prototyping and software development approaches.

Clement, A. & den Besselaar, P. V. (1993). A retrospective look at pd projects. *Communications of the ACM*, *36*(6), 29–37.

Davis, A., Dieste, O., Hickey, A., Juristo, N., & Moreno, A. M. (2006). Effectiveness of requirements elicitation techniques: Empirical results derived from a systematic review. In *RE '06: Proceedings of the 14th IEEE International Requirements Engineering Conference (RE'06)*, (pp. 176–185)., Washington, DC, USA. IEEE Computer Society.

DeLone, W. & McLean, E. (1992). Information Systems Success: The Quest for the Dependent Variable. *Information Systems Research*, *3*(1), 60–95.

Delone, W. H. & McLean, E. R. (2003). The delone and mclean model of information systems success: A ten-year update. *J. Manage. Inf. Syst.*, *19*(4), 9–30.

Engelbrektsson, P. (2002). Effects of product experience and product representations in focus group interviews. *Journal of Engineering Design*, *13*(3), 215–221.

Engeström, Y. (1987). Learning by expanding. *Helsinki: Orienta-Konsultit*.

Ernst & Young (2008). Ict barometer. Technical report.

Floyd, C. (1984). A systematic look at prototyping. In Budde, R., Kuhlenkamp, K., Mathiassen, L., & Zullighoven, H. (Eds.), *Approaches to Prototyping*, (pp. 1–17)., Heidelberg. Springer-Verlag.

Goodwin, J. C. (2005). *Research in Psychology: Methods and Design* (4 ed.). John Wiley & Sons, Inc.

Gordon, V. S. & Bieman, J. M. (1995). Rapid prototyping: Lessons learned. *IEEE Software*, *12*(1), 85–95.

Greenbaum, J. & Kyng, M. (1992). *Design at Work: Cooperative Design of Computer Systems*. Lawrence Erlbaum Associates, Inc. Mahwah, NJ, USA.

Grønbæk, K. (1989). Rapid Prototyping with Fourth Generation Systems - an Empirical Study. *Office: Technology and People*, *5*(2), 105–125.

Grønbæk, K. (1991). *Prototyping and Active User Involvement in System Development: Towards a Cooperative Prototyping Approach*. PhD thesis, Aarhus University.

Hardgrave, B. (1995). When to prototype: Decision variables used in industry. *Information and Software Technology*, *37*(2), 113–118.

Hartwick, J. & Barki, H. (1994). Explaining the role of user participation in information system use. In *Management Science*, volume 40, (pp. 440–465).

Hartwick, J. & Barki, H. (2001). Communication as a dimension of user participation. *IEEE Transactions on Professional Communication*, *44*(1), 21–36.

He, J. (2004). Knowledge impacts of user participation: a cognitive perspective. *Proceedings of the 2004 SIGMIS conference on Computer personnel research: Careers, culture, and ethics in a networked environment*, 1–7.

Heinbokel, T., Sonnentag, S., Frese, M., Stolte, W., & Brodbeck, F. C. (1996). Don't underestimate the problems of user centredness in software development projects - there are many! *Behaviour & Information Technology*, *15*(4), 225–236.

Hirschheim, R. & Klein, H. K. (1989). Four paradigms of information systems development. *Communications of the ACM*, *32*(10), 1199–1216.

Jones, C. (1996). Strategies for managing requirements creep. *Computer*, *29*(6), 92–94.

Keil, M. & Carmel, E. (1995). Customer-developer links in software development. *Communications of the ACM*,

38(5), 33–44.

Kimmond, R. M. (1995). Survey into the acceptance of prototyping in software development. In *RSP '95: Proceedings of the Sixth IEEE International Workshop on Rapid System Prototyping (RSP'95)*, (pp. 147)., Washington, DC, USA. IEEE Computer Society.

Kujala, S. (2003). User involvement: a review of the benefits and challenges. *Behaviour & Information Technology*, *22*(1), 1–16.

Kujala, S. (2005). The role of user involvement in requirements quality and project succes. *Proceedings of International Conference on Requirements Engineering*, *13*.

Lichter, H., Schneider-Hufschmidt, M., & Zullighoven, H. (1994). Prototyping in industrial software projects-bridging the gap between theory and practice. In *IEEE Transactions on Software Engineering*, volume 20, (pp. 825–832).

MacCormack, A., Kemerer, C. F., Cusumano, M., & Crandall, B. (2003). Trade-offs between productivity and quality in selecting software development practices. *IEEE Software*, *20*(5), 78–85.

Markus, M. & Mao, J. (2004). Participation in development and implementation–updating an old, tired concept for today's is contexts. *Journal of the Association for Information Systems*, *5*(11-12), 514–544.

Mattia, A. & Weistroffer, H. (2008). Information system development: A categorical analysis of user participation approaches. *Hawaii International Conference on System Sciences, Proceedings of the 41st Annual*, 452–452.

McConnell, S. (2004). *Code Complete, Second Edition*. Redmond, WA, USA: Microsoft Press.

McKeen, J. D. & Guimaraes, T. (1997). Succesful strategies for user participation in systems development. *Journal of Management Information Systems*, *14*(2), 133–150.

McKeen, J. D., Guimaraes, T., & Wetherbe, J. C. (1994). The relationship between user participation and user satisfaction: An investigation of four contingency factors. In *IS Quarterly*, volume 18, (pp. 427–451).

Neill, C. J. & Laplante, P. A. (2003). Requirements engineering: The state of the practice. *IEEE Softw.*, *20*(6), 40–45.

Nielsen, J. (1993). *Usability Engineering*. Morgan Kaufmann.

Norvig, P. (2007). Warning signs in experimental design and interpretation.

Pirinen, A. & Pekkola, S. (2006). Different perspectives on learning in information system development. In *29th Information Systems Research Seminar in Scandinavia*.

Pomberger, G., Bischofberger, W., Kolb, D., Pree, W., & Schlemm, H. (1991). Prototyping-oriented software development - concepts and tools. *Structured Programming*, *12*, 43–60.

Rettig, M. (1994). Prototyping for tiny fingers. *Communications of the ACM*, *37*(4), 21–27.

Robertson, S. & Robertson, J. (1999). *Mastering the Requirements Process*. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co.

Rudd, J., Stern, K., & Isensee, S. (1996). Low vs. high-fidelity prototyping debate. *Interactions*, *3*(1), 76–85.

Seddon, P. & Kiew, M. (1996). A Partial Test and Development of Delone and Mclean's Model of IS Success. *Australasian Journal of Information Systems*, *4*(1).

Snyder, C. (2003). *Paper Prototyping: The Fast and Easy Way to Design and Refine User Interfaces*. Morgan Kaufmann.

Standish Group (2004). The chaos report. Technical report, Standish Group.

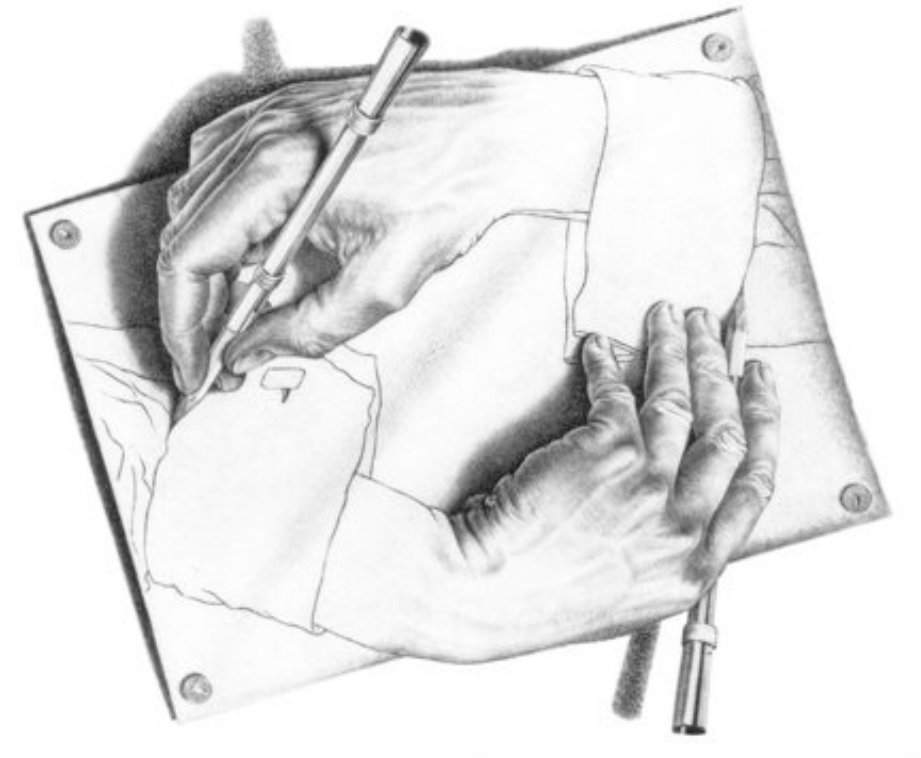
Tiwana, A. (2003). Knowledge partitioning in outsourced software development: A field study. *International Conference on Information Systems (ICIS, Seattle, Washington, 2003)*, 259–270.

Trigg, R. H., Bødker, S., & Grønbæk, K. (1991). Open-ended interaction in cooperative prototyping: a video-based analysis. *Scandinavian Journal of Information Systems*, *3*, 63–86.

Young, R. R. (2004). *The Requirements Engineering Handbook*. Artech House.

Appendices

(in Dutch)



Bijlage A

Protocol experiment

Het experiment wordt in twee dagen uitgevoerd met 18 verschillende personen. De deelnemers zijn verdeeld in vier groepen, waarin verschillende prototyping varianten worden getoetst. Het is van belang dat de prototyping sessies zoveel mogelijk gelijk zijn en alleen verschillen op het gebied van de participatie vorm en de representatie van het prototype.

Om te voorkomen dat er andere verschillen optreden tussen prototyping sessies, veroorzaakt door het *learning effect* of *bias* bij de facilitator, is er een protocol opgesteld waarin de uitvoering van een sessie is vastgelegd. Het protocol beschrijft de achtergrond en het doel van het prototype, de *workflow* van sessies en de manier waarop de facilitator moet omgaan met opmerkingen in het experiment. Het protocol is gebaseerd op een protocol voor prototyping in de methode *Contextual Design* (Beyer & Holtzblatt, 1998) en het rapport voor de usability test die met het prototype is uitgevoerd[1].

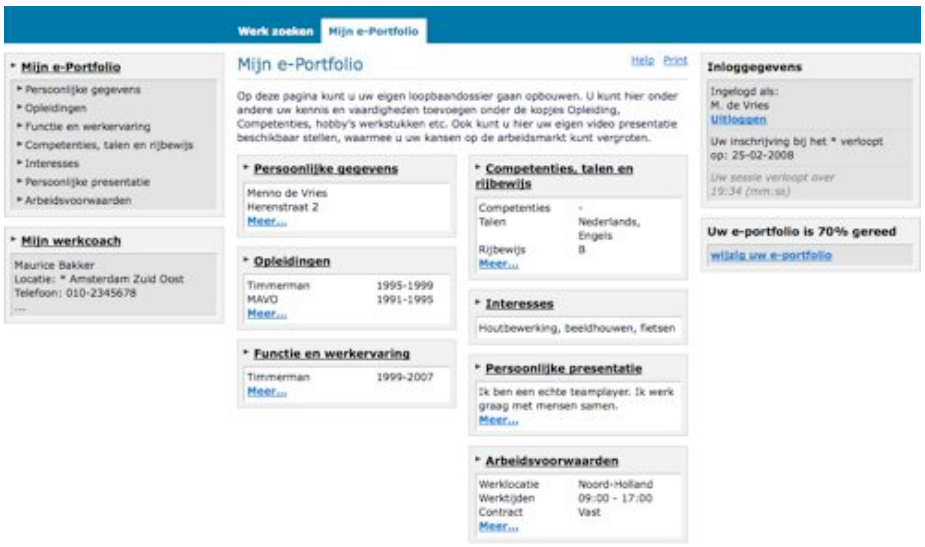
A.1 Het prototype

Het prototype dat wordt gebruikt voor het experiment is een bestaand prototype voor een webapplicatie waarin gebruikers een Curriculum Vitae (CV) kunnen toevoegen en zoeken naar vacatures (zie figuur A.1). Het prototype is gemaakt voor het uitvoeren van een usability test door het afstudeerbedrijf.

Het prototype is gebruikt om te toetsen of gebruikers overweg kunnen met de manier waarop in de webapplicatie gebruik wordt gemaakt van een CV. Het bijzondere aan de applicatie is dat er meerdere CV's kunnen worden aangemaakt die gebaseerd zijn op een selectie van informatie uit één portfolio waar een gebruiker bijvoorbeeld persoonlijke gegevens of werkervaring toe kan voegen.

Het prototype functioneert niet volledig en is beperkt tot het inzien, aanmaken en wijzigen van het portfolio en een of meerdere CV's. De functionaliteit in het prototype wordt gesimuleerd door middel van statische pagina's, de volgorde van het uitvoeren van taken ligt voor het functioneren van het prototype vast.

[1] Rapport Usability test - versie 3.0 - 26 juni 2008, Logica



Figuur A.1: Het prototype van de vacature website

A.2 De prototyping sessies

In het experiment worden vier verschillende prototyping type sessies getoetst: coöperatief prototyping met een hoog detail prototype, prototype demonstratie met een hoog detail prototype, coöperatief prototyping met een laag detail prototype en prototype demonstratie met een laag detail prototype. Het doel van de vier type sessies is gelijk:

> Het verkrijgen van inzicht in de problemen en fouten in het prototype en het verkrijgen van ideeën voor alternatieve oplossingen.

Het verschil tussen coöperatie en demonstratie

De participatie vorm in een prototyping sessie verschilt op het gebied van de verhouding tussen facilitator en deelnemer en de activiteiten die door beiden worden uitgevoerd. In de demonstratie van een prototype wordt de sessie door de facilitator geleid. De facilitator gebruikt het script (zie sectie A.3.3) gedurende de prototyping sessie als leidraad voor de demonstratie. Deelnemers aan de sessie worden vragen gesteld en worden gestimuleerd om problemen aan te geven in het prototype dat zij zien en alternatieve oplossingen voor te dragen.

In tegenstelling tot een prototype demonstratie is er in de coöperatieve prototyping methode sprake van een geëmancipeerde verhouding tussen specialist en gebruiker. Software specialisten en gebruikers werken actief samen om een prototype te ontwerpen en verbeteren (Trigg et al., 1991). In een coöperatieve sessie wordt er in mindere mate gebruik gemaakt van het script, de ervaring en ideeën van de gebruiker zijn leidend voor de taken die met het prototype worden uitgevoerd, gelijk aan de Contextual Design methode[2].

[2] *Don't create scripts - let the user's real work be the script.* (Beyer & Holtzblatt, 1998)

Het verschil tussen veel en weinig detail

De randvoorwaarden van de sessies waar coöperatief prototyping wordt toegepast verschillen afhankelijk van de mate van detail. In de coöperatieve sessies waar gebruik wordt gemaakt van een prototype met weinig detail, op papier, wordt de deelnemer gestimuleerd om op nieuw papier alternatieve oplossingen voor te stellen en het prototype samen met de facilitator te verbeteren (Beyer & Holtzblatt, 1998).

In coöperatieve prototyping sessies waar gebruik wordt gemaakt van een hoog gedetailleerd prototype is er niet de mogelijkheid om aanpassingen te doen aan het prototype. Het gebruikte prototype is dusdanig complex dat aanpassingen in de sessie door de facilitator die het prototype niet heeft ontwikkeld niet te verwezenlijken zijn. Het feit dat dit niet mogelijk is zorgt ervoor dat coöperatief prototyping niet volledig wordt uitgevoerd zoals de methode voorschrijft (Grønbæk, 1991). Desalniettemin is het interessant om te zien wat de *hands-on* participatie van gebruikers met een prototype toevoegt ten opzichte van een demonstratie en daarnaast verschilt met een paper prototyping sessie waarin wel directe aanpassingen kunnen worden gerealiseerd.

A.3 Workflow sessies

De structuur, *workflow*, van de sessies wordt vastgelegd in fases. De fasering van de prototyping sessies is gebaseerd op de methode Contextual Design. Beyer & Holtzblatt (1998) maken onderscheid tussen vier fases in een prototyping sessie: de introductie, transitie, interview en afronding.

A.3.1 Introductie

Tijdsduur: 15 minuten
Verantwoordelijke: facilitator

Een prototyping sessie start met het voorstellen door de facilitator en een korte uitleg over het doel van de sessie. In vervolg op het welkom wordt een enquête (zie hoofdstuk B) overhandigd aan de deelnemer(s). Het doel van de enquête is dat deelnemer(s) worden aangespoord om na te denken over ervaringen met het gebruik van vacature websites in het verleden en gewenste functionaliteit in het prototype, zonder het huidige ontwerp in te zien.

Als de deelnemer(s) de enquête hebben ingevuld wordt de deelnemer(s) gevraagd om kort toe te lichten wat de persoonlijke ervaring is met het plaatsen van een CV op internet en uit te leggen welke problemen ze hierbij hebben ervaren. De volgende vragen kunnen worden gebruikt om de sessie te vervolgen:

- Welke functionaliteit verwacht je op een vacature website?
- Welke functionaliteit vindt je belangrijk op een vacature website?
- Hoe is je ervaring met het aanmaken of plaatsen van een CV op Internet?
- Welke informatie vind jij belangrijk om op te nemen in je CV?

A.3.2 Transitie

Tijdsduur: 5 minuten
Verantwoordelijke: facilitator

In de introductie fase is het van belang dat de facilitator een aanknopingspunt zoekt voor het vervolg van de prototyping sessie, bijvoorbeeld een situatie uit de ervaring van de deelnemer(s). Het aanknopingspunt wordt gebruikt om het prototype te introduceren. Het is niet de bedoeling om tijdens de introductie een volledige *walk-through* van het prototype te doen, maar in een aantal korte zinnen uit te leggen welke functionaliteit het prototype representeert. De introductie van het prototype klinkt bijvoorbeeld alsvolgt:

> Deze website maakt het mogelijk om als werkzoekende een portfolio te maken van uw persoonlijke situatie en werkervaring. Het opgeslagen portfolio kan worden gebruikt om een CV aan te maken, die door werkgevers kan worden gebruikt om u een vacature aan te bieden en helpt in het zoeken van een passende vacature. In deze sessie is het prototype beperkt tot het inzien, aanmaken en wijzigen van een portfolio en CV.

Het prototype wordt altijd vanaf de volgende locatie gestart:
.../usabilitytest/opdr11/1_10_eportfolio.html

A.3.3 Prototype interview

Tijdsduur: 40 minuten
Verantwoordelijke: facilitator

Als het prototype is geïntroduceerd kan het demonstreren of werken met het prototype beginnen, afhankelijk van de participatie vorm of representatie van het prototype (zie sectie A.2). In de demonstratie van het prototype wordt in het experiment gebruik gemaakt van een script voor het prototype interview. Het script wordt gebruikt om de workflow van de verschillende prototyping sessies gelijk te houden.

In het prototype interview worden één voor één secties van het prototype behandeld. Op basis van het protocol voor de reeds uitgevoerde usability test zijn de secties geselecteerd: het bekijken, wijzigen en aanmaken van een portfolio en CV. In de demonstratie van het prototype vult de facilitator zijn eigen gegevens in het prototype in, let op dat het opslaan van de gegevens in het prototype niet wordt ondersteund.

Portfolio

In het portfolio is een overzicht van alle gegevens die mogelijk in een CV kunnen worden opgenomen. De gegevens zijn verdeeld in een aantal categorieën. De volgende vragen kunnen worden gesteld in een prototyping sessie:

- Vind je de categorieën in het portfolio duidelijk?
- Kan je zelf categorieën bedenken die ontbreken in deze lijst, of vind je een bepaalde categorie overbodig?

Persoonlijke gegevens

Een belangrijk onderdeel van het portfolio zijn de persoonlijke gegevens. In een prototyping sessie wordt het onderdeel gedemonstreerd door de gegevens van de facilitator in te vullen. Het doel is om door actief gebruik en personalisatie van het prototype deelnemer(s) aan te moedigen commentaar te geven. Mogelijke vragen zijn:

- Zijn er bepaalde velden die jij mist in het invullen van persoonlijke gegevens?
- Wat vind jij van de manier die in het prototype is gekozen om gegevens in te vullen?

Opleidingen

In het portfolio is het mogelijk om één of meerdere opleidingen op te slaan. In het experiment wordt het toevoegen van een nieuwe opleiding gedemonstreerd, de facilitator gebruikt hierbij een persoonlijke afgeronde opleiding. Het is belangrijk om in de demonstratie aan te geven dat er in de uiteindelijke software geen vrije invoer is voor de naam van de opleiding, maar gekozen moet worden uit een lijst. Als er met de muis over het vraagteken voor de vraag *Welke opleiding heeft u gevolgd?* wordt bewogen verschijnt een uitleg van deze toekomstige functionaliteit.

- Vind je dit een begrijpelijke manier om een opleiding toe te voegen aan je portfolio?
- Wat vind je van de mogelijkheid om een opleiding uit een lijst te kiezen?

Functie en werkervaringen

Het onderdeel *functie en werkervaringen* maakt onderscheid in drie secties: een gewenste functie, werkervaring in het verleden en functies die een gebruiker niet wilt uitvoeren. In de prototyping sessie wordt het verschil uitgelegd tussen de drie secties en een gewenste functie toegevoegd, conform een (fictieve) gewenste functie van de facilitator. Het is belangrijk om op te merken dat er in het invoeren van een functie in de toekomst ook gebruik zal worden gemaakt van een selectielijst.

- Verwacht jij problemen met deze werkwijze voor het invoeren van een gewenste functie of werkervaring?
- Welke functionaliteit mis jij?

Competenties, talen en rijbewijs

De vacature website biedt functionaliteit om competenties, talen en gehaalde rijbewijs certificaten toe te voegen aan het portfolio. De belangrijkste functie in het onderdeel is de categorisering van de competenties. De *competentie groepen* kunnen worden opengeklapt, waarna er de mogelijkheid is om een of meerdere bestaande competenties aan te vinken. Er is daarnaast de mogelijkheid om nieuwe competenties toe te voegen aan een competentie groep.

- Welke competenties zou je zelf opnemen in je CV?
- Wat vind jij van de categorisering van competenties?
- Welke competenties mis jij in de lijst met competenties of welke vind je overbodig?

Curriculum Vitae

Een van de belangrijkste functies van de vacature website is de mogelijkheid om meerdere CV's aan te kunnen maken, passend bij een bepaalde vacature. Het aanmaken van de vacatures geschiedt op basis van de gegevens die zijn ingevuld in het portfolio. In de prototyping sessie wordt het aanmaken van een nieuw, fictief, CV getoond (twee stappen). Er zijn een aantal vragen die naar aanleiding van de demonstratie van de sectie kunnen worden gesteld:

- Wat vind jij van de mogelijkheid om meer dan één CV aan te kunnen maken?
- Vind jij het duidelijk hoe je een CV aan moet maken op de vacature website?

A.3.4 Afronding

Tijdsduur: 5 minuten
Verantwoordelijke: facilitator

In het afronden van de sessie worden de belangrijkste problemen en ideeën voor het prototype die in de sessie aanbod zijn gekomen herhaald. Het gesprek wordt afgesloten met het peilen van de emotionele ervaring van deelnemer(s) met het prototype. De volgende vragen kunnen hierbij worden gebruikt:

- Wat is uw algemene indruk van de website?
- Zou u de website gebruiken voor het plaatsen van uw CV en het zoeken van een vacature?

A.3.5 Dankwoord

Tijdsduur: 5 minuten
Verantwoordelijke: observant

De observant sluit de sessie af, bedankt de deelnemers voor deelname en vraagt de deelnemer(s) om de enquête in te vullen (zie bijlage B).

A.4 Dagindeling

Het prototype experiment wordt uitgevoerd in twee dagen. In tabel A.1 en A.2 wordt een schematisch overzicht weergegeven van de tijdsindeling per dag en de participatie vorm en mate van detail die wordt getoetst.

Tijd	**Persoon**	**M/V**	**Leeftijd**	**Participatie**	**Detail**
11:00	-	M	50	Coöperatief	Hoog
12:30	-	M	27	Coöperatief	Hoog
13:30	-	V	37	Coöperatief	Hoog
15:00	-	M	27	Demonstratie	Hoog
	-	V	43	Demonstratie	Hoog
	-	M	21	Demonstratie	Hoog
	-	M	54	Demonstratie	Hoog
	-	M	38	Demonstratie	Hoog

Tabel A.1: Dagindeling experiment - Donderdag 21 augustus

Tijd	**Persoon**	**M/V**	**Leeftijd**	**Participatie**	**Detail**
10:00	-	M	54	Coöperatief	Laag
11:00	-	M	24	Coöperatief	Laag
12:30	-	V	50	Coöperatief	Laag
13:30	-	V	21	Coöperatief	Laag
15:00	-	M	23	Demonstratie	Hoog
	-	M	21	Demonstratie	Hoog
	-	M	36	Demonstratie	Hoog
	-	M	58	Demonstratie	Hoog
	-	M	28	Demonstratie	Hoog
	-	M	23	Demonstratie	Hoog

Tabel A.2: Dagindeling experiment - Vrijdag 22 augustus

Bijlage B

Enquête experiment

Naam:

Hartelijk dank voor jou deelname aan de evaluatie van de vacature website! Logica heeft een eerste prototype ontwikkeld voor een belangrijke klant, jouw opmerkingen en feedback uit deze sessie zullen worden gebruikt in het verdere ontwerp van dit systeem. Daarnaast wordt deze sessie gebruikt in onderzoek naar de effectiviteit van prototyping. Je bijdrage wordt anoniem verwerkt in de publicatie van het onderzoek.

Het gewenste systeem stelt gebruikers in staat hun CV te maken en te beheren; te zoeken naar vacatures en het CV te publiceren op het internet. Het prototype dat in deze sessie wordt getoond richt zich voornamelijk op het aanmaken en beheren van het CV.

Voor dat we aan de sessie beginnen wil ik je eerst vragen om een aantal vragen over vacature websites te beantwoorden.

1. Wat vond je goed aan de vacature websites die je in het verleden hebt gebruikt? Noem minimaal drie punten.

2. Wat vond je irritant of slecht? Noem minimaal drie punten.

3. Welke functionaliteit moet een vacature website minimaal bevatten? Noem minimaal drie punten.

4. Welke informatie buiten je persoonlijke gegevens, opleidingen en werkervaring wil je in je CV opnemen?

5. Wat was het uiteindelijke resultaat van het gebruik van vacature websites? (omcirkel)
   - Het bedrijf waar ik voor solliciteerde heeft mij uitgenodigd voor een gesprek.
   - Het bedrijf heeft mij niet uitgenodigd.
   - Ik ben benaderd voor vacatures die goed aansloten bij wat ik zocht.
   - Ik ben benaderd voor niet passende vacatures.
   - Overig, namelijk:

6. Hoe vaak wijzig je je CV?

7. Pas je je CV aan, afhankelijk van de vacature? Zo ja, wat wijzig je?

Ik wil jou vragen of je na de prototype evaluatie nog even de tijd wilt nemen om deze enqute in te vullen.

De deelname van mij in het prototyping proces voor de ontwikkeling van een nieuwe vacature website is:

Belangrijk	_	_	_	_	_	_	_	**Onbelangrijk**
Geen belang voor mij	_	_	_	_	_	_	_	**Groot belang voor mij**
Irrelevant	_	_	_	_	_	_	_	**Relevant**
Veel betekenis voor mij	_	_	_	_	_	_	_	**Geen betekenis voor mij**
Onbruikbaar	_	_	_	_	_	_	_	**Bruikbaar**
Waardevol	_	_	_	_	_	_	_	**Waardeloos**
Triviaal	_	_	_	_	_	_	_	**Fundamenteel**
Lonend	_	_	_	_	_	_	_	**Niet lonend**
Maakt mij wel uit	_	_	_	_	_	_	_	**Maakt mij niets uit**
Niet geïnteresseerd	_	_	_	_	_	_	_	**Geïnteresseerd**
Significant	_	_	_	_	_	_	_	**Niet significant**
Vereist	_	_	_	_	_	_	_	**Overbodig**
Saai	_	_	_	_	_	_	_	**Interessant**
Spannend	_	_	_	_	_	_	_	**Niet spannend**
Uitnodigend	_	_	_	_	_	_	_	**Niet uitnodigend**
Gewoon	_	_	_	_	_	_	_	**Fascinerend**
Essentieel	_	_	_	_	_	_	_	**Niet essentieel**
Ongewenst	_	_	_	_	_	_	_	**Wenselijk**
Gewild	_	_	_	_	_	_	_	**Ongewild**
Niet nodig	_	_	_	_	_	_	_	**Nodig**

Wilt u het resultaat van dit onderzoek ontvangen per e-mail?

Bijlage C

Toetsing in de praktijk

Het theoretisch model dat in dit onderzoek is voorgesteld leidt tot inzicht in het effect van verschillende prototyping dimensies op software succes. Literatuurstudie wijst uit dat het onderzoek waarin de theorie wordt bevestigd niet compleet is en vaak ontbreekt aan empirische toetsing.

In dit onderzoek is een poging gedaan om het theoretisch model dat is voorgesteld empirisch te toetsen. In eerste instantie is een zoektocht gestart naar bedrijven en projecten waar prototyping wordt toegepast. De zoektocht heeft uiteindelijk geresulteerd in twee projecten waarin, ondanks de beperkte hoeveelheid data, interessante ontdekkingen zijn gedaan.

In dit hoofdstuk wordt in sectie C.1 ingegaan op de problemen die zijn ervaren in het verzamelen van empirische data voor toetsing van het theoretisch model en worden mogelijke verklaringen gegeven voor het falen van de zoektocht. In sectie C.2 is de analyse van een tweetal prototyping sessies uitgewerkt. De secties C.3 en C.4 beschrijven de conclusies die op basis van de studie kunnen worden gedaan en discussie over validiteit van deze conclusies.

C.1 Problemen met het verzamelen van data

In een zoektocht naar projecten waar prototyping wordt toegepast is contact gezocht met alle 261 werknemers van een divisie binnen het afstudeerbedrijf, een afdeling van 25 personen gespecialiseerd in het ontwerpen van gebruikersinterfaces en een architectuur distributielijst, waarop 614 leden met affiniteit op het gebied van software architectuur en requirements engineering zijn geabonneerd. De zoektocht in het afstudeerbedrijf heeft uiteindelijk geresulteerd in veel geïnteresseerde softwarespecialisten, een vijftal informatieve interviews en twee projecten.

In een poging om het aantal projecten voor empirisch onderzoek te vergroten is de zoektocht uitgebreid naar andere softwarebedrijven in Nederland. In samenwerking met een expert op het gebied van requirements engineering zijn 12 grote softwarebedrijven benaderd en is een requirements engineering conferentie bijgewoond, waarbij contact is gelegd met meerdere experts op dit gebied.

De contacten hebben veel interesse opgeleverd, interviews met 5 experts in één softwarebedrijf, maar ook afwijzingen en uiteindelijk geen projecten.

Het is opvallend dat er, gezien de omvang van de zoektocht, zo weinig projecten zijn gevonden waar prototyping wordt toegepast. Het aantal staat in schril contrast tot onderzoek van Hardgrave (1995); Neill & Laplante (2003) waaruit blijkt dat er in respectievelijk 60 en 71% van de softwareprojecten gebruik wordt gemaakt van een vorm van prototyping. Uit discussies met experts op het gebied van prototyping en reacties in de zoektocht zijn een aantal interessante verklaringen naar voren gekomen voor het slechte resultaat.

Prototyping definitie Het beeld dat bestaat bij het begrip prototyping verschilt sterk per persoon. Prototyping wordt gedefinieerd als een *eerste concept van een softwaresysteem* of *de basis voor software*, maar het begrip wordt vooral geassocieerd met een formeel proces om incrementele software ontwikkeling te faciliteren. In de gehouden interviews is regelmatig gebleken dat mensen verklaarden niet gebruik te maken van prototyping, maar volgens de prototyping definitie die in dit onderzoek wordt gehanteerd wel degelijk werkten volgens een prototyping methode.

In de zoektocht naar prototyping projecten is wellicht ten onrechte de intentie gewekt dat er voor het onderzoek een formeel en volledig prototyping proces gezocht werd. De toepassing van prototyping lijkt in de praktijk voornamelijk informeel te zijn. De verklaring bevestigt de bevindingen van Beynon-Davies et al. (1999). Beynon-Davies et al. stellen dat de toepassing van prototyping verschuift van een initiëel formele softwareontwikkelmethode naar een business-geörienteerde aanpak waarin de term prototyping niet expliciet wordt gebruikt en de focus ligt op het betrekken van gebruikers en snelle software ontwikkeling.

Periode van onderzoek Het beperkt aantal gevonden projecten is voor een groot deel te verklaren door verschil in definitie van het begrip prototyping en het feit dat mensen simpelweg slecht reageren op e-mails via distributielijsten. Er zijn echter opvallend veel reacties ontvangen. In veel gevallen is de ontvangen reactie negatief, met de mededeling dat de fase waarin prototyping wordt toegepast recent is afgesloten.

De zoektocht is gestart in de maand mei. Uit de reacties kan worden afgeleid dat de ontwerpfase van veel softwareprojecten in deze maand is afgerond. Dit zou betekenen dat veel projecten worden gestart in het voorjaar. Uit meerdere interviews is dit bevestigd, een verklaring voor het feit dat veel softwareprojecten in het begin van het jaar starten is het feit dat budgetten bij bedrijven vaak in het voorjaar vrijkomen.

C.2 Case studies

Er zijn uiteindelijk twee projecten gevonden waarin een vorm van prototyping is toegepast en toestemming is gegeven om één of meerdere prototyping sessies bij te wonen. Het geringe aantal projecten maakt het niet mogelijk om de interpretatie van het theoretisch model significant te toetsen en een vergelijking te maken tussen verschillende type prototyping vormen.

Interpretatie van het theoretisch model wijst uit dat het gebruik van een prototype een beperkte invloed heeft op de kwaliteit van software. Een belangrijke verklaring hiervoor is het bestaan van *The Gradient of Resistance*, het bestaand van weerstand tot verandering van het prototype bij de ontwerper. De theorie is door Bowers & Pycock (1994) voorgesteld, maar in de praktijk niet getoetst. Analyse van de prototyping sessies wordt gebruikt om vast te stellen hoe sterk de invloed is van de ontwerper van een prototype op kennisoverdracht in een dergelijke sessie.

Gebaseerd op de interpretatie van het theoretisch model is de volgende hypothese opgesteld, welke in de praktijk wordt getoetst:

> *Hypothese 1:* Het gebruik van een prototype roept weerstand op tot verandering van het prototype bij de ontwerper en beperkt kennisoverdracht in een prototyping sessie.

C.2.1 Overheid

Een niet nader te noemen overheidsinstantie heeft een prototype ontwikkeld voor een webapplicatie waar burgers in staat worden gesteld om informatie op te vragen en wijzigingen door te geven in hun persoonlijke situatie. Het prototype is in een vergevorderd stadium en is recent getest met een representatieve groep gebruikers. Op basis van de gebruikerstest is het prototype aangepast. In de bijgewoonde sessie zijn de aanpassingen ter goedkeuring gepresenteerd aan belanghebbenden.

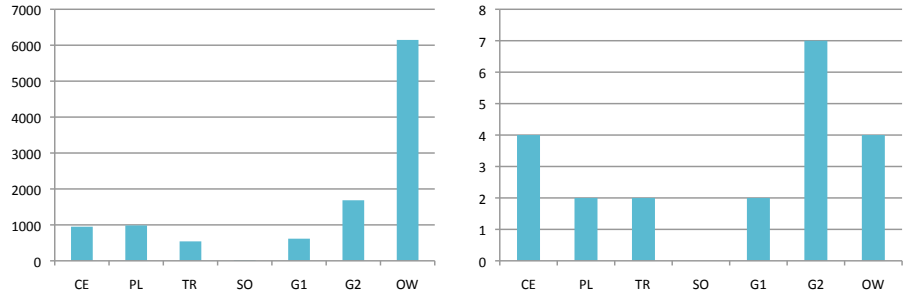
De ontwikkeling van het prototype is uitbesteed aan een ontwerper. De ontwerper is verantwoordelijk voor het correct vertalen van eisen en wensen voor de nieuwe webapplicatie naar een prototype. Het prototype dat wordt ontwikkeld heeft een hoge mate van detail en zal uiteindelijk de volledige functionaliteit tonen. In de prototyping sessie die is geanalyseerd wordt het prototype door de ontwerper gedemonstreerd aan de belanghebbenden van de webapplicatie.

In totaal hebben 6 belanghebbenden en 1 ontwerper (vanaf nu persoon OW) aan de prototyping sessie deelgenomen. De belanghebbenden zijn de concept eigenaar (persoon CE), deelproject leider (persoon PL), een medewerker tekst redactie (persoon TR), software ontwikkelaar (persoon SO) en twee medewerkers (persoon G1 en G2) van de afdeling gebruikersvriendelijkheid. Persoon G1 en G2 verdedigen in de sessie het belang van de gebruiker.

Het toetsen van de hypothese

De overdracht van kennis tussen deelnemers in een prototyping sessie en de ontwerper is niet eenvoudig te meten. Een veelgebruikt meetinstrument voor de kwaliteit van kennisoverdracht in de ontwikkeling van software is het aantal ideeën dat wordt voorgesteld door deelnemers in een sessie (Engelbrektsson, 2002).

Voor de toetsing van de hypothese is het van belang om niet alleen de overdracht, acquisitie, van kennis te meten, maar vooral wat er met de overgedragen kennis wordt gedaan, geëxploiteerd. *The Gradient of Resistance* is met name



(a) Aantal woorden per persoon.

(b) Aantal ideeën per persoon.

Figuur C.1: De inbreng per deelnemer in de prototyping sessie van de overheid.

van invloed op de exploitatie van kennis in een prototyping sessie. Clement & den Besselaar (1993) meten de exploitatie van kennis door te bepalen of een sessieleider een idee accepteert, er kritiek op geeft of het idee negeert.

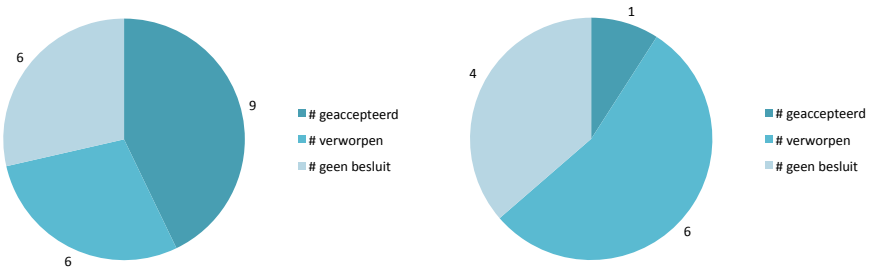
De prototyping sessie is opgenomen met een *voice recorder*. De totale opname van 1 uur en 19 minuten is volledig uitgewerkt in een transcript. Het transcript is gebruikt om iedere opmerking van een deelnemer in de sessie te classificeren als idee, of in het geval van de ontwerper van het prototype als acceptatie van het idee, kritiek op het idee of het uitblijven van een reactie. De kans is groot dat de classificatie in een dergelijke meting subjectief wordt uitgevoerd, om de subjectieve invloed te beperken is de classificatie geformaliseerd. Een opmerking is als idee gekenmerkt als de opmerking resulteert in een wijziging van of aanvulling op het prototype, acceptatie van een idee vindt plaats wanneer de ontwerper positief reageert op een idee en het noteert in een lijst van aanpassingen. Er is sprake van kritiek en het verwerpen van een idee als de ontwerper negatief reageert op een idee en het niet opschrijft.

Resultaten

Er zijn totaal 10939 woorden uitgewisseld in de prototyping sessie, waarvan 56% is uitgesproken door de ontwerper van het prototype. In figuur C.1a wordt het aantal woorden per persoon weergegeven.

In de sessie zijn totaal 8 wijzigingen in het prototype aan bod gekomen, waarvan 3 wijzigingen in een vorige sessie zijn afgesproken. De wijzigingen zijn door de ontwerper in de sessie een voor een voorgesteld. 5 van de wijzigingen zijn door de belanghebbenden geaccepteerd, 3 wijzigingen zijn verworpen of leiden niet tot een besluit. Op de helft van de wijzigingen is kritiek geuit door de belanghebbenden, dit heeft geresulteerd tot het verwerpen van 2 wijzigingen, de andere 2 wijzigingen zijn echter alsnog geaccepteerd of staan nog open.

In aanvulling op de wijzigingen hebben de deelnemers van de sessie totaal 21 ideeën voorgesteld voor nieuwe functionaliteit of wijzigingen van het bestaande prototype. In figuur C.1b wordt het aantal ideeën per persoon weergegeven. 9 van de ideeën worden uiteindelijk geaccepteerd, over de resterende 12 van de



(a) Acceptatie van ideeën. (b) Kritiek op ideeën door de software specialist.

Figuur C.2: Acceptatie en weerstand bij ideeën.

ideeën wordt geen besluit genomen of verworpen (zie figuur C.2a). De ontwerper van het prototype heeft op 11 van de 21 ideeën kritiek. Van de ideeën waar de ontwerper kritiek op heeft wordt er 1 geaccepteerd, 6 ideeën worden expliciet verworpen door de ontwerper en 4 ideeën worden genegeerd (zie figuur C.2b).

Interpretatie

Het is opvallend dat de acquisitie van kennis voor een groot deel, 56% van de uitgesproken woorden, toekomt aan de ontwerper van het prototype. Het feit dat de ontwerper veel aan het woord is is te verklaren door het type prototyping sessie, een prototype demonstratie. In de demonstratie van een prototype leidt de ontwerper en legt hij de functionaliteit van een prototype uit. De ontwerper vraagt echter ook naar feedback van deelnemers in de betreffende sessie. Het grote verschil in het aantal gesproken woorden in de prototyping sessie tussen de ontwerper en deelnemers wordt naast het type sessie wellicht beïnvloed door het feit dat de ontwerper geneigd is het prototype te verdedigen.

G2: Eigenlijk moet dat dan al veel eerder op de website staan. Ik weet eigenlijk niet of dat er in staat hoor, dat verschil tussen website en portal. Maar dat is..
OW: Ik weet niet zo goed hoe ik dat moet begrijpen, eerder in de site. Ik vind dat snel toegankelijk, want vanaf ieder pagina heb je dit menu.
G2: Ja, maar het is wel een essentieel verschil: de website en de portal. Voor ons is dat allemaal duidelijk dat de website informatie is en de portal ga je iets doen, dus persoonlijke situatie. Maar het wordt nergens echt uitgelegd volgens mij, of ik heb het mis.
OW: Nou, op alle pagina's hebben we dit..
G2: Ja, daar hebben we dit staan. Maar dat wil nog niet zeggen dat mensen dan begrijpen: o, dit is informatie en dat is eh.. Waar ik iets kan doen.

Tabel C.1: Prototyping sessie Overheid: Fragment #1

G2: Ja ik zou zeggen zet hem gewoon rechtsonder aan, onder het veld.

OW: Hier wordt er gekozen voor de Windows stijl een beetje he.

G2: Ja. Ja. Een beetje ja.

OW: Wat vind jij ervan, CE?

CE: Ik zou het even zo laten.

OW: Dat antwoord zocht ik ook ja.

G2: Daarom vroeg je het.

OW: Dit zijn van die dingen waarvan ik dacht, dat heeft niet zoveel zin om ..

G2: Ja okee, maar dan kunnen we nergens over discussiëren.

Tabel C.2: Prototyping sessie Overheid: Fragment #2

De resultaten laten zien dat de ontwerper op ruim de helft van de ideeën van deelnemers kritiek heeft. In veel gevallen is de kritiek gebaseerd op het feit dat de ontwerper het prototype verdedigt. Uit een voorbeeldfragment van de sessie in tabel C.1 komt naar voren dat de ontwerper zijn eigen standpunt ten opzichte van een idee voor een aanpassing van de webapplicatie verdedigt. Het betreffende idee wordt uiteindelijk niet geaccepteerd. Het resultaat komt overeen met *The Gradient of Resistance*: deelnemers aan een prototyping sessie hebben ideeën over aanpassingen van het prototype, maar worden geremd door de ontwerper die van mening is dat het prototype correct is.

De invloed van de weerstand tot verandering van het prototype is van grote invloed op de exploitatie van kennis. In een fragment uit de prototyping sessie (zie tabel C.2) is te lezen dat de ontwerper op een zeker moment de exploitatie van kennis tegenhoudt. Analyse van de prototyping sessie wijst uit dat er van de ideeën waar de ontwerper kritiek op heeft, er slechts één wordt geaccepteerd.

Het is natuurlijk goed mogelijk dat de kritiek op ideeën volledig terecht is, maar het tegenhouden van kennisoverdracht kan tot gevolg hebben dat de het effect van participatie van belanghebbenden minder invloed heeft op software kwaliteit (He, 2004). De weerstand kan er daarnaast voor zorgen dat het de deelnemers aan een prototyping sessie weerhoudt om opmerkingen over het prototype te geven. In de betreffende sessie kan de weerstand tot exploitatie van kennis verklaren waarom de inbreng van sommige deelnemers in de sessie zo laag is (zie figuur C.1).

Conclusie

Analyse van een prototyping demonstratiesessie voor een webapplicatie van de overheid geeft inzicht in het bestaan van *The Gradient of Resistance* en de invloed van de weerstand op kennisoverdracht. In de sessie is, in lijn met Bowers & Pycock (1994), zichtbaar sprake van weerstand tot verandering van het prototype bij de ontwerper ervan.

De weerstand tot verandering leidt ertoe dat de ontwerper kritiek heeft op

veel ideeën van deelnemers aan de sessie, waardoor de ideeën niet worden geaccepteerd of genegeerd. Het resultaat is in overeenstemming met onderzoek van Clement & den Besselaar (1993), waarin de auteur op basis van studie van 16 softwareprojecten tot de conclusie komt dat softwarespecialisten vaak geneigd zijn om commentaar en ideeën over functionaliteit van een softwaresysteem door belanghebbenden tegen te spreken of te negeren.

Validatie

De conclusies die op basis van deze ene sessie kunnen worden gedaan zijn zwak. Er kan op basis van de resultaten worden vastgesteld dat de ontwerper kritiek heeft op ideeën en dat het gevolg daarvan is dat aanpassingen naar ideeën van belanghebbenden niet altijd worden geaccepteerd. Er kan echter niet worden vastgesteld in hoeverre de kritiek terecht is en in hoeverre het negeren van ideeën gevolgen heeft voor het succes van de software uiteindelijk.

In een poging om de resultaten te valideren zijn de deelnemers benaderd en gevraagd om aan te geven wat de waarde is van de ideeën die in de prototyping sessie zijn voorgesteld. Op basis van de resultaten kan worden vastgesteld hoe belangrijk de ideeën zijn die zijn verworpen. Wegens gebrek aan tijd en vakantieperiodes is de validatie echter niet uitgevoerd.

C.2.2 Vacature website

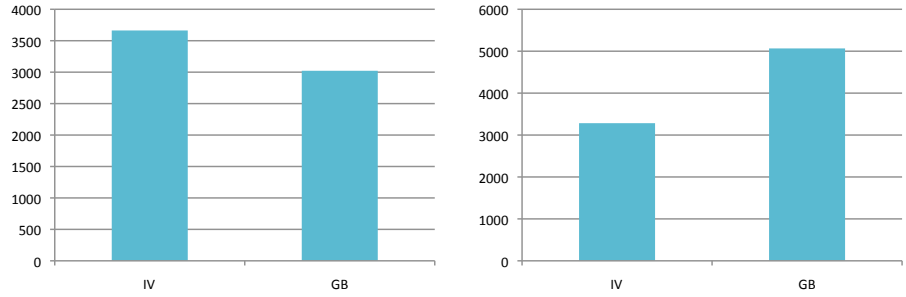
Een afdeling, gespecialiseerd in het ontwerpen van gebruikersinterfaces, binnen het bedrijf Logica heeft een prototype ontwikkeld voor een vacature website. De webapplicatie maakt het mogelijk voor werkzoekenden om een Curriculum Vitae (CV) op Internet te plaatsen en op basis van het CV te zoeken naar geschikte vacatures.

Het prototype wordt gebruikt om te testen of de functionaliteit van de website volledig is en om problemen met de gebruikersvriendelijkheid van de website vroegtijdig vast te stellen. De mate van detail van het prototype is hoog, gelijkwaardig aan de toekomstige website, en de hoeveelheid functionaliteit is beperkt tot opdrachten die in de prototype testen moeten worden uitgevoerd. Het prototype is in individuele sessies met een facilitator (persoon IV), tevens betrokken bij de ontwikkeling van het prototype, getest in een periode van drie dagen met 12 representatieve gebruikers (persoon GB).

Er zijn voor dit onderzoek twee sessies met verschillende gebruikers bijgewoond en geanalyseerd. De sessies zijn net als de case studie bij de overheid opgenomen en uitgewerkt in een transcript. De duur van de twee sessies is in beide gevallen 50 minuten.

Het toetsen van de hypothese

Er is één verschil in de toetsing van de hypothese ten opzichte van de case studie van de overheid. Een belangrijk doel van een prototype test is om te leren begrijpen wat de belangrijkste problemen zijn die gebruikers in de uitvoering van werkzaamheden met een prototype ervaren. De kennis die wordt overgedragen



(a) Prototype test #1.

(b) Prototype test #2.

Figuur C.3: Aantal woorden per persoon, per sessie.

in een prototype test bestaat dan ook voornamelijk uit problemen die worden aangegeven, het is niet het doel van het type sessie om deelnemers te stimuleren om ideeën voor aanpassingen in het prototype voor te stellen (Nielsen, 1993).

De kwaliteit van kennisoverdracht wordt in de case studie gemeten door het aantal genoemde problemen in het prototype en het aantal voorstellen voor aanpassingen van het prototype vast te stellen. Het verschil tussen beide type opmerkingen is het feit dat met een idee een alternatieve oplossing wordt voorgesteld, een probleem vormt slechts een eventuele aanleiding tot een oplossing.

Resultaten

In de prototyping testsessies zijn respectievelijk 6685 en 8349 woorden uitgesproken, 55 en 39% is uitgesproken door de facilitator. In de grafieken in figuur C.3 staan het aantal woorden per deelnemer, per sessie.

In de eerste prototyping sessie heeft de gebruik 7 problemen in het prototype aangegeven en 4 ideeën ter verbetering van het prototype (zie tabel C.3). De facilitator van de prototype test verdedigt bij 3 van de 4 ideeën het prototype ontwerp. De tweede prototyping sessie levert 8 problemen in het prototype op en 8 ideeën voor het aanpassen van het prototype. In 5 van de 8 ideeën verdedigt de facilitator het prototype.

Sessie	**# problemen**	**# ideeën**	**# verdediging**
Prototype test #1	7	4	3
Prototype test #2	8	8	5

Tabel C.3: Het aantal aangegeven problemen en ideeën ter verbetering van het prototype en het aantal keer dat de ontwerper hiertegen in verdediging gaat.

Interpretatie

Een prototype test is er in eerste instantie op gericht om problemen met gebruikersvriendelijkheid in een prototype vast te stellen (Nielsen, 1993). De twee sessies die zijn geanalyseerd laten zien dat het testen van een prototype ook daadwerkelijk voornamelijk problemen, respectievelijk 7 van de 11 en 8 van de 16 opmerkingen, van een prototype in kaart brengt. In aanvulling op de problemen blijken er in een prototype test toch opvallend veel ideeën voor aanpassingen in het prototype te worden genoemd, in de tweede sessie is zelfs de helft van de opmerkingen een voorstel voor verbetering.

Het feit dat er in een prototype test ideeën ter verbetering van een prototype worden voorgesteld is niet verwonderlijk. Het is natuurlijk dat mensen bij het constateren van een probleem vanzelfsprekend nadenken over een mogelijke oplossing. De reactie van de ontwerper op de voorgestelde ideeën is daarentegen wel opvallend.

De resultaten laten zien dat de facilitator, betrokken bij de ontwikkeling van het prototype, in beide sessies probeert ideeën van gebruikers tegen te spreken. In de sessies verdedigt de facilitator het prototype bij respectievelijk 3 van de 4 en 5 van de 8 ideeën. In een voorbeeldfragment (zie tabel C.4) is te zien dat een gebruiker kritiek heeft op een functie in het prototype. De facilitator verdedigt het bestaan van de functie en, hoewel er duidelijk wordt gevraagd naar de mening van de gebruiker, geeft de facilitator het idee dat ondanks de kritiek van de gebruiker, kennis acquisitie, er geen veranderingen in het prototype zullen worden gedaan op basis van de kritiek, exploitatie.

Het doel van een prototype test is om zoveel mogelijk feedback van een gebruiker te krijgen over een prototype (Nielsen, 1993; Snyder, 2003). Het is van belang om een gebruiker te stimuleren om feedback te geven door alle kritiek te aanvaarden en pas in een later stadium te interpreteren. Het verdedigen van een prototype kan door *The Gradient of Resistance* een gebruiker weerhouden feedback te geven.

Conclusie

Het bestaan van *The Gradient of Resistance* wordt ook in een tweetal prototype testsessies voor een vacature website zichtbaar. In een prototype test is het belangrijk dat er zoveel mogelijk kennis van deelnemers wordt verkregen ten aanzien van het prototype, de rol van de *interviewer* is alleen faciliterend. Analyse van de genoemde sessies wijst uit dat er zelfs in een prototyping methode met de betreffende vorm van participatie weerstand tot verandering van het prototype bij de ontwerper optreedt.

De invloed van *The Gradient of Resistance* op kennis exploitatie is een prototype test minder groot op de kwaliteit van software. Een prototype test wordt vaak bijgewoond door één of meerdere observatoren of opgenomen voor latere observatie. Als de kennisoverdracht van een gebruiker in een sessie door de facilitator niet wordt geëxploiteerd, dan kan de observator de kennis wel degelijk opvangen en exploiteren.

GB: Ja dat snap ik maar waarom dat überhaupt mogelijk is? Want als ik iemand zoek als werkgever, als ik een sollicitatie binnen krijg, waar iemand niet, waarin iemand niet volledig vertrouwen in mij heeft of in mijn firma en dus zijn persoonlijke gegevens niet aan mij laat zien dan zou ik zeggen: Nou wegwezen. Ja krijg die baan niet.

IV: Ja. Nee.

IV: Ja maar als werkgever of als werkzoekende zou u deze functie ook niet gebruiken als ik u zo hoor?

GB: Nee. Ik zou dan mijn..

IV: Gewoon de volledige informatie gewoon dus met volle vertrouwen aan de werkgever uh..

GB: Ja. De werkgever uh..als hij niet weet waar ik woon en hij moet me dan weet ik het reisgeld betalen omdat ik 150 kilometer daar vandaag woon

IV: Ja.

GB: dan zou ik zeggen, die heb ik niet nodig. Dus ik snap niet dat dat uhh uberhaupt mogelijk is.

IV: Ja ik begrijp u, volkomen. Alleen ja, we hebben die mogelijkheid er, omdat sommige mensen er voor kiezen om hun prive gegevens niet direct op hun CV te tonen. Dus uhh.. Dus de mogelijkheid is er.

GB: Ja. Ja.

IV: Ja en het gaat er eigenlijk niet om of u er gebruikt van maakt of niet..

GB: ja jajajaja

Tabel C.4: Prototyping sessie Vacature website: Fragment #1

C.3 Conclusie

Een poging om de interpretatie van het theoretisch model in de praktijk te toetsen is niet geslaagd. Een zoektocht naar projecten waar een vorm van prototyping wordt toegepast heeft twee projecten opgeleverd, waarvan respectievelijk 1 en 2 prototyping sessies zijn bijgewoond. Een verklaring voor het lage aantal gevonden projecten is het feit dat er een groot verschil blijkt te zijn in de definitie van prototyping die wordt gehanteerd en de methode vaak informeel wordt toegepast zonder dat men zich ervan bewust is.

In de prototyping sessies die zijn geanalyseerd wordt bevestigd wat als hypothese op basis van het theoretisch model wordt gesteld. Het gebruik van een prototype roept weerstand op tot verandering van het prototype bij de ontwerper en beperkt kennisoverdracht in een prototyping sessie. *The Gradient of Resistance* stelt dat een gebruiker ideeën heeft over aanpassingen en toevoegingen aan een bestaand prototype, maar wordt geremd door de ontwerper van het prototype (Bowers & Pycock, 1994). In de analyse van zowel een prototype demonstratie als prototype test wordt het bestaan van *The Gradient of Resistance* waargenomen.

De invloed van de weerstand tot verandering op kennisoverdracht is tweeledig. Ten eerste beperkt de weerstand de exploitatie van kennis. In de sessies worden ideeën van deelnemers aan de sessie door de facilitator tegengesproken

en genegeerd. Het resultaat is in overeenstemming met onderzoek van Clement & den Besselaar (1993). Ten tweede kan *The Gradient of Resistance* een negatieve invloed hebben op de acquisitie van kennis. Een deelnemer kan zich door weerstand van de facilitator in een sessie geremd voelen om opmerkingen te plaatsen over het prototype.

C.4 Validiteit

In de prototyping sessies die zijn geanalyseerd wordt het bestaan van *The Gradient of Resistance* en de invloed op kennisoverdracht waargenomen. De conclusies die op basis van de analyse van de 3 sessies kunnen worden gemaakt zijn echter zwak. Het aantal bijgewoonde sessies is te laag om de resultaten te kunnen generaliseren.

Het optreden van een facilitator is ten eerste van grote invloed op het proces en de kwaliteit van het resultaat uit een prototyping sessie (Miranda & Bostrom, 1999). De complexiteit van een prototype dat wordt gedemonstreerd of getest modereert ten tweede de invloed van gebruikersparticipatie op software kwaliteit (McKeen & Guimaraes, 1997). In de derde plaats is de kwaliteit van samenwerking in een groep van invloed op de kennisoverdracht die plaatsvindt in een prototyping sessie (He, 2004).

De weerstand tot verandering en de invloed van de weerstand op kennisoverdracht wordt in de analyse van de prototyping sessies gemeten door de reactie van de facilitator op ideeën van deelnemers te analyseren. Hoewel het aantal ideeën veel wordt gebruikt om de kwaliteit van kennisoverdracht vast te kunnen stellen (Engelbrektsson, 2002), is het een subjectief meetinstrument. Kennisoverdracht is niet alleen succesvol als er veel kennis wordt overgedragen, maar met name als er relevante kennis wordt overgedragen. In de analyse van de prototyping sessies is voorgesteld om deze methode als validatie toe te passen, maar dit is wegens het uitblijven van reacties uiteindelijk niet gelukt.

Bibliografie

Beyer, H. & Holtzblatt, K. (1998). *Contextual Design: Defining Customer-centered Systems*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Beynon-Davies, P., Tudhope, D., & Mackay, H. (1999). Information systems prototyping in practice. *Journal of Information Technology*, *14*(1), 107–120.

Bowers, J. & Pycock, J. (1994). Talking through design: Requirements and resistance in cooperative prototyping. In *CHI '94: Proceedings of the SIGCHI conference on Human factors in computing systems*, (pp. 299–305)., New York, NY, USA. ACM.

Clement, A. & den Besselaar, P. V. (1993). A retrospective look at pd projects. *Communications of the ACM*, *36*(6), 29–37.

Engelbrektsson, P. (2002). Effects of product experience and product representations in focus group interviews. *Journal of Engineering Design*, *13*(3), 215–221.

Grønbæk, K. (1991). *Prototyping and Active User Involvement in System Development: Towards a Cooperative Prototyping Approach*. PhD thesis, Aarhus University.

Hardgrave, B. (1995). When to prototype: Decision variables used in industry. *Information and Software Technology*, *37*(2), 113–118.

He, J. (2004). Knowledge impacts of user participation: a cognitive perspective. *Proceedings of the 2004 SIGMIS conference on Computer personnel research: Careers, culture, and ethics in a networked environment*, 1–7.

McKeen, J. D. & Guimaraes, T. (1997). Succesful strategies for user participation in systems development. *Journal of Management Information Systems*, *14*(2), 133–150.

Miranda, S. & Bostrom, R. (1999). Meeting Facilitation: Process versus Content Interventions. *Journal of Management Information Systems*, *15*(4), 89–114.

Neill, C. J. & Laplante, P. A. (2003). Requirements engineering: The state of the practice. *IEEE Softw.*, *20*(6), 40–45.

Nielsen, J. (1993). *Usability Engineering*. Morgan Kaufmann.

Snyder, C. (2003). *Paper Prototyping: The Fast and Easy Way to Design and Refine User Interfaces*. Morgan Kaufmann.

Trigg, R. H., Bødker, S., & Grønbæk, K. (1991). Open-ended interaction in cooperative prototyping: a video-based analysis. *Scandinavian Journal of Information Systems*, *3*, 63–86.