

# Design Reasoning Improves Software Design Quality

Antony Tang<sup>1</sup>, Minh H. Tran<sup>1</sup>, Jun Han<sup>1</sup>, and Hans van Vliet<sup>2</sup>

<sup>1</sup> Swinburne University of Technology, Melbourne, Australia  
{atang, mtran, jhan}@ict.swin.edu.au

<sup>2</sup> VU University, Amsterdam, The Netherlands  
hans@cs.vu.nl

**Abstract.** Making justifiable decisions is a critical aspect of software architecture design. However, there has been limited empirical research on the effects of design reasoning on the quality of software design. The goal of this work is to investigate if there is any quality improvement to software design when design reasoning is applied. We conducted an empirical study involving twenty designers, the designers were asked to design a user interface and their designs were scored and compared. The results showed that the test group that was equipped with design reasoning produced a higher quality design than the control group, especially for inexperienced designers.

**Keywords:** Design Reasoning, Software Architecture Design, Usability.

## 1 Introduction

Software designers tend to base their judgments on prior beliefs and intuition rather than a logical reasoning process. This tendency is common to human thinking and has influenced the performance of many people's reasoning and decision making [1]. In this paper, we demonstrate that software design is subject to the same cognitive bias, and therefore can affect the quality of its results. Through an empirical study, we show that the design quality can be improved with a simple design reasoning approach.

Recent research, especially in the area of software architecture, has shown that design reasoning and design rationale are important [2]. Design reasoning supports designers in making justifiable decisions by explicitly modeling design rationale as a first-class entity [3]. Functional and quality requirements are considered in such methods, and decision making techniques such as trade-off analysis are used to select a solution that best suits the design criteria. It is argued that design rationale should be captured explicitly, together with the decisions taken and the resulting design. They constitute the software architectural knowledge [4].

This research explores how design reasoning affects the quality of design outcome. In exploring this issue, we have carried out an empirical study with the usability quality attribute. Usability is an important quality attribute in software architecture [5, 6]. It is concerned with whether users are able to use a system to perform their tasks effectively, efficiently and with satisfied experience.

We hypothesize that by applying a design reasoning process, designers would come up with a more usable UI. The study was conducted with two groups of designers who

have similar design experience. The control group carried out the design as they usually do, whilst the test group carried out the tasks using a design reasoning approach. The results have demonstrated that the use of a reasoning approach has, on average, improved design quality. Especially for relatively inexperienced designers, the improvement is noteworthy. For them, design reasoning provides a framework for deliberation and supports building up and maintaining a mental image of the ongoing design. Experienced designers have less need for such assistance, they simply “know” how to proceed [7].

The remaining of the paper is organized as follows. Section 2 discusses the concepts of design reasoning and the related work in usability design reasoning. Section 3 presents the empirical study and the findings. Section 4 discusses the lessons that we have learned from applying design reasoning to software design. We conclude the paper in section 5.

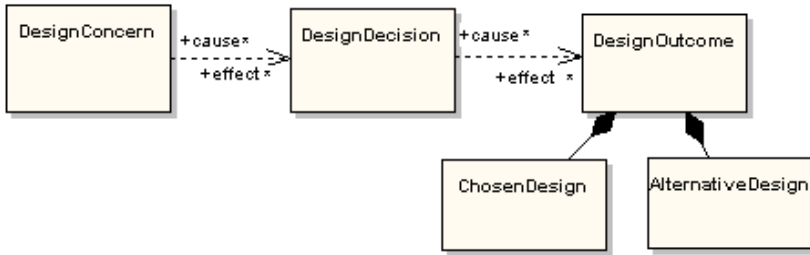
## 2 Related Work

### 2.1 Design Reasoning

Researchers in psychology have proposed that there are two distinct cognitive systems underlying reasoning. System 1 (heuristic system) comprises a set of autonomous subsystems that include both innate input modules and domain-specific knowledge acquired by a domain-general learning mechanism. System 2 (analytic system) allows reasoning according to logical standards [1]. Together they form the dual process theory that explains people’s “rational thinking failure” when people rely heavily on prior beliefs and intuition rather than a logical reasoning process [8, 9]. These findings are also confirmed in a more recent study on decision making in software design where designers make use of rational and naturalistic decision making tactics [10].

In software development, design reasoning is an important process that designers use in developing a solution. Designers in the software industry often rely on intuition and experience to make design decisions. The drawback of such an approach is that the quality of decisions would heavily depend on the experience and expertise of the individuals. Since design rationale plays an important role in making design decisions [2, 11], understanding what constitutes design rationale is important to producing a good design. Rittel and Webber [12] view design as a process of negotiation and deliberation. They suggest that design is a “wicked problem” in which there is no well-defined set of potential solutions, so it is important that a designer learns how to handle and weigh alternatives.

There are different approaches to design reasoning. One approach is by way of argumentation. The basic argumentation-based representation is to use nodes and links to represent knowledge and relationships. Examples of this approach are QOC [13], DRL [14] and gIBIS [15]. A second approach is by way of using rationale template to capture design reasoning. This approach incorporates standard templates into the design process to facilitate design rationale capture. This approach is oriented towards the practical implementation of design rationale in industry. Examples are Architecture Decision Description Template [16] and Views and Beyond [17]. A third approach is a hybrid of the first two approaches. Examples of this approach are AREL [18] and Archium [4].



**Fig. 1.** AREL – A Design Reasoning Model

Whilst there are different approaches to design reasoning, the concepts presented in AREL are common to current research thinking [4, 19]. In the rest of this section, we introduce the AREL model, which concepts we used in the empirical study. In this model, there are three key elements to be considered: design concerns, design decisions and design outcomes (Fig. 1).

Design concerns are concerns that motivate the creation of a design solution. Design concerns are the causes for design decisions to be made. A design concern may be a system requirement, a business goal, a quality attribute such as usability, a circumstance that influences a design, or an existing design component that exerts some design constraints.

A design decision is made by a designer when assessing why a particular design is created or chosen. Capturing this knowledge is important because it justifies the design and explains the reasons to those who do not have intimate knowledge of the design - users, testers and maintainers. The key information contained in the design decisions are the design issues, design assumptions, design constraints, and design rationale for the selection or rejection of a design option. The links (Fig. 1) between design concerns and design decisions indicate what design concerns are considered in a decision. The results of the design decisions, i.e. design outcomes, are linked to the design decisions because the outcomes are the results of a decision.

Design outcomes are the results of a design decision: chosen design is the design that has been selected, and it contains the design elements such as components, classes and database schemas that would be implemented; alternative design is the design options that have been considered but rejected. Capturing all the design options, including the rejected alternative design, are important for three reasons: (a) a comprehensive consideration of all available options shows that the designer has not omitted any viable design option; (b) documenting design options can support design backtracking if an initial design solution is unviable when more details are considered; (c) identifying different design options allow design trade-off analysis to be performed to justify the selection of the most appropriate design option.

## 2.2 User Interface Usability

The quality of UI designs has been considered as an important aspect of software architecture. Research has shown that in UI design, designers are required to make decisions on selecting design options in one way or another [20-24]. However, existing UI design lifecycle models provide very limited guidelines that help

designers reason about design options and make justifiable design decisions. UI design reasoning is something that has always been assumed and intuitive.

Recent studies have pointed out that the outcome of UI design includes both the resulting interface itself and a rationale for why the interface is chosen. Howard's exploratory study [22] identifies key elements, including environment, focus and agents, which are often taken into account when designers make tradeoff decisions. Howard's study also models a behavioral process of which designers make an argument in choosing between two key elements. MacLean et al. [24] examine a representation of design rationale. They have developed a semi-formal notation that allows designers to represent explicitly design options and reasons for choosing from amongst them. However, to the best of our knowledge, there is no empirical evidence concerning how design reasoning influences UI design quality.

### 3 An Empirical Study

The objective of this study was to explore the effects of design reasoning on the quality of design. To do so, we asked the participants to design a user interface for a commercial system. The case was carefully chosen and simplified to make sure that it makes sense to the participants and it was not so simple that it could be designed without careful considerations.

We have been working with an automotive company to develop a Web-based system to monitor test vehicles of a fleet. The key function of the system is to allow car engineers to monitor electronic signals collected from the electronic control units (ECUs) in a vehicle. Through a UI, an engineer is able to prepare a monitoring schedule which specifies the information to be monitored. Each monitoring schedule must contain a minimum of 1 request and a maximum of 99 requests (i.e. requirement  $R_1$ ). Engineers would search and select the electronic signals from a search list of 1700 signals when specifying requests (i.e. requirement  $R_2$ ). A request specifies what electronic signals need to be monitored and monitoring start and stop conditions (i.e., requirement  $R_3$ ). Other requirements such as the insertion and deletion of monitoring conditions and signals were given to the participants but we do not describe them in detail here.

#### 3.1 Participants

The study involves twenty participants. We used a convenient sampling method to invite practitioners in the software industry and academia to participate in the study. The participants were allocated randomly to two groups: *test group* and *control group*. The average design experience of the test group and the control group are 8.95 years and 8.40 years, respectively.

#### 3.2 Experiment Procedure

Both the control group and the test group were asked to design a UI for the monitoring system. The groups were given the following: (a) a set of requirements for the design; (b) usability requirements; and (c) UI controls that can be used in the design. We carried out the experiment with each participant individually.

The control group carried the design as they usually do. The test group was briefed about the design reasoning process and they were asked to apply the principles of

design reasoning. In the briefings, we described the design reasoning principles of AREL without referring to its formal model. During the experiment when the participants reach a design decision point, they need to explain their design options and issues, and justify why they chose a particular design option over other alternatives. As the participants justified their design decisions, the interviewers did not give any hints on how to design nor engaged in discussing the quality of the participants' design. However, the interviewers would ask the following questions to ensure that reasoning was applied: "What are the issues in the decision?" and "What are the options to deal with the issues?"

Participants in both the test and control group were asked to use a think-aloud protocol [25] to describe their design strategies. At the end of the design session, the participants were interviewed for their comments. In the interview, we used the Retrospective Think Aloud (RTA) technique [26] to gather participants' comments on their own designs after they had completed the tasks. We timed each design session, starting from when the participants commenced the design process until they completed the design, excluding the briefing and the interview.

Both quantitative and qualitative data were collected in the study. The quantitative data included details of the participants' experience, duration taken to complete their tasks, the levels of their satisfaction and confidence in their own designs and the quality scorings of their designs. The qualitative data was collected from the participants' think-aloud process, our assessment of the participants' design, our observation of the participants' design process, and the participants' comments.

### 3.3 Findings

We analyzed the test results from three perspectives: the quality of the design outcomes, the design process and the participants' feedbacks.

#### 3.3.1 Design Outcomes

With each UI design, we assess the quality of the design based on three (out of ten) UI design heuristics proposed by Nielsen [27]: (a) consistency; (b) flexibility; (c) accessibility. For instance, we assess design consistency by inspecting if participants used UI controls (e.g. scroll-bars, buttons) consistently across the design. For the purpose of this study, we have selected only three most relevant usability heuristics.

For each participant's design, we rated the usability based on the selected heuristics. For each heuristic, we used a 5-point Likert scale ranging from 0 to 4, with 4 being the best design. Thus, the top score of a design is 12 and the worst score is 0. The rating process was done by comparing how well each design conforms to the heuristics. When scoring the designs, we scored and compared the designs irrespective of which group they come from to ensure that the scorings were consistent and unbiased.

We hypothesize that the test group who equipped with design reasoning would produce better quality design than the control group. If  $m1$  is the average score of the test group and  $m2$  is average score of the control group, the hypotheses are:

$$H_0: m1 = m2$$

$$H_1: m1 > m2$$

**Table 1.** Test and Control Group Design Quality Scores

	n	Mean Score	Std. dev.	Wilcoxon Test
Test Group	10	9.10	1.52	$p = 0.02$
Control Group	10	7.10	1.91	

The null hypothesis  $H_0$  states that the quality of the UI design created by the control group is the same as the test group. The average scores of the control group is 7.10, and that of the test group is 9.10. The one-tailed Wilcoxon<sup>1</sup> [28] test shows that there is a significant difference in design quality between the test and control groups with  $p < 0.025^2$  (see Table 1). Since the one-tailed Wilcoxon test is significant, we reject  $H_0$  and accept the alternative hypothesis  $H_1$ . The conclusion is that the application of a design reasoning process has improved the quality of a UI design.

In order to understand what quality aspects of the design are different between the two groups, we analyze each participant's design. There are two key design issues involved and both of them are concerned with usability:

- ( $R_1$ ) catering for between 1 and 99 monitoring requests; and
- ( $R_2$ ) determining how to search for 1700 vehicle signal(s) and selecting them for monitoring and monitor triggering.

**Test Group.** All participants in the test group used the rationale-based approach to design for the given requirements. In designing for requirement  $R_1$ , the participants selected either a scrollable tab or a side-located expandable list to display and select a request. The participants had initially considered different design options such as a pop-up list, a list-table in the center of the page and a dropdown list. These options had been discarded after the participants considered the usability issues as part of the design reasoning process.

In designing for requirement  $R_2$ , all participants except one in the test group used a pop-up window to specify the monitoring signals. Although other design options such as a dropdown list and a button-control had been considered by the participants, they decided that only one compromised solution is viable. This is because of the combination of constraints that are present: limited screen real-estate, the need to copy specified signals to multiple controls, and reusable programs. Some participants explored each branch of an option in detail and backtracked when the options became unviable. Overall, all participants have created a similar design. They have articulated similar issues related to usability. Most of them have identified a similar set of design options with minor variations. These minor variations are the placements of controls such as "buttons" and "tables".

There was one exception in the group. This design used a menu driven approach. Although the UI was still usable, it did not conform to the Nielsen UI design heuristics and therefore the scores of this design were lower.

<sup>1</sup> Wilcoxon's test is a non-parametric test suitable for comparing ranked data that makes no assumption about their distribution .

<sup>2</sup> We used the standard test of significance at 0.05. For one-tailed test of  $H_1$  in our case, the significance is at 0.025.

**Control Group.** In comparison to the test group, the control group produced much more diverse and less usable designs. First, as for designing UI to handle multiple requests (i.e., requirement  $R_1$ ), four varieties of design were proposed, including:

- *A textual list:* Requests are organized in the form of a hyperlink list on one side of the screen, whilst the rest of the screen estate displays details of a request.
- *Graphical icons:* Requests are represented as graphical icons numbered from 1 to 99. In this design, it takes the entire screen estate to show 99 icons.
- *A textbox:* There is no list to show an overview of all requests. Users retrieve details of a request by entering a unique identification of the request into the provided textbox. This design saves screen estate, but requires users to remember identifications of all requests.
- *A dropdown list:* A dropdown list shows all requests. This design saves screen estate, but scrolling down a long list of requests could be inconvenient.

Amongst these design variations, only the textual list is usable, and four out of ten participants in the control group ended up with this design. In the other six designs where the usability was low, the participants designed for multiple requests after they had finished designing for an individual request. In these cases, they did not reconsider if their individual request design fits multiple requests.

Second, for searching and selecting signals to specify a request (i.e., requirement  $R_2$ ), the control group derived three discrete designs, including:

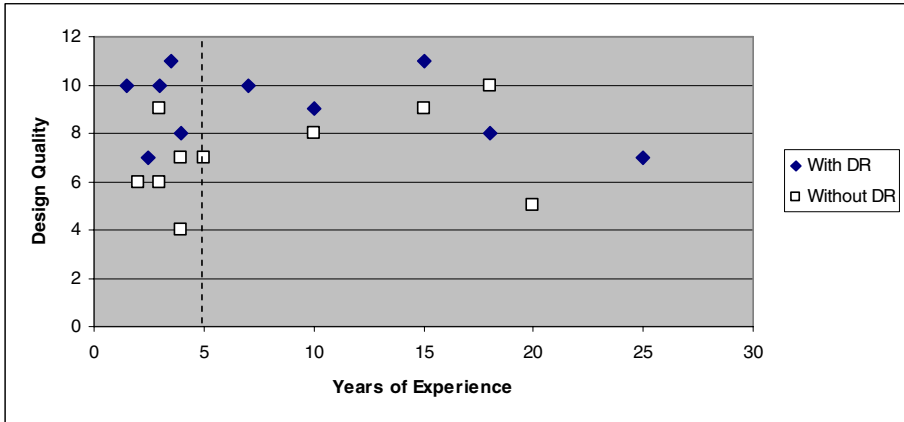
- *Pop-up window:* Buttons are included in different sections of a request (e.g., start condition, monitoring list and stop condition). When users click on a button, a pop-up window is displayed allowing users to select one or more signals to add to a particular section of the request.
- *Sequential pages:* Users must first search and select signal(s) that they want to monitor, and move to different pages to paste the information.
- *Tabbed windows:* Tabs are used to provide different functionalities. For example, the first tab shows a grid of all available signals from which users can choose, the second tab shows a list of users' selected signals, and the third tab shows start and stop monitoring conditions. Users click between them to copy signal information.

Out of these three designs, the pop-up window mechanism is the most suitable for the system, because it allows users to select signals with minimal mouse clicks and errors. Only five out of ten participants of the control group used a pop-up window as a means of searching and selecting signals.

### 3.3.2 Designers' Experience and Design Quality

Using the quality scores of the participants in the test group and the control group, we analyzed the relationships between their quality scores and their design experience. The results are shown in Fig. 2. The solid diamonds depict those who used design reasoning, and the hollow squares depict those who did not use design reasoning.

We notice that above the 5 year experience mark, the quality scores of the two groups are similar. They mostly score between 8 and 10 with some outliers. However, there is a noticeable difference in quality between the two groups of participants



**Fig. 2.** Design Quality Scores and Years of Experience

having less than 5 years of experience. The majority of the participants in the control group in this category scores between 6 and 8, and the majority of the test group in this category scores between 8 and 10. These results indicate that design reasoning helps less experienced designers to design better.

It is interesting to observe that the two most experienced designers did not produce the best designs. Both entered into the information technology field in the era of batch processing systems, way before graphical user interfaces (GUI) became an issue. Their design knowledge was formed in that pre-GUI era, and seems not changed all that much thereafter. This was confirmed by their actions and thinking-aloud during the experiments.

### 3.3.3 Design Process

We compare the duration that the two groups took to finish their tasks. On average, the test group took 39.30 minutes and the control group took 29.40 minutes. The Wilcoxon test shows no significant difference between the time spent by the test and the control group with  $p > 0.05$  (see Table 2). That means both groups took a similar amount of time to finish their tasks.

**Table 2.** Test and Control Group Design Time

	n	Mean Time (min)	Std. dev.	Wilcoxon Test
<b>Test Group</b>	10	39.30	10.86	$p = 0.113$
<b>Control Group</b>	10	29.40	10.08	

In addition, we considered the design processes of the two groups as described below.

**Test Group.** During the study, the test group was required to state their design options and design issues explicitly, and to justify why they made a particular decision at every design point. For example, when they explained the issues of the

initial design, they would say something like: “*how do I organize the UI to show 99 requests when they cannot all be displayed at the same time*” or “*how do I copy data from the signal search screen in a way that is easy to use*” or “*how many clicks are required to get the job done*”. They contemplated their initial design and reasoned about what it could and could not do. In some cases, participants believed that the initial design was adequate. However, when the participants consciously tried to find more design options, they often came up with alternative designs. There were many cases in the experiments in which such alternatives became the final design. However, there were cases when the design alternatives had helped to reinforce that the initial design was more appropriate when all alternative designs seem to be inferior.

After verbalizing the issues, the participants of the test group seemed to have a mental picture of those issues. The participants often considered how these issues conflict or work with each other in the design. The explicit verbalization of design issues and options helps the thought process when the participants started to formulate design options and to backtrack when certain design issues cannot be resolved. For instance, one participant explored the design issue of listing the 99 requests first and then he examined the issue of displaying and editing a single request, and finally the searching of signals. At every decision point, he would backtrack to assess if the chosen design options would work. This backtracking and verification of design options seemed quite natural to the participants of the test group when the issues and options were explicitly stated.

**Control Group.** The participants of the control group were not asked to state design issues and justify their decisions. We observed different design behavioral patterns depending on the experience of the designers. The first pattern was that most participants’ initial design became their final design, especially for designers who are less experienced. Unlike the test group, the control group’s design approach was based heavily on their intuition and on their first impression. The participants appeared to adhere to their initial designs, from where they continued to design for additional requirements. After each decision point, the inexperienced participants especially, spent little efforts on reassessing the consequences of additional changes to the initial design.

The second pattern was that even though the control group was aware of the usability guidelines, most of them did not consider the usability requirements at every decision point. We realized that the participants talked about the usability requirements initially but they became less conscious of them as the design progressed. The more experienced designers in this group were the exceptions.

### 3.3.4 Participants’ Feedbacks

In the follow-up interviews after the design session, all the participants were asked to comment on two things:

- (a) “*What are your key considerations for the design?*”
- (b) “*Do you have any comments on the design process you went through in this exercise?*”

In response to question (a), all participants of the test group mentioned that the usability of the UI was a key issue because of the complexity of the requirements and

the limited screen real-estate. Most participants said that the ease of use and understanding was another key issue and argued that minimal user clicks and minimal UI screens should be provided. Some of the participants also considered that the design should be reusable, particularly the signal search function.

Similar to the test group, the participants of the control group also commented that usability requirements were key to their design. In fact, a list of important usability concerns drawn from the control group was very similar to that of the test group. A convergence in the participants' comments on question (a) indicated that even though both groups of participants were aware of the usability requirements, they had very different approaches to tackling them. The participants in the test group were asked to explicitly state their design issues and design options, and their final designs were more consistent and more usable. Whereas with the control group, the participants carried out the design the way they normally do, the results among the participants were less consistent and showed an inadequate level of usability.

In response to question (b), some participants of the test group commented that well stated design issues helped them think through the design. Nine out of the ten participants in this group mentioned that the exploration of design options had helped their design. When they were asked why this was so, the general suggestion was that the design options allowed them to assess what would and would not work. In the control group, the inexperienced participants had very little comments on their design process, whilst experienced designers in this group were able to describe their requirement analysis and design process.

After the participants had completed their tasks, they were asked to rate their *satisfaction* with their own designs, using a seven-point Likert scale where 1 is not satisfied and 7 is fully satisfied. The test group reported an average of 5.7 and the control group reported an average of 4.9. When the participants were asked how *confident* they were on the usability of their design, using a seven-point Likert scale where 1 is not confident and 7 is fully confident, the test group reported an average of 5.1 and the control group reported an average of 5.5.

Although the test group was more satisfied with their design, they were slightly less confident about it than the control group. We cannot offer any explanations for this outcome, except by showing that the differences between the two group's ratings are statistically insignificant. The Wilcoxon test results show that there is no significant difference between the two groups' ratings on either the satisfaction or confidence of their design,  $p = 0.142$ , and  $p = 0.34$  respectively. This finding shows that participants from both groups were similarly confident and satisfied with their own designs despite the differences in the design qualities (as reported in Section 3.3.1).

## 4 Discussions of the Findings

The primary objective of this study is to analyze how design reasoning influences the quality of design. From the findings of the experiment, we have made a number of observations on how the participants tackled the design.

## 4.1 Discussions

Participants of both groups studied the requirements and the usability guidelines before creating their design. Analyses have shown that the test group has produced better quality UI designs than the control group in general. The following is a summary of the differences between the test group and the control group:

**Reasoning awareness.** By stating the design rationale, the participants of the test group have made explicit the reasoning underlying their designs. This imposed justification process had made the participants more cognizant of whether their decisions were correct. For instance, after spelling out that usability issues that concerned them, they had to find ways to ensure that their design was reasonable in dealing with the usability issue. As such, the test group was more aware of the usability requirement in the design compared with the control group. This result could reflect on the importance of reasoning of quality requirements in software architecture design.

Additionally, the reasoning approach induced the participants of the test group to explicitly reason about their design in a structured manner. Therefore, they were probably more careful in assessing their solutions in order to provide reasonable justifications. This contrasts with the participants of the control group who mostly used their intuition and knowledge to design. The control group's objective was to complete the design and satisfy the requirements without having to justify them.

**Usability awareness.** The participants of the test group identified usability as a key issue to be addressed in the design, and they consistently revisited this issue in the reasoning process. On the other hand, the participants of the control group considered usability in the early part of the study and then they were less conscious about it towards the end of a design session. It implies that an explicit design reasoning can help make designers aware of quality requirements throughout the design process.

**Initial design impression.** We observed that participants from both the test and control group formed initial impressions of a design solution initially. After exploring the design issues and options, participants in the test group may shift from the initial impressions of the design based on their design reasoning. On the other hand, the initial design impression played a more dominant role in the control group, especially with inexperienced designers, the initial design often became their final design. This result indicates that the initial design impression can be dominant, but a reasoning approach would help designers consider the design issues and options more carefully, allowing the designers to move away from the dominant belief to a design that is more appropriate.

**Design backtracking.** The participants of the test group backtracked their design and reconsidered their previously made decisions much more often than those of the control group. We suggest that it is because the test group was asked to explicitly state their issues, options and design rationale, and such acts forced them to address the issues that have been outlined, thereby achieving a level of systematic design reasoning. The control group generally did not reconsider previously made decisions as they built their design. Individual requirements were addressed but issues that arose from the conflicting requirements were not identified.

**Level of satisfaction and design quality.** The level of satisfaction and the level of confidence between the test group and the control group were not significantly different. This is despite that the test group had carried out the design process more thoroughly and produced higher quality designs. We also did not observe differences between the more experienced and less experienced designers. The results have shown that the level of satisfaction and the level of confidence of a designer on his/her design are not good indicators of the design quality.

**Design time.** There is no significant difference between the average time it takes for the test group and the control group to complete their tasks. It implies that the effort (in terms of time) spent by the test group is not significantly higher than that of the control group. Thus, we have not found evidence to indicate that using reasoning in design adds significant overhead to the design process.

**Experience level and design quality.** We have examined the design experience of the two groups and have found them to be similar. However, as shown in Table 1, the test group performed better on average than the control group. This is due to the higher scores achieved by less experienced designers in the test group (see Fig. 2).

In the test group, the design outcomes of experienced (i.e. over 5 years) and inexperienced (i.e. equal or less than 5 years) participants do not show much difference. They both produced good quality design. The inexperienced participants on average took longer to complete their design. In contrast to the test group, there is an observable difference in design quality between the experienced and inexperienced participants in the control group. Put differently, in the group of inexperienced participants, those that used design rationale consistently performed better than those that did not (see Fig. 2).

These results suggest that by using a design reasoning approach, less experienced designers could benefit from it to achieve a better quality designs. All designers, inexperienced and experienced, were briefed equally of the first principles of usability. Design reasoning has helped inexperienced designers in the test group to apply these principles successfully, and achieve what expert designers can do from mere experience, but design reasoning has shown little difference between experienced designers in the two groups. This suggests that experienced designers have the intuitions and insights to look for the right issues and options, as demonstrated by [7].

## 4.2 Limitations

This experimental study was based on a sample of twenty designers with industrial experience. We used a convenient sampling method to find the participants, i.e. the participants are the people whom we had access to and they were not randomly selected. The sample size in this study is small, and so there are limitations on the interpretations of the results.

The participants of the test group were explicitly required to state their design issues, design options and reasoning. Such reminders may have directed them to think more thoroughly, as such one could argue that the presence of the interviewers may bias the results. However, the interviewers did not provide any design hints, and the design decisions were deliberated entirely by the participants based on their knowledge

and reasoning abilities. Hence, we argue that the test results from the experiments are valid.

There are different experimental variables in such empirical studies that cannot be strictly controlled, e.g. participants' familiarity with the technologies involved. To overcome this limitation, we analyzed qualitatively what the designers have done and said about their designs to ensure that these variables do not affect the validity of the results. As for the quantification of the scores, the limitation is the bias the researchers may introduce. To overcome this, we have cross-checked all the designs to ensure that there is a consistency scoring across all designs.

## 5 Conclusions

Recent research works have argued that the explicit representation of design rationale is useful and can lead to better design outcomes. Yet there has been limited research to examine design reasoning's impact on design quality. Using usability as a software architecture quality attribute, we have studied how design reasoning influences the design quality, especially differentiating between experienced and inexperienced designers.

We have used an empirical study to examine the design quality of two groups of designers, one equipped with design reasoning and one without. The results of the experiment have shown statistically that a design reasoning approach improves the quality of design.

Designers who explicitly reason about their design decisions produce on average better designs. Furthermore, design reasoning appears to help inexperienced designers more than they do help experienced designers. The designers who do not use explicit design reasoning produce diverse results, and some of the designs have low usability, especially in the case of inexperienced designers. Therefore, we conclude that design reasoning helps inexperienced designers to better apply first design principles and to deliver a better design, by providing them with a deliberation framework and a mental image of the ongoing design.

These findings have provided encouraging empirical results to support further investigation into incorporating design reasoning in the software architecture design process.

**Acknowledgments.** This work is supported in part by the Australian Collaborative Research Centre for Advanced Automotive Technology and the Swinburne University of Technology Visiting Professor Award Scheme 2008.

## References

1. De Neys, W.: Implicit conflict detection during decision making. In: Proceedings of the Annual Conference of the Cognitive Science Society, vol. 29, pp. 209–214 (2007)
2. Tang, A., Barbar, M.A., Gorton, I., Han, J.: A survey of architecture design rationale. *Journal of Systems and Software* 79(12), 1792–1804 (2006)

3. Bosch, J.: Software Architecture: The Next Step. In: Oquendo, F., Warboys, B.C., Morrison, R. (eds.) EWSA 2004. LNCS, vol. 3047, pp. 194–199. Springer, Heidelberg (2004)
4. Jansen, A., Bosch, J.: Software Architecture as a Set of Architectural Design Decisions. In: Proceedings 5th IEEE/IFIP Working Conference on Software Architecture, pp. 109–120 (2005)
5. Bass, L., John, B.E.: Linking usability to software architecture patterns through general scenarios. *The Journal of Systems and Software* 66(3), 187–197 (2003)
6. Golden, E., John, B.E., Bass, L.: The value of a usability-supporting architectural pattern in software architecture design: a controlled experiment. In: Proceedings of the 27th International Conference on Software Engineering (ICSE 2005), pp. 460–469 (2005)
7. Cross, N.: Creative Thinking by Expert Designers. *The Journal of Design Research* 4(3) (2004)
8. Epstein, S.: Integration of the cognitive and the psychodynamic unconscious. *American Psychologists* 49, 709–724 (1994)
9. Evans, J.S.: In two minds: dual-process accounts of reasoning. *Trends in Cognitive Sciences* 7(10), 454–459 (2003)
10. Zannier, C., Chiasson, M., Maurer, F.: A model of design decision making based on empirical results of interviews with software designers. *Information and Software Technology* 49(6), 637–653 (2007)
11. Bratthall, L., Johansson, E., Regnell, B.: Is a Design Rationale Vital when Predicting Change Impact? – A Controlled Experiment on Software Architecture Evolution. In: Second International Conference on Product Focused Software Process Improvement, pp. 126–139 (2000)
12. Rittel, H.W.J., Webber, M.M.: Dilemmas in a general theory of planning. *Policy Sciences* 4(2), 155–169 (1973)
13. Maclean, A., Young, R., Bellotti, V., Moran, T.: Questions, Options and Criteria: Elements of Design Space Analysis. In: Moran, T., Carroll, J. (eds.) *Design Rationale - Concepts, Techniques, and Use*, pp. 53–105. Lawrence Erlbaum, Mahwah (1996)
14. Lee, J., Lai, K.: What is Design Rationale? In: Moran, T., Carroll, J. (eds.) *Design Rationale - Concepts, Techniques, and Use*, pp. 21–51. Lawrence Erlbaum, Mahwah (1996)
15. Conklin, J., Begeman, M.: gIBIS: a hypertext tool for exploratory policy discussion. In: Proceedings of the 1988 ACM conference on Computer-supported cooperative work, pp. 140–152 (1988)
16. Tyree, J., Akerman, A.: Architecture Decisions: Demystifying Architecture. *IEEE SOFTWARE* 22(2), 19–27 (2005)
17. Clements, P., Bachmann, F., Bass, L., Garlan, D., Ivers, J., Little, R., Nord, R., Stafford, J.: *Documenting Software Architectures: Views and Beyond*. Addison-Wesley, Reading (2002)
18. Tang, A., Jin, Y., Han, J.: A rationale-based architecture model for design traceability and reasoning. *Journal of Systems and Software* 80(6), 918–934 (2007)
19. Ali-Babar, M., Gorton, I., Jeffery, D.R.: Capturing and Using Software Architecture Knowledge for Architecture-Based Software Development. In: Proceedings of the Quality Software International Conference (QSIC 2005), pp. 169–176 (2005)
20. Carroll, J.M., Rosson, M.B.: A case library for teaching usability engineering: Design rationale, development, and classroom experience. *Journal on Educational Resources in Computing* 5(1), 1–22 (2005)

21. Mayhew, D.J.: The usability engineering lifecycle: a practitioner's handbook for user interface design. Morgan Kaufmann Publishers, San Francisco (1999)
22. Howard, S.: Trade-off decision making in user interface design. *Behaviour & Information Technology* 16(2), 98–109 (1997)
23. Norman, D.A.: Design principles for human-computer interfaces. In: Proceedings of the SIGCHI conference on Human Factors in Computing Systems, pp. 1–10. ACM Press, New York (1983)
24. MacLean, A., Young, R.M., Moran, T.P.: Design rationale: the argument behind the artifact. In: Proceedings of the SIGCHI conference on Human factors in Computing Systems, pp. 247–252. ACM Press, New York (1989)
25. Erikson, T.D., Simon, H.A.: Protocol Analysis: Verbal Report as Data. The MIT Press, Cambridge (1985)
26. Guan, Z., Lee, S., Cuddihy, E., Ramey, J.: The validity of the stimulated retrospective think-aloud method as measured by eye tracking. In: Proceedings of the SIGCHI conference on Human Factors in Computing Systems, pp. 1253–1262 (2006)
27. Nielsen, J.: Ten Usability Heuristics. (2007), [http://www.useit.com/papers/heuristic/heuristic\\_list.html](http://www.useit.com/papers/heuristic/heuristic_list.html)
28. Walpole, R.E., Myers, R.H.: Probability and Statistics for Engineers and Scientists. Macmillan Publishing Co., Inc, Basingstoke (1978)