Social Factors Relevant to Capturing Design Decisions

Carmen Zannier, Frank Maurer
University of Calgary, Department of Computer Science, Calgary, AB, CAN)
{zannierc, maurer}@cpsc.ucalgary.ca

Abstract

We present results from a qualitative study of design decision making that used interviews, observations and participatory observations to describe inherent traits of software design decision makers. We find that designers do not always strive for optimal design solutions, that designers do not always consider alternatives when making design decisions, and that alternatives are considered more often in groups of people having a casual conversation. We highlight that tool support for capturing design rationale and intent should first recognize the way decisions are inherently made in software environments and we provide a summary of our results as an indicator of requirements for such tools.

1. Introduction

We examine social aspects of decision making and the impact of these aspects on tool support for capturing design intent and rationale. Our understanding of some of the social aspects of design decision making comes from three qualitative multi-case studies we conducted, each asking the question: how do software designers make design decisions? We use our descriptions of design decision making to recommend requirements for tool support that captures design intent and rationale.

Our empirical studies and this paper are important for capturing design decisions because we believe decision making to be a highly cognitive and highly social process. Existing literature supports this idea [7][8][10][11] and we believe tool support for capturing design decisions should understand those social and cognitive processes, before tools are designed for supporting these activities. Our work provides an introduction to understanding these social and cognitive processes.

In our studies we examined design decision making in small software organizations (<= 10 developers), and we examined design decisions at varying levels of abstraction (i.e. source code, class hierarchies, architectural models). Many of our participants were familiar with or used agile methods. We define software design decision making as the selection of an option among zero or more known and unknown options concerning the design of a software system.

Section 2 reviews background material. Section 3 describes our 3 multi-case studies. Section 4 provides the main results and Section 5 addresses the impact on capturing design intent. Section 6 concludes this work.

2. Background

2.1 Decision Making

We have reviewed twelve approaches to decision making to provide a foundation for examining software design decisions [23]. This list of approaches is not exhaustive, but it demonstrates the numerous perspectives through which decision making can be analyzed. Our empirical studies have examined decision making through an evaluation of rational decision making [15], naturalistic decision making [13], real options analysis [4][20] and explanation based decision making **Error! Reference source not found.**.

2.2 Software Design Studies

A survey of software design studies suggests that six related factors impact software design: expertise, mental modeling, mental simulation, continual restructuring, preferred evaluation criteria and group interactions. Expertise is the knowledge and experience software designers have in design [1]. Existing studies showed expertise is fundamental to design productivity [6], and that higher expertise resulted in an improved ability to create internal models and run mental simulations [1][2]. Mental modeling is the creation of internal or external models by a designer. A mental model is capable of supporting mental simulation [1]. Mental simulation is the "ability to imagine people and

objects consciously and to transform those people and objects through several transitions, finally picturing them in a different way than at the start" [13]. Mental simulations occurred throughout the software design process at varying levels of quality dependent upon the skill of the designer and the quality of the model on which the mental simulation ran [1][6][7][8]. Continual restructuring is the process of turning an ill-structured problem into a well-structured problem. The term "preferred evaluation criteria" refers to the minimal criteria a subject adopts to perform continual restructuring [13]. The use of preferred evaluation criteria occurred on an individual level or group level [21][7][6]. Group interactions are the dynamics of group work in software design. The terms "distributed" and "shared" cognition suggest that individual mental models coalesce via group work, resulting in a common, or overlapping, mental model [7][21].

2.3 Qualitative Studies

Much of empirical software engineering is quantitative which uses hypothesis testing, control variables, and primarily quantitative metrics [12]. Laboratory experiments are a prime example of a quantitative study, requiring much control in manipulating examined variables. Qualitative empirical studies leverage the context of a situation as critical to understanding, are often emergent in theory-building and use primarily qualitative metrics and descriptions as data [16]. The term "qualitative inquiry" is often used "as a blanket designation for all forms of social inquiry that rely primarily on qualitative data." [18] The term empirical refers to the group of "epistemological theories that accept the premise that knowledge begins with sense experience" [18]. Real-world case studies are an example of a qualitative empirical study. The difference between qualitative empiricism and quantitative empiricism does not lie solely in the type of data analyzed (e.g. quantitative data can be used in case studies) but in the overall approach to experimentation [16].

3. Our Empirical Study

We conducted three multi-case studies. The purpose of each study was to describe design decision making through a given lens (i.e. from a given perspective) thereby contributing to answering the larger research question: How do software designers make design decisions? Aspects of qualitative empiricism recommend conducting a study with as few preconceptions as possible, to minimize researcher bias [16]. Some even recommend not conducting literature reviews [16]. The practicality of this, however, is challenging – we all have experience we bring to new situations. Thus, for our studies, we made at least some of our preconceptions explicit, by defining the perspective through which we examined software design decision making. We say we had "a lens on", some existing description of, or a prescription for decision making, and compare it to what we found in each study.

In the first study the lens was a comparison of rational decision making and naturalistic decision making. Properties of multi-attribute decision theory (a rational approach to decision making) [15] were compared to properties of recognition-primed decision theory (a natural approach to decision making) [13]. The purpose of the first study was to describe design decision making through the presence or absence of the components of rational and natural decision making. In the second study the lens used was a description of real options theory [20][4]. Real options theory has been prescribed for solving software design decisions [20][4]. The purpose of the second study was to describe design decision making through the presence or absence of the components or real options theory. In the third study the lens used was a description of explanation based decision making, an approach to decision making used in jurors for court cases [17]. (nb. this is not the same as "explanation building" in case studies [22]). The purpose of the third study was to describe design decision making through the components of explanation based decision making and to improve the use of explanation based decision making by fostering communication in team meetings.

Each study used a different approach to collecting data. In the first study we conducted 25 interviews with software designers. We asked them to describe a design change they had championed. Our interview format was semi-structured, meaning the themes of each question were the same for all participants, but the wording of the question was not. In the second study we conducted 9-10 day observations at 3 different software organizations. We watched software developers work. We were silent for the first day, then asked questions for the second day, for each developer we sat with at an organization. In the third study we participated in a development team's discussion of design decisions they had to make for a new application. This occurred at a fourth organization and our participation lasted 6 weeks. Our analysis varied slightly for each study but used content analysis, explanation building and cross case analysis [14][22].

4. Results

We present three results. The first is that designers do not consistently strive for an optimal design solution, or a boundedly optimal design solution. The second is that the approach taken to solve a design decision problem depends on the strength of the mental model the software designer holds. The third result is that alternatives are considered through conversation.

4.1. Optimal Design Solution?

In our interviews we found software designers do not consistently strive for an optimal design solution when making a design change. We found this in three ways. First, when conducting content analysis, we coded interview transcripts with the code **Satisfice** and with the code **Weight**. **Satisfice** meant that a designer identified some decision as being satisfactory, and is a property of naturalistic decision making [13]. **Weight** meant that a designer compared alternatives, with the intent of selecting the alternatives with the most advantages. **Weight** is a property of rational decision making [15]. We found as many occurrences of the code **Satisfice** as we found the code **Weight**, quantitatively showing an equal pursuit of optimal design solutions and satisfactory design solutions [23]. Second, we found quotes representative of the idea of the code **Satisfice**, which qualitatively showed the pursuit of satisfactory design solutions. For example:

"… it just had to work, it didn't have to have all the fancy clowns and all the bells and whistles it just really had to work."

"…you should have a very full bag of tricks, you should know as many things as you can, but you shouldn't reach very deep into that bag for solutions."

Third, we used explanation building to show that 12 of our 25 interview participants followed satisficing when referring to the design or indicators that a design change needed to be made [22][23]. These three points contributed to the point that designers do not consistently strive for optimal design solutions. The idea of goodness of fit [5], or the idea of good enough software [3] is well known in software development communities. Empirically supporting these anecdotal summaries is a contribution of our work. .

4.2. Considering Alternatives?

In examining the general approach to decision making we looked for one of two approaches: consequential choice or serial evaluation. Consequential choice is defined as the concurrent comparison and trade-off evaluation of more than one option in a decision [15][13]. Serial evaluation is the sequential evaluation of options (n.b. no tradeoff evaluation and no concurrency) in a decision [13]. We found decision alternatives were considered when the decision was "ill structured", or the decision maker did not have a strong mental model of the design [23].

Software design is a problem structuring activity accomplished throughout the software development lifecycle [8][9][19][23]. Our results show that the less-structured the problem, the more software designers considered alternatives. The more structured the problem, the more designers used serial evaluation [23]. A quote from our interviews is below.

"…if I'm quite confident that I know of a good solution that does what I need I will just do that and that's a question of experience. If I've seen it before, and I know where it's going then I just go ahead and get there. A lot of the time though, all I have are these rules and I don't necessarily know in advance, if I apply this rule, where am I going to end up? I know that I can do four things here, I don't know where any of those four things will lead me. So let's just see what happens. It's a lot of very small experiments with the goal of accumulating the experience and heuristics I can use next time to inform my decision."

4.3. Conversation

Our third result came primarily from our observations. Our observations strongly suggest that environments that fostered continual direct conversation led to more use of consequential choice (rather than serial evaluation) than environments that did not foster continual direct communication. We observed decision making at three organizations. The first company fostered minimal continual direct conversation and used consequential choice in 4 of the 12 decision events we observed (33.3%). The second company fostered much continual direct conversation and used consequential choice in 9 of 11 decisions we observed (81.8%). The third company did not foster continual direct conversation and used consequential choice in 2 of 5 decisions observed (40%).

We emphasize that our observations showed consequential choice when more than one developer contributed to the decision. In the 15 decision events where consequential choice was used, 14 of these involved more than 1 person contributing to the decision. At the first company more than one person was involved in decision making in 3 of the 12 decision events we observed. At the second company more than one person was involved in decision making in 9 of 11 decision events we observed. At the third company more than one person was involved in decision making in 1 of the 5 decision events we

observed. While we recognize that there are many possible contributors to the approach to decision making, at this point we believe additional people communicating about a design provides more ideas to the solution and makes the problem less structured than it would otherwise be. This suggests that a decrease in discussing design decisions may lead to a decrease in the use of consequential choice, which is, we believe, a decrease in the *opportunity* to select a good decision alternative among many decision alternatives.

5. Impact on Decision Support Tools

We discuss the importance of these results in light of tool support to capture design intent. Our first result, that designers do not always strive for an optimal design solution, means decision support tools should capture decisions that are made in a satisficing manner. There should be room for "just because" rationale or rationale that acknowledges satisfactory solutions. The second result, that the approach depends on the structure of the problem or the strength of a designer's mental model, means that decision support tools should be interwoven with tools that manage existing models of applications under design, and accommodate serial evaluation when those models are well-defined. The third result, that alternatives were considered primarily when conversation among colleagues occurred, means that decision support tools should recognize the social context in which a decision occurs and the tools should be easily integrated into these social contexts (nb. not the other way around). Often, "the emphasis on finding and describing 'knowledge structures' that are somewhere 'inside' the individual encourages us to overlook the fact that human cognition is always situated in a complex socio-cultural world and cannot be unaffected by it" [11]. Decision support tools should support these socio-cultural contexts. For example, tool support for decision capture could focus on single display groupware support (multiple users interacting simultaneously on a single device) instead of developing personal tools for capturing rationales.

6. Conclusion

We summarized results from an empirical study on design decision making to provide direction for decision support tools that recognize the inherent way that designers work. If we wish to capture the intent and rationale behind design decisions we must recognize that sometimes design decisions are made "just because", sometimes they are sub-optimal, sometimes they are made without considering alternatives and sometimes they are made during casual conversations, where tools are not used.

10. References

[1] Adelson B, et al. "The Role of Domain Experience in Soft. Design"; *IEEE Trans. Soft. Eng* 11 11 Nov. 1985.

[2] Ahmed S et al. "Understanding Differences Between How Novice & Experienced Designers Approach Design Tasks"; *Res. Eng. Design* 14, 1-11 2003.

[3] Bach J; "The Challenge of Good Enough Software", *American Programmer* 1995

[4] Boehm B.W, Sullivan, K.J; "Software Economics, A Roadmap"; *Int. Conf. on Software Engineering*, Proc. Conference on the Future of Soft. Eng, Limerick, IE pp: 319 – 343, 2000

[5] Booch G; "Goodness of Fit", *IEEE Software*, 23 6 2006

[6] Curtis B, et al. "A Field Study of the Soft. Des. Process for Large Systems", *Comm. ACM*, 31 11 Nov. 1988.

[7] Gasson S; "Framing Design: A Social Process View of Information System Development"; *Proc. Int. Conf. Information Systems*, Helsinki Finland, 224-236; 1998.

[8] Guindon; Designing Design Process *HCI*5 305-344 1990

[9] Guindon; Knowledge Exploited by Experts during Soft. Sys. Design *Int. J. Man-Mach. Stud*. 33, 279-304; 1990.

[10] Highsmith; Agile Proj.Management; Add-Wesley 2004

[11] Hutchins E; *Cognition in the Wild*, MIT Press, Cam. MA, 1995

[12] Juristo N, Moreno A.M; *Basics of Software Engineering Experimentation*; Kluwer Academic Pub., Boston MA; 2001

[13] Klein; *Sources of Power*, MIT Press Camb. MA; 1998

[14] Krippendorff; *Content Analysis*; V5 Sage. Lond. 1980

[15] Luce et al. *Games & Decisions*, John Wiley & Sons NY 1958

[16] Patton M.Q; *Qualitative Research & Evaluation Methods* 3rd Ed.; Sage Pub, CA; 2002.

[17] Pennington et al "A Theory of Explanation-Based Decision Making"; In: Klein et al, editors. *Decision making in action*. Norwood, NJ: Ablex Pub; 1993.

[18] Schwandt T.A, *Dictionary of Qualitative Inquiry* 2nd Ed., Sage Publications, CA, 2001

[19] Simon H; "The Structure of Ill Structured Problems"; *AI* V4, 181-201; 1973

[20] Sullivan et al; "Software Design as an Investment Activity: A Real Options Perspective" *Real Options & Business Strategy*, Trigeorgis, London, ENG 215-261, 1999.

[21] Walz D.B, et al. "Inside a Software Design Team"; *Comm. ACM*, V.36 No.10 Oct 1993.

[22] Yin, R.K; *Case Study Research: Design & Methods* 3rd Ed. Sage Publications, CA, 2003

[23] Zannier C et al; "A Model of Des. Decision Making based on Empirical Results Interviews Soft. Designers" *Understanding Social Side of Soft. Eng. Special Issue J. Info. & Soft. Tech*., to appear Spring 2007