

Creativity and Rationale in Software Design

John Daughtry
The Pennsylvania State
Univ.
daughtry@psu.edu

Janet Burge
Miami University
burgeje@muohio.edu

John M. Carroll
The Pennsylvania State
Univ.
jcarroll@ist.psu.edu

Colin Potts
Georgia Institute of
Technology
cp25@mail.gatech.edu

DOI: 10.1145/1457516.1460354

<http://doi.acm.org/10.1145/1457516.1460354>

Abstract

An NSF sponsored workshop on Creativity and Rationale in Software Design was held at University Park, PA in June, 2008. The participants represented the spectrum of software design, which was reflected in the discussions. This report summarizes the workshop with respect to the discipline of software engineering.

Keywords: Rationale, Design Rationale, Software Engineering, Creativity

Introduction

Creativity is often associated with innovation and inspiration—the conception of a brilliant new idea or discovering a novel approach to solving a problem. According to this view, there should be no clear reasons for a creative discovery. Rather, creative design proceeds through ineffable, non-reproducible leaps. Rationale, in contrast, consists of the reasons behind the decisions that are made while designing. It explains and justifies why the design is the way it is and what criteria were considered. A simple, but probably incorrect account of the relationship between creativity and rationale would therefore imply that creativity responds to romantic unfettered impulses whereas rationale is the result of cool reasoning. A less sensational account would accept that creativity and rationale are compatible if sometimes uneasy bedfellows, sometimes competing and sometimes complementary. Rationale can support creativity, by providing mechanisms to encourage and capture ideas, but may also inhibit it, by requiring designers to explain and reason explicitly while creating. Both are clearly necessary in successful design processes, and so the aim of this workshop was to explore the complicated relationship between these two differing faces of design.

Seven questions served to orient discussions at the workshop:

1. When and how can design rationale evoke creativity in design? For example, does/can design rationale function differently (more effectively) in end-user design, participatory design, pair programming/agile design, or open source design communities?
2. When and how can design rationale fail to evoke, or even undermine, creativity?
3. How can the construction of design rationale be construed and experienced as a creative activity? And how can this be enhanced?
4. What tools and methods for rationale can support or enhance the creativity of design products? For example, how much structure should design rationale tools provide/impose to maximize creative outcomes (e.g., contrast QOC, gIBIS, and design blogs).
5. How might valuing the creativity of rationales inspire new forms of design rationale? What would be characteristics of such new forms of rationale?
6. How can design rationale be used in the classroom to motivate and instruct students about reflection, idea generation, and evaluation?
7. What are useful models, theories, and frameworks for understanding and managing the relationship between rationale and creativity in design?

Workshop participants prepared for the workshop and explored these questions in position statements giving their view of creativity and rationale. These statements represented a broad range of ideas including: education, knowledge management, a designerly way of knowing, and generative rationale.

Discussion and Themes

The essence of the workshop's guiding questions lies in questions one and two; when and how can rationale evoke, fail to evoke, or undermine creativity? Creative endeavors are typically viewed as the product of inspiration, not analysis. Recording rationale, especially in a structured semi-formal representations, can prematurely narrow the breadth of design reasoning [2]. From this perspective, requiring the designer to articulate each decision, the alternatives considered, criteria used, and tradeoffs considered can appear to inhibit the creative process.

Other arguments, however, suggest that rationale can enhance creativity. Rationale can support idea generation by providing a mechanism to capture, explore, and develop ideas generated during design. Rationale can also support subsequent evaluation of these ideas as well as of the design itself. Rationale can also support design explanation by capturing the designer's intent and how it affected the final design.

Figure 1 is a representation of the views of this bivalent relationship between design rationale and creativity, as presented in the position papers. Each of these perspectives were explored in more detail over the course of the workshop within various contexts.

Learning from Practice

Creativity and rationale can each be viewed as objective entities or as socially constructed and situated, concepts. Indeed, two developers can (and often do) understand the same design decisions in different ways. And, what one design community deems as creative design may be dismissed as routine or paradigm-bound design in another community.

Often discussed at the workshop was the notion that, if we are to make progress in understanding the relationship between creativity and rationale in software design, we should study practitioners actively engaged in designing software. In particular,

we need to develop an operative definition of creativity and rationale that has utility within the modern software development environment.

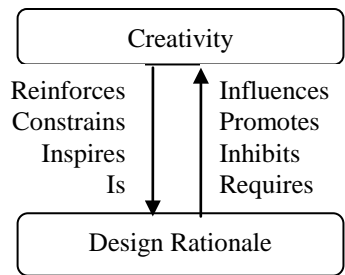


Figure 1 - Design Rationale and Creativity

Creativity is a term that is common within our conventional and conversational language. And, it is through this definition that we recognize the tension between creativity and rationale. Despite its common usage, however, one finds it difficult to define creativity explicitly and comprehensively. As one workshop participant noted, various dictionaries and encyclopedias vary from containing 1 to over 50 distinct definitions of creativity.

When trying to relate creativity with rationale we must be able to operationalize the concepts. However, we must both operationalize our terms grounding our usage in the *lingua franca* of the software design community, and, at the same time, operationalize in a way that allows us to situate our work within the broader research communities of creativity and design.

Towards this end, it would be productive for us to seek a better understanding of creativity as defined by software professionals. What do *they* describe as being creative and rational? Who do *they* describe as being creative and rational? Why do *they* describe these objects/people/processes as creative and rational? The answers to these questions were not assumed or asserted in the workshop, but rather explicitly left undefined throughout our discussions. While this led to some confusion, it allowed us to move with some degree of fluidity around the many perspectives. Nevertheless, to make further progress toward a research program for creativity and rationale in software engineering, we must develop operative definitions for creativity and rationale that are grounded at once in the state-of-practice and in the state-of-research.

While we seek to operationalize creativity and rationale, we can also seek an understanding of how developers describe their work and designed artifacts. Do developers relate creativity with rationale in their own descriptions? Do they describe a particular rationale process as being creative? Do they describe the rationale itself as being creative? Do they describe an artifact in terms of its rationale and deem it creative? Do they describe a creative thing in terms of its rationality? We should learn from those that navigate the space without needing operationalized concepts as we operationalize them ourselves.

In studying description, we should focus on the social vs. individual nature of these descriptions. Indeed, it is reasonable to presume that rationale is socially constructed along with a design in, for example, a multi-disciplinary team. How does a software design professional convey his or her description to another? How do these individuals construct meaning and understanding with

respect to creativity and rationale? And, how do these descriptions change with respect to the audience and over time?

Akin to description is modality. As one workshop participant noted, descriptions can take many forms; in sculpture, an ineffable description that is common is via the sense of touch. Rather than describing a flaw or feature, one touches it and feels it. This sense of touch is less likely in the invisible world of bits and bytes, but descriptions and rationale among software design professionals may take other forms.

Finally, we should seek to identify exemplars of creativity and rationale that are at once salient and relevant to practice to pursue a research agenda relating creativity with rationale in software design. What are some exemplars for rationale evoking, failing to evoke, and undermining creativity? The balance between salience and relevance leads one towards practitioners.

Exploring the Design Space

A theme arising throughout the workshop was how the relationship between creativity and rationale is related with the tools and techniques used when designing. Two examples often raised were modality and iteration.

The modality of software design communication can easily be circumscribed as documentation. But software designers do talk to each other when they design. One participant noted that, in sculpture, touch is an important means of communication. In order to explain what is wrong with a sculpture, an instructor places the students hand on one that is correct. Perhaps touch is more effective than spoken word.

Unlike sculptors, we are not in tactile contact with our medium as software designers. But what special modalities are available to us? What could be made available? Which of these evoke, fail to evoke, or even undermine creativity and in which contexts? If there are nonverbal aspects of creativity in software design, which is likely, then how do they find expression in design reasoning? Do we undermine creativity through restricting communication to written symbols? Are there aspects of creativity and rationale that are modality independent?

The increased prominence of evolutionary software design also significantly alters how we must perceive creativity and rationale. It means that we must find ways to represent differences in time and composition of components. How does the rationale in version 1.0 impact or influence the creativity in version 3.0?

Further, we must seek a better understanding of modality with respect to time. Which modalities stand the test of time with respect to not only evolutionary design but also long-term dramatic shifts within the socio-technical culture within which we as software designers operate?

The informal creativity literature is replete with techniques for encouraging practitioners to become “unstuck” in their thinking. Despite the romantic view of creativity as an unfettered impulse, these tend to follow highly and artificially structured procedures. One general mechanism behind many of these techniques is that of analogy and comparison, particularly incongruous or surprising comparisons. According to Koestler’s [8] informal and pioneering work on creative thinking in humor, art and science, *bisociation*, the bringing together of fertile but incompatible conceptual frameworks, is the basis of creative discovery and accounts for the sense of surprise or catharsis that often accompanies creative realization. Given this marriage of the cognitive unconscious

perceptual and search processes to the emotional responses of creative thought, one question that the participants investigated is the need to develop techniques that enable the search for and recovery of rationale that are rewarding to practitioners rather than being chores and that exploit the ability of designers to seek and find connections and analogies.

Kinds of Design Rationale

Design rationale (DR) can come in many forms ranging from very informal representations to very structured ones. Semiformal argumentation notations, such as IBIS (Issue-Based Information Systems) [9] capture rationale in the form of issues raised, alternatives considered, and the reasons for and against the design alternatives. An alternative approach is scenarios claims analysis [5] where DR is captured in the form of scenarios that make claims about the design. Irrespective of its form, rationale matters in software engineering [4].

Rationale capture and use can be categorized along a number of dimensions. The use of rationale can be prescriptive, where the rationale is used to assist with decision-making by evaluating decisions as they take place or by providing recommendations based on either previous designs or earlier versions of the current design. Rationale can also be viewed as being primarily descriptive—a richer documentation of the design process than is provided by standard design documentations. Another categorization is *when* the rationale is captured. Is it prospective, captured during design to assist with the designing process, or retrospective, captured after design to capture what happened? Retrospectively generated rationale can document the results of the creative process but can only assist with creativity of future designs.

Designers often do not capture rationale, because the capture is viewed as time consuming, costly, and tedious. There is not much data on the cost of capture, but there have been some studies on its usefulness. Scenario-based rationale has been shown to be successful at supporting design explanation, but argumentation-based rationale was less successful [7]. Some studies of using argumentation-based rationale had mixed results [2] while others indicated that it was a valuable design resource [1]. Most DR techniques and tools have received little formal evaluation, most providing evaluation in the form of case studies only [3].

The workshop raised many questions on the ability of rationale to support creativity. Can DR be used to support creativity? To what extent do existing methods/techniques for rationale enhance/inhibit creativity? There have not been studies comparing the creativity of designs created with or without the assistance of rationale. Such a study would be challenging since it would require metrics for evaluating creativity as well as a way to account for differences in creative ability for the designers studied. What skills and competencies are required to capture/use design rationale and how to those relate to creative ability? Are there circumstances under which design rationale is more supportive of creativity? One frequently mentioned use of rationale is to support collaboration. Does collaboration encourage creativity or is there a risk of group-think inhibiting creativity? Can rationale serve as a mechanism to force groups of people to analyze and debate ideas rather than jumping to a premature consensus? Can the conflicts between design criteria captured in rationale provoke creative solutions to these conflicts?

Our Challenge

While there was significant variety in perspectives among participants, they agreed that creativity and rationale have mutual influence and should be considered in parallel. This allows us to investigate not only the effect of rationale on creativity but also explore creative ways to use rationale.

As articulated by one breakout group during the workshop, the grand challenge for this research area is to create a framework where rationale and creativity could be connected. At the same time, the secondary challenge is the pursuit of heuristics, rules, and concrete examples where rationale serves design with respect to creativity (i.e. – provocation and undermining). The connection between creativity and rationale is complex and nuanced, but a relevant and just pursuit if our long-term aims include working towards level 3 theories of software engineering (all embracing) as defined by [10] and articulated by [6].

Acknowledgements

This workshop was sponsored by the National Science Foundation CreativeIT program under grant 0742392.

References

- [1] Aurisicchio M., M. Gourtovaia, R. Bracewell, and K. Wallace (2008): How to Evaluate Reading and Interpretation of Differently Structured Engineering Design Rationales, *Artificial Intelligence in Engineering Design, Analysis, and Manufacturing*, 22(4), 345-358
- [2] Buckingham Shum, S., and N. Hammond (1994): Argumentation-based Design Rationale: What Use at What Cost? *International Journal of Human-Computer Studies*, 40(4), 603-652.
- [3] Burge, J. (2008): Researching Under Uncertainty, *Artificial Intelligence in Engineering Design, Analysis, and Manufacturing*, 22 (4), 311-324
- [4] Burge, J., J. Carroll, R. McCall, and I. Mistrik. (2008): *Rationale-Based Software Engineering*. Heidelberg: Springer-Verlag.
- [5] Carroll, J., and M. Rosson, (1992): Getting Around the Task-Artifact Cycle: How to Make Claims and Design by Scenario. *ACM Trans. Inf. Syst.* 10(2), 181-212.
- [6] Carroll, J. and P.A. Swatman, (2002): Structured-case: A Methodological Framework for Building Theory in Information Systems Research, *European Journal of Information Systems*, 9: 235–242
- [7] Haynes, S.R. (2006): Three Studies of Design Rationale as Explanation. In *Rationale Management in Software Engineering* (Du-toit, A., McCall, R., Mistrik, I., & Paech B., Eds.), Heidelberg: Springer-Verlag, pp. 53-71.
- [8] Koestler, A. (1964): *The Act of Creation*, New York: Macmillan.
- [9] Kunz, W., H. Rittel, (1970): Issues as Elements of Information Systems. Working Paper 131, Center for Urban and Regional Development, University of California, Berkeley
- [10] Sjøberg, D., T. Dybå, B. Anda, and J. Hannay. (2008): *Building Theories in Software Engineering. Guide to Advanced Empirical Software Engineering* (Eds. Forrest Shull, Janice Singer, and Dag I.K. Sjøberg). Heidelberg: Springer-Verlag, 312-336