Artificial Intelligence for Engineering Design, Analysis and Manufacturing (2008), *22*, 309–310. Printed in the USA.
Copyright © 2008 Cambridge University Press 0890-0604/08 $25.00
doi:10.1017/S0890060408000206

GUEST EDITORIAL

Special Issue: Design Rationale

JANET E. BURGE[1] AND ROB BRACEWELL[2]
[1]Computer Science and Systems Analysis Department, Miami University, Oxford, Ohio, USA
[2]Engineering Design Centre, Department of Engineering, University of Cambridge, Cambridge, United Kingdom

The process of designing can be viewed as making a series of design decisions. These decisions, the alternatives considered, the reasons for and against the decisions, and the dependencies among all of these form the rationale for the design. This *design rationale* provides a history of the design process as well as capturing the intent behind the decisions made.

Design rationale has been an active area of research since the 1970s with the introduction of issue-based information systems (IBIS) as a means for representing argumentation-based rationale (Kunz & Rittel, 1970). Rationale research has been applied to many design disciplines including engineering design (Lee, 1997), human–computer interaction (Moran & Carroll, 1996), and software engineering (Dutoit et al., 2006)(Burge et al., 2008).

Despite over 30 years of research, there are still few rationale systems used in practice. There is a strong consensus that rationale is very valuable, but there is an equally strong concern that the costs of its capture may be too high. In order to justify the costs of its capture, it is essential to establish ways in which rationale can be useful that exceed the simple provision of additional design documentation.

The seven articles in this Special Issue address the usefulness of rationale. One article points out how much is still unknown about both the cost of capturing the rationale and how it can be most useful. It concludes that successful technology transfer cannot occur until more is known about where rationale is really needed and how well current approaches meet those needs. The remaining articles describe research into the uses of rationale that include capturing and maintaining design constraints, understanding existing designs, supporting design research, providing traceability for customer needs during the early design stage, capturing knowledge generated during building redesign, and supporting the reuse of high-level software designs. These articles describe a wide spectrum of design rationale uses and methodologies in multiple domains and at multiple levels of formality.

"Researching Under Uncertainty," by Burge, describes the controversy over the usefulness of rationale versus the potential cost of its collection. The arguments both for and against rationale are presented along with a list of rationale approaches and how they were evaluated. Although there has been significant research activity proposing rationale approaches over the last 30 years, little data on the actual use and usefulness of rationale have been collected. The article also presents the results of a survey of software practitioners and their predictions of how rationale could be used. The results of this survey appear counterintuitive and indicate the need for more emphasis on gathering and reporting data on the uses for rationale and its usefulness.

"Constraint Capture and Maintenance in Engineering Design," by Ajit, Sleeman, Fowler, and Knott, tackles the important task of ensuring that designs are consistent with their specification and design rules defined by the organization developing the design. The relationship of the design rules to the problem can be defined as constraints. It is necessary to capture these constraints and to understand the conditions (assumptions and context) under which a constraint applies. These conditions form part of the rationale for the constraint. The methodology that is described supports constraint maintenance by detecting inconsistencies, subsumption, and redundancy as well as performing fusion between constraints and assisting with constraint refinement.

"How to Evaluate Reading and Interpretation of Differently Structured Engineering Design Rationales," by Aurisicchio, Gourtovaia, Bracewell, and Wallace, describes an empirical study on reading and interpreting technical documentation that is provided to aerospace trainees in a structured text format provided by DRed (using IBIS-style argumentation) and as unstructured text design definition reports. The article provides the results of this evaluation as well as a methodology for similar empirical tool evaluations. The experiment participants were provided with information in one of the two formats and asked a series of questions about it. The results indicate that the more structured DRed format has a positive impact on both the answers to the questions and the time required.

Reprint requests to: Janet E. Burge, Computer Science and Systems Analysis Department, Miami University, Oxford, OH 45056, USA. E-mail: burgeje@muohio.edu

https://doi.org/10.1017/S0890060408000206 Published online by Cambridge University Press

"Scientific Design Rationale," by Haynes, Bach, and Carroll, applies design rationale to the science of design by describing an empirical study of the roles that design rationale can play in design research. The analysis of the study results identified five affordances in which design rationale supported design research: *marshalling theory*, where design rationale was used to trace theories to the decisions and resulting artifact features; *synthetic science*, where design rationale supported the use of expertise and knowledge in designing; *research apparatus*, where design rationale was a mechanism for observing and analysis of the design; *boundary representation*, where the design rationale provided a common representation to communicate the research activities and results; and *description, explanation, prediction, discovery*, all uses of design rationale during design research.

"Antecedence and Consequence in Design Rationale Systems," by Agouridas and Simons, is concerned with how design rationale systems can better support the early design stage. During this stage it is crucial to analyze the needs of the customer and trace them to their *antecedents*, which in effect are the rationale for those needs. It is also necessary to determine how the stakeholder needs will be fulfilled, which are their *consequences*. This article presents a conceptual framework, which is a set of ontological primitives, for evolving requirements in response to changing stakeholder intentions by capturing the entities and the antecedent–consequent relationships involved as the problem formulation changes.

"Chunks, Lines, and Strategies," by Lindekens and Heylighen, focuses on capturing the rationale developed by architects while they are redesigning a building. The observation of architects during redesign identified three mechanisms for recording their rationale: *chunks*, which capture dependent fragments of the design discussion, such as a specific part of the design like a door or window; *lines*, which capture "lines of thought" followed during redesign (which may include several chunks of discussion); and *strategies*, which steer the different lines of thought. Strategies could potentially be reused in multiple design projects. The approach was evaluated by creating an on-line repository of rationales, providing them to a set of architects (students, practitioners, and academics), and assessing their ability to represent the design process followed during a competition design and their ability to explain that process.

"Kuaba Approach: Integrating Formal Semantics and Design Rationale to Support Design Reuse," by de Medeiros and Schwabe, supports the reuse of model-based software designs utilizing an argumentation-based design rationale representation model integrated with a metamodel of the design method that describes the artifact being designed. The intent is to be able to reuse software design at a high level where rationales can be used to aid in designing a new system. The Kuaba ontology extends the IBIS argumentation structure (Kunz and Rittel, 1970) by integrating it with descriptions of the resulting artifacts as well as a design history describing additional information about the decisions such as who made them and when. The ontology, represented in F-logic, also provides rules for validating the design rationale.

The seven articles selected for this Special Issue were chosen after two rounds of reviews. In the first round each article was peer reviewed by multiple reviewers, and in the second round the revised articles were reviewed by the Guest Editors. We thank the reviewers and the authors for their hard work as well as Professor David Brown, the *AIEDAM* Editor In Chief, for his advice and patience during the process of compiling and editing this Special Issue.

REFERENCES

Burge, J.E., Carroll, J.M., McCall, R., & Mistrík, I. (2008). *Rationale-Based Software Engineering*. Heidelberg: Springer–Verlag.

Dutoit, A., McCall, R., Mistrík, I., & Paech, B., Eds. (2006). *Rationale Management in Software Engineering*. Heidelberg: Springer–Verlag.

Kunz, W., & Rittel, H. (1970). *Issues as Elements of Information Systems*, Working Paper 131. Berkeley, CA: University of California, Berkeley, Center for Urban and Regional Development.

Lee, J. (1997). Design rationale systems: understanding the issues. *IEEE Expert: Intelligent Systems and Their Applications 12(3)*, 78–85.

Moran, T., & Carroll, J., Eds. (1996). *Design Rationale Concepts, Techniques, and Use*. Hillsdale, NJ: Erlbaum.

Janet E. Burge is an Assistant Professor in the Miami University Computer Science and Systems Analysis Department. She has worked in industry as a researcher and software developer for over 20 years. She received her PhD and MS in computer science from Worcester Polytechnic Institute and her BS in computer science from Michigan Technological University. Dr. Burge's major research interests are in software engineering and artificial intelligence. Her primary research area is in design rationale, with a focus on design rationale for software maintenance. She is a coauthor of the book *Rationale-Based Software Engineering*.

Rob Bracewell is a Senior Research Associate and Assistant Director of the Cambridge Engineering Design Centre and a Mechanical Engineer. He has conducted research in the fields of exploitation of ocean wave energy, mechatronics, and advanced robotics. Dr. Bracewell's recent main interests are in creating software tools to help designers, generating and evaluating innovative design solutions, and unobtrusively capturing the rationale behind their decisions. His software tool, DRed, which was researched and developed in close collaboration with Rolls-Royce plc, has rapidly gained acceptance as the standard tool for capturing and communicating rationales for design and problem diagnosis activities across the company.

https://doi.org/10.1017/S0890060408000206 Published online by Cambridge University Press