Making it Easy for Designers to Provide Design Rationale

Brent Reeves and Frank Shipman

Department of Computer Science and Institute of Cognitive Science
Campus Box 430
University of Colorado, Boulder, Colorado 80309
{brentr,shipman}@cs.colorado.edu

Introduction

Design rationale systems can support several goals, such as making designs less expensive, improving the quality of artifacts, or helping more people to become "good" designers [Jarczyk, Loeffler, Shipman 92]. Most of the literature on design rationale has described what the goals of the systems are and how the system addresses these goals. However, if designers, for whatever reasons, will not use a system then the goals of the system will not be achieved. In this paper, we therefore explore the difficulty of getting designers to add and modify design rationale. In the end, designers will use what serves their design task.

We look at two ways of making it easier for designers to add rationale to the system. First, we suggest the inclusion of the rationale in the design itself. Although this is not possible in domains where design does not occur on the computer, there are many domains where this is possible. Second, we suggest the use of free text without any formalized structure (e.g., IBIS) as the initial form of new rationale.

In the next section we describe the problem we are addressing in more detail. Following this we describe our proposed solutions for this problem and a prototype system that we have built that instantiates our ideas.

Problem Statement

While there are numerous methodologies for representing design rationale [Toulmin 58], [Kunz, Rittel 70], [McCall 79], [Lee 90], [MacLean, Young, Moran 89] and many systems [Conklin, Begeman 88], [Fischer, McCall, Morch 89], [Streitz, Hannemann, Thuring 89], [McCall et al. 90], [Lee 90], [Marshall et al. 91], there have been few recorded successes [Yakemovic, Conklin 90]. This lack of success stories is congruent with the general experiences of computer supported cooperative work systems [Grudin 88], [Markus, Connolly 90].

For design rationale systems to be used the designers must perceive the benefits provided by the system as greater than the costs involved in using the system. While this may seem like a trite
tautology, as system designers we must reflect on this cost/benefit decision that is being made by the designers.

When the designers do not feel they have the time to spend on the luxury of recording design rationale, the only way rationale will get in the system is if its addition becomes part of the design task. One way of making the recording part of the design process is by supporting communication between designers. The system could record the natural discussions about the artifact as they occur throughout the project. Since it is impossible to predict precisely what information will be needed later for design rationale, it seems best to keep all discussions and debates about design tradeoffs as they occur. An important question open to empirical investigation is, "How related are today’s arguments and tomorrow’s misunderstandings?" We think they are highly related and that current discussions about the design will serve later as the best design rationale.

Another way for the design rationale to be integrated with the design process is that a knowledge-based system could infer rationale about proposed changes. This is similar to critiquing systems that point out sub-optimal configurations to the designer [Fischer et al. 91a]. For design rationale, however, a system would need knowledge of atomic actions and likely motivations for those actions, as well as knowledge of components used in the artifact.

The approach we have taken in our system is to let users embed discussions about the design in the design, next to the aspect of the design of interest. Rationale does not immediately need to be in a structured format, but can be added as plain text annotations. Over time, we expect some of these discussions to migrate to a formal representation of the design rationale.

By integrating the design rationale in the design we have made the rationale less of a separate entity and more of an extension of the design. Also, since design is most often done by teams, this mechanism can record communication between designers that might otherwise have occurred through other channels. Once this conversation is in the system, there is the opportunity to organize it later into a structured format.

Text annotations provide a mechanism for more information to remain implicit and so reduces the designer’s cognitive overhead in providing rationale. This approach respects the tremendous difficulty of capturing tacit knowledge and is necessary for the addition of information not understood well enough for a more structured representation, such as the first encounter with a new problem in a design. Currently we are looking at tools to support the evolution of text to more structured representations.

To summarize, the problem we are addressing is how to make it easier for designers to communicate with each other about the artifact being designed, and thus to provide design rationale. Our solution emphasizes the integration of design rationale in the design artifact and the use of informal text as an initial representation for that rationale. These ideas have been implemented in a prototype system, XNETWORK, a design environment for the design of computer networks.
A Prototype System

Our approach to solving the problems described above is based on experience with writing cooperative problem solving systems and builds on the framework described elsewhere [Fischer, McCall, Morch 89; Fischer et al. 92]. As a result, we describe here only briefly the components of this architecture that are relevant to the current discussion. Previous work on design environments has integrated the construction of form with the argumentation about decisions through the combination of a construction component with a hypermedia argumentation component [Fischer, McCall, Morch 89].

XNETWORK, shown in Figure 1, goes beyond this framework of connecting components that deal with a particular type of knowledge in isolation. XNETWORK allows for the integration of partial or whole design artifacts, PHI-style argumentation, and textual annotation within a single view. All the knowledge in the system is available to all the components.

One aspect of this architecture relevant to the discussion of the addition of design rationale is the annotation component that is embedded in the construction space. This allows designers to
attach comments to the design artifacts. By integrating the construction and the discussion about the construction, the most crucial aspects of an argument are placed where they are most influential.

Over time, we expect textual notes in the workspace to migrate to another view of the design, or into an argumentation page. In previous design environments the argumentation component [Fischer et al. 91b] captured design rationale and evolved over time to reflect changes in technology and policy. Generic as well as project-specific information was contained. XNETWORK extends this use of the argumentation component by providing argumentation pages that can be used as in previous design environments, but also can contain pieces of designs and
the textual discussion of the design. Figure 2 shows an argumentation page containing PHI structured argumentation which, in turn, contains a piece of the design seen in Figure 1, along with its discussion.

The integration of these components is possible because of a homogeneous underlying knowledge representation. One cannot simply tack on an annotation or design rationale system and achieve the integration necessary to support the evolution of the design artifact and the discussion about that artifact. The system must be designed from the ground up to support the diverse types of knowledge that must be represented in a system. The artifact must support the sharing of substantive, as well as procedural and annotative activities [Trigg, Suchman, Halasz 86]. From formal knowledge about domain-specific data types, through semi-formal argumentation and to free-text notes: all are part of an evolving artifact.

From a pragmatic viewpoint, everything in the system must be a "first class object" so that any object can reference, and thereby reason about, other objects.

From a design-reuse viewpoint an integrated environment can make a larger variety of analogies, since it has access to different kinds of knowledge. It knows not only that a textual note exists at some absolute coordinates on the viewing plane, but that the note is a critique of, say, a certain combination of design units.

As far as information retrieval is concerned, an integrated system can handle queries of the sort, "show me the discussions about routers," "show me the subnets about which there was disagreement." Domain independent annotation systems cannot provide this kind of functionality, since the objects are not aware of their place in the context of the whole design.

**Summary**

Designers have always preferred working on the design artifact itself: arguing about various options by direct manipulation. When this becomes impossible (architecture, networks) representations are constructed that serve the purpose of grounding the design discussion. Domain-dependent annotation is required to ground the discussion and provide leverage to knowledge-based systems. Tools must be provided to support the merging of these notes into the existing argumentation space.

The focus of our work is the reduction in the effort required of the designer to provide rationale. This problem’s importance comes from difficulties in getting systems supporting design rationale accepted by designers. By providing for the design rationale to be in the design as well as for the use of textual annotations for the initial addition of rationale we attempt to reduce the costs involved for the designer in using our system.

**References**

[Conklin, Begeman 88]
[Fischer et al. 91a]

[Fischer et al. 91b]

[Fischer et al. 92]

[Fischer, McCall, Morch 89]

[Grudin 88]

[Jarczyk, Loeffler, Shipman 92]

[Kunz, Rittel 70]

[Lee 90]

[MacLean, Young, Moran 89]

[Markus, Connolly 90]

[Marshall et al. 91]
[McCall 79]

[McCall et al. 90]

[Streitz, Hannemann, Thuering 89]

[Toulmin 58]

[Trigg, Suchman, Halasz 86]

[Yakemovic, Conklin 90]