Structuring cases in a case-based design aid
Craig Zimring, Sonit Bafna, Ellen Do

Abstract

Architectural designers use a wide variety of reasoning and decision-making aids such as rules, prototypes and specific examples. In this paper we discuss how we have structured cases in a case-based design aid called ARCHIE to allow designers to navigate through the database in a flexible manner yet maintain links between general statements and specific examples. In particular, we focus on the way in which design problems are structured, aimed at helping both novice and experienced designers recognize tacit knowledge.

Introduction

Recent analyses of ill-defined design domains have emphasized the idiosyncratic and recursive nature of design and particularly of early conceptual design (Chandrasekaran, 1990). These characteristics are particularly important for architectural design, where the under-determined nature of architectural design tasks, and centuries of tradition, have created numerous loosely related systems of goals, methods, precedents and ways of evaluating outcomes (Archea, 1987; Schon, 1988).

In this paper, we focus on two aspects of architectural design that have implications for building design aids:

1. Architectural design is multi-constituency in that designers must acknowledge and reconcile the intentions of multiple stakeholders. Learning which intentions are critical and how they can be supported is an important part of conceptual design. (The large number of intentions that an architect or design team must reconcile in a large project makes this process particularly complex. For example, in a recent hospital project the design process explicitly involved over 40 user groups and 150 consultants.)

2. Architectural designers use a complex toolkit derived from their memory and experience as well as from external materials. For example, during design they may use models, explicit and tacit rules, prototypes, and specific examples (Archea, 1987; Schon, 1988). The relationship between these types of reasoning aids are quite complex. A designer may make a set of design decisions because she thinks a building like the Empire State Building will suit her needs, or because she has a general knowledge of neo-Gothic skyscrapers, or because she is familiar with rules of thumb about height-to-width ratios in skyscrapers. She may have learned these rules directly or derived them from specific cases or from prototypes.

1 College of Architecture, Georgia Institute of Technology, Atlanta, Georgia 30332-0155

To appear in Proceedings for Third Congress on Design Computing
American Society of Civil Engineers, June 1996, Anaheim
For the past several years we have been building tools to aid novice and experienced architectural designers. This program of research, called ARCHIE, is the product of collaboration between architects, computer scientists, design researchers and environmental social scientists. ARCHIE is a “case-based design aid” (CBDA).” As the name implies, a CBDA is “case-based”: it provides access to past experience so that human designers can adapt the cases for use in a present situation. It is an “aid” in that it does not automate design; rather, it takes advantage of the ability of human designers to flexibly apply information to new situations, and focuses instead on providing easy, natural access to a wide range of appropriate information.

Written by Eric Domeshek using Common Lisp on the Macintosh, ARCHIE uses case-based reasoning as an underlying theory and technology. Case-based Reasoning (CBR) is a memory-centered cognitive paradigm within Artificial Intelligence based on the idea that people are good at figuring out what to do in new situations largely because they are able to remember what happened in similar past situations and adapt those cases to the current problem (Kolodner, 1993). CBR is being developed into a technology for building systems that assist human users by presenting them with useful information chosen from organized memories of past experiences (Bareiss, Ferguson & Fano, 1991; Kolodner, 1991; Domeshek, 1992).

We have presented a fuller description of ARCHIE elsewhere, and readers interested in learning more about the technical implementation of the program may want to consult those sources (see for example, Domeshek & Kolodner, 1991; Domeshek, 1994; Domeshek, 1992). At present, the ARCHIE case base includes cases about community libraries, courthouses, tall buildings, and handicapped access.

Archie presents cases that are broken into several kinds of information that represent evaluated cases: descriptions of designs, presented as text materials and graphics of various kinds such as building plans, sections, elevations, and other diagrams; problems, which are structured as two intentions with an implementation where at least one outcome is not realized; design responses, which are general strategies a designer might consider for reconciling the intentions; stories, which are brief descriptions of how the problem or solution has played out in a specific building. A sample screen showing a problem response and story is presented in Figure 1.

In Archie, designers can search for case examples of descriptions and evaluations of specific buildings or for more statements of problems, and can move from the specific to the general and vice-versa. Users of the system can retrieve problems, responses or designs by specifying their interests and using index retrieval. They can also take advantage of hypertext links between problems, responses and stories, and between designs and stories. Hence, a designer can search for all stories about courthouse acoustics or can search for the description of the design of a given courthouse and all associated stories.

Writing problems, stories and responses

At the heart of ARCHIE is the link between stories, problems and responses. Take for example, the following story that emerged from interviews with users in a post-occupancy evaluation of a courthouse:

The new courthouse had an airport-style metal detector in the small entrance lobby. The equipment and its associated security station dominated the lobby, so jurors and trial participants reported that they felt under great suspicion in space intended to grandly welcome citizens to a public building. In addition, the noise created by people waiting to pass through the space surfaced in terrazzo, stone, and glass reverberated loudly, creating high levels of stress for staff and visitors.
This story reflects a dilemma facing designers of modern US courthouses: as important public buildings, courthouses should be accessible and welcoming to the public, yet the nature of the proceedings requires that the entry be tightly controlled. Each person who enters must be carefully searched for weapons. The actual implementation in the case above resulted in an overwhelming sense among visitors that control, rather than democratic access was paramount. The story also reflects a collision of two intentions: the use of materials was intended to reflect grandeur and achieve low maintenance, yet did not accommodate the acoustic needs of people who used the space.
The designers and clients did not understand the implications of the design of the security system.

A particular issue that Archie addresses, regarding design theory, is that of defining these “problems”. We have performed numerous post-occupancy evaluations (POEs) of buildings, and have found that several problems in dealing with intentions may limit a project’s overall success: 1) Some stakeholders’ intention may be overlooked. For instance, the architect working with a high level client representative may not recognize the perspective of the maintenance staff. 2) The design team may make a conscious trade-off, such as between functionality and cost, but may not fully recognize the implications of this trade-off. 3) The strategy for resolving conflicting intentions may be appropriate, but the implementation may be flawed.

Design, and particularly conceptual design, is not simply a problem-solving exercise, but rather one of problem-definition. Empirical evaluations provide the opportunity to put goals in sharp focus by linking them to specific design decisions. We believe that structuring cases in ways that make these kinds of design problems explicit we can help novice designers understand the key goals and constraints that experienced designers may incorporate into their designs using their large store of tacit knowledge. Also, however, this may make such tacit knowledge explicit in such a way that it is accessible to experts.

The formulation of “problems,” therefore, is intended to ensure two points:
1) to bring to light, from the expert designer’s repertoire, a particular piece of design knowledge which is not likely to be found in textbooks, or source books on the subject, and
2) to make explicit particular relationships between design factors, which can be known only through occurrences in design and construction practice.

There are two counts on which this is significant:
1) Making implicit design decisions explicit exposes novices to alternatives and their implications and can thus offer crucial design lessons. As we have seen above, the kind of knowledge that Archie seeks to present is of the case based kind. Much of this kind of knowledge is internalized within the expertise of experienced designers and is relatively inaccessible to a novice designer. The question of making this knowledge explicit is a particularly intractable one, since such knowledge is largely unsystematic or at least does not follow the epistemological structuring that design and construction knowledge is generally presented in (classification of aspects of building construction into different systems such as structural system, mechanical system and so on, for instance). It appears to us that breaking down this knowledge into the kind of format we have discussed above—stories, responses and problems and specifically of structuring the problems as instances of conflicting design decisions—should be a particularly productive way of making the latent expertise of designers available to novices and students.
2) Making explicit the design decisions also makes “experts” aware of design decisions that they may be taking habitually or unconsciously, and thus may act as a “check” on their design process. Tacit knowledge is useful and efficient but may lead to automatic and inappropriate decisions if the expert misreads the pattern.

**Future steps**

We have created several hundred stories, problems, responses and designs over several years. However, our initial expectation was that we could create a case base of this size in several months, and that we would have a base of several thousand items by this time. We have found that scaling up is hard to do, partly because it requires a profound knowledge of the domain that comes from both evaluation and experience, and the team has not always had this knowledge. We are experimenting with ways to improve the efficiency of our knowledge engineering by using video interviews and other methods.
Another goal is to link a sketch program to ARCHIE, so that when designers sketch, their graphics can be recognized as indicators of interests and the system can search for related cases. This area of work involves developing a sketching program that allows designers to rapidly express their ideas during conceptual design and use this sketching as a way to access information. For example, the designer should be able to find relevant stories in the case base, copy their diagrams into the sketching program, and enhance, embellish, and develop the diagrams into more rigorous drawings. Meanwhile, ARCHIE could be watching in the background, taking opportunities to display examples of relevant designs. As part of a smoother transition between the stages of conceptual diagram and design development, we would like to program the sketching environment to maintain the spatial relations in a diagram -- such as adjacency, proximity, relative size, and alignment constraints -- as the user moves from bubble diagrams to more precise design representations. We plan to extend the diagram indexing scheme to include more of the case base (Gross, Zimring & Do, 1994).

Acknowledgments

The Archie project and the related ideas have involved many people, and benefited from the contributions of many others not directly involved. Éric Domeshek wrote the ARCHIE and the CBR shell called “Design Muse”. Janet Kolodner was the Principal Investigator of grant from the Defense Advanced Research Projects Agency, monitored by ONR under contract N00014-91-J-4092 and supplied much of the conceptual development of case-based reasoning. EduTech has supported the project under Project D48-566. The idea of organizing POE stories into Archie cases was developed as part of a larger effort to explore application of case based reasoning in design. Michel Conan, of the Centre Scientifique et Technique du Bâtiment, developed much of the framework for case writing. At Georgia Tech, in addition to Ashok Goel and Richard Billington who helped shape the Archie projects at the earliest stages, we have been fortunate to have architecture students Ameen Farooq, Husam Khalil, Ali Malkawi, and Osman Ataman, CS students Anna Zacherl and Vijaya Narayanan in our team for both programming and POE studies. All views expressed are those of the authors.

References


