HUMAN TECHNOLOGY

An Interdisciplinary Journal on Humans in ICT Environments ISSN: 1795-6889

www.humantechnology.jyu.fi Volume 6 (1), May 2010, 38–54

JUXTAPOSING DESIGN REPRESENTATIONS FOR CREATIVITY

Alistair Sutcliffe
Manchester Business School
University of Manchester, UK

Abstract: *This paper argues that the influence of design rationale on creativity is best achieved by concurrent use of scenarios, prototypes and models. A framework of cognitive affordances is introduced to discuss the merits and limitations of each representation. The paper concludes by discussing how different representations might complement each other in creative scenario-based design.*

Keywords: *scenarios, prototypes, cognitive affordances, design representations.*

INTRODUCTION

It is often argued that creative design is best supported by examples of good design, thought probes, and stimulating artifacts (Cross, 2000; Gaver, Beaver, & Benford, 2003). In contrast, the methodical engineering approach to design emphasizes a systematic process, models, and the reuse of design knowledge, criticizing less systematic approaches as "craft" (Dowell & Long, 1998). Design rationale may provide a middle ground between the two approaches as an easy-to-use notation that can stimulate creativity while preserving some of the generality and rigor of models. I will investigate the contributions that different design representations can make to the creative design process from the viewpoint of cognitive reasoning processes. The relative merits of design rationale, scenarios, models, and prototypes are investigated in terms of their roles in the design process and cognitive affordances.

The integrated use of different representations will be illustrated by the scenario-based requirements analysis method (SCRAM; Sutcliffe & Ryan, 1997). SCRAM advocates a combination of design rationale, scenarios, and early prototypes as a means of effective requirements analysis and design exploration. More recently we have used a merge of SCRAM and scenario-based design (Carroll, 2002) with a similar combination of design representations in eScience health informatics domains (Sutcliffe et al., 2007). The following section of this paper describes the properties of different design representations. Next, I discuss how the representations can support creative reasoning, with the following section elaborating the theme by investigating cognitive affordances. Then I review how representations can be integrated into the design requirements discovery process. Integration is illustrated with the SCRAM method, followed by a brief review of other approaches to creative design support. The paper

URN:NBN:fi:jyu-20105241906

concludes by reviewing the potential for juxtaposing different design representations for creative design, as well as requirements specification of systems.

DESIGN REPRESENTATIONS

This section reviews the role of the more common design representations in creative design from a human–computer interaction (HCI) perspective and from the more analytic view of software engineering.

Scenarios

One of the key distinctions between scenarios and any model is that the former are grounded examples of specific experience, whereas models are more abstract representations of phenomena in the real world. Unfortunately, the term *scenario* has been abused in the literature and a large number of definitions exist (see Rolland et al., 1998). Indeed, much of the scenario literature, especially in the software engineering tradition (Kaindl, 1995), is in fact describing event–sequence traces through state transition models. In object-oriented design it becomes difficult to distinguish between use cases, alternative paths through use cases, and scenarios, which are just another path through a use case (Cockburn, 2001; Graham, 1996; Jacobson, Christerson, Jonsson, & Overgaard, 1992).

Scenarios have several roles in design; according to Carroll, one of these is a "cognitive prosthesis," or an example to stimulate the designer's imagination. Scenarios and other techniques, such as claims, are lightweight instruments that guide thought and support reasoning in the design process (Carroll, 2002). Carroll has articulated several different roles for scenarios in the design process, including envisionment for design exploration, requirements elicitation, and validation (Carroll, 1995). Usage scenarios illustrate problems for analysis and initiating or visioning scenarios stimulate design of a new artifact, while projected use scenarios describe future use of an artifact that has been designed (Sutcliffe & Carroll, 1998). Scenarios can promote creative reasoning by stimulating examples and vivid illustration of real-life problems.

One problem with scenarios is that extreme examples might bias reasoning towards exceptional and rare events, or towards the viewpoint of an unrepresentative stakeholder. These biases are an acknowledged weakness of scenarios; however, some proposed scenarios are deliberately exceptional to provoke constructive thought (Djajadiningrat, Gaver, & Frens, 2000). Although scenarios are useful as cognitive probes for design, this is not their only role.

Scenarios arguably are the starting point for all modeling and design, and contribute to several parts of the design process. For instance, Potts (1999) has advocated scenarios to validate or check the acceptability of designs. The process of generalization inevitably loses detail, and the analyst has to make judgments about when unusual or exceptional behaviors are omitted, or explicitly incorporate them in task models as branches in action sequences. Hence one criticism that can be leveled at scenarios is that gathering detail comes at the price of effort in capturing and analyzing a "necessary and sufficient" set of scenarios.

Models

A prime role of models, either in the HCI tradition of task modeling or in software engineering (e.g., use cases, class diagrams, activity sequence diagrams, UML, etc.), has been to specify the system and represent the problem space to support design reasoning.

One criticism of models is that they do not capture the richness of interaction that occurs in the real world, compared with scenario narratives that concentrate on contextual description (e.g., Kuutti, 1995; Kyng, 1995). For instance, software engineering and task models may be criticized for not representing the relationships between agents, activity, and organizational structures, although these concepts are described in sociotechnical system design frameworks such as ORDIT (Eason, Harker, & Olphert, 1996). Meanwhile, a more comprehensive modeling language can be found in the i* requirements engineering method that analyzes the dependencies between agents, tasks, goals, and resources (Mylopoulos, Chung, & Yu, 1999; Yu, 1993). Models can expose design dilemmas and inconsistencies and thereby support the generation of creative solutions; however, how well models expose problems depends on the clarity of their notations and the reasoning mechanisms associated with the model. Models show an abstract view of problems so they might be accused of having a narrow scope of phenomena and omit detail, whereas scenarios might be able to represent phenomena in more detail, but they do so in an ad hoc manner and leave the responsibility of generalization to the analyst. Of course, models can be used with scenarios, and this theme is elaborated later in this paper.

Design Rationale

The essence of design rationale (DR) is to represent argumentation and knowledge within the design process. Hence DR can be viewed as models that are specialized to represent the problem space for decision making, including evidence for evaluating alternative designs. Various forms of DR have appeared since their genesis in Toulmin's (1958) argumentation semantics, notably issue-based information systems and the diagrammatic form gIBIS (Conklin & Begeman, 1988), which represents issues (design problems to be solved), alternatives (possible solutions), and evidence that supports or detracts from each alternative. The most influential HCI variant of DR recapitulates the semantics as questions (design problem), options, and criteria (QOC; MacLean & McKerlie, 1995; MacLean, Young, Bellotti, & Moran, 1991). DR can also be used to express generalizable knowledge accumulated during iterative design. Psychological DR, or claims (Carroll & Rosson, 1992), uses a simpler semantic representation of the claim (problem statement), a solution (expressed as an artifact/design pattern), and arguments divided into upsides and downsides. Claims may be used to support reasoning during the design process (Carroll, 2002) or present reusable knowledge by recording the results of evaluation, including the problem that motivated a general design principle—called a claim—with trade-offs expressed as upsides and downsides (Carroll, 2000; Sutcliffe & Carroll, 1999). When DR is used to support the design process, the trade-off concerns for a claim about DR representations might invite comparison of the cost of representing the design space versus the advantage gained in more effective reasoning. The juxtaposition of alternatives is a key affordance for creative reasoning. For collaborative decisions, DR diagrams can function as a shared representation to focus discussion, although the costs may well outweigh

the benefits. The uptake of DR in industry has been slow. When representing reusable knowledge, the benefits of DR may be potentially larger, but reuse depends on an effective knowledge management and retrieval system.

Prototypes

This category includes a variety of design representations, ranging from paper (or computer-based) storyboards to mock-ups/concept demonstrators with limited scripted functionality and prototypes with a partial software implementation. Prototypes stimulate creative design because they engage the user (designer) with the material of the product, be that software or hardware. Experimentation becomes part of the implementation unless a rigorous specification in detail of the implementation process is adopted, as practiced in software engineering. The prototype artifacts all result from the creative design process and, unlike models and DR, show the user concrete aspects of a design. Prototypes, mock-ups, and storyboards are probably the most common ways of representing the problem space for creative design exploration. This applies not only in software-related products but also in many other areas of creative design. The variation between the techniques lies in the media used (paper, video, computer media, interactive software), the cost of production, and the fidelity and extent of the representation of the intended design. While very early creative brainstorming may be used to map out a space of ideas and concepts, once these have been prioritized, design realization becomes necessary to progress the user–designer dialogue. The power of the prototype lies in anchoring the focus of discussion in a concrete example, and stimulating user reaction to specific features.

REPRESENTATIONS AND REASONING PATHOLOGIES

In this section, the merits of different representations are reviewed in light of how they can stimulate and support creative design. Scenarios use language and concepts that are readily accessible to users and domain experts, whereas tasks and other conceptual models are expressed in a specialized language that users have to learn. Because scenarios invoke specific memory schema associated with experience or similar stories, they help to recruit specific knowledge (Carroll, 2000; Sutcliffe, 2002). This tunes our critical faculties, since detail tends to provide more subject matter to detect inconsistencies and errors when we reason about models and specifications.

In contrast, models are harder to comprehend because they represent abstract generalizations. While people naturally form categorial abstractions of physical things (Rosch, Mervis, Gray, Johnson, & Boyes-Braem, 1976), we are less efficient at forming categories of abstract concepts and functions (Hampton, 1988). Unfortunately, formation of conceptual-functional categories is a necessary part of the generalization process, so users can find reasoning with simple conceptual models, such as data flow diagrams, difficult (Sutcliffe & Maiden, 1992). Once learned, models become memory schema that represent abstract concepts removed from everyday experience, so their effectiveness depends on how well connected they are to more specialized memory schema representing scenario-based knowledge. The importance of the connection becomes clear when we try to validate models. Without any connection to specialized knowledge, I can accept the validity of the general

concept simply because it has a wide scope of meaning. For example, I might accept the proposition that <all birds can fly> as a true type definition of the class <birds> in the absence of more specific knowledge of penguins, kiwis, rheas, ostriches, and dodos. Models therefore need to be integrated examples and scenarios and, furthermore, cannot exist profitably without them; indeed, human categorial memory is probably an integration of abstract models and specific examples (Lakoff & Johnson, 1999).

While scenarios might be effective in grounding reasoning, their downsides lie in reasoning biases and partial mental model formation. Confirmation bias is a well-known weakness of human reasoning (Johnson-Laird & Wason, 1983). We tend to seek only positive evidence to support hypotheses, so scenarios can be dangerous in supplying us with minimal evidence to confirm our beliefs. While problem statement scenarios and anti-use cases (I. Alexander, 2002) can counteract confirmation bias, we need to be wary of this downside. Another potential pathology is encysting, more usually described by the saying "can't see the wood for the trees." Since scenarios are detailed, they can bias people away from the big picture of important design issues and towards obsession with unnecessary detail. Models exist to counteract this pathology. Partial mental model formation is another weakness when we test hypotheses without sufficient reasoning (Simon, 1973). Scenarios can encourage this pathology by reassuring us that we have covered all aspects of the problem with a small number of scenarios. This exposes the Achilles heel of scenario-based reasoning: It is difficult, if not nearly impossible, to be confident that a necessary and sufficient set of scenarios has been gathered to escape from the partial mental model problem.

Prototypes and other concrete design realizations share many of the same pathologies with scenarios, such as encysting and confirmation bias, since users might be prone to accepting a design to please the designer. This may be critical when the power relationships give designers a de facto authority over users, which they should strive to avoid. Groups of users may also be prone to suppress criticism of a design and agree with the consensus, following a group-think bias. However, prototypes do afford concrete representations and detail that users can react to, as well as anchoring discussion to specific issues/features, which can facilitate users' participation in the creative design process.

Many of the same criticisms can be leveled at DR as a genre of models. Although DR represents the decision space with specific issues in some detail, arguments may make little sense without the background knowledge contained in other representations. Also, DR may bias problem exploration by presenting a ready-made set of alternatives. Furthermore, unless the author of the DR diagram is careful, the diagram can embed biases from the author's viewpoint in the relationships between the alternatives and supporting/detracting evidence (Karsenty, 1996; Sutcliffe, 2002; Sutcliffe & Ryan, 1997).

AFFORDANCES AND REPRESENTATIONS

While analysis of general properties of representations can provide some insight into their potential contribution towards supporting design, a more detailed view is necessary to unpack the nature of cognitive affordances. The term *affordance* was borrowed by Norman (1999) from Gibson's (1986) concept of physical features that suggested or afforded intuitive understanding, for instance, cliffs suggest the danger of falling. As Norman realized, when the

concept of affordances is applied to design features, the meaning of the term becomes more complex, since it has to account for the general suggestibility of the external form towards some purposeful use and the cognitive internalization of the external form into an individual's plan of action; for example, a slider control on a user interface suggests movement of the control itself that then changes another component, such as panning a display.

A useful distinction, therefore, is to examine the external appearance of a representation (or design) and its integration into action plans after people have interpreted its meaning. To illustrate this line of inquiry, I will compare three exemplars of design rationale: gIBIS, QOC, and claims. The first two have a similar external form but differ in their semantics. Claims, in contrast, have a different (text-based) external form and semantics.

gIBIS diagrams, as illustrated in Figure 1, have a simple tree/network structure that can be traced from the root node representing the issue, to the branches (two or more design/solution alternatives), and then to leaf nodes representing supporting arguments. The graphical form intuitively suggests composition and relationships, as do most hierarchy diagrams. The semantics of the diagram nodes are easily explained so the representation can be used to trace relationships from the issues through each alternative solution to the supporting (or detracting) arguments. Thus the representation "affords" comparison of alternative solutions by pathway tracing. Furthermore, the external representation reduces working memory loading since different pathways can be reviewed at will.

QOC (see Figure 2) has a similar graphical notation, so the diagram also affords pathway tracing of questions or different design options. However, criteria and arguments have subtly different semantics. Criteria are more terse and represent concepts by which trade-off decisions can be made, rather than arguments that record the results of reasoning about different alternatives. Criteria therefore invite more in-depth reasoning about the options and their relationship to one or more criteria; hence, QOC may stimulate more creative thought by provoking reasoning. This conjecture would require experimental study to assess the quality of reasoning invoked by each representation; nevertheless, the comparison illustrates how graphical forms and the semantics attached to diagrams might influence reasoning.

Claims, also termed psychological design rationale by Carroll (Carroll & Rosson, 1992), do not share the diagram representation; instead, formatted text is used to illustrate the structure

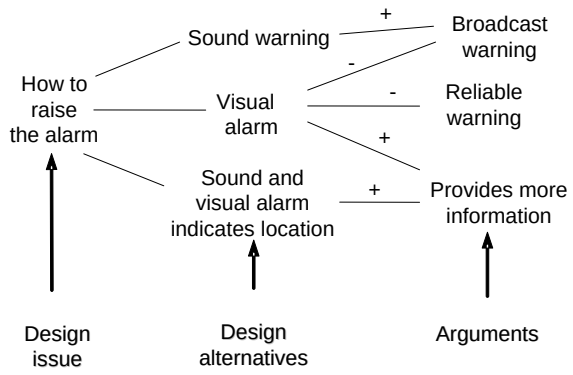

Figure 1. In the gIBIS design rationale, the + or – signs denote arguments that either support or hinder a particular alternative.

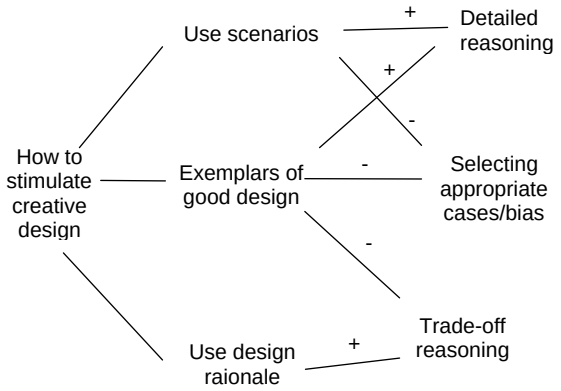

Figure 2. The QOC (questions, options, criteria) form of design rationale, applied to trade-offs between representations to support creative design.

of a claim (see Figure 3). The basic components of claims are (a) the claim (essentially a design principle); (b) upside and downside trade-offs that may arise from application of the claim; (c) a scenario of use; and (d) an artifact illustrating a design that embeds the claim. The juxtaposing of alternatives has been moved from alternative designs to the assessment criteria or arguments. Claims present essentially only one design alternative and then positive and negative arguments about its merits. While DR provides more structured arguments, claims use the combination of a design solution (a generalized design principle) with examples of use illustrated in scenarios and artifacts. Claims may therefore stimulate creative thought by the challenge posed from the general assertion about a design treatment (the claim), concrete illustrations of its interpretation and use, and the results of previous design experience recorded in the upsides and downsides. Design patterns (Borchers, 2001) follow a similar format with forces, scenarios and illustrations of exemplar design for the pattern.

So how do other representations compare with the affordances of DR? Models share diagrammatic notation with DR but have many more morphologies and semantics, ranging from the simple (e.g., use case diagrams) to the very complex (e.g., i* requirements modeling language; Yu, 1993). It is notable that most semiformal modeling notations rely on a restricted

Claim ID: Colour-coded Telegraphic Display
Author: Singley, M.K.; Carroll, J.M.
Artifact: MoleHill tutor - Goalposter tool
Description: A colour-coded telegraphic display of goals
Upside: Provides persistent feedback on the correctness of actions, as well as access to further information
Downside: Learners must learn the display's feature-language and controls
Scenario: The presentation of individual goals in the window is telegraphic, several words at most. However, the learner can expand any of the telegraphic goals (through a menu selection) to display a fuller explanation of why the goal is worthwhile pursuing or not. Thus the system provides both shorthand feedback on correctness and access to further help.

Figure 3. Claim showing components in structured text format.

number of graphical formats that afford intuitive interpretations, namely hierarchies (task models, class diagrams), networks (data flow diagrams, activity sequence diagrams), and timelines (Gantt charts, interaction diagrams). Complexity arises when diagram notations become overloaded with symbols to represent a large variety of relationships and objects, as illustrated in Figure 4.

Scenarios, in contrast, use natural media (text and image) for representing concrete examples of experience. Prototypes also represent concrete examples of designs as physical artifacts, with storyboards and mock-ups providing representations of the physical form early in the design process.

Cognitive affordances therefore emerge from intuitive understanding or representations in a variety of media, coupled with reasoning about the content of those representations. DR and models provide abstract representations of knowledge and design trade-offs to support creative reasoning, while scenarios and prototypes give grounded examples from which to abstract more general principles. However, creative thought should generate innovative designs, but these need to be based on general principles; otherwise, design is limited to a craft-style incremental improvement of specific examples (Dowell & Long, 1998). I argue that a combination of representations is the most productive way to stimulate creative design, a challenge addressed in the next section.

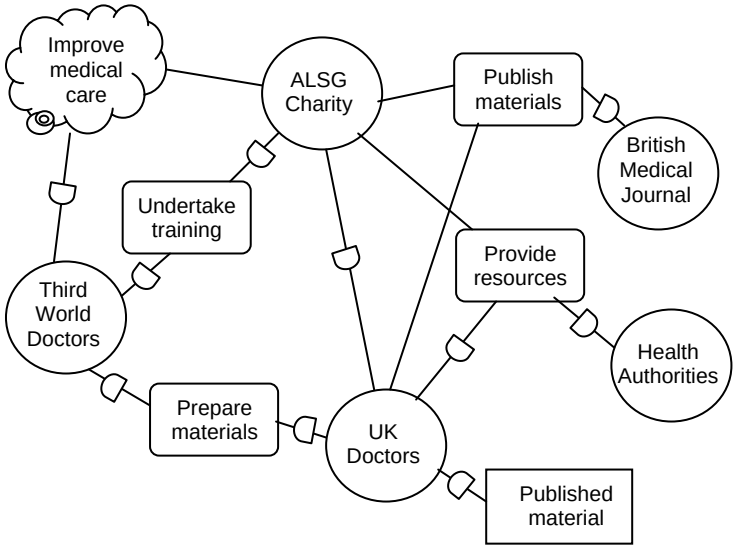

Figure 4. An i* strategic dependency model illustrating complex modeling.
The circles denote agents, boxes are resources, rounded boxes are goals, and clouds are quality goals.
The D symbols stand for “depends on” relationships.

INTEGRATING REPRESENTATIONS

One productive juxtaposition of scenarios and models is to use scenarios as test data to validate design models. This approach has been actively researched in the inquiry cycle (Potts, Takahashi, & Anton, 1994), which recommended using scenarios as specific contexts to test the utility and acceptability of prototype system output. By questioning the relevance

of system output for a set of stakeholders and their tasks described in a scenario, the analyst can discover obstacles to achieving system requirements. Input events can be derived from scenarios to test validation routines and other functional requirements. This process stimulates reasoning by integrating two physical representations, operating a prototype to produce output with scenarios of potential use. HCI uses scenarios in a similar manner in usability evaluation, although the role of scenarios is not articulated so clearly. Nevertheless, task or test scripts in evaluation methods (Monk & Wright, 1993) are scenarios.

Claims have evolved through several iterations of more or less integrated representations. In their original form, claims united scenarios, illustrating a problem with DR presenting the upsides and downsides of usability arguments as trade-offs for applying a design principle with a concrete example of an implementation. Claims are situated in a context by a scenario of use and the artifact that helps designers understand how to apply usability arguments. Since claims have a domain-specific anchor in the artifact context, insight into more general design implications and trade-offs may be gained if they can be integrated with models.

By associating claims in this manner, the designer can have the best of both worlds. Claims with their associated artifacts and scenarios provide grounded examples of design advice while models represent a more general context within which to consider the implications of the design decision. This view of claims is similar to the schema of patterns that recommend that design advice is presented in the context of a motivating problem, and with an example of its application (Borchers, 2001). Although patterns do have a clause that indicates the range of problems the design advice can be applied to, this scoping is ad hoc. Advocates of patterns proposed relationships between individual patterns constructed into a hypertext-like pattern network or language (C. Alexander, Ishikawa, & Silverstein, 1977) to set the context. Unfortunately, pattern languages tend to be incomplete. Claims have been integrated with models that may be specific to the application, or generalized models of tasks to stimulate reuse of knowledge (Sutcliffe & Carroll, 1999); see Figure 5.

The scope of the claim is defined by models that may be related to particular applications, for example, task models, class diagrams for a telephone fault-finding application, or more generic models capturing a range of applications (e.g., generic models of diagnostic tasks, including fault finding). One of the problems with integrating claims with

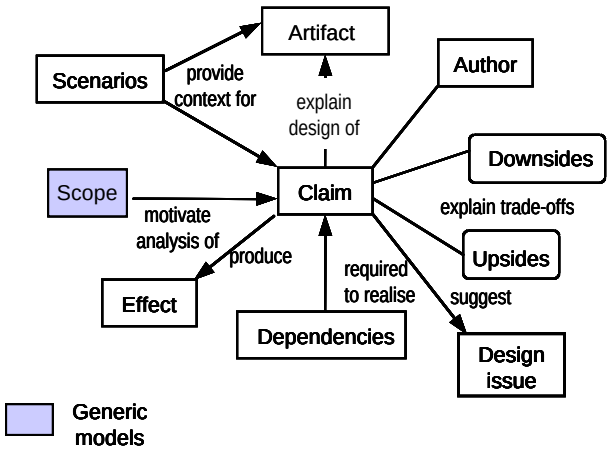

Figure 5. An extended claims schema, associating claims with reusable generic models and supporting arguments.

models and other arguments lies in the complexity of the number of representations, which in turn necessitates further guidance about how the representations may be combined in the design process. More elaborate representations therefore run into the criticism leveled at the engineering approach: The complexity of models and process advice militates against the creative freedom necessary in design. In the following section, I introduce the SCRAM method, which attempts to tread a middle path between creative use of multiple representations and a systematic approach.

The SCRAM Method

The SCRAM method (Sutcliffe & Ryan, 1997) for analyzing the requirements for interactive systems provides one way forward for integrating representations. The approach is based on integrating three representations:

Prototypes or concept demonstrators provide a designed artifact that users can react to;

Scenarios, in which the designed artifact is situated in a context of use, thereby helping users relate the design to their work/task context;

Design rationale, where the designer's reasoning is deliberately exposed to the user to encourage user participation in the decision process.

The representations are combined with a method to provide process guidance, composed of advice on setting up sessions, and more detailed guidance on fact acquisition and requirements validation. The method consists of the following phases:

1. Initial requirements capture and domain familiarization. This is conducted with conventional interviewing and fact-finding techniques to gain sufficient information to develop a first-concept demonstrator.
2. Specification and development of the concept demonstrator. I define a concept demonstrator as a very early prototype with limited functionality and interactivity, so it can only be run as a script to illustrate a typical task undertaken by the user. *Scripts* illustrate a scenario of typical user actions with effects mimicked by the designer. Concept demonstrators differ from prototypes in that no real functionality is implemented and the user cannot easily interact with the demonstrator since many functions are not implemented.
3. Requirements analysis-design exploration session. The users involved in the initial requirements capture interview are invited to critique the concept demonstrator and interview the designer. The session is recorded for subsequent analysis.
4. Session analysis. Data collected are analyzed and conclusions are reported back to the users. This frequently leads to a further iteration of revising the concept demonstrator and another analysis session.

The end point of the method delivers the concept demonstrator, a set of analyzed DR diagrams expressing users' preferences for different design options, and specifications as text, diagrams, or more formal notations, depending on the designer's choice. In addition, video of the analysis sessions is available for requirements traceability analysis.

The walkthrough method employs scenario scripts that describe an imaginary work situation for the user and a typical key task. The session is started with an introduction and verbal

summary of the situation described in the scenario narrative, for example, “Imagine you are in your office and a production order arrives ....” One developer operates the concept demonstrator while the explainer-rapporteur asks questions at key points in the demonstration script.

At key points in the sequence, a designed response to a requirement is illustrated. This is best explained by reference to the example used for the validation, which is covered later. Figure 6 illustrates a screen dump from a shipboard emergency management system. The user’s requirement is for timely and appropriate information to support decision making. The operational steps accompanying Figure 6 are

User: identify the hazard location

System: shows location of fire

User: sound alarm

User: find location of fire-fighting crews

System: displays crew information and location on the diagram

User: decide appropriate instructions to give to crew

System: displays a checklist of actions.

The key point in the task is how to instruct the emergency team on where to go and how to deal with the hazard, in this case a fire. The concept demonstrator illustrates one design option. Alternative solutions expressed in a DR format are illustrated in Figure 7. The user’s attention is drawn to the design options, in this case providing complete information for decision support. The first option displaying comprehensive information is illustrated with the demonstrator; this is followed by option 2, provision of more restricted but relevant information for the task, by identifying the team nearest the fire; and then the final option to give the emergency team autonomy and broadcast the location of the fire. The users are asked to rate each option and consider the trade-off criteria. The diagram also functions as a recording medium, since ranking of options, additional ideas, and notes can be scribbled on top of the diagram. Indeed, in many cases, discussion may promote redrawing the diagram.

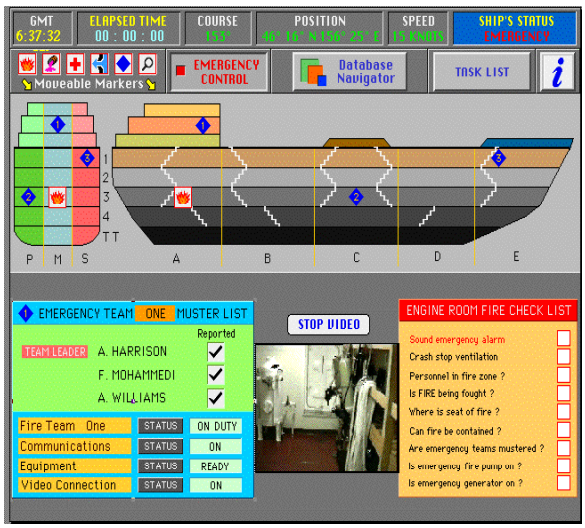

Figure 6. Concept demonstrator showing the “show emergency teams and hazard location” design option for the Muster emergency teams task.

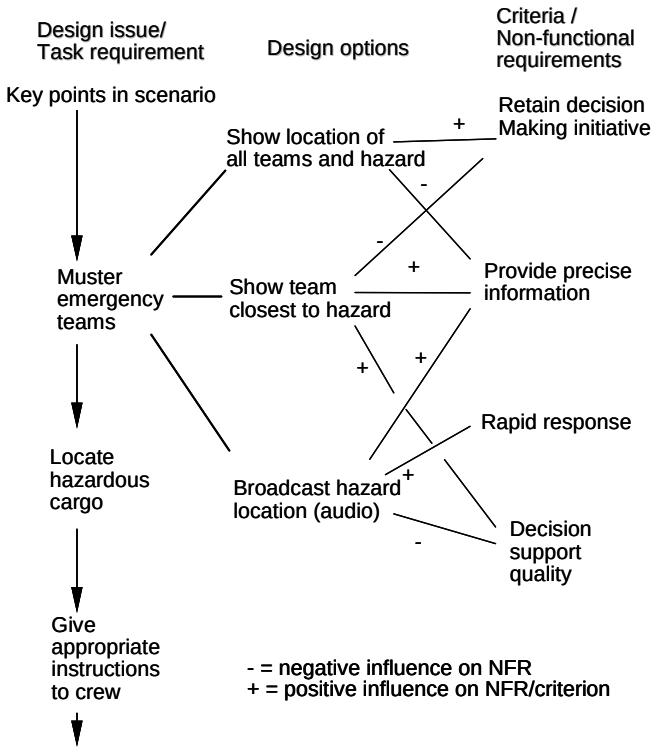

Figure 7. A design rationale diagram used as a key point in the concept demonstrator script.

The DR diagram is used as a shared artifact to promote discussion, and gesture is used where possible to illustrate differences between the options by pointing to the screen. One obvious problem is bias towards the option implemented in the demonstrator. This can be counteracted by using storyboard sketches of the other options and by more vigorous critiquing by the developers of the implemented version. In particular, use of the criteria is a powerful way of promoting critical thought. The motivation for using design rationale is to explore the possible solution space with the user. Rationale diagrams enable this to be done cost-effectively, since only one version of the demonstrator is produced. However, should additional resources be available, alternative versions of the artifact can be implemented and both versions illustrated at the key point.

Evaluation of SCRAM demonstrated that more detailed requirements and design feedback were captured using the method than with conventional requirements analysis techniques without multiple representations (Sutcliffe & Ryan, 1997). The method also stimulated creative reasoning about new design solutions through the process of critiquing the concept demonstrator and alternative solutions that were presented.

Creative Combinations

Although integrated representations show considerable promise in stimulating design reasoning, the creativity and cognition literature suggests that challenging content, shifting viewpoints, metaphor, and analogies also play important roles in creative reasoning (Cross, 2000, 2002; Karsenty, 1996). Selecting concrete examples and setting up contrasts may therefore be an important extension to DR and integration of representations (Buxton, 2007).

DR affords configuration of challenges and contrasts by its comparison of trade-offs between alternatives; however, to challenge thought requires a considerable shift in the

traditional view of DR as a discussion forum for trade-offs to active design of rationale. This could be explored by creating unusual solutions or criteria that challenge the conventional assumptions about a design. Contrasts might be borrowed from value-based design (Friedman, 1997), sketching stories (Buxton, 2007), or extreme characters (Gaver et al., 2003) to produce challenges. Another example is the emergence of antipatterns (I. Alexander, 2002) as a means of stimulating counterintuitive reasoning. Combinations of representation and content could provide a design space for creative exploration, using probes based on analogies and metaphors that alter viewpoints on a design problem with cultural probes and unusual examples of design (Gaver et al., 2003) to stimulate thought.

Choice of how many and which representations to combine is a complex question for further research. I expect the answer may be "horses for courses," meaning, for wide-ranging creative design with green-field applications, sketches, storyboards, and scenarios (Buxton, 2007; Moggridge, 2006) may be the best choice, although as Nigel Cross (2002) noted, expert designers still reuse basic knowledge in the form of "first principles." Such knowledge could be passed on as DR or claims. In more constrained contexts, creative reasoning may be better supported with more detailed representations, models, and specifications (Kaindl, 1995; Paterno, 1999). Although multiple representations were effective in SCRAM (see previous section), there were limits; I did not integrate models with the other representations, since the management of artifacts, DR, and scenarios within one session was already complex. A future view on the representation creativity problem may be to evaluate how representations contribute to the "common ground" (Clark, 1996) between the parties in design conversations. Representations need to promote shared understanding between the parties according to their prior knowledge and the design problem in hand.

CONCLUSIONS

The argument advocated in this paper is that the constructive tension between different types of representation is productive since they have different affordances for abstract reasoning and detailed critiquing. Unfortunately, there are obstacles in the way of using multiple representations, even though many advocate them in HCI and software engineering (Mylopoulos et al., 1999; Paterno, 1999; Sutcliffe, 2000). In reality, it is difficult to get practitioners to accept complex representations, and even simple ones get misused and customized to individuals' needs. Take MacLean et al.'s (1991) QOC variant of DR as an example. This is a simple representation of a design question, alternative solutions, and evaluation criteria for the solutions. However, QOC has been difficult to introduce into new communities of practice (MacLean & McKerlie, 1995), and similar problems have been encountered with the gIBIS (Conklin & Begeman, 1988) version of DR (Buckingham Shum, 1996; Sutcliffe & Ryan, 1997). Carroll, in his more recent work, has simplified claims (Carroll, 2000; Rosson & Carroll, 2001), abandoning complex formatting. Claims are presented as simple design principles, in association with a motivating scenario and occasionally an artifact. In terms of process, Carroll advocates a more creative view of design, with scenarios playing roles of "thought prostheses" and challenges for design.

So is there a synthesis for the model-analytic and creative-exploration approaches to design? A partial answer is acknowledging the "horses for courses" argument. A differentiation

between formal model-analytic and creative-exploratory design approaches will always be necessary for applications that range between safety critical, on the one hand, and those oriented toward entertainment, education, and general commerce on the other. So different combinations of design presentations have contributions to make in different phases of design and design contexts. Scenarios and prototypes can stimulate thought and provoke argument on detail, whereas models give the wider, more abstract context for design reasoning. DR can provide the link between the two, although its effectiveness in supporting decision making or just documenting the result is an open question. Finally, design representations enable knowledge to be reused effectively in a generalized form as models, claims, principles, and guidelines.

Combinations of representations using prototypes, scenarios, and claims are advocated in scenario-based design (Carroll, 2002), and these have been successfully applied in eScience applications (Thew et al., 2008; Thew et al., 2009), as well as in Carroll's development of collaboration tools for eCommunity applications (Carroll & Rosson, 1996). SCRAM integrated DR with scenarios and early prototypes and this proved to be an effective combination for critiquing designs and stimulating further design ideas (Sutcliffe & Ryan, 1997). Prototypes with scenarios and formatted question lists have been successfully applied in requirements analysis (Sutcliffe, Gault, & Maiden, 2005), which, while not using DR explicitly, did present issue lists and alternatives to provoke design reasoning. The scenario presenter tool evolved from earlier research on automated support for design reasoning with tools that produced question prompts linked to specific locations in a scenario or use case, thus giving more active support for design reasoning (Sutcliffe, Maiden, Minocha, & Manuel, 1998). A combination of scenarios, prototypes, and lightweight design representations appears to have evolved in several research strands suggesting that combining representations has some utility.

The more general question that requires considerable future research is how juxtaposing contrasts in the content can augment the combination of different representations. Challenges to reasoning from usual content are known to induce creative reasoning; however, how such content can be produced for specific situations is not clear. Seeding the design environment with stimulating content (Fischer, 1996) assumes considerable insight into future problems. Although premade solutions might inhibit creative reasoning if designers just take the easy option of reusing previous solutions, cognitive probes in the forms or personae, and extreme characters (Djajadiningrat et al., 2000), can stimulate thought. In future work I will use common ground (Clark, 1996) as a theoretical framework for exploring cognitive probes and DR, as well as a combination of representations to support creative design reasoning.

REFERENCES

Alexander, C., Ishikawa, S., & Silverstein, M. (1977). *A pattern language*. Oxford, UK: Oxford University Press.

Alexander, I. (2002). Initial industrial experience of misuse cases in trade-off analysis. In *Proceedings of RE-02 IEEE Joint International Conference on Requirements Engineering* (pp. 61–70). Los Alamitos CA: IEEE Computer Society Press.

Borchers, J. (2001). *A pattern approach to interaction design*. Chichester, UK: Wiley.

Buckingham Shum, S. (1996). Analyzing the usability of a design rationale notation. In T. P. Moran & J. M. Carroll (Eds.), *Design rationale: Concepts, techniques and use* (pp. 185–215). Hillsdale NJ, USA: Lawrence Erlbaum Associates.

Buxton, W. (2007). *Sketching user experiences: Getting the design right and the right design.* Amsterdam: Elsevier.

Carroll, J. M. (Ed.). (1995). *Scenario-based design: Envisioning work and technology in system development*. New York: Wiley.

Carroll, J. M. (2000). *Making use: Scenario-based design of human-computer interactions*. Cambridge, MA, USA: MIT Press.

Carroll, J. M. (2002). Making use is more than a matter of task analysis. *Interacting with Computers, 14, 619*–627.

Carroll, J. M., & Rosson, M. B. (1992). Getting around the task-artifact framework: How to make claims and design by scenario. *ACM Transactions on Information Systems, 10,* 181–212.

Carroll, J. M., & Rosson, M. B. (1996). Developing the Blacksburg Electronic Village. *Communications of the ACM, 39,* 69–74.

Clark, H. H. (1996). *Understanding language*. Cambridge, UK: Cambridge University Press.

Cockburn, A. (2001). *Writing effective use cases*. Boston: Addison-Wesley.

Conklin, J., & Begeman, M. L. (1988). gIBIS: A hypertext tool for exploratory policy discussion. *ACM Transactions on Office Information Systems, 64,* 303–331.

Cross, N. (2000). *Engineering design methods: Strategies for product design*. Chichester, UK: Wiley.

Cross, N. (2002). Creative cognition in design: Processes of exceptional designers. In *Proceedings of the 4th Conference on Creativity and Cognition* (pp.14–19). New York: ACM Press.

Djajadiningrat, J. P., Gaver, W. W., & Frens, J. W. (2000). Interaction relabelling and extreme characters: Methods for exploring aesthetic interactions. In D. Boyarski & W. A. Kellogg (Eds.), *Conference Proceedings: DIS2000 Designing Interactive Systems: Processes, Practices Methods and Techniques* (pp. 66–71). New York: ACM Press.

Dowell, J., & Long, J. L. (1998). A conception of the cognitive engineering design problem. *Ergonomics, 41,* 126–139.

Eason, K. D., Harker, S. D. P., & Olphert, C. W. (1996). Representing socio-technical systems options in the development of new forms of work organisation. *European Journal of Work and Organisational Psychology, 5,* 399–420.

Fischer, G. (1996). Seeding, evolutionary growth and reseeding: Constructing, capturing and evolving knowledge in domain-oriented design environments. In A. G. Sutcliffe, D. Benyon, & F. Van Assche, (Eds.), *Domain knowledge for interactive system design; Proceedings: TC8/WG8.2 Conference on Domain Knowledge in Interactive System Design* (pp. 1–16). London: Chapman & Hall.

Friedman, B. (Ed.). (1997). *Human values and the design of computer technology*. Cambridge, UK: Cambridge University Press.

Gaver, W. W., Beaver, J., & Benford, S. (2003). Ambiguity as a resource for design. In V. Bellotti, T. Erickson, G. Cockton, & P. Korhonen (Eds.), *CHI 2003 Conference Proceedings: Conference on Human Factors in Computing Systems* (pp. 233–240). New York: ACM Press.

Gibson, J. J. (1986). *The ecological approach to visual perception*. Hillsdale, NJ, USA: Lawrence Erlbaum Associates.

Graham, L. (1996). Task scripts, use cases and scenarios in object oriented analysis. *Object-Oriented Systems, 3,* 123–142.

Hampton, J. A. (1988). Disjunction in natural categories. *Memory and Cognition, 16,* 579–591.

Jacobson, I., Christerson, M., Jonsson, P., & Overgaard, G. (1992). *Object-oriented software engineering: A use-case driven approach*. Reading, MA, USA: Addison Wesley.

Johnson-Laird, P. N., & Wason, P. C. (1983). *Thinking: Readings in cognitive science*. Cambridge, UK: Cambridge University Press.

Kaindl, H. (1995). An integration of scenarios with their purposes in task modelling. In G. M. Olson & S. Schuon (Eds.), *Designing Interactive Systems: DIS 95 Conference* (pp. 227–235). New York: ACM Press.

Karsenty, L. (1996). An empirical evaluation of design rationale documents. In *Proceedings of CHI'96 Conference: Human Factors in Computing Systems* (pp. 150–156). New York: ACM Press.

Kuutti, K. (1995). Workprocess: Scenarios as a preliminary vocabulary. In J. M. Carroll (Ed.), *Scenario-based design* (pp. 19–36). New York: Wiley.

Kyng, M. (1995). Creating contexts for design. In J. M. Carroll (Ed.), *Scenario-based design* (pp. 85–108). New York: Wiley.

Lakoff, G., & Johnson, M. (1999). *Philosophy in the flesh: The embodied mind and its challenge to western thought*. New York: Basic Books.

MacLean, A., & McKerlie, D. (1995). *Design space analysis and user-representations* (Technical Report EPC-1995-102). Cambridge, UK: Xerox Research Centre Europe.

MacLean, A., Young, R. M., Bellotti, V., & Moran, T. P. (1991). Questions, options and criteria: Elements of design space analysis. *Human-Computer Interaction*, *6*, 201–250.

Moggridge, B. (2006). *Designing interaction*. Cambridge, MA, USA: MIT Press.

Monk, A. G., & Wright, P. (1993). *Improving your human-computer interface: A practical technique*. Boston: Prentice Hall.

Mylopoulos, J., Chung, L., & Yu, E. (1999). From object-oriented to goal-oriented requirements analysis. *Communications of the ACM*, *42*, 31–37.

Norman, D. A. (1999). *The invisible computer: Why good products can fail, the personal computer is so complex, and information appliances are the solution*. Cambridge, MA, USA: MIT Press.

Paterno, F. (1999). *Model-based design and evaluation of interactive applications*. Berlin, Germany: Springer-Verlag.

Potts, C. (1999). ScenIC: A strategy for inquiry-driven requirements determination. In *Proceedings: 4th IEEE International Symposium on Requirements Engineering* (pp. 58–65). Los Alamitos CA, USA: IEEE Computer Society Press.

Potts, C., Takahashi, K., & Anton, A. I. (1994). Inquiry-based requirements analysis. *IEEE Software*, *11*, 21–32.

Rolland, C., Ben Achour, C., Cauvet, C., Ralyte, J., Sutcliffe, A. G., & Maiden, N. A. M. (1998). A proposal for a scenario classification framework. *Requirements Engineering*, *3*, 23-47.

Rosch, E., Mervis, C. B., Gray, W., Johnson, D., & Boyes-Braem, P. (1976). Basic objects in natural categories. *Cognitive Psychology*, *7*, 573–605.

Rosson, M. B, & Carroll, J. M. (2001). *Usability engineering: Scenario-based development of human computer interaction*. San Francisco: Morgan Kaufmann.

Simon, H. A. (1973). The structure of ill-structured problems. *Artificial Intelligence*, *4*, 181–201.

Sutcliffe, A. G. (2000). Bridging the communications gap: Developing a lingua franca for software developers and users. In *Actes du XVIIIe Congres: INFORSID* (pp. 13–32). Toulouse, France: Inforsid.

Sutcliffe, A. G. (2002). *User-centred requirements engineering*. London: Springer.

Sutcliffe, A. G., & Carroll, J. M. (1998). Generalizing claims and reuse of HCI knowledge. In H. Johnson, L. Nigay, & C. Roast (Eds.), *People and computers XIII; Proceedings: BCS-HCI Conference* (pp. 159–176). Berlin, Germany: Springer-Verlag.

Sutcliffe, A. G., & Carroll, J. M. (1999). Designing claims for reuse in interactive systems design. *International Journal of Human-Computer Studies*, *50*, 213–241.

Sutcliffe, A. G., Gault, B., & Maiden, N. A. M. (2005). ISRE: Immersive scenario-based requirements engineering with virtual prototypes. *Requirements Engineering*, *10*, 95–111.

Sutcliffe, A. G., & Maiden, N. A. M. (1992). Analysing the novice analyst: Cognitive models in software engineering. *International Journal of Man-Machine Studies*, *36*, 719–740.

Sutcliffe, A. G., Maiden, N. A. M., Minocha, S., & Manuel, D. (1998). Supporting scenario-based requirements engineering. *IEEE Transactions on Software Engineering*, *24*, 1072–1088.

Sutcliffe, A. G., & Ryan, M. (1997). Assessing the usability and efficiency of design rationale. In S. Howard, J. Hammond, & G. Lindgaard (Eds.), *Proceedings: Human Computer Interaction INTERACT-97* (pp. 148–155). London: Chapman and Hall.

Sutcliffe, A. G., Thew, S., Venters, C., De Bruijn, O., NcNaught, J., Proctor, R., & Buchan, I. (2007). ADVISES project: Scenario-based requirements analysis for e-science applications. In *Proceedings of UK All Hands Conference on e-Science, Nottingham.* Available on CD-ROM.

Thew, S., Sutcliffe, A. G., De Bruijn, O., McNaught, J., Procter, R., Venters, C., & Buchan, I. (2008). Experience in e-science requirements engineering. In *Proceedings: 16th IEEE International Requirements Engineering Conference* (pp. 277–282). Los Alamitos CA, USA: IEEE Computer Society Press.

Thew, S., Sutcliffe, A. G., Procter, R., De Bruijn, O., McNaught, J., Venters, C., & Buchan, I. (2009). Requirements engineering for e-science: Experiences in epidemiology. *IEEE Software*, *26*(1), 80-87.

Toulmin, S. (1958). *The uses of argument*. Cambridge, UK: Cambridge University Press.

Yu, E. S. K. (1993). Modelling organisations for information systems requirements engineering. In S. Fickas & A. C. W. Finkelstein (Eds.), *Proceedings: 1st International Symposium on Requirements Engineering* (pp. 34–41). Los Alamitos, CA, USA: IEEE Computer Society Press.

Author's Note

All correspondence should be addressed to:
Alistair Sutcliffe
Manchester Business School
University of Manchester
Booth Street East
Manchester M15 6PB, UK
ags@manchester.ac.uk

Human Technology: An Interdisciplinary Journal on Humans in ICT Environments
ISSN 1795-6889
www.humantechnology.jyu.fi