University of Colorado at Boulder

Center for LifeLong Learning and Design (L3D)

Department of Computer Science

ECOT 717 Engineering Center
Campus Box 430
Boulder, Colorado 80309-0430
(303) 492-1592, FAX: (303) 492-2844

Turning Breakdowns into Opportunities for Creativity

Gerhard Fischer

Department of Computer Science and Institute of Cognitive Science
University of Colorado
Campus Box 430
Boulder, CO 80309
gerhard@cs.colorado.edu

Knowledge-Based Systems, Vol. 7, No. 4 December 1994, pp. 221-232

Turning breakdowns into opportunities for creativity

Gerhard Fischer

Design is a creative activity. Computational artifacts change the way people design. The author's research in design and design support systems has been firmly rooted in a cooperative problem solving approach that tries to amplify the creativity of designers by the use of domain-oriented design environments. Evolving design artifacts represented within the environment are able to 'talk back' to both system builders and future users, and act as representations for mutual understanding between various stakeholders. The domain orientation allows design activities to be conversations with the materials of the design situation.

However, in many situations, artifacts do not speak for themselves. To address this problem, the environments were augmented with computational agents that identified *breakdowns* which might have remained unnoticed without them. These breakdowns provide opportunities for enhancing the creativity of designers by giving them support in reframing problems, attempting alternative design solutions, and exploring relevant unknown background knowledge.

Keywords: human creativity, cooperative problem solving, domain-oriented design environments

One of our primary research goals is to design and develop computational artifacts that empower humans, rather than to build expert systems that replace them [Fischer 90; Fischer, Nakakoji 92]. We have pursued this approach not only because automation approaches have failed in many domains and serious doubts have been articulated about 'in principle' limitations of expert systems [Suchman 87; Winograd, Flores 86], but also, primarily, because we are convinced that humans enjoy doing, deciding, and learning and that people's creativity unfolds in response to experiencing breakdowns in their activities.

There is overwhelming evidence that we learn from our mistakes, whether as learners [Papert 80], as skilled domain workers [Fischer 91], as scientific communities [Popper 65], or as humankind as a whole [Lee 92]. Petroski [Petroski 85] observed that 'the colossal disasters that do occur are ultimately failures of design, but the lessons learned from those disasters can do more to advance engineering knowledge than all the successful machines and structures in the world'. We hope that our ongoing research efforts in designing, building, and evaluating design environments will increase the benefits of these environments for the amplification of human creativity. Beyond trying to understand creative processes in general and the importance of breakdowns in particular, we have focused our work on creating instrumental versions of computational domain-oriented design environments (DODEs) [Fischer 92], which assist humans in noticing and understanding breakdowns and exploiting them as sources for creative explorations. Our interest is in understanding how designers and hence how computational systems can best support them. (We are not pursuing the question of how computers can be made creative [Boden 91].) As is the case with any cognitive artifact, the existence of design support systems significantly influences and changes design [Norman 93], illustrating the fact that human activities and thinking are dependent upon interchanges with the world. Our approach builds upon the claim that creativity is significantly influenced by people reacting and making sense out of breakdowns as they occur in the creation and evolution of artifacts and in reflection about them.

In this paper, we will first characterize the nature of breakdowns in design, provide a brief characterization of methods and support mechanisms to amplify designers' creativity, and illustrate the specific roles of breakdowns in this context. A concrete example (a specific design environment and a brief scenario) is presented to illustrate how our theoretical framework guides the development of computational artifacts, and how the existence of these artifacts creates the breakdowns used to further refine our conceptual framework.

BREAKDOWNS IN DESIGN

Design is creative, dealing with ill defined and wicked problems [Simon 81; Rittel 84]. Each design problem is unique in some aspect, and therefore each solution or solution attempt provides room for creativity. Partial but critical feedback in design is provided by the 'back-talk

Department of Computer Science and Institute of Cognitive Science, University of Colorado, Campus Box 430, Boulder, CO 80309, USA
Paper received 1 December 1993. Revised paper received 23 May 1994. Accepted 16 June 1994

0950-7051/94/04/0221–12

of the situation' [Schoen 83]. While engaging in 'a conversation with the design material', designers can become aware of a breakdown. This awareness is triggered by evaluation and appreciation of the current design stage (artifact) in terms of the task at hand (goal). The evaluation is done either by the designers themselves, or by outside agents such as design teachers, colleagues or computational agents (e.g. critics) in DODEs [Fischer *et al.* 91a]. The occurrence of a breakdown will not only change the solution (e.g. the artifact under construction), but also challenge the framing of the problem in the same way (thereby contributing to the integration of problem framing and problem solving [Rittel 84]. Defining breakdowns in this way is related to but not identical to Heidegger's notion that objects and properties are not inherent in the world, but arise only in the event of breaking down, in which case they change from 'ready-to-hand' to 'present-at-hand' [Winograd, Flores 86]. Seen in this way, breakdowns reveal to us the nature of our understanding, our practices, and our tools. By providing opportunities for reflection and sources for discovery, they function in a positive rather than negative way.

Description of various breakdown situations

A first objective in many design activities is to incrementally establish a shared understanding between all stakeholders (i.e. clients, skilled domain workers, software developers and users (see *Figure 5* further below)). To overcome the 'symmetry of ignorance' [Rittel 84] between the stakeholders, design activities are undertaken to produce explicit design representations (e.g. scenarios, mockups, partial specifications, prototypes, seeds) that ground communication of design intent and design critiques [Ostwald, Nakakoji 94].

There are numerous *causes* for breakdowns. Slips or oversights (implementation errors) occur when designers act differently from their intentions, whereas mistakes (specification errors) are based on inappropriate intentions [Lewis, Norman 86]. In most computational environments, support mechanisms to identify and signal breakdowns are restricted to violations *within* the computational environment. Error messages generated by a compiler are the most straightforward breakdowns to identify, because the syntax and semantics of programming languages can be precisely defined. Systems such as LISP-CRITIC [Fischer 87] can identify breakdown situations with respect to the use (according to style, and cognitive and machine efficiency criteria) of the programming language (the LISP-CRITIC system plays a role with LISP programs that is similar to that played by a human technical editor with a paper or book). The domain orientation of our environments allows us to capture breakdowns as they occur in the *problem domain* in which designers may violate design principles by overlooking them or being unaware of them, or may violate unknown and unarticulated constraints [Lee 92].

Breakdowns in cooperative problem solving systems

In cooperative problem solving systems, breakdowns are not as detrimental as in expert systems, because humans are part of the overall system and can step in if necessary. One can never anticipate or 'design away' all of the misunderstandings and problems that may arise during the use of these systems. We need to recognize and develop system resources for dealing with the unexpected. 'The problem is not that communicative trouble arises that does not arise in human-to-human communication, but rather that when these inevitable troubles do arise, there are not the same resources available for their detection and repair' [Suchman 87]. A cooperative agent needs to understand the nature of open problems, the intentions of the problem solver, and the fact that goals are modified during the problem solving process.

Breakdowns in computational design environments are not primarily *experienced* by the designers of the systems (i.e. the software designers), but by their users (i.e. the domain designers, such as kitchen designers [Fischer *et al.* 91b], computer network designers [Fischer *et al.* 92], user interface designers [Lemke, Fischer 90], or voice dialog designers [Repenning, Summer 92; Harstad 93]). It is impossible to avoid breakdowns for numerous reasons. Design for an 'ideal situation' is impossible; for example, work-oriented design [Ehn 88] stresses that it is essential to design for the work that people do rather than for a disembodied, idealized description of the work process. Human knowledge is *tacit* [Polanyi 66], and it surfaces only in concrete problem situations (these observations have led us to the development of the 'seeding–evolutionary-growth–reseeding' model described later in the paper).

Various *resources* may be available to deal with breakdowns as they occur. The breakdown may just be noted and explained by a brief descriptive message. Beyond this, mechanisms may exist to allow designers to access argumentation, design rationale, design principles, and previous design cases to reflect, analyze and explore a breakdown situation. It is important that these information structures are contextualized with respect to the task the designer is pursuing.

AMPLIFYING THE CREATIVITY OF DESIGNERS

Numerous researchers have stressed the critical role of breakdowns, especially in design activities. Our thinking and our work have been influenced by the work of Ehn [Ehn 88], Norman [Norman 93], Polanyi [Polanyi 66], Rittel [Rittel 84], Schoen [Schoen 83], Simon [Simon 81], Suchman [Suchman 87], and Winograd and Flores [Winograd, Flores 86]. The major design principles derived from the frameworks that shape our own research are summarized in the following sections. They are particularly influenced by the following characterization of design activities by Schoen ([Schoen 83], p 79):

> The designer shapes the situation in accordance with his initial appreciation of it, the situation, 'talks back', and he responds to the situation's back-talk. In a good process of design, this conversation with the situation is reflective. In answer to the situation's back-talk, the designer reflects-in-action on the construction of the problem.

Design is a conversation with the materials of a design situation

We operationalized this principle by creating DODEs in support of human problem-domain communication

[Fischer, Lemke 88]. The 'materials' of the design situation are not low-level computer abstractions but objects with which domain workers are familiar that are part of the practice of their specific communities. DODEs allow designers to focus on the task rather than on the medium. They are an attempt to turn the proportion of effort around. Most humans using computers spend 80% of their time 'fighting the machine', and 20% on their task. The objective of DODEs is to allow people to spend at least 80% of their time on the task by establishing a common preunderstanding which lets users communicate with a minimum of words and conscious effort.

Situations need to talk back

Our ability to notice the shortcomings by (visual) inspection and careful analysis is limited. A certain amount of 'back talk' will be provided by the design situation itself (especially by environments supporting human problem-domain communication). However, the 'back talk' can be and needs to be greatly enriched by additional mechanisms:

- feedback from other stakeholders involved in the design process and human critics,
- computational critics [Fischer *et al.* 91a],
- simulation components which illustrate the behavior of an artifact [Repenning, Sumner 92].

In providing additional feedback, it is important that the 'back talk' is relevant to the actual design situation and that it is articulated in a way the designer can understand [Fischer *et al.* 93]. Specification components [Nakakoji 93] can increase the relevance of feedback to the task at hand by providing mechanisms for a partial characterization of the particular design situation.

Reflection in action

Action (as construction) and reflection (as argumentation) is integrated by the critiquing component in our systems [McCall, Fischer, Morch 90; Fischer *et al.* 91b]. This integration is important because (a) it creates a context-sensitive mechanism that provides entry into an issue base where argumentation relevant to the constructive design situation can be found, and (b) it makes designers aware that they may need additional information. Whereas reflective processes are triggered by violations of principles of design, our systems support reflection on the principles of design themselves as well. Our work has shown that the value of stand-alone design rationale systems which are isolated from the artifact under construction (such as gIBIS) [Conklin, Begeman 88] and DesignRationale [MacLean, Young, Moran 89]) can be greatly enhanced by integrating construction and argumentation with a critiquing component [Fischer *et al.* 91b].

Integration of problem framing and problem solving

The hardest and most important problem in design is not that of solving a given problem, but that of figuring out what problem to solve. The strong intertwining between problem framing and problem solving provides one of the major sources and challenges for creativity. This view denies the objective existence of problems. It focuses our efforts on searching *for* a problem space rather than just *within* a problem space, and it questions all design methodologies which are founded on a separation of analysis and synthesis (e.g. the specification driven approaches used in software engineering). It also emphasizes the importance of putting problem owners in charge, because they have the authority and the knowledge to redefine the problem on the fly [Lave 88].

To gain a deeper understanding of the integration of problem framing and problem solving, we have conducted an empirical study of a success model of cooperative problem solving between people in a large hardware store [Fischer, Reeves 92]. The following dialog shows an interaction in which a customer wanted to buy heaters. Through a collaborative problem solving effort with a sales agent, the problem was reconceptualized from one of 'adding heat' to one of 'retaining heat'. This appears to be a trivial reframing and hardly worthy of notice, but understanding this kind of reframing is crucial to the understanding and amplification of creativity; the problem *itself* was redefined.

Customer: 'I want to get a couple of heaters for a downstairs hallway.'

Sales agent: 'What are you doing? What are you trying to heat?'

Customer: 'I'm trying to heat a downstairs hallway.'

Sales agent: 'How high are the ceilings?'

Customer: 'Normal, about eight feet.'

Sales agent: 'Okay, how about these here?'

(***They proceed to agree on two heaters.***)

Customer: 'Well, the reason it gets so cold is that there's a staircase at the end of the hallway.'

Sales agent: 'Where do the stairs lead?'

Customer: 'They go up to a landing with a cathedral ceiling.'

Sales agent: 'OK, maybe you can just put a door across the stairs, or put a ceiling fan up to blow the hot air back down.'

Design knowledge is tacit

Competent practitioners usually know more than they can say [Polanyi 66]. Their tacit knowledge is triggered by new design situations and by breakdowns that occur as they engage in a design process. This implies that design worlds are not closed but open-ended, and they require support for end-user modifiability [Fischer, Girgensohn 90; Candy, Edmonds, O'Brien 94] and the evolution of design environments (starting with seeded

environments) in response to new design problems [Fischer *et al.* 94].

Saying the 'right' thing at the 'right' time in the 'right' way

The challenge of building computational environments is primarily not just to provide more information, but to say the 'right' thing at the 'right' time [Fischer *et al.* 93]. There is a spectrum of how accurately and closely related the information presented should be to the designer's task at hand. At one end of the spectrum is information retrieved on the basis of precise queries formed by designers. At the other end of the spectrum is arbitrary information (related only in the most general terms) presented to the designer [Owen 86]. Creative design should be both *innovative* and *valuable*. If designers are given arbitrary information, they are likely to get innovative ideas as a result of the information, but the ideas may not be valuable. While this may be true in the overriding number of cases, we should not overlook the value of serendipity and especially pseudoserendipity [Roberts 89]. By recognizing the importance of background knowledge for ill defined design problems, the unexpected and unintended encounters one has in browsing can at times be of much greater importance than efficient precise recall.

Distributed cognition

Design environments embed the design activities in rich information spaces and provide mechanisms to contextualize these information spaces to the task at hand. This approach contrasts sharply with many other approaches which model design and creativity purely in terms of cognition by focusing on only (a) the disembodied mind and (b) knowledge in the head rather than on the mutual exploitation of knowledge in the head with knowledge in the world [Norman 93]. We externalize (by acting) to find out what is in our head [Simon 81; Edmonds 94]. We need our artifacts to talk back to us — and this back talk needs to be as rich and as task-relevant as possible.

Systems that support distributed cognition are desirable for the following reasons: (a) they support mutual intelligibility (reciprocal recognizability of our actions, enabled by common conventions for the expression of intent, and shared knowledge about typical situations), (b) they support communicative economy (the premises or rationale of an action can be assumed to be shared, and they can be left unspoken), and (c) they allow tools and artifacts to become ready-to-hand and invisible, allowing users to communicate more directly with the task.

COMPUTATIONAL ENVIRONMENTS THAT SUPPORT TURNING BREAKDOWNS INTO OPPORTUNITIES FOR CREATIVITY

Domain-oriented design environments have emerged in our research work as computational environments that support turning breakdowns into opportunities for creativity. They are semiformal systems that integrate object-oriented hierarchies of domain objects, rule-based critiquing systems, case-based catalog components, and argumentative hypermedia systems. They are representational media that support communications and negotiations between all the stakeholders and between the designers and their work in progress. They do limited reasoning and interpretations, trigger breakdowns, deliver information, and support the exploration of the rationale behind the artifact.

To ground the discussion, this section first presents a scenario with a specific design environment, followed by a description of the domain-independent architecture underlying DODEs. Next, a process model for the evolution of DODEs will be described, and some of the specific contributions and challenges created by DODEs will be mentioned.

Scenario

The Voice Dialog Design Environment (VDDE) (see *Figure 1* [Repenning, Sumner 92]) will be used to illustrate our conceptual framework. Voice dialog interfaces consist of a series of voice prompted menus. Users press buttons on a telephone keypad and the system responds with appropriate voice instructions. Current interface design technologies for voice dialog systems are based on flowcharts. It is difficult for designers, customers and end users of these systems to anticipate what the (audio) interaction will sound like by simply looking at a static visual diagram. For breakdowns to be experienced, simulations are needed which can serve as representations for mutual understanding by allowing designers, customers and end users to 'experience' the actual audio interface.

Earlier versions of VDDE did not contain a critiquing component, limiting the 'back talk' to the designers. Voice dialog design is complicated by the fact that there are different rule sets which should be obeyed by a design. VDDE-Critics [Harstad 93] is a development effort to add critics to VDDE to signal additional breakdowns for the designers. *Figure 2* shows the user interface of the system which allows designers to tailor the 'breakdown' characteristics of the system to their personal needs by

- selecting the rule set and the associated argumentation to be used,
- determining the intrusiveness of the critiquing mechanisms with the critiquing thermometer,
- determining the design component to be critiqued (a conceptual unit versus the overall design).

Figure 3 shows the argumentation behind the artifact that empowers designers to agree or disagree with the specific design choices and their associated design rationale. VDDE supports (a) a reflective practice for the designer through a critiquing enhanced conversation with the situations and (b) the relation between designers and their clients by providing artifacts which increase mutual understanding by the identification of breakdowns.

In summary, the scenario illustrates the following specific breakdown situations captured within DODEs:

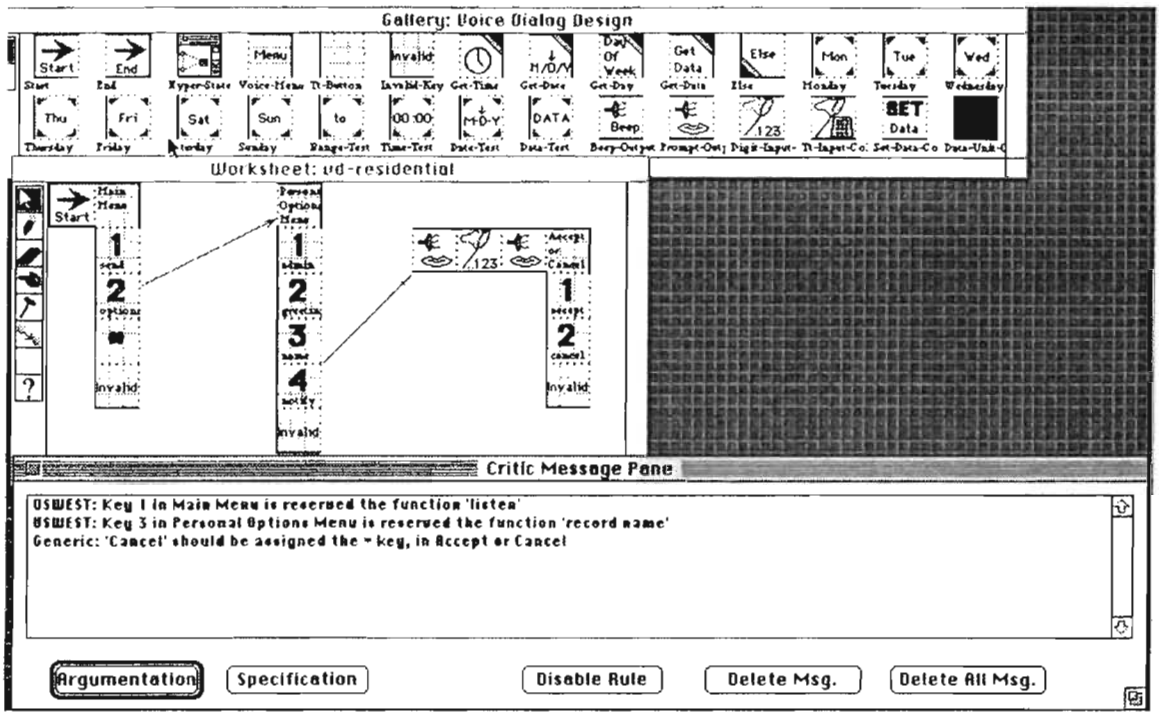

Figure 1 Voice Dialog Design Environment
[The VDDE allows domain designers to create graphic specifications. The top window is a gallery of domain-oriented components. The middle window is a worksheet where designers create a specific design. The behavior of the design can be simulated at any time. Design simulation consists of a visual trace of the execution path combined with audio feedback of all prompts and messages encountered. The bottom window displays the critiquing message for the design under construction.]

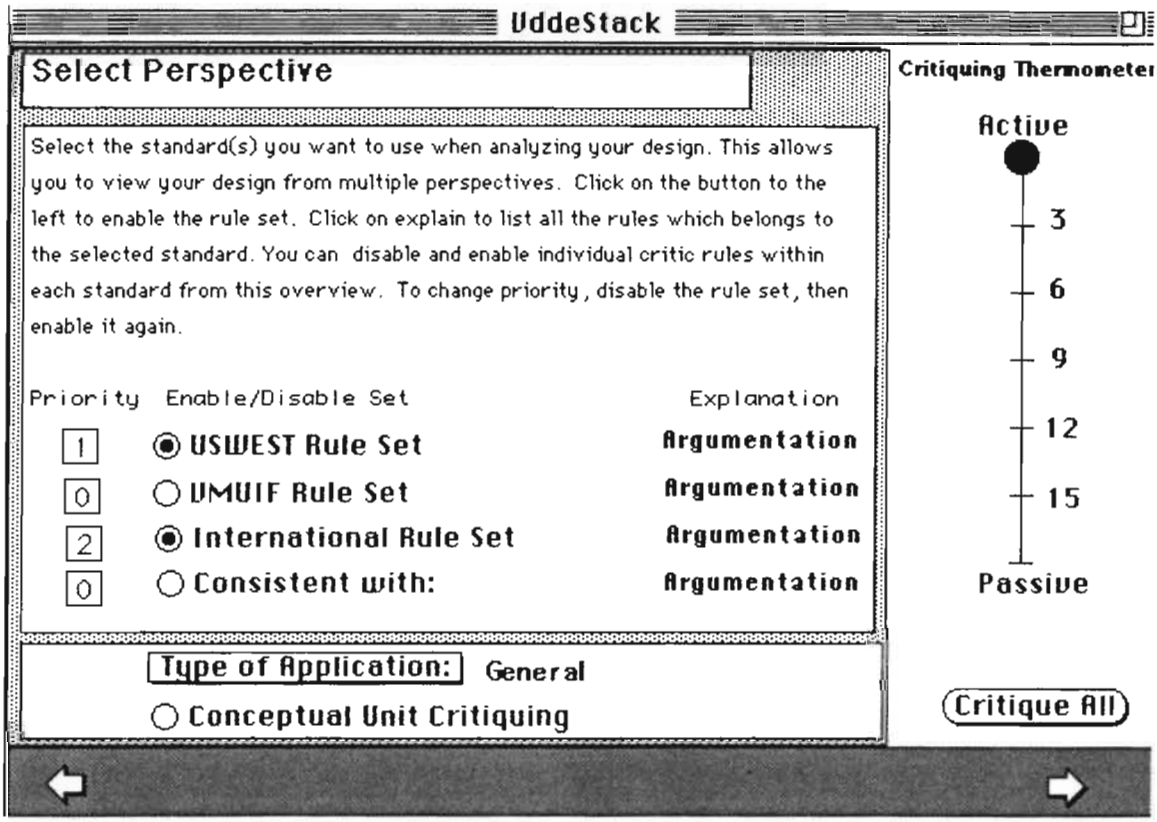

Figure 2 Specifying breakdown characteristics of VDDE

- the domain-orientation enhances the conversation with the materials of a design situation,
- simulations let stakeholders experience behavior and see the consequences of their assumptions,
- critiquing signals violation of rules and controversial design decisions,
- argumentation provides the arguments behind rules and artifacts to empower designers to disagree.

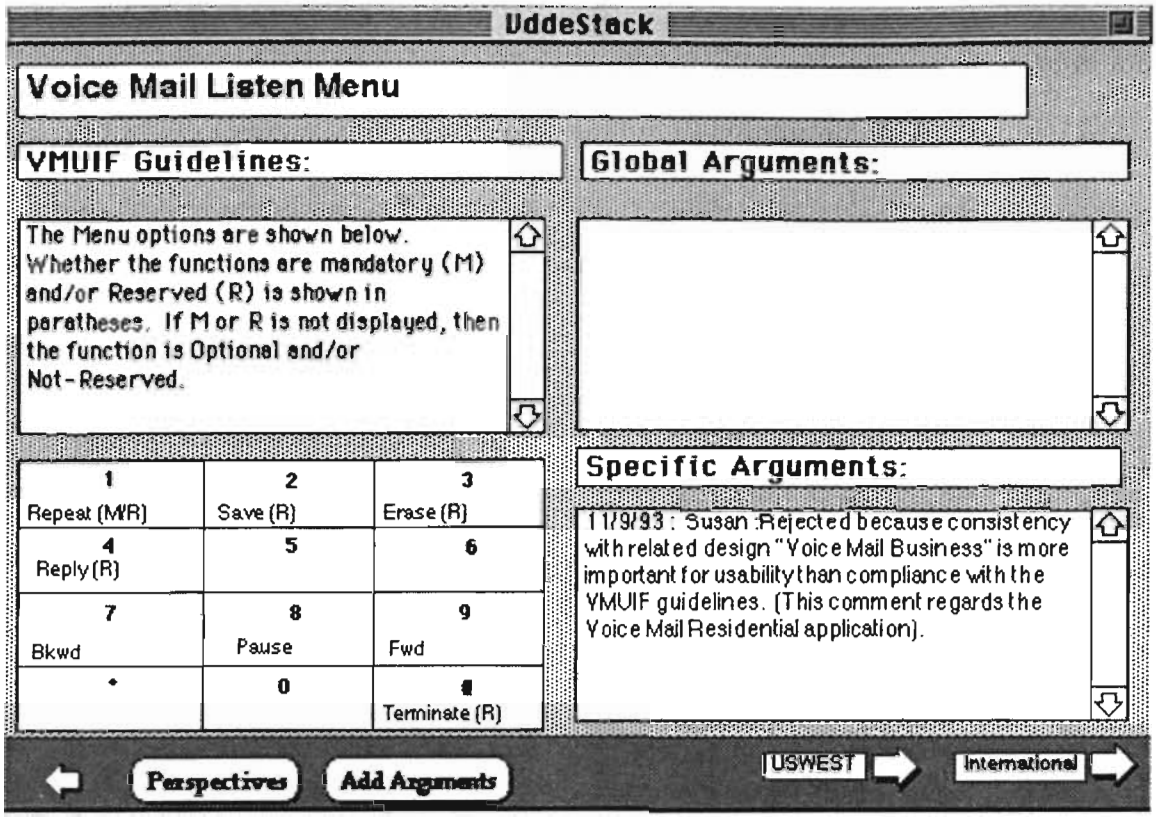

Figure 3 Argumentation component of VDDE

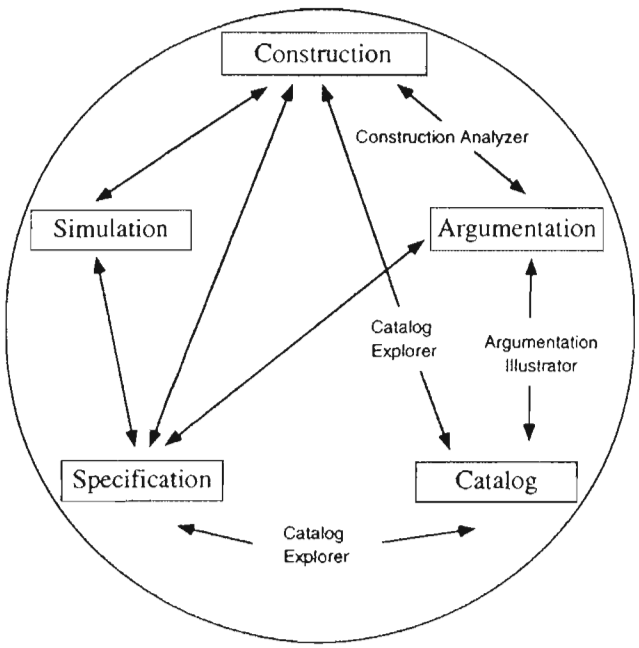

Figure 4 Multifaceted architecture for DODEs
[The various DODEs which we have developed over the last few years do not always contain all components. The VDDE system illustrated with the scenario contains the construction, simulation and argumentation component. A DODE for kitchen design [Fischer, Nakakoji 92] contains all components except a simulation component.]

Domain-oriented design environments

DODEs [Fischer 92] have been evolved in our research over several years to instantiate the conceptual framework outlined in the previous sections. They contain the following major components (see *Figure 4*):

- A *construction component* (see the gallery in *Figure 1*) is the principal medium for implementing design. It provides a gallery of domain abstractions and supports the construction of artifacts using direct manipulation and other interaction styles.
- a *specification component* [Nakakoji 93] allows designers to describe some required characteristics of the design at a high level of abstraction and assigns weights of the importance to each specified item. Specifications are expected to be modified and augmented during the whole design process, rather than to be fully articulated before the design is started. They are used to prioritize the information spaces in the system with respect to the emerging task at hand.
- An *issue-based argumentative hypermedia system* (see *Figure 3*) captures the design rationale. Information fragments in the hypermedia issue base are based on issue-based information systems [McCall 91] and are linked according to what information serves to resolve an issue relevant to a partial construction [Fischer *et al.* 91b; Morch 94].
- A *catalog* provides a collection of prestored design objects which illustrates the space of possible designs in the domain [Fischer, Nakakoji 92]. Catalog examples are sources of information as reminders for amplifying designers' creativity by bringing the designers new ideas and perspectives for their design [Riesbeck, Schank 89; Kolodner 93].
- A *simulation component* allows the stakeholders to experience the behavior of an artifact (e.g. the audio feedback of prompts and messages in VDDE). It supports 'what-if' games to let designers simulate usage scenarios with the artifact being designed.

Simulation complements the argumentative component.

The architecture derives it essential value from the *integration* of its components and links between the components. Used individually, the components cannot achieve their full potential. Used in combination, each component augments the value of the others, forming a synergistic whole. Links among the components of the architecture are supported by various mechanisms (see *Figure 4*). The major mechanisms include the following:

- The Construction Analyzer is a critiquing component [Fischer *et al.* 91a] that detects and critiques partial solutions constructed by designers on the basis of domain knowledge of design principles (see *Figure 1*). The firing of a critic signals a breakdown to designers, warns them of potential problems in the current construction, and provides them with an entry into the neighborhood of relevant information in the argumentative hypermedia system in which the corresponding argumentation lies (thereby supporting the requirement to say the 'right' thing at the 'right' time in the 'right' way).
- The Argumentation Illustrator [Fischer *et al.* 91b] helps designers to understand the information given in argumentative hypermedia by using a catalog design example as a source of concrete realization. Explanations given as argumentation are often highly abstract and conceptual. Concrete design examples that illustrate the explanations in context help designers to understand the concept.
- The Catalog Explorer [Fischer, Nakakoji 92] helps designers to search the catalog space according to the task at hand. It retrieves design examples that are similar to the current construction situation and orders a set of design examples chosen for their appropriateness to the current specification.

Design, as supported by the multifaceted architecture, iterates through cycles of specification, construction, evaluation, and reuse in the working content. At each stage in the design process, the partial design embedded in the design environment serves as a stimulus for suggesting what users should attend to next. The direction to new subgoals permits new information to be extracted from memory and reference sources, and leads to new steps toward the development of the design. The integration of various aspects of design enables the situation to 'talk back' to users [Schoen 83].

Critiquing and simulation components are necessary to increase the 'back talk' of the situation. 'Artifacts do not speak for themselves' [Rittel 84], because designers have insufficient knowledge and experience to fully understand the conversation with the materials of the situation. Critiquing mechanisms [Fischer *et al.* 91a] serve as 'interpreters' that support designers in seeing and understanding the 'back talk' of the situation. When a critic fires, reflection does not occur simply on the basis of the message. Designers 'listen to' the design material with the help of the interpreter in the form of a critic. Critics provide a computational mechanism that allows designers to think about what they are doing while this thinking can still make a difference. Relevancy to the task at hand (saying the 'right' thing at the 'right' time) plays an important role here, because the more given information is relevant to the current problem situation, the more understandable the information is for a human.

Process model for creation of DODEs

Our work on creating DODEs has indicated that a promising model for creating these environments involves three major phases (see *Figure 5*) and [Fischer *et al.* 94]: seeding, evolutionary growth, and reseeding. A *seed* for a domain-oriented design environment is created through a participatory design process between environment developers and domain designers by incorporating domain-specific knowledge into a domain-independent architecture for design environments. *Evolutionary growth* takes place as domain designers use the seeded environment to undertake specific projects in response to clients' needs. *Reseeding* is a process that reinvolves the environment developers in helping domain designers to better organize, formalize, and generalize knowledge added during the use phases.

Seeding

A seed is built by customizing the domain-independent design environment architecture (see *Figure 4*) to a particular domain through a process of knowledge construction. Although the goal is to construct as much knowledge as possible during seed building, for complex and changing domains complete coverage is not possible. In addition, design knowledge is tacit and design worlds are open-ended [Polanyi 66], users of design environments must be able to extend them in response to breakdowns. Domain designers and environment developers bring to their encounter a body of understanding which they can only very partially communicate to one another, much of which they cannot describe to themselves. Therefore, the seed is explicitly designed for redesign by capturing design knowledge during use [Girgensohn 92].

Domain designers must participate in the seeding process because they have the expertise to determine when a seed can support their work practices. Rather than expecting designers to articulate precise and complete system requirements prior to seed building, we view seed building as knowledge *construction* (in which knowledge structures are collaboratively designed and built), rather than as knowledge *acquisition* (in which knowledge is transferred from domain experts to an environment developer and finally expressed in formal rules and procedures). New seed requirements are elicited by constructing and evaluating domain-oriented knowledge structures.

A seed is a collection of knowledge and procedures that is capable of growing through interaction with domain designers during day-to-day use. It stimulates, focuses, and mediates discussion during the incremental growth phase. The seed must be capable of capturing the information elicited from the use of the system. There is no absolute requirement for the completeness, correctness, or specificity of the information in the seed. In fact, it is often its shortcomings in these respects that provoke input from designers.

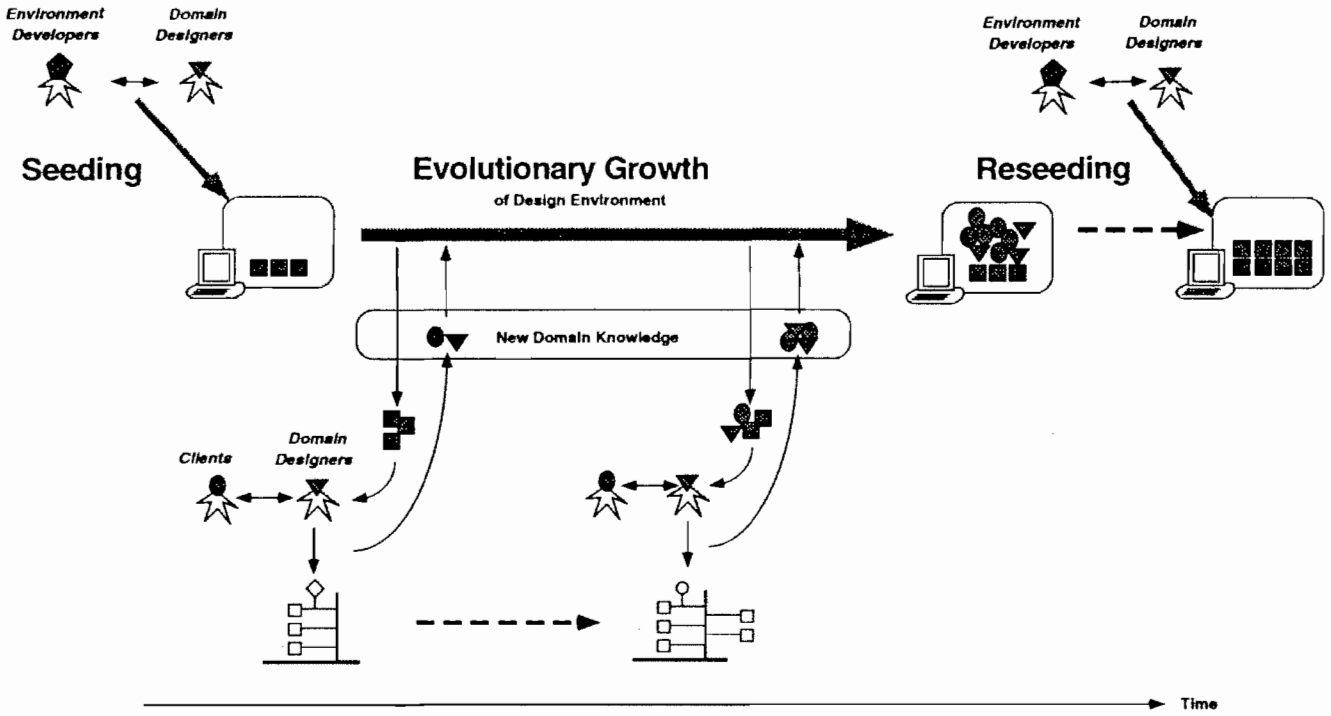

Figure 5 Seeding, evolutionary growth, and reseeding: a process model for DODEs
[During seeding, environment developers and domain designers collaborate to create a design environment seed. The seed should contain a mechanism for all the components shown in *Figure 4*. During evolutionary growth, domain designers create artifacts in response to the needs of clients that add new domain knowledge to the seed. In the reseeding phase, environment developers again collaborate with domain designers to organize, formalize, and generalize new knowledge.]

Evolutionary growth through use

The seed will *evolve* in response to its extensive use in new design projects where the new design projects will create breakdowns, illustrating the limitations of the existing environment. 'Use' is not just use, because requirements fluctuate, change is ubiquitous, design knowledge is tacit, and design environments need to evolve [Henderson, Kyng 91]. This evolution is primarily driven by using the existing environment to develop new designs that uncover its limitations through *breakdowns*. Examples of such breakdowns which we have observed in the domain of kitchen design [Fischer, Nakakoji 92] are (a) an appliance (e.g. a microwave) is missing from the gallery, and (b) a critiquing rule is too general (e.g. appliances should be against the wall, which is not applicable for island kitchens). These breakdowns are perceived by the people who use the programs, not by the professionals who have developed them in the first place. Integrating this knowledge into the system requires support for end-user modifiability and end-user programming [Fischer, Girgensohn 90; Nardi 93; Eisenberg, Fischer 94; Eisenberg 94].

Our approach to this challenge is to view the design environment seed as a medium for communication as well as design. Our critique of current design systems is that they function as 'keepers of the artifact', in which one deposits representations of the artifact being designed. However, our experience has shown that designers integrate designing and discussing in such a way as to make separate interpretation difficult [Reeves 93]. Talking *about* an artifact requires talking *with* the artifact. Therefore, later interpretation of the discussion requires that the discussion be embedded in the context in which it was originally elicited, requiring that design artifacts must not be artificially separated from the communication about them.

Evolutionary growth during system use is a process of adding information related directly or indirectly to the artifact being designed. The artifact is the foundation for evolutionary growth. During the growth phase, the designers who use the system are primarily focused on their task at hand. Information input is highly situation specific: tied to a specific artifact and stated in particular rather than in general. For a while, information grows in an orderly manner, but eventually order breaks down and the system begins to degrade in usefulness.

Reseeding

Acquiring design knowledge is of little benefit unless it can be delivered to designers when it is relevant. Periodically, the growing information space must be structured, generalized, and formalized in a reseeding process, which increases the computational support the system is able to provide to designers [Shipman, McCall 94].

The task of reseeding involves environment developers working with domain designers. After a period of use, the information space can be a jumble of annotations, partial designs, and discussions mixed in with the original seed and any modifications performed by the domain designers. To make this information useful, the environment developers work with the domain designers in a process of organizing, generalizing, and formalizing the new information and updating the initial seed.

Reseeding is necessary when evolutionary growth stops proceeding smoothly. During reseeding, the system's information is restructured, generalized and formalized to serve future design tasks. The reseeding process creates a forum to discuss what design information captured in the context of specific design projects should be incorporated into the extended seed to support the next cycle of evolutionary growth and reseeding. Tools contained in design environments support reseeding by

making suggestions about how the information can be formalized.

Impact of DODEs on design and creativity

Representations for mutual understanding

Our DODEs serve as representations for mutual understanding for environment developers, domain designers and end users (see *Figure 5*) and 'languages of doing' [Ehn 88] in which an evolving artifact is embedded. By externalizing goals and plans in concrete artifacts, 'objects-to-think-with' and 'objects-to-point-to' are created that (a) allow mutual understanding of the domain as well as computing concepts, (b) ground and focus discussion, and (c) talk back to both software designers (system builders) and domain workers (future users). Breakdowns experienced by *future users* include the following situations: (a) the prototype does not perform as specified (this is caused by communication difficulties between designers and users), (b) the prototype performs as specified, but not as expected (this is caused by the inability of users to articulate their goals and needs), and (c) there is a possibility that new insights are triggered by the evolving artifact itself (demonstrating that specifications are never complete).

Breakdowns experienced by software designers may be caused as follows: (a) the interpretation of partial specifications leads to breakdowns, showing insufficient mutual understanding, as well as incompleteness and inconsistency of the specifications, and (b) the existing design environment is found inadequate for the new task at hand, leading to potential extensions and insights for evolving the design environment.

Evolution and end-user modifiability

Design is more often modification than innovation [Basalla 88]. Breakthroughs and fundamentally new approaches are rare, because designers are people, and people's imaginations and knowledge are limited and knowledge is evolutionary [Popper 65].

Obeying general rules and design standards is necessary but not sufficient for good design. DODEs must support the designer in seeing the situation at hand as *unique*. Domain standards can help constrain designs, but at the same time standards alone do not determine a design solution. For interesting design domains, generic design rules can only play a part in determining the design. A critiquing system that criticizes all designs based on the same standards is based on *standard critics* [Fischer *et al.* 93]. Design domains often have a basic set of rules that all artifacts in that domain should follow. Standards that apply to all designs in a domain are important for designers to understand and follow. Although standard critics are good at enforcing a set of rules that may be applied to all designs for a domain, they do not fully support design processes. *Conditional critics* allow a design environment to evaluate design situations in accordance with partial specifications [Fischer *et al.* 93]. The partial specification represents a set of goals articulated by the user. Each specification item corresponds to a set of critics, which detect design situations relevant to that specification item. The set of specification items chosen by the designer determines which critics are active. Only active critics participate in the evaluation of design situations. The partial specification is a resource for both the system and the designer (and is thereby an important part of the shared knowledge): (a) it allows the system to generate design-specific (rather than domain-specific) critiquing, and (b) it allows the designer to understand the design in terms of its unique characteristics rather than its common ones.

Breakdowns as a driving force in the evolution from programming languages to end-user modifiable design environments

Our current conceptual framework and prototypes of design environments have a long history. Their evolution was driven by breakdowns uncovered by (a) a more elaborate conceptual framework, (b) the use of the environments, and (c) assessment and evaluation studies. The work started several years ago in an effort to support human problem-domain communication with domain-oriented construction kits [Fischer, Lemke 88]. Construction kits enable designers to create artifacts quickly, but they provide no feedback on the quality of a design. Augmenting construction kits with a critiquing component turned them into design environments. Critics [Fischer *et al.* 91a] identify potential problems in the artifact being designed. Their knowledge is based on design principles of the domain. (Designers might violate these principles by believing in different arguments, out of ignorance or because of a temporary oversight.)

Our original assumption was that designers would have no difficulty in understanding these critic messages. User experiments [Lemke, Fischer 90] demonstrated that the short messages the critics presented to designers did not reflect the complex reasoning behind the corresponding design issues. To overcome this shortcoming, we initially developed a static explanation component for the critic messages based on the assumption that there is a 'right' answer to a problem. However, the explanation component proved unable to account for the deliberative nature of design problems [Rittel 84; McCall 91]. By combining construction and argumentation, *integrated* design environments support 'reflection-in-action' as a fundamental process underlying design activities [Schoen 83; Fischer, Nakakoji 92].

However, even integrated design environments have their shortcomings. Design in real-world situations deals with complex, unique, uncertain, conflicted, unstable situations in practice. Design knowledge as embedded in design environments will never be complete because design knowledge is tacit (i.e competent practitioners know more than they can say [Polanyi 66]), and additional knowledge is triggered and activated by situations and breakdowns. These observations require computational mechanisms in support of *end-user modifiability* [Fischer, Girgensohn 90].

ASSESSMENT

Our approach is in line with the framework outlined by Basalla [Basalla 88], in which an artifact is the fundamental unit for the study of technology, and that continuity

prevails throughout the made world. How can novelty then appear in the midst of the continuous? Basalla argues for human imagination, socioeconomic and cultural forces, diffusion of technology, and the advancement of science as different sources of novelty.

In our own work, we have focused on the creation of partial external representations of domains in DODEs. Stakeholders in design processes are engaged in a conversation about possibilities, where individuals overcome their 'blindness' through interactions with others. The creativity of the individual (or of a specific group) is transcended by engaging in a participatory design effort [Brown, Duguid 91]. Challenges for creativity are the 'unpredictabilities' of real world situations, where the existing systems and the designers' preconceived notions break down in actual use situations [Henderson, Kyng 91]. As articulated in our process model (see *Figure 5*), the development of DODEs proceeds in a cycle from design to experience and back again. It is impossible to anticipate all of the relevant breakdowns and their domains during a seeding process. They emerge gradually in practice. System development methodologies supporting creativity must empower stakeholders to act in response to breakdowns.

Do critics enhance or hinder creativity?

Critics [Fischer *et al.* 91a] have the potential to make designers conform to the established 'wisdom of the trade'. Some people may argue that this may limit creativity. However, our own experiences and numerous other investigations [Popper 65; Petroski 85; Basalla 88] indicate that creativity and new solutions result from overcoming the shortcomings in existing solutions. By making these shortcomings explicit (e.g. through the process of using existing design environments to achieve new tasks), we strongly believe that opportunities for creativity are created.

Our user studies [Nakakoji 93] have repeatedly demonstrated that designers (amateurs as well as experts) do not agree with the advice or complaints of the computational critics. The critiquing messages were used by designers as a starting point for reflective processes (activating tacit knowledge [Polanyi 66]), leading to an articulation of their own arguments. Therefore, critiquing is more than just reminding; it provides a starting point for the articulation of additional knowledge, and can lead to new insights.

Few of the standard 'principles of design' are inviolable. They are merely rules of thumb whose appropriateness must be judged by the designer in each new situation. In other words, the designer must not merely reflect on how to apply principles, but also on whether the principles should be applied as is, modified for the particular situation, or simply abandoned.

Exploiting the unique possibilities of computers as a medium

In noncomputational environments, 'seeing' can be enhanced by training or supported by a human. Computational environments create the unique opportunity to put some of the subjective 'seeing' burden on the computation. The environments need to remediate for the perceptually untrained, by engaging them in reflective conversation at the level they can handle, and teaching them to 'see'. Beyond remediation for lack of perceptual training, there are many facts about designs that even the most well trained can never see directly, but that computation can visualize for them. If designers are willing to annotate their work products, computers can deliver this additional information to future designers [Fischer *et al.* 91b].

Motivation

Creativity requires an intense concern and personal involvement with some domain. It also requires enough self-confidence and immunity to peer pressure (whether this pressure is presented in personal interactions or in computational mechanisms such as critics and argumentation) to break the grip of standard practice.

Future work

Our research work itself is based on the conceptual framework outlined in this paper: the integration of action, assessment, and reflection. We acted for many years by building prototype systems. We create breakdowns in our thinking by assessing and evaluating the strengths and weaknesses of our prototypes, leading us to reflect about our work. This reflection has created a large number of interesting questions to be pursued in the future [Fischer, Nakakoji 92]:

- Are there differences in the performance, quality, and creativeness of the product when the system is used with and without critics, the catalog, and the simulation components?
- What is the tradeoff between running the system in a critiquing mode and running it in a constraint mode, when the latter would prevent certain problems from arising (e.g. by enforcing building codes), whereas the former would provide designers with opportunities to deal with breakdowns?
- What is the tradeoff between various intervention strategies, such as the balance between displaying enough information and disrupting the work process? When are designers willing to suspend the construction process to access relevant information? How can both modes of operation be integrated?
- Are mechanisms such as checklists and task agendas [Lemke, Fischer 90] needed and useful to address breakdowns occurring in the process rather than in the product?
- Does 'making information relevant to the task at hand' prevent serendipity [Roberts 89]?
- If an environment can always supply the information that the situation demands, why would designers bother to learn the information [Fischer 91]?
- Under what conditions will designers challenge or extend the knowledge represented in the system? How can they be motivated to do so [Fischer *et al.* 92]?

- To what extent can the 'back talk' be embedded directly into the artifact, or does it need to be handled by a separate discourse, such as feedback from critiquing and simulation components?
- To what extent are situations and reflective-conversations controlled by media properties?
- How can we achieve a balance between technical rationality (e.g. using plans and rules) and reflective action [Ehn 88; Suchman 87]? People do use plans, such as milestone charts and business plans. In particular, people working in teams might profit from systematic agreements. Even if one agrees that 'design is more than the application of standard principles', one cannot infer that principles cannot be useful.

CONCLUSIONS

> The way in which knowledge progresses, and especially our scientific knowledge, is by justified (and unjustifiable) anticipations, by guesses, by tentative solutions to our problems, by *conjectures*. These conjectures are controlled by criticism; that is, by attempted *refutations*, which include severely critical tests . . . Criticism of our conjectures is of decisive importance: by bringing out our mistakes it makes us understand the difficulties of the problem which we are trying to solve [Popper 65].

Learning from mistakes is in the center of a theory of knowledge and of its growth, the essential elements of creativity. Although Popper's thinking is primarily focused on historical creativity (i.e. ideas that are fundamentally novel with respect to the whole of human history), his arguments are equally relevant to psychological creativity (i.e. ideas that are fundamentally novel with respect to the individual mind that had the idea) [Boden 91]. Breakdowns generated by human and computational critics are opportunities to increase our creativity in our thinking as well as in the artifacts that we design.

ACKNOWLEDGEMENTS

The author would like to thank the members of the Human–Computer Communication Group at the University of Colorado, USA, who contributed to the conceptual framework and the systems discussed in this paper. The research was supported by the US National Science Foundation under grant MDR-9253425, by the HCI program of ARPA, and by grants from the NYNEX Science and Technology Center, and from Software Research Associates.

REFERENCES

[Basalla 88] G Basalla *The Evolution of Technology* Cambridge University Press, USA (1988)

[Boden 91] M Boden *The Creative Mind: Myths & Mechanisms* Basic Books (1991)

[Brown, Duguid 91] J S Brown, P Duguid 'Organizational learning and communities-of-practice: toward a unified view of working, learning, and innovation' *Organization Science* Vol 2 No 1 (1991) pp 40–57

[Candy, Edmonds, O'Brien 94] L Candy, E Edmonds, S O'Brien 'End user knowledge manipulation and creativity' *in* T Dartnall (Ed.) *Artificial Intelligence and Creativity* Kluwer (to be published)

[Conklin, Begeman 88] J Conklin, M Begeman 'gIBIS: a hypertext tool for exploratory policy discussion' *Transactions Office Information Systems* Vol 6 No 4 (1988) pp 303–331

[Edmonds 94] E Edmonds 'Cybernetic serendipity revisited' *in* T Dartnall (Ed.) *Artificial Intelligence and Creativity* Kluwer (to be published)

[Ehn 88] P Ehn *Work-Oriented Design of Computer Artifacts* Almquist & Wiksell, Sweden (1988)

[Eisenberg 94] M Eisenberg 'Programmable applications for the arts: computational tools for hand, eye and mind' *Knowledge-Based Systems* Vol 7 No 4 (1994) pp 239–246

[Eisenberg, Fischer 94] M Eisenberg, G Fischer 'Programmable design environments: integrating end-user programming with domain-oriented assistance' *Human Factors in Computing Systems CHI'94 Conference Proceedings* ACM (1994) pp 431–437

[Fischer 87] G Fischer 'A critic for LISP' *Proceedings 10th International Joint Conference on Artifical Intelligence* Morgan Kaufmann, USA (1987) pp 177–184

[Fischer 90] G Fischer 'Communications requirements for cooperative problem solving systems' *International Journal of Information Systems* Vol 15 No 1 (1990) pp 21–36 (special issue on knowledge engineering)

[Fischer 91] G Fischer 'Supporting learning on demand with design environments' *Proceedings International Conference on the Learning Sciences* Association for the Advancement of Computing in Education, USA (1991) pp 165–172

[Fischer 92] G Fischer 'Domain-oriented design environments' *Proceedings 7th Annual Knowledge-Based Software Engineering (KBSE-92) Conference* IEEE Computer Society Press, USA (1992) pp 204–213

[Fischer *et al.* 91a] G Fischer, A C Lemke, T Mastaglio, A Morch 'The role of critiquing in cooperative problem solving' *ACM Transactions Information Systems* Vol 9 No 2 (1991) pp 123–151

[Fischer *et al.* 91b] G Fischer, A C Lemke,R McCall, A Morch 'Making argumentation serve design' *Human Computer Interaction* Vol 6 No 3–4 (1991) pp 393–419

[Fischer *et al.* 92] G Fischer, J Grudin, A C Lemke, R McCall, J Ostwald, B N Reeves, F Shipman 'Supporting indirect, collaborative design with integrated knowledge-based design environments' *Human Computer Interaction* Vol 7 No 3 (1992) pp 281–314 (special issue on computer supported cooperative work)

[Fischer *et al.* 93] G Fischer, K Nakakoji, J Ostwald, G Stahl, T Sumner 'Embedding computer-based critics in the contexts of design' *Human Factors in Computing Systems INTERCHI'93 Conference Proceedings* ACM (1993) pp 157–164

[Fischer *et al.* 94] G Fischer, R McCall, J Ostwald, B Reeves, F Shipman 'Seeding, evolutionary growth and reseeding: supporting incremental development of design environments' *Human Factors in Computing Systems CHI'94 Conference Proceedings* ACM (1994) pp 292–298

[Fischer, Girgensohn 90] G Fischer, A Girgensohn 'End-user modifiability in design environments' *Human Factors in Computing Systems CHI'90 Conference Proceedings* ACM, USA (1990) pp 183–191

[Fischer, Lemke 88] G Fischer, A C Lemke 'Construction kits and design environments: steps toward human problem-domain communication' *Human–Computer Interaction* Vol 3 No 3 (1988) pp 179–222

[Fischer, Nakakoji 92] G Fischer, K Nakakoji 'Beyond the macho approach of artificial intelligence: empower human designers — do not replace them' *Knowledge-Based Systems* Vol 5 No 1 (1992) pp 15–30

[Fischer, Reeves 92] G Fischer, B N Reeves 'Beyond intelligent interfaces: exploring, analyzing and creating success models of cooperative problem solving' *Applied Intelligence* Vol 1 (1992) pp 311–332 (special issue intelligent interfaces)

[Girgensohn 92] A Girgensohn 'End-user modifiability in knowledge-based design environments' *PhD Dissertation* Department of Computer Science, University of Colorado, USA (1992) (also available as *TechReport CU-CS-595-92*)

[Harstad 93] B Harstad 'New approaches for critiquing systems: pluralistic critiquing, consistency critiquing, and multiple intervention strategies' *Technical Report CU-CS-685-93* Department of Computer Science, University of Colorado, USA (1993)

[Henderson, Kyng 91] A Henderson, M Kyng 'There's no place like home: continuing design in use' *in* J Greenbaum, M Kyng (Eds.) *Design at Work: Cooperative Design of Computer Systems* Lawrence Erlbaum, USA (1991) pp 219–240

[Kolodner 93] J L Kolodner *Case-Based Reasoning* Morgan Kaufmann, USA (1993)

[Lave 88] J Lave *Cognition in Practice* Cambridge University Press, UK (1988)

[Lee 92] L Lee *The Day the Phones Stopped* Donald I Fine Inc., USA (1992)

[Lemke, Fischer 90] A C Lemke, G Fischer 'A cooperative problem solving system for user interface design' *Proceedings AAAI-90 Eighth National Conference Artificial Intelligence* AAAI Press/MIT Press, USA (1990) pp 479–484

[Lewis, Norman 86] C H Lewis, D A Norman 'Designing for error' *in* D A Norman, S W Draper (Eds.) *User Centered System Design, New Perspectives on Human–Computer Interaction* Lawrence Erlbaum, USA (1986) pp 411–432

[MacLean, Young, Moran 89] A MacLean, R Young, T Moran 'Design rationale: the argument behind the artifact' *Human Factors in Computing Systems CHI'89 Conference Proceedings* ACM, USA (1989) pp 247–252

[McCall 91] R McCall 'PHI: a conceptual foundation for design hypermedia' *Design Studies* Vol 12 No 1 (1991) pp 30–41

[McCall, Fischer, Morch 90] R McCall, G Fischer, A Morch 'Supporting reflection-in-action in the Janus design environment' *in* M McCullough *et al.* (Eds.) *The Electronic Design Studio* MIT Press, USA (1990) pp 247–259

[Morch 94] A I Morch 'Designing for radical tailorability: coupling artifact and rationale' *Knowledge-Based Systems* Vol 7 No 4 (1994) pp 253–264

[Nakakoji 93] K Nakakoji 'Increasing shared understanding of a design task between designers and design environments: the role of a specification component' *PhD Dissertation* Department of Computer Science, University of Colorado, USA (1993) (also available as *TechReport CU-CS-651-93*)

[Nardi 93] B A Nardi *A Small Matter of Programming* MIT Press, USA (1993)

[Norman 93] D A Norman *Things That Make Us Smart* Addison–Wesley, USA (1993)

[Ostwald, Nakakoji 94] J Ostwald, K Nakakoji 'EVA: a medium for conceptual coordination in software development'

[Owen 86] D Owen 'Answers first, then questions' *in* D A Norman, S W Draper (Eds.) *User Centered System Design, New Perspectives on Human–Computer Interaction* Lawrence Erlbaum, USA (1986) pp 361–375

[Papert 80] S Papert *Mindstorms: Children, Computers and Powerful Ideas* Basic Books, USA (1980)

[Petroski 85] H Petroski *To Engineer Is Human: The Role of Failure in Successful Design* St Martin's Press, USA (1985)

[Polanyi 66] M Polanyi *The Tacit Dimension* Doubleday, USA (1966)

[Popper 65] K R Popper *Conjectures and Refutations* Harper & Row, USA (1965)

[Reeves 93] B N Reeves 'The role of embedded communication and artifact history in collaborative design' *PhD Dissertation CU-CS-694-93* Department of Computer Science, University of Colorado, USA (1993)

[Repenning, Sumner 92] A Repenning, T Sumner 'Using agentsheets to create a voice dialog design environment' *Proceedings 1992 ACM/SIGAPP Symposium Applied Computing* ACM Press (1992) pp 1199–1207

[Riesbeck, Schank 89] C Riesbeck, R C Schank *Inside Case-Based Reasoning* Lawrence Erlbaum, USA (1989)

[Rittel 84] H W J Rittel 'Second-generation design methods' *in* N Cross (Ed.) *Developments in Design Methodology* John Wiley, USA (1984) pp 317–327

[Roberts 89] R M Roberts *Serendipity: Accidental Discoveries in Science* John Wiley, USA (1989)

[Schoen 83] D A Schoen *The Reflective Practitioner: How Professionals Think in Action* Basic Books, USA (1983)

[Shipman, McCall 94] F Shipman, R McCall 'Supporting knowledge-base evolution with incremental formalization' *Human Factors in Computing Systems INTERCHI'94 Conference Proceedings* ACM (1994) pp 285–291

[Simon 81] H A Simon *The Sciences of the Artificial* MIT Press, USA (1981)

[Suchman 87] L A Suchman *Plans and Situated Actions* Cambridge University Press, UK (1987)

[Winograd, Flores 86] T Winograd, F Flores *Understanding Computers and Cognition: A New Foundation for Design* Ablex, USA (1986)