Towards a System Architecture Supporting Contextualized Learning

Gerhard Fischer,[1] Andreas C. Lemke,[1] and Raymond McCall[2]

Dept. of Computer Science,[1] College of Environmental Design,[2] and Inst. of Cognitive Science[1,2]
Engineering Center ECOT 7-7, University of Colorado, Boulder, CO 80309-0430
{gerhard, andreas}@boulder.colorado.edu, mccall_r@cubldr.colorado.edu

Abstract

We have developed a conceptual framework and a demonstration system that contextualize (or situate) learning in the context of real-world work situations. The conceptual framework is based on the following requirements: the choice of tasks and goals must be under the control of the user, not the system. The environment must be able to situate learning, allow situations to "talk back," support reflection-in-action, identify the instructional information relevant for tasks at hand, and turn breakdowns from disasters into opportunities for learning. Learning must not disrupt or interfere with solving a problem, and new information to be learned must help to accomplish the task at hand.

Our demonstration system JANUS (developed for the domain of architectural design) is built on an *integrated architecture*: a knowledge-based construction component, a hypermedia-based argumentation component, a set of critics, and a catalog of precedent solutions. Contextualized learning is supported by the critics that link construction and argumentation, and precedent solutions from the catalog that situate argumentation. Evaluation of JANUS and the underlying conceptual framework have shown that this approach combines some of the best features of open-ended learning environments and tutoring systems.[1]

Introduction

We have developed a prototype and an architecture for knowledge-based design environments that contextualize learning. Our system uses knowledge-based construction kits and hypermedia to contextualize learning, i.e., to enable users *to learn within the context of their work on real-world problems*. With such systems, learning does not take place in a separate phase and in a separate place but is integrated into the work process. The system lets users construct solutions to their own problems, advises them when they are getting into trouble, and provides directly relevant information. By letting users see for themselves the usefulness of new knowledge for actual problem situations, the system improves the motivation to learn.

In this paper, we discuss the necessity and rationale to support contextualized learning, develop a conceptual framework for the problem and describe JANUS as a demonstration system illustrating the approach. From the specific system we derive a generalized architecture for such systems. Our evaluations of our effort identified the limitations of our current work and suggested a number of extensions for future research.

The Need to Contextualize Learning

Doing problem solving and design well has always been difficult, but the explosive growth of technology is greatly magnifying this difficulty. In technologically oriented design fields, the knowledge base needed for design is growing and changing at an alarming rate (Draper, 1984; Fischer, 1988). Learning everything in advance is impossible because there are too many things to learn. Humans cannot keep up with developments in their own fields, much less in other fields of potential relevance. The large and growing discrepancy between the amount of potentially relevant knowledge and the amount an individual can know and remember puts a limit on progress in design. Overcoming this limit is a central challenge for developers of support systems.

Designers can no longer depend on learning everything they need to know before they enter the world of work. They must be able to learn on demand and learning must be contextualized. At least two things are required of a system that allows learning to take place within the context real problem-solving situations. First of all, instruction must relate to and serve the actual task situation at hand where the choice of tasks is primarily under the control of the problem solver, not the system. Secondly, learning must not disrupt or otherwise interfere with solving the problem. The system must avoid the "production paradox," (Carroll, Rosson, 1987) where learning is inhibited by lack of time and working is inhibited by lack of knowledge. The designer must regard the time and effort invested in learning to be immediately worthwhile for the task at hand — not merely for some putative long-term gain.

Contextualizing learning shares goals with the concept of "cognitive apprenticeship" of (Collins, Brown, Newman, 1989). They characterize their approach as follows:

> Perhaps as a byproduct of the relegation of learning to schools, skills and knowledge have become abstracted from their uses in the world. In apprenticeship learning, on the other hand, target skills are not only continually in use by skilled practitioners, but are instrumental to the accomplishment of meaningful tasks.

Contextualizing learning requires more support than *tutoring systems* or *open learning environments* can provide. The strength of tutoring systems (Anderson, Reiser, 1985; Psotka, Massey, Mutter, 1988; Polson, Richardson, 1988; Wenger, 1987) lies in their ability to teach basic concepts and skills of a problem domain. However, they cannot be designed to closely match the concrete problem solving situations of users. Problems

[1]This research was partially supported by grants No. DCR-8420944 and No. IRI-8722792 from the National Science Foundation, grant No. MDA903-86-C0143 from the Army Research Institute, and grants from the Intelligent Interfaces Group at NYNEX and from Software Research Associates (SRA), Tokyo.

presented by tutoring systems are often well-defined and prespecified by the system rather than ill-structured and arising out of real-world contingencies. With tutoring systems, learners are left to acquire task application knowledge on the job, after the completion of formal training. It is left to the learner to relate such training to real-world problem situations.

Open learning environments (e.g., LOGO (Papert, 1980)) do not suffer from the problem that presentation of instructional material is system-controlled without regard to the learner's situation, but these systems provide limited support in helping learners detect mistakes or overcome breakdowns. Misconceptions may accumulate into chains in which each later misconception is based on a previous one. Learners get trapped on suboptimal plateaus because they fail to discover the knowledge needed for better design.

Contextualized learning has the potential to overcome these shortcomings. Since the learning context is triggered by the actual working context, applicability of the knowledge is clear. Elaboration of new knowledge and inferential recall is facilitated by the familiarity of the working situation. Learning in work situations fosters efficient impasse-driven learning (VanLehn, 1988). Computer-based contextualized learning environments could tailor instruction to the special learning needs of different users and could monitor users' success in applying knowledge.

A Conceptual Framework for Contextualized Learning

Our understanding of contextualized learning is based on an analysis of learning as situated and of seeing reflection-in-action as a crucial process in design.

Situated Learning. Situating learning requires integrating learning into situations for which that learning is useful. Instructional information needs to be related to the specific tasks which it serves and doing so while those tasks are being undertaken. General concepts and principles have to be mapped onto specific instances.

A growing number of researchers (Lave, 1988; Schoen, 1983; Schoen, 1987; Suchman, 1987; Winograd, Flores, 1986) argue that no amount of knowledge of principles suffices to account for successful action in real-world problem situations. An additional skill is needed: *contextual elaboration*, i.e., the ability to go beyond general procedural prescriptions to devise specific courses of action and to make intelligent exceptions to principles. Learning is not truly situated unless the learner can perform such contextual elaboration.

Reflection-in-Action. Underlying the *reflection-in-action* approach (Schoen, 1983) is an "action-breakdown-repair model" of the relationship between action and rational thought. Design is seen as repeated alternation between situated action and reflection. The former is governed by a non-reflective thought process and proceeds until it breaks down. A breakdown occurs when the designer realizes that non-reflective action has resulted in unanticipated consequences — either good or bad. Schoen describes this feedback as "the situation talking back." Reflection is used to repair the breakdown, and then (non-reflective) situated action continues. Reflection-in-action is different from pre-planning and post-mortem analysis because it takes place within the *action present*, i.e., the time period during which the decision to act has been made but the final decision about how to act has not. This is the period during which reflection can still make a difference to what action is taken. Schoen gives extensive arguments and examples that it is during reflection-in-action that instructional information can best be presented.

Integrating Construction and Argumentation. The concepts of situated learning and reflection-in-action need to be further operationalized and augmented if they are to provide a basis for a system architecture. We have done this conceptual elaboration in conjunction with the building and testing of a prototype which provided an "object-to-think-with." These efforts led us to interpret action as construction and reflection as argumentation. Construction is the process of shaping the solution — e.g., manipulating the form of a building. Argumentation is the reasoning about the problem and its solution.

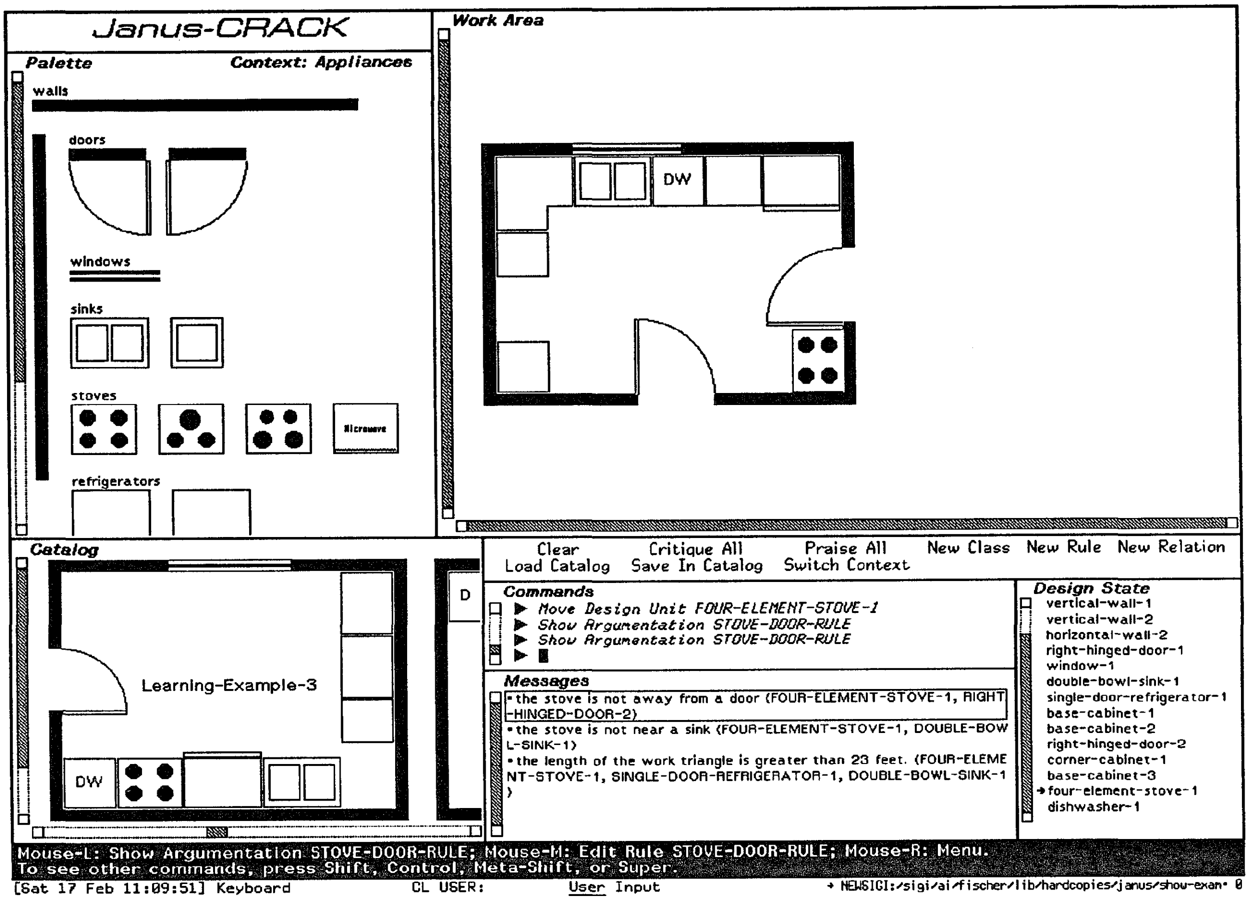
The problem of contextualizing design learning thus becomes one of integrating construction and argumentation. In particular, the system must help designers to do the following:

- to see where their construction knowledge is inadequate (to perceive breakdowns, to "let the situation talk back");
- to find the argumentative knowledge they need for such situations (ideally, all the knowledge and only the knowledge needed for the tasks at hand);
- to understand how generalized principles of design relate to their particular construction situations;
- to understand how to perform the contextual elaboration needed to go beyond principles — i.e., to make intelligent exceptions and perform detailed situated construction.

A Demonstration System: JANUS

JANUS (Fischer, McCall, Morch, 1989) is a integrated design environment comprising two subsystems: JANUS-CRACK (see Figure 1), a knowledge-based construction component and JANUS-VIEWPOINTS (see Figure 2), a hypermedia argumentation component. The former consists of a domain-oriented construction kit, a set of rule-based critics that "judge" the object under construction, and a catalog of completed designs. The domain of JANUS is architectural design, specifically kitchen design. The critics (Fischer et al., 1990) in JANUS-CRACK evaluate design decisions and provide feedback when design principles are violated.

JANUS supports the construction of an artifact either "from scratch" by combining the primitives from the palette or by modifying an already constructed artifact from its catalog of previously designed kitchens. The designer can browse through this catalog until an interesting design is found. This can then be brought into the



```
▶ Move Design Unit FOUR-ELEMENT-STOVE-1
▶ Show Argumentation STOVE-DOOR-RULE
▶ Show Argumentation STOVE-DOOR-RULE
▶ █
```

```
•the stove is not away from a door (FOUR-ELEMENT-STOVE-1, RIGHT
-HINGED-DOOR-2)
•the stove is not near a sink (FOUR-ELEMENT-STOVE-1, DOUBLE-BOW
L-SINK-1)
•the length of the work triangle is greater than 23 feet. (FOUR-ELEME
NT-STOVE-1, SINGLE-DOOR-REFRIGERATOR-1, DOUBLE-BOWL-SINK-1
)
```

```
 vertical-wall-1
 vertical-wall-2
 horizontal-wall-2
 right-hinged-door-1
 window-1
 double-bowl-sink-1
 single-door-refrigerator-1
 base-cabinet-1
 base-cabinet-2
 right-hinged-door-2
 corner-cabinet-1
 base-cabinet-3
→four-element-stove-1
 dishwasher-1
```

```
Mouse-L: Show Argumentation STOVE-DOOR-RULE; Mouse-M: Edit Rule STOVE-DOOR-RULE; Mouse-R: Menu.
To see other commands, press Shift, Control, Meta-Shift, or Super.
[Sat 17 Feb 11:09:51] Keyboard          CL USER:          User Input          → NEWSIGI:/sigi/ai/fischer/lib/hardcopies/janus/show-exam• 0
```

Figure 1: JANUS-CRACK

This screen image shows JANUS-CRACK, the construction component of JANUS. Building blocks (design units) are selected from the *Palette* and moved to desired locations inside the *Work Area.* Designers can reuse and redesign complete floor plans from the *Catalog*. The *Messages* pane displays critic messages automatically after each design change that triggers a critic. Clicking with the mouse on a message activates JANUS-VIEWPOINTS and displays the argumentation related to that message.

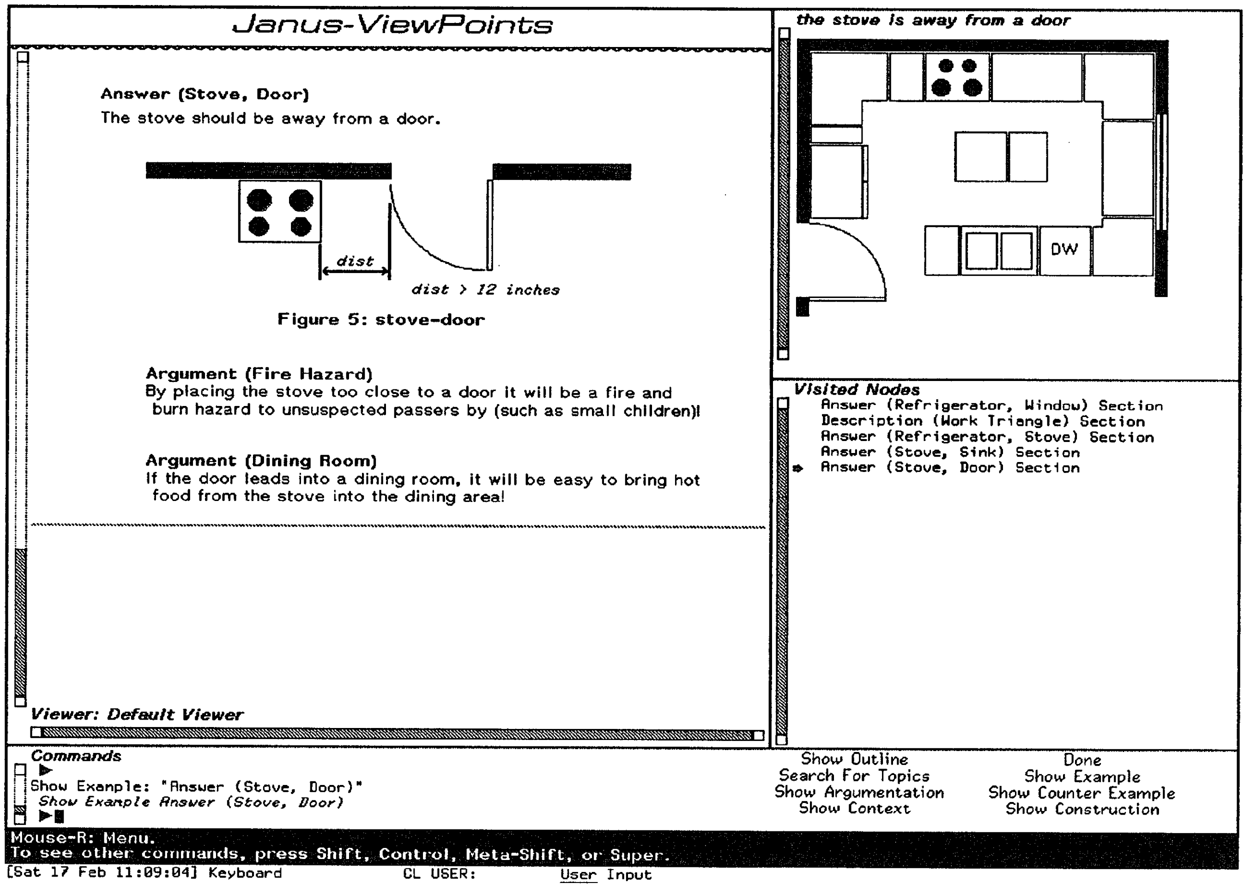
work area and modified to the designer's liking.

JANUS-VIEWPOINTS (see Figure 2) is an issue-based hypermedia system (McCall, 1987) containing answers, arguments and subissues. Its arguments use design knowledge ranging from building codes describing safety rules to work flow rules such as the work triangle rule. The system allows designers to learn about what issues have to be resolved, about possible answers to these issues, and why and when these answers are appropriate.

The knowledge-based critiquing mechanism is the main means for bridging the gap between construction and argumentation. This means that JANUS-CRACK and JANUS-VIEWPOINTS are coupled by using JANUS's critics to provide the designer with immediate entry into the exact place in the hypermedia network where the argumentation relevant to the current construction task lies. This solves two problems: It greatly enriches the argumentative information for construction, it allows hypermedia-based argumentation to be used during construction and it contextualizes learning by exposing designers to directly relevant knowledge.

The catalog contains both positive and negative "learning examples" (see Figure 1). The positive examples in the catalog can be used to learn design principles and explore their argumentative background by bringing them into the work area and applying the *Praise All* command to them. This results in the critics pointing out design principles that are *not* violated. The negative examples violate design principles embedded in JANUS. After such an example is brought into the work area, critics will fire providing the learner a list of the violated design principles. Both praise and criticism provide the users with entry points into the hypermedia system, where the argumentative background of the principles can be explored by navigation.

The integrated system supports contextualized learning by providing argumentative information for construction effectively, efficiently and without designers' having to (1) realize they need information, (2) suspect that needed in-



Answer (Stove, Door)
The stove should be away from a door.

Figure 5: stove-door

Argument (Fire Hazard)
By placing the stove too close to a door it will be a fire and burn hazard to unsuspected passers by (such as small children)!

Argument (Dining Room)
If the door leads into a dining room, it will be easy to bring hot food from the stove into the dining area!

Visited Nodes
Answer (Refrigerator, Window) Section
Description (Work Triangle) Section
Answer (Refrigerator, Stove) Section
Answer (Stove, Sink) Section
➡ Answer (Stove, Door) Section

```
Commands
▶
Show Example: "Answer (Stove, Door)"
 Show Example Answer (Stove, Door)
▶■
Mouse-R: Menu.
To see other commands, press Shift, Control, Meta-Shift, or Super.
[Sat 17 Feb 11:09:04] Keyboard          CL USER:        User Input
```

Figure 2: JANUS-VIEWPOINTS

This screen image of JANUS-VIEWPOINTS shows an answer to the issue of where to locate the kitchen stove with respect to doors and graphically indicates the desirable relative positions of the two design units. Below this is a list of arguments for and against the answer. The example in the upper right corner (activated by the "Show Example" command in the "Commands" pane) contextualizes an argumentative principle in the context of a specific design (retrieved by the system from the catalog).

formation is in the system or (3) know how to retrieve it. The system also contextualizes the general principles contained in the hypermedia system by dynamically showing examples illustrating these principles in the context of specific designs.

A System Architecture to Contextualize Learning

The system building effort around JANUS served as a starting point for a generalized architecture for design environments in support of contextualized learning. The major components of such an architecture are (see Figure 3): a construction kit, argumentative hypermedia, critics, and a catalog.

Construction Kit. The construction kit provides a palette of domain abstractions for construction of artifacts. The domain abstractions preserve the situatedness of work in the problem domain. Designers can think about problem situations without the distraction of having to think about communicating with the computer.

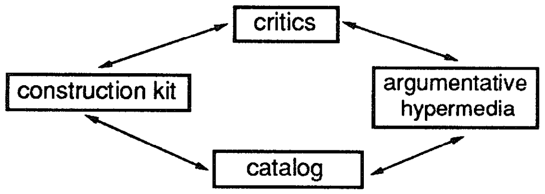


Figure 3: A System Architecture Supporting Contextualized Learning

Critics. Successful design requires that the *situation talks back* (Schoen, 1983). However, for designers who do not have extensive experience in the domain, the situation is often mute unless the learning environment has a component that speaks up and points out issues that the designer may otherwise not consider. Critics can fulfill this role. Critics point out suboptimal aspects of the artifact and retrieve relevant issues in the issue base. Critics thus contextualize learning in two ways: by identifying learning needs through real-time analysis of actual construction situations and by presenting just the argumentative information useful for those needs.

Argumentative Hypermedia. Contextualized learning can only be effective if the flow of work is not disrupted. This requires rapid and timely access to relevant information in the issue base, a collection of argumentation on recurring issues in the problem domain. Issue-based hypermedia (McCall, 1987) make designers aware of issues, possible answers, and argumentation, and allows them to incrementally specify personal design constraints at various levels.

The PHI (Procedural Hierarchy of Issues) approach to issue-based hypermedia structures information according to its relevance to design tasks (issues). It contains principles of design together with the argumentative background necessary for contextual elaboration. The rich connectivity of hypermedia allows immediate access to explanatory, elaborative and other types of information. Hypermedia also accommodates individual differences in prior knowledge and learning styles — crucial contextual factors. Users with little knowledge can explore the hypermedia network in depth, while knowledgeable users can ignore links to known information.

Catalog. The catalog is an evolving collection of artifacts accumulated by the actual use of the system. It illustrates the space of possible designs in the domain. For instructional purposes, the catalog can be enriched with positive and negative learning examples (see Figure 1). The catalog examples are contextualized that the system dynamically selects and presents those examples that not only illustrate the concept but most closely correspond to the construction situation at hand (see Figure 2). Catalog examples provide a link back from argumentation to construction by making abstract principles concrete and ready to be integrated into the artifact under construction. Catalog examples support *case-based reasoning* (Riesbeck, Schank, 1989; Rissland, Skalak, 1989) to complement the generalized argumentative reasoning in the issue base. To this end, catalog examples are annotated with design rationale that allows designers to assess the relevance of the example to the situation at hand. This allows them to provide examples for how to make intelligent exceptions to design principles and to do other types of contextual elaboration — which by definition go beyond what can be stated in principles. The user can independently analyze catalog examples with the critics to discover design principles in the context of specific construction situations.

Evaluation, Limitations, and Future Work

Evaluation. Our evaluation using professional and amateur designers showed that contextualized learning can be supported by JANUS. Users extend their knowledge in a demand-driven way and breakdowns in the systems were perceived as learning opportunities. Users were able to understand the purposes and uses of the new knowledge encountered. The different components of the architecture enabled users to learn new knowledge in multiple contexts: tied to the contexts of its uses and independent of any particular context.

Limitations. Our evaluation also showed several shortcomings of the current system. One is that JANUS allows views of the artifact only at the individual room level and not at a higher or lower level of aggregation. A second is that it does not support functional simulation. This is a technique frequently used by designers to understand the consequences of design decisions — increasing the possibilities of *letting the situation talk back* — by simulating use of the design. A third is that JANUS does not allow designers to volunteer their goals, preferences, and specifications (Fischer, Stevens, 1987) and therefore fails to realize the full potential of critics and the argumentation component to give advice tailored to the actual problem situation.

Extended Architecture. In order to overcome these limitations we are in the process of developing an extended architecture containing the following components:

- *Specification component.* Design is a process of trading off competing goals such as minimizing cost or maximizing reliability or extensibility, and the final artifact depends on which tradeoffs are acceptable. Without knowing and adapting to such characteristics of the specific problem situation, the system's appearance will remain abstract. A specification component would allow the designer to input these characteristics, and the system can use them to better situate its information structures by filtering out argumentation, critics, and catalog examples that are not relevant to the specified problem situation.
- *Simulation component.* Evaluations of the JANUS system by expert designers have demonstrated a need for simulating usage scenarios with the artifact being designed. Such functional simulations can take the form of deterministic mathematical models as well as informal what-if games. Functional simulation enhances further the capability of the construction situation to talk back to the designer.
- *Dynamic Recomputation of Hypermedia Networks.* To situate the issue and argumentation structure, the issue base must be *active*, that is, it must filter out information that is irrelevant to the current construction situation, even if that information is relevant for other hypothetical situations. The situated information structure then is much more manageable. Concretely, an issue base can be better situated by suppressing issues that are made irrelevant by answers to other issues, and, secondly, by inferring answers from information the designer has previously specified and from design decisions that have already been made.

New Application Domains. In order to test the generality of our architecture, we will test this approach with other application domains. Our previous work has concentrated on the domain of kitchen design. A first test of the generality was conducted by designing and implementing with the same basic architecture a system for user interface design (Lemke, 1989). We are currently in the planning stage to apply the approach to more general architectural design issues (e.g., design of buildings) *in combination* with a technical domain (e.g., the design of high-speed digital communication networks within these buildings).

Future Issues for Investigation. With these new features added to our design environments, we will be able to investigate issues such as the following: (1) Will there be substantial differences in performance if the system is used with and without critics, catalog, and simulation component? (2) What are the most important components in our architecture that enable situated learning? How should these components be structured? (3) What intervention strategies should the system use for displaying enough information at the right time without disrupting the work process? (4) When are designers willing to suspend the construction process to access relevant information? (5) When will designers challenge or extend the knowledge represented in the system? and (6) When are subjects willing to enhance the knowledge structures (Fischer, Girgensohn, 1990)?

Conclusions

In order for designers to cope with the deluge of knowledge for design they must integrate learning into their everyday work. With the described research, we have created an initial conceptual framework for such contextualized learning. This framework was illustrated with a prototype system utilizing argumentative hypermedia and knowledge-based critics. From this a preliminary architecture for integrated knowledge-based design environments was developed. Evaluation has shown the benefits and limitations of this approach and, in particular, showed that additional components are needed to contextualize learning more fully. In our future research we will develop and test these new components. Our goal is to develop systems which become means for lifelong learning. Ultimately, such environments for contextualized learning might allow learning to become a more personal, less alienating process not separate from the rest of our life.

Acknowledgements. The authors would like to thank Andreas Girgensohn, Anders Morch, Kumiyo Nakakoji, and Jonathan Ostwald who have made major contributions to the JANUS system.

References

J.R. Anderson, B.J. Reiser (1985). The LISP Tutor. *BYTE, 10*(4), 159-175.

J.M. Carroll, M.B. Rosson (1987). Paradox of the Active User: In J.M. Carroll (Ed.), *Interfacing Thought: Cognitive Aspects of Human-Computer Interaction* (pp. 80-111). Cambridge, MA: The MIT Press.

A.M. Collins, J.S. Brown, S.E. Newman (1989). Cognitive Apprenticeship: Teaching the Crafts of Reading, Writing, and Mathematics: In L.B. Resnick (Ed.), *Knowing, Learning, and Instruction* (pp. 453-494). Hillsdale, NJ: Lawrence Erlbaum Associates.

S.W. Draper (1984). The Nature of Expertise in UNIX. *Proceedings of INTERACT'84, IFIP Conference on Human-Computer Interaction*, 182-186. Amsterdam: Elsevier Science Publishers.

G. Fischer (1988). Enhancing Incremental Learning Processes with Knowledge-Based Systems: In H. Mandl, A. Lesgold (Eds.), *Learning Issues for Intelligent Tutoring Systems* (pp. 138-163). New York: Springer-Verlag.

G. Fischer, A.C. Lemke, T. Mastaglio, A. Morch (1990). Using Critics to Empower Users. *Human Factors in Computing Systems, CHI'90 Conference Proceedings (Seattle, WA)*, 337-347. New York: ACM.

G. Fischer, A. Girgensohn (1990). End-User Modifiability in Design Environments. *Human Factors in Computing Systems, CHI'90 Conference Proceedings (Seattle, WA)*, 183-191. New York: ACM.

G. Fischer, R. McCall, A. Morch (1989). JANUS: Integrating Hypertext with a Knowledge-Based Design Environment. *Proceedings of Hypertext'89*, 105-117. New York: ACM.

G. Fischer, C. Stevens (1987). Volunteering Information -- Enhancing the Communication Capabilities of Knowledge-Based Systems. *Proceedings of INTERACT'87, 2nd IFIP Conference on Human-Computer Interaction (Stuttgart, FRG)*, 965-971. Amsterdam: North-Holland.

J. Lave (1988). *Cognition in Practice*. Cambridge, MA: Cambridge University Press.

A.C. Lemke (1989). *Design Environments for High-Functionality Computer Systems*. Unpublished doctoral dissertation, Boulder, CO: Department of Computer Science, University of Colorado.

R. McCall (1987). PHIBIS: Procedurally Hierarchical Issue-Based Information Systems. *Proceedings of the Conference on Architecture at the International Congress on Planning and Design Theory*. New York: American Society of Mechanical Engineers.

S. Papert (1980). *Mindstorms: Children, Computers and Powerful Ideas*. New York: Basic Books.

M.C. Polson, J.J. Richardson (Eds.). (1988). *Foundations of Intelligent Tutoring Systems*. Hillsdale, NJ: Lawrence Erlbaum Associates.

J. Psotka, L.D. Massey, S.A. Mutter (Eds.). (1988). *Intelligent Tutoring Systems: Lessons Learned*. Hillsdale, NJ: Lawrence Erlbaum Associates.

C. Riesbeck, R.C. Schank (1989). *Inside Case-Based Reasoning*. Hillsdale, NJ: Lawrence Erlbaum Associates.

E.L. Rissland, D.B. Skalak (1989). Combining Case-Based and Rule-Based Reasoning: A Heuristic Approach. *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, 524-530.

D.A. Schoen (1983). *The Reflective Practitioner: How Professionals Think in Action*. New York: Basic Books.

D.A. Schoen (1987). *Educating the Reflective Practitioner*. San Francisco, CA: Jossey-Bass Publishers.

L.A. Suchman (1987). *Plans and Situated Actions*. New York: Cambridge University Press.

K. VanLehn (1988). Toward a Theory of Impasse-Driven Learning: In H. Mandl, A. Lesgold (Eds.), *Learning Issues for Intelligent Tutoring Systems* (pp. 19-41). New York: Springer-Verlag.

E. Wenger (1987). *Artificial Intelligence and Tutoring Systems*. Los Altos, CA: Morgan Kaufmann Publishers.

T. Winograd, F. Flores (1986). *Understanding Computers and Cognition: A New Foundation for Design*. Norwood, NJ: Ablex Publishing Corporation.