JANUS: Integrating Hypertext with a Knowledge-based Design Environment

Gerhard Fischer*
Raymond McCall*
Anders Morch**

*University of Colorado, Boulder
Boulder, Colorado 80309

**Intelligent Interfaces Group
NYNEX Science and Technology
White Plains, NY 10604

ABSTRACT

Hypertext systems and other complex information stores offer little or no guidance in helping users find information useful for activities they are currently engaged in. Most users are not interested in exploring hypertext information spaces *per se* but rather in obtaining information to solve problems or accomplish tasks. As a step towards this we have developed the JANUS design environment. JANUS allows designers to construct artifacts in the domain of architectural design and at the same time to be informed about principles of design and the reasoning underlying them. This process integrates two design activities: construction and argumentation. Construction is supported by a knowledge-based graphical design environment and argumentation is supported by a hypertext system. Our empirical evaluations of JANUS and its predecessors has shown that integrated support for construction and argumentation is necessary for full support of design.

KEYWORDS

Hypertext, knowledge-based systems, construction, argumentation, informed design, human problem-domain communication, construction kits, design environments, issue-based information systems (IBIS), procedural hierarchy of issues (PHI) methodology.

INTRODUCTION

We have worked for a number of years on the development of computer systems, including hypertext systems, to support designers in a wide range of fields, including architecture and software design. Hypertext systems and other complex information stores generally offer little or no help to designers in finding information useful for activities they are currently engaged in. This is a major deficiency, because designers are not interested in exploring hypertext information spaces *per se* but rather in obtaining information to solve problems or accomplish tasks they are dealing with. As a step towards correcting this deficiency we have developed the JANUS design environment. JANUS allows architectural designers to graphically construct artifacts by direct manipulation and at the same time to receive information useful to what they are doing from hypertext activated by knowledge-based agents. The information they receive

concerns principles of design and the reasoning underlying them. This work has implications for a number of issues in hypertext research, including what types of links there should be, what style of user interface should be used, the "lost in hyperspace" problem, retrieval of information without browsing [Aksc88], as well as integration of hypermedia with AI technology [Hala88].

The Information Problem in Design

Designers in a wide range of fields--including architectural, industrial, engineering and software design--need better access to information. They seldom have in their heads all the knowledge needed for good design. Even expert designers can no longer master all the relevant knowledge, especially in technologically oriented design, where growth and change of the knowledge base are incessant. Designers typically cannot keep up with developments in their own fields much less in other fields of potential relevance.

While the quantity and complexity of design information continually increases, designers' abilities to locate and manage useful information does not. The large and growing discrepancy between the rates of information generation and use puts a limit on progress in design. Overcoming this limit is a central challenge for creators of design information systems and thus an important topic for hypertext research.

Conventional information systems offer little hope of providing designers with adequate access to information. Retrieval in such systems is based on content search, which typically means keyword and free-text search. This approach only becomes useful if designers know three things that they very often do not know:

1) that they need information, i.e., that their knowledge is inadequate for certain tasks,
2) that the needed information is likely to exist and be in the information system, and
3) how to retrieve it, e.g., what keywords to use.

Conventional retrieval also has problems in effectiveness and efficiency. Its effectiveness is limited by a seemingly unavoidable tradeoff between the two standard measures of effectiveness: precision and recall. The former denotes the fraction of retrieved items which are relevant; the latter, the fraction of relevant items in the information base which are retrieved. No conventional approach can come close to finding all those items and only those which are relevant. Typically, users must wade through large amounts of retrieved items to find small amounts which are useful. The inefficiency of conventional retrieval is problematic for designers because of both the effort and the time involved. These interfere with design by creating distractions and interruptions.

JANUS substantially reduces these problems for design by integrating a knowledge-based system with an issue-based hypertext system. In this paper we describe JANUS using examples of its application to the domain of kitchen design. We begin by explaining the two independent lines of research which led to the creation of JANUS. The genealogy of JANUS is shown in Figure 1.

One line of research pursued the development of issue-based hypertext to support the design activity of argumentation; the other pursued the development of knowledge-based design environments to support the design activity of construction. Argumentation is the activity of reasoning about a design problem and its solution. It is predominantly verbal but can be partly graphic. Construction is the activity of creating the solution's form. In many design fields, such as architecture and industrial design, construction is a graphic activity and is traditionally done by drawing.

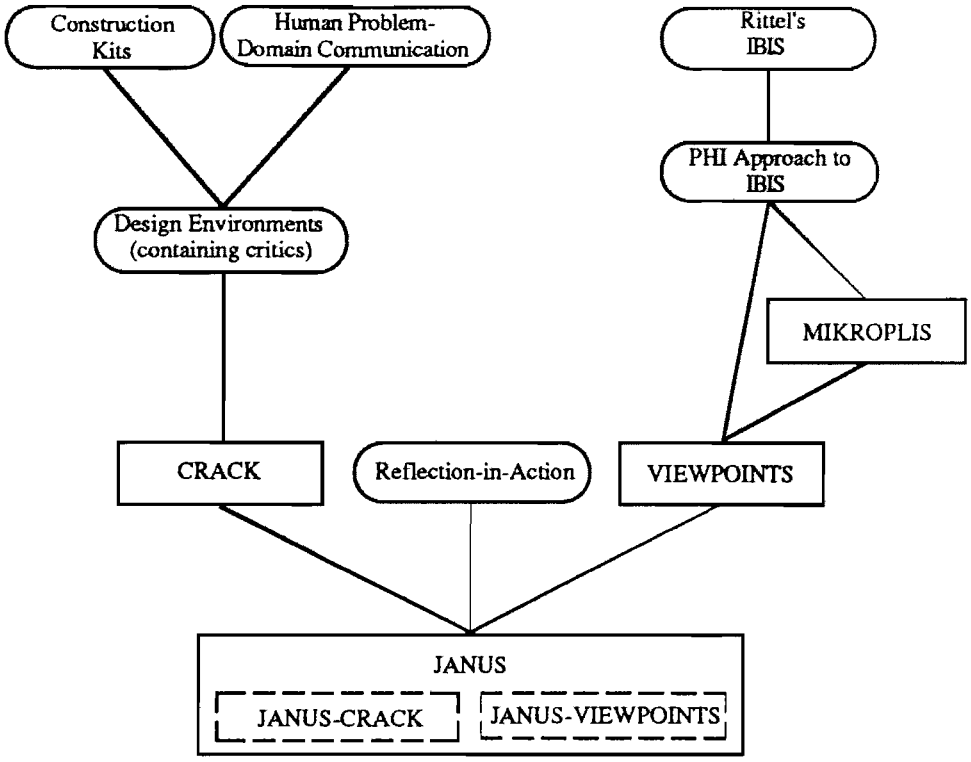

FIGURE 1. Conceptual frameworks (in ovals) and system-building efforts (in rectangles) leading to JANUS.

ISSUE-BASED HYPERTEXT SUPPORTING ARGUMENTATION

IBIS and PHI

Research aimed at developing systems supporting designers' argumentation activities was originated by Rittel [Ritt72]. His aim in doing so was to improve design by improving the reasoning on which it is based. This work ultimately inspired the development of hypertext systems such as gIBIS [Conk87][Conk88] and the ones described in this article.

The central concept behind Rittel's work was that of Issue-Based Information Systems (IBIS) [Kunz70]. In an IBIS, information is organized around the discussion of questions, referred to as *issues*. The method of discussion is deliberation, i.e., considering arguments for and against various proposed answers to the issues. The issues in an IBIS are connected to each other by a variety of relationships such as *is more general than*, *leads to*, and *is similar to*.

The Procedural Hierarchy of Issues (PHI) approach [McCa87] extends IBIS by broadening the scope of the concept *issue* and by altering the structure relating issues, answers and arguments. In Rittel's IBIS, an issue is a design question which can be deliberated. In PHI, an issue is *any* design question, whether deliberated or not. PHI replaces the various inter-issue relationships in Rittel's IBIS with *serves* relationships, defined as follows:

> Issue A serves issue B if the answers to A influence what answers are given to B, i.e., if the answering of B depends in some way on the answering of A.

The main serve relationship in PHI is the *subissue of* relationship, defined as follows:

Issue A is a subissue of issue B if A serves B and B is raised before A.

Unlike the original IBIS, PHI does not rely on deliberation alone for resolving issues. A second crucial process is the recursive decomposition of issues into subissues. This characteristically results in a tree-like structure of an issue with subissues, such as that shown in Figure 2. PHI also allows hierarchies of answers with subanswers and arguments with subarguments.

ISSUE 101:
What should the arrangement of the various components of the kitchen be?
SUBISSUES:
102: What are the various components of the kitchen that need to be arranged?
103: What should the locations of these various components be?
SUBISSUES:
104: What should the locations of the various architectural features be?
SUBISSUES:
105: What are the various architectural features?
106: What should the location of the walls be?
107: What should the location of the doors be?
108: What should the location of the windows be?
109: What should the location of the plumbing be?
110: What should the location of the equipment area be?
SUBISSUES:
111: What is the longest usable wall length?
112: What should the location of the eating area be?
113: What should the circulation pattern be?
114: What should the locations of the various components of the equipment area be?
SUBISSUES:
115: What are the components of the equipment area that need to be arranged?
116: What should the location of the cleanup center be?
117: What should the location of the cooking center be?
130: What should the location of the storage center be?
118: What should the location of the preparation center be?
119: What should the locations of the various components of the cleanup center be?
123: What should the locations of the various components of the cooking center be?
127: What should the locations of the various components of.the storage center be?
131:What should the locations of the various components of the preparation center be?

FIGURE 2. A Hierarchy of issues with subissues. This hierarchy was created with the MIKROPLIS hypertext system. The numbering of issues is by the order in which they were proposed.

An advantage of PHI over the original IBIS is that it more completely and accurately models the task structure of the design process and the information useful for tasks. According to the theory on which PHI is based, every task in a design project can be modeled as an attempt to resolve an issue [McCa86]. The information useful for design tasks can then be found by following the links connecting issues to their deliberative discussions--i.e., answers, subanswers, arguments and subarguments--and to the issues which serve them.

The above-mentioned problems of information retrieval can be overcome for some but not all design tasks by structuring hyperdocuments according to the principles of PHI. The link structure of a PHI hyperdocument allows designers to retrieve information useful for current tasks effectively and efficiently by means of navigation. The link labels in screen displays also tell designers

1) of the existence of information useful for current design tasks,

2) the basic nature of this information and

3) how to retrieve it (by clicking on it).

PHI Hypertext

We have developed two stand-alone PHI hypertext systems. One is MIKROPLIS [McCa83]; the other is VIEWPOINTS [Fisc89]. MIKROPLIS is a text-only system which superficially resembles an outline processor but can handle graphs with thousands of nodes. At the nodes are texts of essentially arbitrary length. The contents of clusters of linked nodes are displayed in outline format (shown in Figure 2). Retrieval is by navigation and/or use of an English-like applicative query language providing both structure search and content search [Hala88]. This allows retrieval of complex substructures of the hypertext network using numeric identifiers, indexed terms or substrings in combination with descriptions of network structure. VIEWPOINTS is implemented in HyperCard on the Macintosh. It has fewer retrieval capabilities than MIKROPLIS but can store graphics as well as texts at nodes. A screen image from VIEWPOINTS is shown in Figure 3.

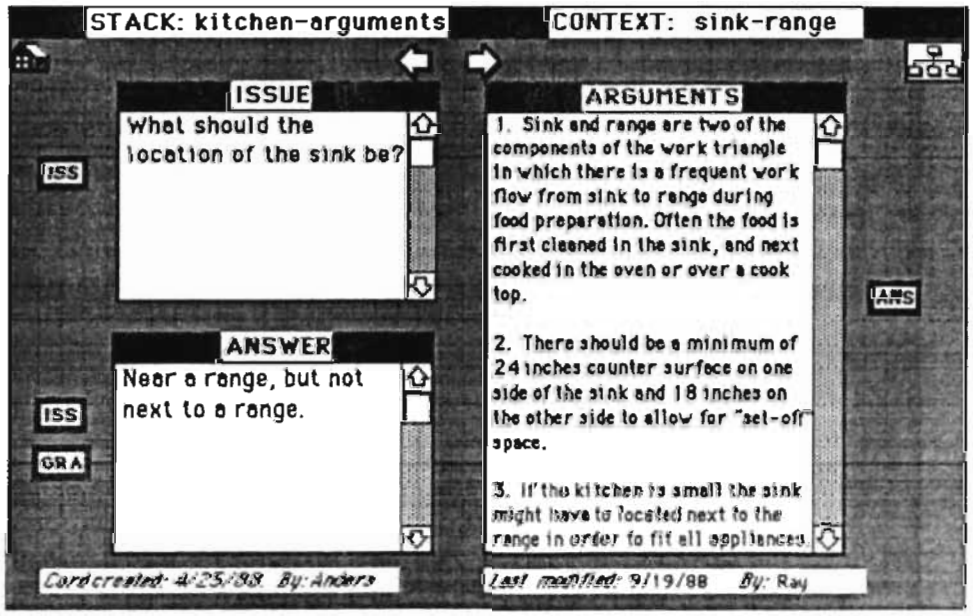

FIGURE 3. A screen image from the VIEWPOINTS hypertext system.

KNOWLEDGE-BASED DESIGN ENVIRONMENTS SUPPORTING CONSTRUCTION

Human Problem-Domain Communication

To shape the computer into a truly usable and useful tool, users must be able to work directly on their problems and tasks. The goal of human problem-domain communication [Fisc88a] is to eliminate computer-specific programming languages and instead to build layers of abstractions with which domain specialists--such as kitchen

designers--can feel comfortable. Human problem-domain communication provides a new level of quality in human-computer communication because the important abstract operations and objects in a given area are built directly into the computing environment. The user can therefore operate with personally meaningful abstractions which reduce the transformation distance between problem-oriented and system-oriented descriptions.

CRACK (Critiquing Approach to Cooperative Kitchen Design) [Fisc88b] is a knowledge-based design environment which represents a step toward human problem-domain communication in architectural design. A screen image of CRACK is shown in Figure 4.

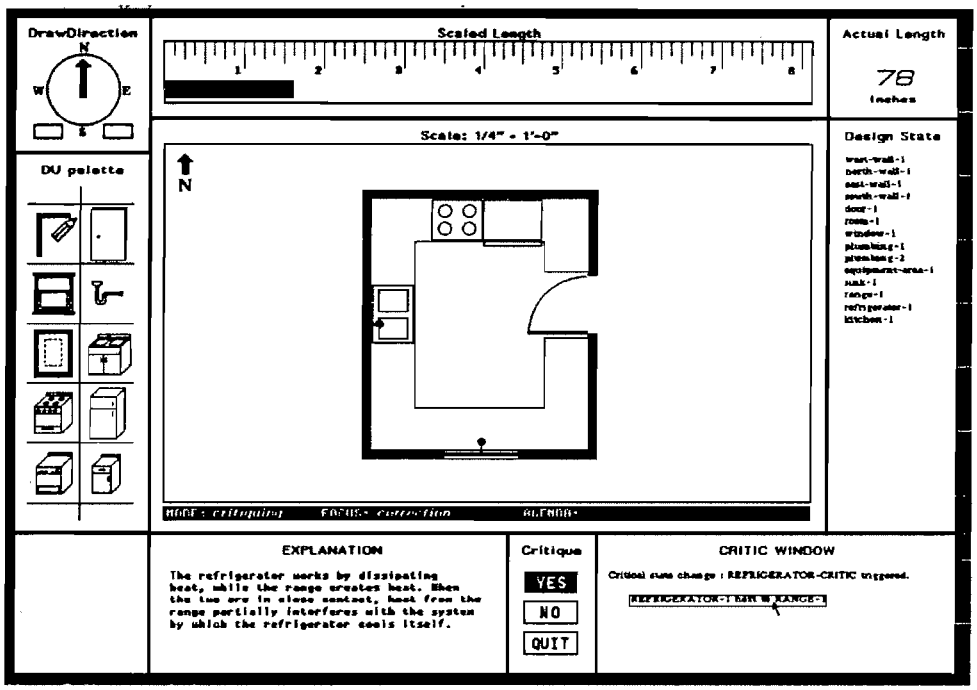

FIGURE 4. A Screen image from CRACK.

Construction Kits

Construction kits are system components that represent steps toward human problem-domain communication by providing a set of building blocks that model a problem domain. The building blocks in CRACK, called design units (DUs), define a design space (the set of all possible designs that can be created by combining them using the "language of kitchen design") and a design vocabulary (represented by the "Palette", on the left in Figure 4). Construction kits are domain-specific programming languages that help users to formulate solutions to complex problems and that allow them to create complex artifacts without having to master the many details inherent in general purpose programming languages. They eliminate a number of prerequisite skills, thus allowing users much more time to practice and work in their actual areas of interest and expertise.

Beyond Construction Kits: Critics and Design Environments

Human problem-domain communication and construction kits are necessary but not sufficient for good design. Upon evaluating prototypical construction kits we found that they do not by themselves assist the user in constructing interesting and useful artifacts in the application domain. To do this they would need knowledge for distinguishing "good" designs from "bad" designs together with explanations which

justify this knowledge. Kitchen design, for example, involves more than selecting appliances from a palette; it involves knowing how to combine these simple building blocks into functional kitchens. Knowledge about kitchen design includes design principles based on building codes, safety standards, functional preferences and other considerations.

Design environments, such as CRACK, combine construction kits with critics. Critics use knowledge of design principles to detect and critique partial solutions constructed by the designer. Critics in CRACK are state-driven condition-action rules which are triggered when non-satisficing partial solutions are detected. The critics display criticism (such as, "Refrigerator next to range") in the "Critic Window" (lower right corner in Figure 4). If designers want more information, they can make requests for explanations (lower left corner in Figure 4) and suggestions by clicking with the mouse on the text of the criticism. The critics create the text displays dynamically and in the proper context as users are constructing the design.

LIMITATIONS OF THE TWO TYPES OF SYSTEMS

The above-described systems have been tested in use long enough for their basic capabilities and limitations to become apparent. MIKROPLIS and VIEWPOINTS have been used to create issue-bases for a variety of problem domains, including software, architectural and urban design as well as environmental and health care policy making. CRACK has been tested in use by both student and professional designers.

A clear lesson which has emerged from the use of MIKROPLIS and VIEWPOINTS is that PHI hypertext alone cannot fully realize the goal of informing designers while they design.

PHI hypertext works as long as designers are actually working on resolving issues with the hypertext system. While working in this predominantly verbal mode, the designer is automatically shown the information known to be relevant to the issue being worked on. Unfortunately, there is a crucial type of design activity which cannot effectively be handled in PHI hypertext: construction.

Our students consistently proved unable to do construction within PHI itself. But, once they left PHI hypertext to work on construction, the hypertext exhibited the characteristic weakness of conventional retrieval systems--i.e., designers got information from it only if they knew to ask for such information and knew how to ask correctly. The PHI link structure could no longer automatically alert them to the existence of information useful for current tasks. Thus, during construction PHI hypertext failed to achieve its goal of adequately informing designers.

The inability to do construction in PHI even occurred when students did PHI without use of computers, thus it is not an artifact of the hypertext implementations. This result was surprising at first, since video-taped protocols revealed hand-drawn construction to involve continual deliberation and decomposition of issues. The explanation for this apparent paradox is that PHI's argumentation is *about* construction, *not the same as* construction. Construction and argumentation are distinct modes of design. Both are required for good design, but they cannot be done simultaneously. This conclusion nicely fits Schoen's theory of design as reflection-in-action [Schoe83], with his concepts of reflection and action corresponding, respectively, to argumentation and construction.

Tests of CRACK revealed that it also did not go far enough toward achieving its goal, i.e., the goal of informing its users about construction. CRACK's rules for kitchen design were not intended to be inflexible commandments, but rather rules of thumb to which exceptions can and should be made. While users were in principle free to ignore CRACK's criticism, in practice they had difficulty judging when and how to do so.

CRACK's brief "explanations" failed to reveal the complex argumentative background--including assumptions, conditions and controversies--behind its critiques. Without this background it was difficult for users to make the intelligent exceptions characteristic of good design.

Evaluations of these systems together with studies of designers by ourselves and others show that integrated support for construction and argumentation is essential for full support of design. Schoen's theory of reflection-in-action states that constant alternation between action and reflection, i.e., between construction and argumentation, is the hallmark of good design. The shift to argumentation occurs when designers need to evaluate what has been constructed, are stuck in problematic situations or need to think about what to do next. The critiques given by CRACK allow this but in far too limited a way. Some way is needed of coupling systems like CRACK and VIEWPOINTS to overcome their deficiencies. JANUS provides such a coupling.

JANUS: AN INTEGRATED DESIGN ENVIRONMENT

The JANUS Concept

JANUS' concept for integrating CRACK and VIEWPOINTS derived from the observation that CRACK's criticism is really a limited type of argumentation and can be mapped directly into PHI form. In particular, the construction actions can be seen as attempts to resolve issues. For example, when a designer is positioning the refrigerator in the kitchen the issue being resolved is "Where should the refrigerator be located?"--or some paraphrase of this. Criticism which appears in CRACK's "Critic Window" (lower right in Figure 4) corresponds to a general answer to the construction issue. For example, "REFRIGERATOR-1 next to RANGE-1" (in Figure 4) corresponds to the answer, "Do not put the refrigerator next to the range." Finally, what appears in CRACK's "Explanation" window (lower left in Figure 4) is an argument for this answer.

The fact that CRACK's criticism can be put in PHI form shows that its knowledge-based critiquing mechanism actually connects construction with argumentation--albeit very limited argumentation. This means that CRACK and VIEWPOINTS could be coupled by using CRACK's critics to provide the designer with immediate entry into the exact place in the hypertext network where the argumentation relevant to the current construction task lies. This would solve two problems: it would greatly enrich the argumentative information provided by critics and it would allow PHI hypertext to be used during construction. Such a combined system could provide argumentative information for construction effectively, efficiently and without designers' having to

1) realize they need information,

2) suspect that needed information is in the system or

3) know how to retrieve it.

JANUS is the integration of the two systems, CRACK and VIEWPOINTS, in a common computing environment: the Symbolics Genera Software Environment. JANUS is named for the Roman god who had two faces looking in opposite directions and who symbolized a variety of dualities, such as beginning and ending, rise and fall, entry and exit. In the JANUS design environment the two faces correspond to construction and argumentation--and to the coupled software systems JANUS-CRACK and JANUS-VIEWPOINTS.

JANUS-CRACK: Janus' Construction Component

JANUS-CRACK supports the construction of an artifact either "from scratch" or by modifying an already constructed artifact. To construct "from scratch," the designer chooses building blocks from a design units "Palette" (upper left in Figure 5) and

positions them in the "Work Area" (upper right in Figure 5). Protocol studies have shown that designers use design units at different levels of abstraction in the same project [Akin78][Fisc88b]. Therefore, JANUS-CRACK allows the designer to select different palettes each of which contains design units at a different level of abstraction. In the current version of JANUS-CRACK there are two such palettes, one containing specific appliances (shown in Figure 5) and one containing the more abstract design units of "work centers," such as cleanup center, cooking center and storage center.

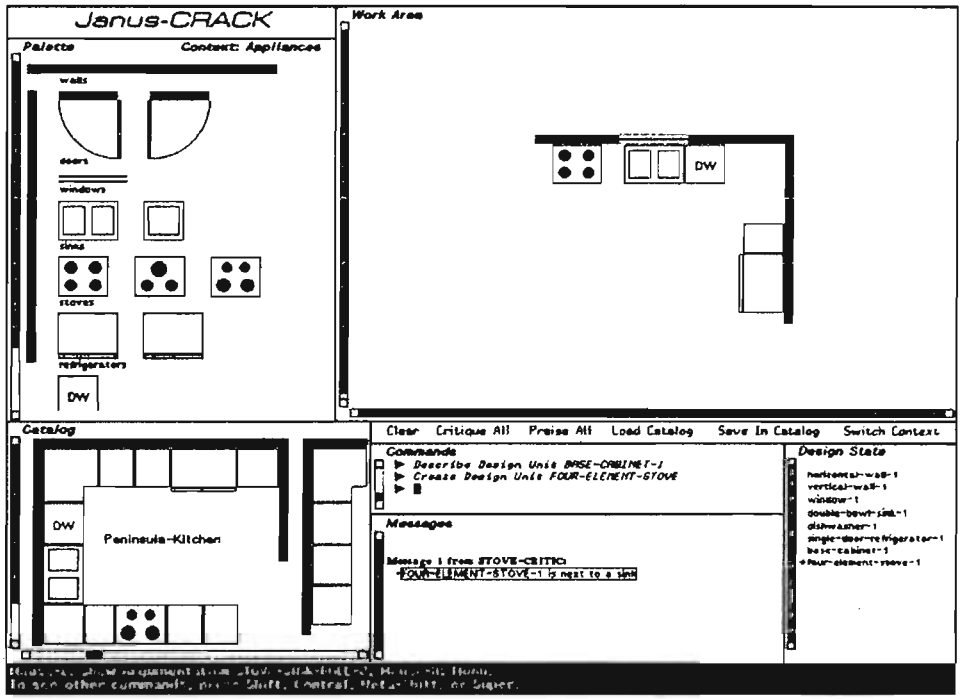

FIGURE 5. A screen image from JANUS-CRACK.

To construct by modifying an existing artifact, the designer uses the "Catalog" (lower left in Figure 5), which contains many previously designed kitchens. The designer can browse through this catalog of examples until an interesting design is found. This design can then be selected and brought into the "Work Area," where it can be modified to the designer's liking. In this way the designer can avoid many of the difficulties of starting "from scratch."

There are currently two types of examples in the catalog: "good" designs and "bad" designs. The former satisfy all the rules of kitchen design and will receive no negative criticism from the system's critics. People who want to design without knowing the underlying principles might want to select one of these, since minor modifications of these will be probably result in little or no negative criticism from the critics.

The "bad" designs in the catalog are there to support learning about principles of design--in this case principles of kitchen design. By bringing these into the "Work Area" the user can subject them to critiques by the system--either by repositioning design units in the selected example or by activating the "critique all" command, which fires all critics. This will show what principles of kitchen design are incorporated into the system. It will also provide the user with entry into the hypertext system where the larger argumentative background of the principles can be explored by navigation.

The "good" designs in the catalog can also be used to learn design principles and explore their argumentative background. This can be done by bringing them into the "Work

Area" then using the "praise all" command. This command causes the system to display all of the rules which the selected example satisfies. This positive criticism also provides entry points into the hypertext argumentation.

JANUS-VIEWPOINTS: Janus' Argumentation Component

The PHI hypertext component of JANUS is implemented using Symbolics' Concordia and Document Examiner software. Concordia is a hypertext editor [Walk88] which we used to create the issue base. The Document Examiner [Walk87] provides functionality for on-line presentation and browsing of the issue base by users. To use this software for PHI we had to map the documentation structure of the Document Examiner--including sections, subsections, subsubsections and fragments--into the argumentative structure of PHI--including issues, subissues, answers, subanswers, arguments and subarguments. Concordia allowed us to do this and to display argumentation in the same sort of outline format used in MIKROPLIS. Document Examiner allows users to navigate in the issue base by using the mouse in several different types of displays.

Many displays in JANUS-VIEWPOINTS are composites of related texts representing "argumentative contexts." The "answer" context (in the main "Viewer" pane in Figure 6) contains an answer and its arguments. The "issue" context (in the "Outline" pane, upper right in Figure 6) contains an issue with its answers and their arguments.

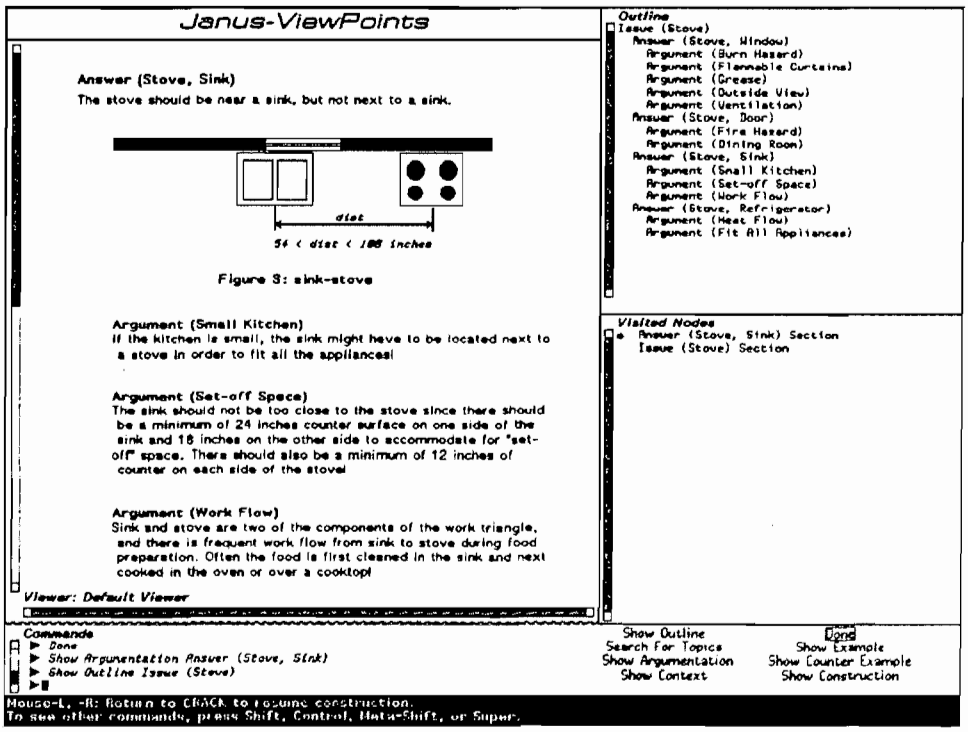

FIGURE 6. A screen image from JANUS-VIEWPOINTS.

When users enter JANUS-VIEWPOINTS from JANUS-CRACK they are brought into a section of the PHI issue base determined by and relevant to their current construction situation. They do not have to hunt for relevant information. Their point of entry into the hypertext network will contain relevant information. But since the argumentation on an issue can be large and complex, they can use the initial display of relevant information as the starting place for a navigational journey through the issue base. Each traversal of a PHI link will take them to additional relevant information which argues the current construction issue in more detail. Thus the chances of becoming "lost in

hyperspace" are reduced. Upon completion of the examination of the relevant argumentation the designer can return to construction and complete the current task.

Critics as Hypertext Activation Agents

JANUS' knowledge-based critics serve as the mechanism linking construction to argumentation. As is the case with CRACK, the critics in JANUS have knowledge about how design units should and should not be arranged. They "watch over the shoulders" of designers doing construction and critique their work, displaying the criticism in the "Messages" pane (center bottom in Figure 5) if design principles are violated. In doing so they also identify the argumentative context in JANUS-VIEWPOINTS which is appropriate to the current construction situation.

For example, when a designer is positioning a stove in the "Work Area" as in Figure 5, the stove critic fires and detects that the stove is too near the sink. It therefore displays the appropriate criticism in the "Messages" pane--as is also shown in Figure 5. To see the argumentation surrounding this rule the designer has only to click on the text of this criticism with the mouse. The argumentative context shown in Figure 6 is then displayed.

Evaluation of JANUS.

We have evaluated JANUS with subjects ranging from neophyte to expert designers and from neophyte to expert computer users. We found that no user group had any significant overall advantage in using the system. Design students were more familiar with the general application domain but learned to use the system without much difficulty after some initial practice. Computer science students were able to understand the critics and learned from them to create reasonable kitchen designs. Users uncertain about criticism from the system or interested in more background information about design principles entered the hypertext system by clicking on the criticism. No users got "lost" in the hyperdocument; none failed to complete the construction task.

CONCLUSIONS AND FUTURE WORK

Conclusions

JANUS represents not merely a system building effort but a theoretical framework for application of hypertext to design. Our preliminary evaluation of the JANUS system has demonstrated that hypertext can be used in conjunction with knowledge-based design environments to inform designers effectively, efficiently and without their having to know

1) that they need information, i.e., that their knowledge is inadequate for certain design tasks,
2) that the needed information exists and is in the system, or
3) how to retrieve it.

PHI hypertext systems like VIEWPOINTS and design environments like CRACK individually help to improve design, but they have significant limitations. Integrating them in a system like JANUS overcomes many of these limitations. PHI's task-oriented links make navigation effective in locating information for design tasks once the verbal representations of these tasks as issues have been located. Construction kits support human problem-domain communication; adding intelligent critics to create design environments promotes better construction. When PHI hypertext and design environments are combined as in JANUS, the critics alert the designer to the existence of information relevant to current construction tasks and provide immediate and effortless entry into the exact location in the hypertext network where the useful information lies.

Future Work

In the course of developing JANUS we have found a variety of topics for future research. Roughly speaking, these can be categorized into issues dealing with 1) the construction system (JANUS-CRACK), 2) the argumentation system (JANUS-VIEWPOINTS) and 3) the connections between these two.

Within the construction system alone there are many issues which need further exploration. One is that of what would be needed if the construction task were scaled up--e.g., from a kitchen to a house to an apartment building to a cluster of apartment buildings. It seems that the current flat structures of the palette, the catalog and the construction state (in the "Work Area") would be inadequate. What then should these structures be? How should their information be managed? Should hypertext be used? If so, how should this construction hypertext system relate to the argumentation hypertext system?

Within the argumentation system there is a pressing need for authoring to be integrated with browsing. Allowing *ad hoc* authoring during browsing would enable the designer to annotate the issue base, record decisions on issues and generally personalize the argumentation. This in turn would create the need for certain basic kinds of inference mechanisms. For example, if the designer has rejected the answer "dining area" to the issue "What functional areas should the kitchen contain?" then the system should probably not display any issues, answers or arguments which presuppose or assume that the kitchen has a dining area.

Construction and argumentation might usefully be connected in a number of additional ways. Catalog examples could be used to illustrate argumentation, and argumentation could be used to help in selecting examples from the catalog. Argumentation could be used to help designers select items from the palette--or even to select palettes. Designers might want to see. the relevant argumentation *before* they place items in the work area. Subissue structures might be used to suggest the nature and order of construction tasks.

Finally, while kitchen design has been a useful starting point, the real significance of the JANUS approach lies in its applicability to a wide range of problem domains. We therefore plan to test it in several other domains, including software design.

ACKNOWLEDGEMENTS

We would like to thank our colleagues and students who helped us critically evaluate the usefulness of the systems described in this paper. The research was supported in part by Grant No. MDA903-86-C0143 from the Army Research Institute and by grants from NYNEX Corporation and Software Research Associates (SRA).

REFERENCES

[Akin78] Akin, O. "How Do Architects Design?" in **Artificial Intelligence and Pattern Recognition in Computer-Aided Design**, J. Latombe (ed.), North-Holland, New York, 1978.

[Aksc88] Akscyn, R.; McCracken, D.; and Yoder, E. "KMS: A Distributed Hypermedia System for Managing Knowledge in Organizations," **Communications of the ACM**, Vol. 31, No. 7, July, 1988, pp. 820-835.

[Conk87] Conklin, J.; Begeman, M., "gIBIS: A Hypertext Tool for Design Deliberation," **Hypertext'87 Papers**, University of North Carolina, Chapel Hill, NC, November 1987, pp., 1987, pp. 247-251.