gIBIS: A Hypertext Tool for Exploratory Policy Discussion

Jeff Conklin and Michael L. Begeman

MCC
Software Technology Program
3500 West Balcones Center Drive
Austin, Texas 78759-6509
ARPA: conklin@MCC.COM begeman@MCC.COM

ABSTRACT.

This paper describes an application specific hypertext system designed to facilitate the capture of early design deliberations. It implements a specific method, called Issue Based Information Systems (IBIS), which has been developed for use on large, complex design problems. The hypertext system described here, gIBIS (for graphical IBIS), makes use of color and a high speed relational database server to facilitate building and browsing typed IBIS networks. Further, gIBIS is designed to support the collaborative construction of these networks by any number of cooperating team members spread across a local area network. Early experiments suggest that the IBIS method is still incomplete, but there is a good match between the tool and method even in this experimental version.

INTRODUCTION.

There is a growing recognition that hypertext is an ideal model on which to base a support environment for the system design process. In the MCC Software Technology Program we have been working on a hypertext project called the *Design Journal* which is aimed at providing a team of system designers a medium in which all aspects of their work can be computer mediated and supported. This includes the traditional documents such as requirements, specifications, high level design, and the design document itself, but it also includes such things as interviews with users, scenarios, design reviews, designers' early notes and sketches, design decisions and rationale, internal design constraints, meeting minutes, etc. The Design Journal places particular emphasis on the capture of the design rationale as the central aspect of the process which may serve to integrate all of the other documentation. In addition, our research is directed at the *upstream* of the design process, where most of the information is informal, and for which there is little technical support.

By design rationale we mean the design problems, alternative resolutions (including those which are later rejected), tradeoff analysis among these alternatives, and record of the tentative and firm commitments that were made as the problem was discussed and resolved. Our research has two thrusts: (i) to understand the internal structure of design decisions, and the higher level dependencies which grow up among decisions, and (ii) to address the interface problems inherent in capturing large amounts of informal design information and in providing effective methods for indexing and retrieval within that information. As part of the former thrust we have been developing our own theory about the structure of design decisions, called *ISAAC*. As part of the latter thrust we have built a running prototype of the Design Journal, called *gIBIS*. At the time of the design of gIBIS, however, the ISAAC theory was not yet ready to be encoded as a running tool. Instead, gIBIS is based on a similar though somewhat simpler model of design deliberation called *Issue Based Information Systems*, or *IBIS*.

THE IBIS METHOD.

The IBIS method was developed by Horst Rittel [RIT70], and is based on the principle that the design process for complex problems, which Rittel terms "wicked" problems, is fundamentally a conversation among the stakeholders (e.g. designers, customers, implementers, etc.) in which they bring their respective expertise and viewpoints to the resolution of design *issues*. Any problem, concern, or question can be an issue, and may require discussion (if not agreement) in order for the design to proceed. Indeed, in the IBIS model it is this "argumentation" which constitutes the design process. (This does not preclude "arguing" with oneself, and gIBIS works as well for monologues as for dialogues.) Rittel developed this model over 15 years ago, and has used it successfully in diverse design situations such as architectural design, city planning, and planning at the World Health Organization.

The IBIS model focuses on the articulation of the key *Issues* in the design problem. Each Issue can have many *Positions*. A Position is a statement or assertion which resolves the Issue. Often Positions will be mutually exclusive of each other, but the method does not require this.

Each of an Issue's Positions, in turn, may have one or more *Arguments* which either support that Position or object to it. Thus each separate Issue is the root of a (possibly empty) tree, with the children of the Issue being Positions and the children of the Positions being Arguments.

There are nine kinds of links in IBIS. For example, a Position *Responds-to* an Issue, and this is the only place the Responds-to link can be used. Arguments must be linked to their Positions with either *Supports* or *Objects-to* links. Issues may *Generalize* or *Specialize* other Issues, and may also *Question* or *Be-suggested-by* other Issues, Positions, and Arguments.

A typical IBIS discussion begins with someone posting an Issue node containing a question such as "How should we do X?". That person may also post a Position node proposing one way to do X, and may also post some Argument nodes which support that Position. Another user may post a competing Position responding to the Issue, and may support that with their own Arguments. Others may post other Positions, or Arguments which support or object to any of the Positions. In addition, new Issues which are raised by the discussion may be posted and linked into the nodes which most directly suggested them. Figure 1 shows a state transition diagram specifying all of the legal moves within the IBIS method.

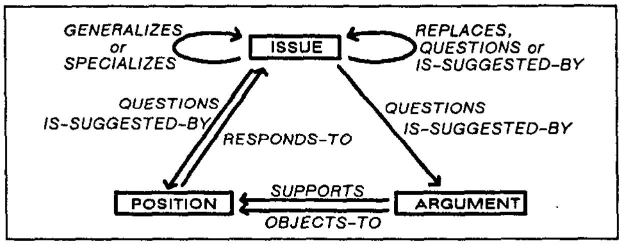


Figure 1: The set of legal rhetorical moves in IBIS.

There is no stopping rule, nor is there in the IBIS method a particular way of registering that an Issue has been resolved by agreement upon some Position. Rather, the goal of the discussion is for each of the stakeholders to try to understand the specific elements of each others' proposals, and perhaps to persuade others of his own point of view. The method makes it harder for discussants to make unconstructive rhetorical moves, such as "argument by repetition" and name calling, and it supports other more constructive moves, such as seeking the central issue, asking questions as much as giving answers, and being specific about the supporting evidence of one's viewpoint.

In implementing gIBIS we have made certain changes and extensions to the IBIS method to allow needed flexibility. However, we tried to change the method as little as possible. The IBIS method has been in use by Rittel in various design and planning activities for many years, and we felt it was important to push his method as far as it would go, understanding its strengths and weaknesses, before we started making any radical extensions.

The extensions to IBIS in the current gIBIS tool are: (1) an additional "Other" type for nodes and links, as an "escape" mechanism for users who could not find a way to express a thought within the IBIS framework; (2) an additional "External" type for nodes that contain non-IBIS material, such as requirements documents, design sketches, or code; and (3) the ability to let Positions "specialize" or "generalize" other Positions, and likewise with Arguments.

THE gIBIS TOOL.

There were three technological themes guiding our design of gIBIS. The first was an interest in exploring the capture of design history: the decisions, rejected options, tradeoff analysis ... in short, the rationale behind the design itself. For this purpose, the IBIS framework seemed a good starting point. The second theme was an interest in supporting computer mediated teamwork, and particularly the various kinds of design conversations that might be carried on via networked computers, a la email or news [EVE86, HOR86]. And thirdly, we needed an application in which we would have a sufficiently large information base that we could investigate navigation (i.e. search and browsing) of very large information spaces. All of these factors lead us to this application, and to the more specific requirements discussed below.

As can be seen in Figure 2, the basic gIBIS interface is divided into 4 tiled windows: a graphical browser on the left, a structured index into the nodes on the top right, a control panel below the index window, and an inspection window in which the attributes and contents of nodes and links can be viewed. This interface is somewhat unusual in that the only way to view the contents of a node or link is to select it, causing it to be immediately displayed in the single inspection window.†

The browser.

The browser provides a visual presentation of the IBIS graph structure. Nodes and their interconnecting links are displayed on a canvas of virtually unlimited size. Most of the browser is dedicated to a local view of the network: a "zoomed in" view of the current area of interest which shows the full detail of the nodes and links. The lower-right portion of the browser is reserved for a global overview of the data: a "zoomed out" abstraction of the entire network in which node labels, link type icons, and the secondary links which make up the network's fine structure are filtered out. In addition to giving an overview of the entire network, the global view also indicates the scope and position of the current local view by a rectangular overlay (in this example, the local view extends down from the top left corner of the global view).

† The examples used throughout these sections are taken from the issue group "gibis-issues" which we have used to capture many of the design issues for the gIBIS tool itself. The reader is warned not to confuse statements which appear in the issue group (e.g. "The gIBIS tool does not allow users to select links".) with the current state of the tool. (This particular IBIS conversation resulted in an implementation for selectable links.)

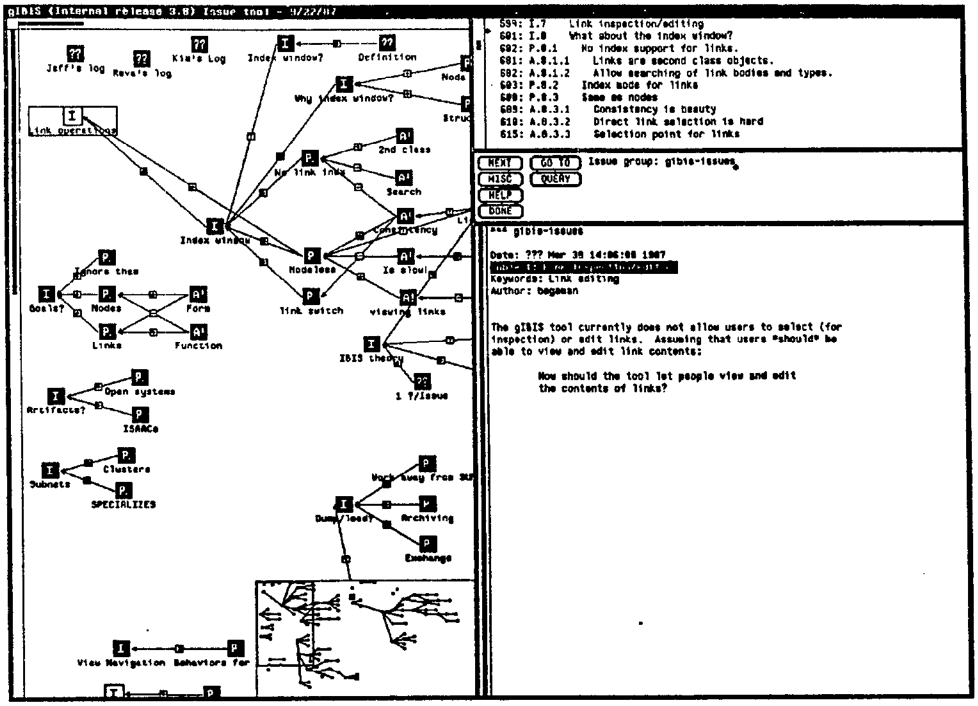
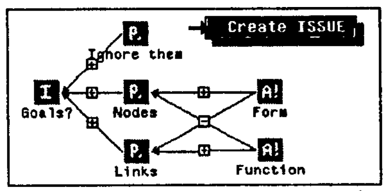


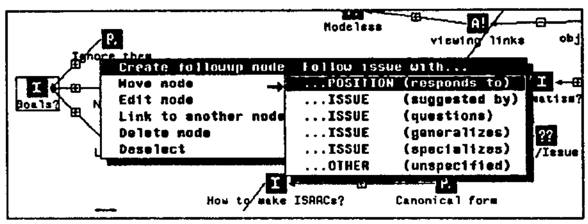
Figure 2: The gIBIS Interface.

The canvas can be scrolled within the window area of the browser by the use of traditional scrollbars (seen at the top and left side of the browser) or more directly by "snap scolling" - a method where the user clicks the mouse anywhere within the local view to center that location in the window. This method allows the user to easily fine-tune the positioning of the display, and to scroll diagonally without having to reposition two independent scrollbars. Scrolling to an area outside of the local view is also possible by directly repositioning the local view indicator in the global view window. Simply dragging the rectangle to a new area within the global view causes the local view to be updated appropriately.

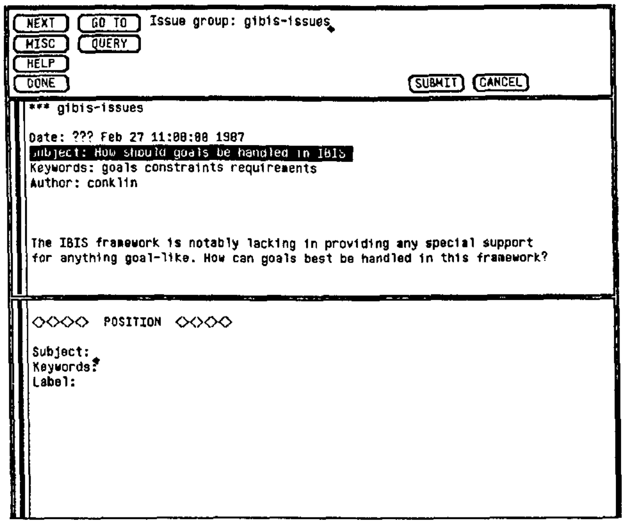
The browser supports a direct manipulation [NOR86] style interface to the nodes and links (i.e. display objects). Display objects can be selected simply by "clicking on them" with the left button of our 3-button mouse. Selecting a display object causes it to become highlighted and boxed in the browser, its contents to appear in the inspection window (see Figure 2), and its index line to be scrolled to the top of the index window. A right-click on the mouse causes context-sensitive menus to be displayed. It is by these menus that objects are created, edited, deleted, moved, and so forth. As an example, let's begin with the case where the user presses the menu button when no object is selected. The following menu appears



indicating that the only legal operation is Issue creation (i.e. the beginning a new IBIS structure). By contrast, if a node of type Issue is selected, the menu changes to reflect the legal operations on Issues.

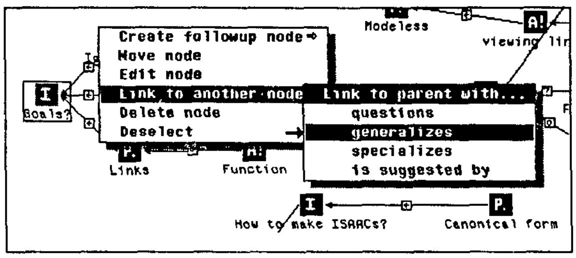


In this example, the user is choosing to create a followup node of type Position which, when populated and submitted, will be placed to the right of the selected Issue and will automatically be linked to it by a link of type Responds-to. Upon making this menu selection, the inspection window (lower right) divides itself in half and a Node Creation window appears under it which is preloaded with a structured template to be filled in.

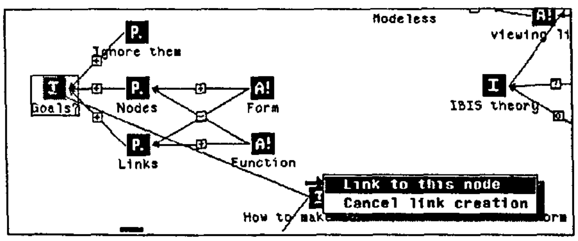


The user fills in the template's structured fields (e.g. Subject, Keywords, ...), and then provides an optional description of the node's topic (i.e. an unstructured node body). When the node has been completed, the user pushes the SUBMIT button in the control panel (which appears only during Node Creation/Editing), the node gets parsed and stored, and the browser and index windows are updated to include it.

When users follow the "Link to another node" menu item, a pullright menu appears which constrains them to select from the set of legal outgoing link types for the currently selected node.



Choosing the link type, the new link appears stretching from the source-node to the current mouse position. The user moves the mouse to the destination node (the link follows the mouse by "rubber banding" across the canvas), and the user then drops the end of the link on its destination again by menu selection.

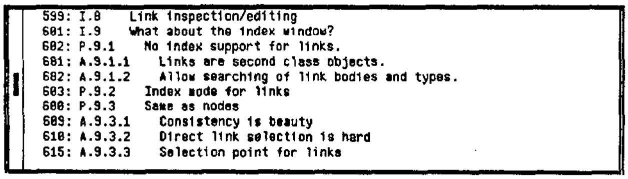


In addition to being able to select nodes and links, canonical IBIS subnets (i.e. a single Issue followed by its set of Positions, followed in turn by their set of Argument nodes) are selectable as an entity as well. The tool provides support for the movement and automatic layout of these subnets as a whole. Further, gIBIS allows aggregation of these subnets into a single composite *IPA* node which provides additional structure to represent an analysis of the competing Positions and commitment to one of them (i.e. Issue resolution).

While it has a structure and body all its own, the *IPA* node by default inherits its label, subject and keywords from the root Issue of the underlying subnet. Selecting the composite causes traversal of the underlying IBIS subnet, composing an "inherited" body which is shown in the inspection window along with the text which is specific to the composite (see Figure 3). Since the inherited body of the composite can grow to be quite long for an aggregation of a large IBIS subnet, users can suppress (or reveal) its inclusion in the inspection window by use of a function key.

The node index window.

The node index window provides an ordered, hierarchical view of the nodes in the current IBIS network. The network is traversed following Primary links (discussed later) in depth-first order starting from each Issue (i.e. the root of each canonical IBIS subnet). The Issues, Positions and Arguments are given sequence numbers like those one would expect to find in an outline editor[HER85]; for example, the following figure shows the Subject line for Issue 8 (I.8) which has no children, I9 whose first Position node (P.9.1) has 2 Argument nodes as children (A.9.1.1 and A.9.1.2), and so forth. Issues are simply ordered by creation date. The integers in the leftmost column are unique object identifiers and can be ignored. The view configuration panel allows the user to tailor the index information to reflect not only by Subject, but also Author, Keyword, or Node Label.


599: I.8 Link inspection/editing
601: I.9 What about the index window?
602: P.9.1 No index support for links.
681: A.9.1.1 Links are second class objects.
682: A.9.1.2 Allow searching of link bodies and types.
603: P.9.2 Index mode for links
600: P.9.3 Same as nodes
609: A.9.3.1 Consistency is beauty
610: A.9.3.2 Direct link selection is hard
615: A.9.3.3 Selection point for links

Nodes can be selected through the index as well as the browser. Clicking on a node's index line causes that node to become current: its icon is highlighted in the browser, the canvas is scrolled (if necessary) to make the node visible in the local view, and the node's contents appear in the inspection window. Through this index window based access, we have provided a second browsing method which provides a linear, compressed view of the data in the network.

The control panel.

The control panel is composed of a set of buttons which extend the tool's functionality beyond simple node and

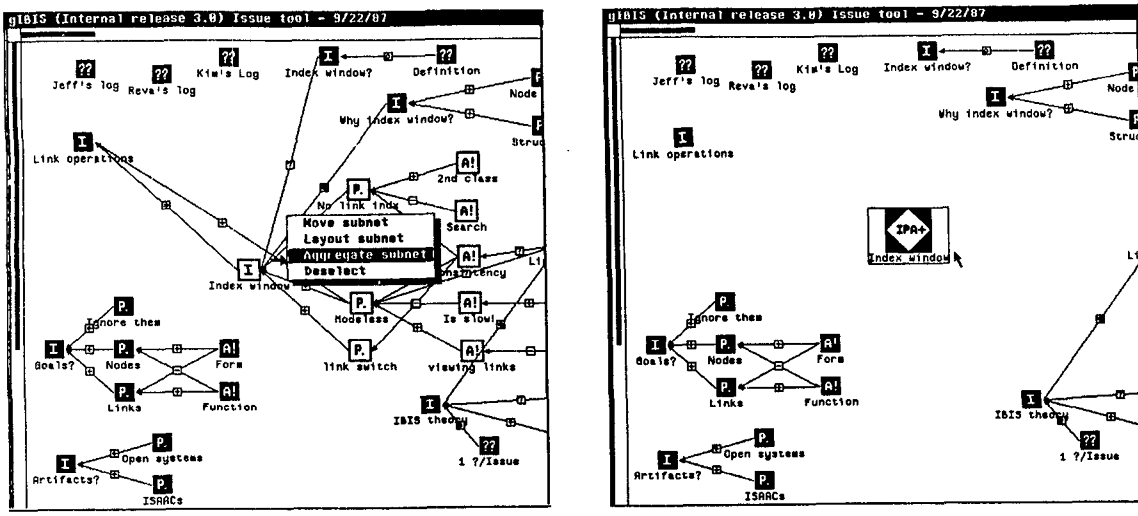
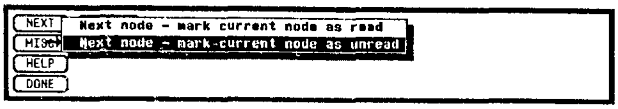


Figure 3: A canonical IBIS subnet before and after aggregation.

link creation. Each button has a menu hidden behind it which extends or tailors its basic function. The NEXT button, for example, will normally cause gIBIS to record that you have read the current node before displaying the next node in the network, but pressing the right (menu) mouse button while over the NEXT button causes this menu to appear – a slight extension of the basic functionality which leaves the current node marked unread.

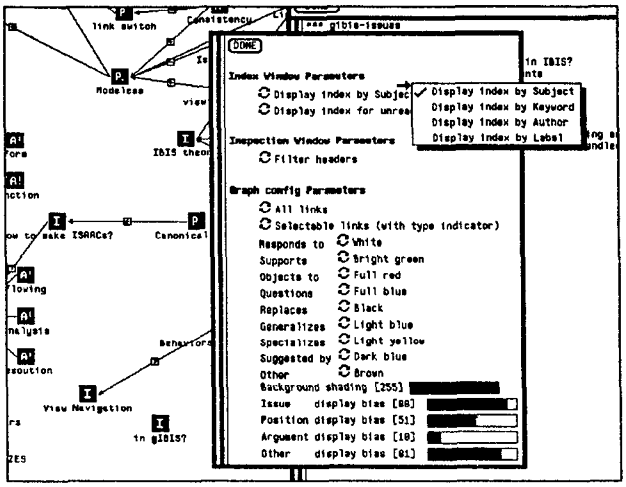


For those functions which have no extended function, the menu is simply a longer explanation of the functionality provided. For example the GOTO button which causes gIBIS to load a particular issue group's data into the browser has a simple help menu behind it which instructs the user to "Enter an issue group name and push this button."

Interface configurability.

The MISC button hides a grab-bag of functionality. Of the functions available, we will describe one in depth: the TOOL CONFIG item which allows the user to tailor particular aspects of the interface.† Upon selecting this item, a new window appears (see following figure) which contains the gIBIS configuration parameters, their current settings, and constraints on their legal settings.

† Other functions behind the MISC button allow users to send tool gripes/suggestions to the developers, mark all nodes as having been read/unread, linearizing and printing the IBIS net, and [un]subscribing to issue groups.



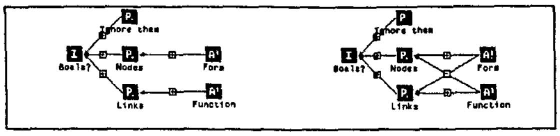
The Tool Configuration Window.

Configuration parameters are divided according to which window they affect: the index, browser, or inspection window. The adjoining figure shows a user modifying the node attribute upon which the index window will be built. We wish to emphasize two major points about the browser now: the concept of a "primary" link, and the use of color.

Primary and Secondary links.

Recall that when a node is created, it is usually automatically linked into the existing network of nodes. This automatic link – the *first* link which connects a node into the network – is considered to be that node's primary link. The user may later connect that node to others in the network using the linking facility described above; but all subsequent links are considered to be *secondary* links, and are distinguished from the primary link both visually and navigationally.

Filtering the secondary links from a canonical IBIS subnet results in a hierarchy, and this hierarchy is the basis for the index window's structured linearization. Take the case where three Positions were created in response to an Issue. Two of the Positions have supporting arguments. In this example, the Positions were mutually exclusive, so each Argument also objected to the other Position, and hence the authors created secondary links to make this explicit.



We have found that it is easier for the casual browser of an IBIS network to understand the network if, on first pass, the secondary links are turned off so that the browser only displays the primary links. The NEXT button leads the user through the network in the canonical IBIS order (the same sequence as the index window), and the primary-link browser view reinforces their understanding of how the current node relates to the surrounding conversational structure. For subsequent passes, the user may wish to enable the visibility of secondary links in order to understand the cross-relationships which the authors of the network have encoded. (In keeping with the design philosophy of tightly coupled windows, selecting a node with the NEXT button causes the same scrolling/highlighting as selection via the browser or index window.)

The use of color.

gIBIS was designed for use on SUN workstations with color monitors. Based on this, we chose to make use of color to indicate node and link type information, as well as some special node states such as "currently selected" and "matches the current query." We also gave users the ability to configure the tool to customize the type-color mapping.

This flexibility caused some trouble at first and we quickly proposed (and then encoded in the tool) a set of standardized color mappings. Having colored nodes and links turns out to be one of the most compelling aspects of the tool. Users can quickly learn the type mappings for the most commonly used nodes and links, and type identification then becomes a rapid, unconscious activity. While users occasionally change their mappings using the TOOL CONFIG panel for special purposes (like making some links invisible for presentations), they most commonly set their mapping and leave it alone.

Later, we adopted a set of users with monochrome monitors, so we provided iconic information which duplicates the information encoded by color. While the tool by default presents both color and iconic information to the user, both can be suppressed. Most frequently, the color user will suppress link icons in order to make the browser appear less cluttered.

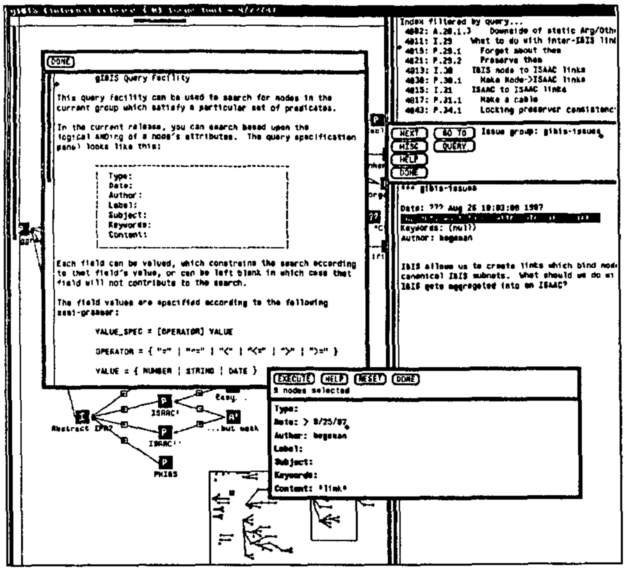
The use of color presents its own set of problems though. The technique of type-to-color mapping is obviously limited to those users who have color display devices and are not themselves color blind. It is also limited to situations where the number of mappings remains rather small. In our application there are nine link types, and the feeling is that we are near the limit of people's ability to reliably perform the mapping. By adding the link type icons, the mapping complexity drops and more link types could be "safely" added.

More surprising, however, is the large machine-to-machine variation among the color monitors. The variation in overall brightness, convergence, and RGB gun saturation has eliminated the possibility of using a single, standardized set of color mappings for all machines. Color settings which produce bright, highly defined images on one screen can look very dark, muddy and indistinct on another. To address this, we have provided the four sliders at the bottom of the TOOL CONFIG window which allow users to "fine tune" the colormap on their machines. While this approach lets users construct colormaps which adequately distinguish the types, we doubt whether we could ever reliably use more subtle shading schemes to communicate type information.

Search and query.

The last control panel feature we wish to discuss is the QUERY button. Pressing it results in the appearance of a small query-construction window as seen centered in the next figure. The query window itself contains a small control panel and a query specification section which is query by example in that the user creates a proto-node against which those in the current IBIS net will be matched. The following example searches the network for those nodes authored by "begeman" after 8/25/87 which contained the string "link" in their body. Upon pressing the EXECUTE button, the query is parsed, evaluated, and the query's results are displayed both in the browser (selected nodes are turned a bright yellow in both the local and global views) and in the index window (the window only shows the index lines for those nodes which satisfied the query). The user can then examine those nodes using the standard navigation techniques described above. Pressing the HELP button reveals yet another window (obscuring the browser window below) which contains instructions on how to formulate queries, the query grammar, and a number of examples.

This query specification technique allows users to easily formulate node content searches based upon the logical ANDing of predicates over node attributes. While we have thought about extending the grammar to allow full boolean expressions over the predicates (and have designed a visual interface for the specification of structural queries), there has been so little demand for this that we have chosen to focus our implementation resources elsewhere. We feel that these more sophisticated queries may be required when the networks become very large, but experience shows that the simple query engine which has been provided is sufficient for searching over networks of moderate size.



The query control and help windows.

Some key gIBIS requirements.

gIBIS is primarily a vehicle for the exploration of Issue-based methodologies for the capture of design rationale. It was intended from the start to be used by small teams of people collaborating on "real" projects within the Software Technology Program at MCC. Because of this, we had a number of constraints to design to:

1) *The tool had to be reliable.*
We recognized that since people would be using gIBIS to capture information which was important to them (i.e. not for "toy" problems/experiments), our data storage had to be very reliable. Losing or corrupting an IBIS network was considered to be an intolerable fault.

2) *The tool had to support multiple concurrent users.*
As a tool to facilitate team collaboration, gIBIS had to provide true multiuser support. This meant shared, coordinated access to centralized IBIS networks, automatic notification of significant changes (e.g. new nodes) to the nets, access control and locking to prevent multiple updaters from corrupting the data, and "lightweight" IBIS groups that teams could create and share at their own volition. The user community had already been using the USENET news network to hold machine-mediated group conversations, and was expecting richer and more powerful functionality from gIBIS.

3) *The tool had to perform reasonably well.*
With the goal of producing a real tool for use on real problems, we had to provide reasonable performance from the beginning. Accessing an existing node or link's contents needed to be almost instantaneous, while larger tasks such as loading a new IBIS network into the browser or performing a query could take longer (10-15 seconds for a large network seemed reasonable). Our basic guideline was that the performance of the tool should not break a user's rhythm of creating and browsing a network.

4) *We had to implement gIBIS with very limited resources.*
Because the entire project team consisted of $2^1/_2$ people (1 author working full time on the methodology, the other author and a student working on the tool), we had to import as much functionality as possible. Wanting to concentrate our efforts on the interface and the method, we chose to build gIBIS on top of an existing relational DBMS.

The Choice of a RDBMS.

Choosing a relational DBMS as our storage manager provides us with concurrency control, record level locking, reliable data storage, fast access methods and a reasonable search engine for free. In addition, our DBMS (UNIFY) provides us with an uninterpreted data type - basically a field into which we can store arbitrarily long passages of text, digitized voice, graphics bitmaps, or whatever. We are therefore able to store the body of a node as an integral part of a record in the database – something which many of today's DBMSs do not support. In retrospect, we feel that the decision to implement gIBIS on top of a DBMS has allowed us to focus on our research topic, and has saved us many months of development effort.

Unfortunately, the DBMS does not provide an adequate notification mechanism (*triggers* in DBMS parlance) to alert an application when a table or set of records gets modified (e.g. when a new node gets added to an Issue group). To overcome this, we had to build our own notification layer on top of the database. This layer keeps track of the state of the DBMS *with respect to each individual user*. When the database gets modified in such a way as to cause a change in any user's view of the data, those affected users are sent a notification and their copy of gIBIS updates their view appropriately. In this way, gIBIS provides an effective, tight coupling between its users and their views of the evolving Issue networks.

References to external data.

Using a DBMS as our storage manager presented one major drawback however: closing the system. In essence, all of the objects which the Issue networks reference needed to reside within the DBMS. Unfortunately, many objects which give rise to Issue-based discussions (like requirements or architecture documents) as well as those which result from these networks (such as code and documentation - the *artifacts* of design) are external to the database and hence out of reach. For this reason, we felt compelled to create a special *surrogate* type of node which allows gIBIS to reference external objects in a "blind faith" sort of way. A surrogate has two parts: a pointer to the external object (usually a fully-qualified pathname to a file) and an optional display program which gIBIS should invoke to display the object. If the default display program is invoked, the external object is assumed to be a text file and is loaded into gIBIS' standard inspection window. If, on the other hand, the user specifies a display program, that program is invoked and passed the external pathname as an argument. Using this facility, external data and programs are smoothly integrated into gIBIS. Some examples of external data which we have seen include simple textual documents, static graphic figures, dynamic simulations, a spreadsheet, and even a full-scale hypertext network managed by MCC's PlaneText hypertext system.

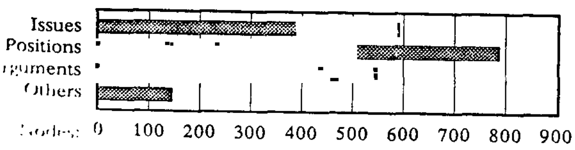
OBSERVATIONS.

In this section we wish to present some trends and observations of the uses, strengths, and weaknesses of this hypertext tool. These are preliminary findings. In this section we have tried to be as candid as possible about the weaknesses and research problems of both the IBIS method and the gIBIS tool. We hope that this candor does not create an overly negative impression about what we feel is a very positive research effort.

It is important to keep in mind that none of our users was -- at least initially -- more than passingly familiar with the IBIS method itself, so there was quite a bit of learning and experimenting going on while users constructed their networks. Indeed, it could be said that most of our users regarded *themselves* as experimenters, exploring different ways of working and using the tool, conventions, etc.

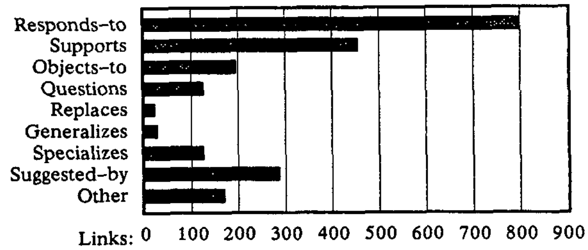
Network structure.

In this section we describe the usage of the gIBIS tool during a one year period, from mid February 1987 to mid February 1988, in terms of statistics on the networks. Thirty two people participated in the creation of 33 issue groups in all. (One issue group, not presented here, was used for an experiment in which the gIBIS hypertext facilities were used but the IBIS method ignored.) As of February 1988, 2091 nodes had been created in roughly equal numbers of Issues, Positions, and Arguments across all of the issue groups.



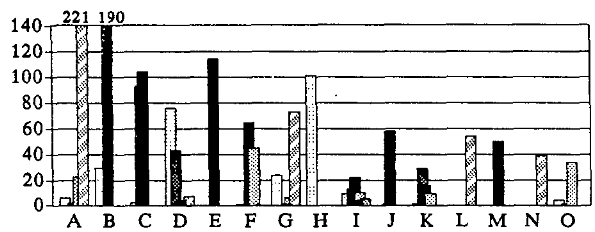
Thirty one percent of the Issue nodes had no Positions and the remainder of the Issue nodes had, on average, 1.9 Positions. On the other hand, 59 percent of the Position nodes had no Argument nodes, and the remaining Positions had an average of 1.7 Arguments.

Connecting these nodes were 2214 links, in the following proportions:



While it is still to soon to draw any conclusions from these numbers, they do at least indicate that users had a greater tendency to post supporting Arguments than objecting ones, and that it was somewhat more natural to specialize nodes than to generalize them.

Finally, we wish to indicate the levels of individual participation within the fifteen largest issue groups during this trial period. Each issue group is shown with a code letter (e.g. "A"), and the bars above that designator show the number of nodes contributed by each participant, with one bar per participant. Thus, issue group "B" had two participants, one of whom posted 190 nodes, the other of whom posted 30. As the figure suggests, many issue groups were constructed largely by a single participant, while a few issue groups had reasonably balanced participation. This reflects a pattern of gIBIS usage that falls



into two categories: some people used the tool primarily as an isolated hypertext tool for structured thinking and design, others used the tool primarily as a vehicle for structured communication.

The usefulness of explicit rhetorical structure.

One early surprise about the IBIS method was that, while most people felt before they learned to use it that IBIS was awkward and overspecialized, many users of the gIBIS tool have come to regard IBIS as a powerful method for research thinking and design deliberation. Users who worked alone in an issue group reported that the Issue-Position-Argument framework helped to focus their thinking on the hard, critical parts of the problem, and to detect incompleteness and inconsistency in their thinking more readily. Users who collaborated in issue groups reported that the structure that it imposed on discussions was very useful, and served to expose "axe grinding, hand waving, and clever rhetoric." They also valued the tendency for assumptions and definitions to be made explicit.

Some of these advantages can be traced to the *semi-structured* nature of IBIS networks [MAL86]. The writer is aided in structuring a complete message without any constraint on expressibility, while the reader is provided with recurrent structure in the textual material that aids both search and comprehension. Both reader and writer are aided by the explicit rhetorical structure of IBIS, which makes apparent at least the general structure of an unfolding discussion. Indeed, we feel that a distinct advantage stems from the particular structure that IBIS provides. That is, there is a good match (though this will be difficult to prove) between some of the cognitive structures and processes of design and the 3 node types and 9 link types that compose IBIS.

However, as we press gIBIS into service in an ever wider variety of design applications, we find that there are some major shortcomings. There is no specific node type for goals and requirements, and several users have requested support for these. There is no particular support for making a decision (or reaching concensus) among the various positions of an issue, and there is no way to indicate that such a decision has been made. Design decisions usually result in the addition of solution elements to the design itself (e.g. code, module structure, etc.), but these elements are not supported by gIBIS and must be stored externally to the tool, i.e. artifacts cannot now be integrated with (i.e. linked to) the decisions that lead to them. All of these extensions were anticipated during our theoretical work on the Design Journal, though our experiences with gIBIS are suggesting some changes to our theory of design rationale.

The synergy of tool and method.

We have observed an interesting synergy -- a mutual facilitation between tool and method. The noncomputerized IBIS method is cumbersome, and would not have reached the popularity that it has here in our lab without the high speed gIBIS tool to support it. On the other hand, gIBIS is not the only hypertext system available in our environment, and yet it has achieved a wider and more prolonged usage in a much shorter time than has PlaneText [CON87], the other system. We speculate that there is a particularly good match between the requirements of the IBIS method and the hypertext facilities of the gIBIS tool.

For example, one of the clear successes of this project has been the use of color to indicate the type of the IBIS nodes and links. Perhaps this is in part because there are just a few distinct node and link types in IBIS, and each has a reasonably well-defined semantics, so that the browser display can use bright primary colors which, after some familiarization, come to have a strong association with the semantics. Evidently, despite its narrow design and rigid functionality, gIBIS provides facilities which can be quickly learned and appreciated by researchers working on ill-defined design problems. This experience has lead us to begin the design of a general hypertext system which would be a *toolkit for building gIBIS-like systems*. The basic functionality for such a system includes: typed nodes and links (user defined); customizable interface views of the network that exploit type and other information; high-speed real-time interaction among coworkers in the hyperdocument; rapid global search of the entire network, both for node contents and network structure; and strong support for classificational heirarchies and composite nodes.

The dangers of premature segmentation.

One common but subtle difficulty in hypertext systems is that it is sometimes unnatural to break ones' thoughts into discrete units, particularly when the problem is not well understood and those thoughts are vague, confused, and shifting. With gIBIS this effect is pronounced, because the IBIS method imposes a rather austere selection of node and link types on the user. In particular, design conversations often feature commitments of the form, "Let's try X -- it has advantage Y." Notice that this is a Position and its supporting Argument, with no Issue articulated for the Position. Some users have complained that they don't always see the Issue or Position immediately, and that they would like to have a "proto-node" to simply record ideas before structuring them.

To some extent this complaint is to be expected: the tool supports a method which demands that one think within a particular framework (e.g. focusing on issues without necessarily resolving them), and this can be disruptive. However, even some users who are well familiar with the IBIS method still insist that they occasionally require support for recording unstructured material.

As John Brown [BRO82] and John Smith [SMI86] have both noted for the writing process, the early phase of consideration of a writing or design problem is critical and fragile, and must be allowed to proceed in a vague, contradictory, and incomplete form for as long as necessary. However, any insights and breakthroughs should be immediately (and reversibly) capturable, and the tool should support the emergence of a coherent structure as that develops in the designer's mind.

Ultimately, of course, it will be valuable to have teased apart these elements into separate issues, positions, and arguments. But in the moment of struggling to solve the problem, the cognitive overhead required to segment the "muck" into discrete thoughts, identify their types, label them, and link them is prohibitive. We are considering providing a "brainstorming" mode in which it is easy to jot down snippets of text (and perhaps graphical sketches), providing only minimal organization to these elements. This will lead to the development of tools to aid in the structuring of this "raw" material into the IBIS framework.

Capturing issue resolution.

A somewhat complementary problem exists at the other end of the deliberation process: how should the resolution of an issue be represented and displayed? The IBIS method suggests that an issue is resolved by selecting (it does not matter how) one of the positions that respond to it as being "the right answer," or at least "the position we are committing to for now." This could be represented as marking the Position node as "SELECTED," and could be displayed simply by marking such Position nodes distinctively in the browser, e.g. by giving them a somewhat different color from unselected Positions.

We have recently added this feature, but do not yet have enough experience with it to report on user acceptance. We combined indicating resolution with the aggregation in IPA nodes, so that once an issue's discussion is aggregated into an IPA node one can indicate that the Issue is resolved. At the moment, this is done by changing the value of the "Resolved:" field to TRUE and adding a short piece of text indicating which of the Issue's Positions was the one selected as the resolution.

However, we suspect that it will not always be sufficient to simply flag the selected Position. One reason is that the rationale for adopting a particular conclusion may require more explanation -- it may be that not all of the argumentation occurred within the gIBIS tool, or that there is a broader perspective for the resolution than that in which the pros and cons of the established Positions were argued. Similarly, the resolution of an issue sometimes transcends the fixed options which were originally perceived to be available. Such emergent resolutions often combine elements of the original options, and often they abandon assumptions or presuppositions that were hidden in those options. Sometimes when such "breakthroughs" occur there is no need for further discussion -- it is clearly the right solution. gIBIS will need to allow for such leaps in the argument without unduly constraining the Issue to a well structured resolution. This may be as simple as providing the kind of free text annotation of an Issue-Position-Argument tree described above, or it may require a facility for marking such discussions as "irrelevant in light of Position X".

A problem with context in non-linear documents.

One of the chief elements of our experimentation with gIBIS is to investigate the use of hypermedia as a medium for cooperative work. In some cases where several users worked cooperatively in a shared issue group an unexpected problem emerged. Unless each author was careful to write clearly and completely, the readers found that, while they had a sense of understanding the individual nodes, they could not follow the thread of the writer's thoughts as it wound through several dozen nodes. That is, there was the sense that the hypertext tool forced ideas to be expressed in a fined-grained, separated manner, and that this obscured the larger idea being developed by the author.

In one respect this is the familiar problem of cognitive overload common to many hypertext systems: the freedom of choice inherent in branching documents simply requires greater care from the writer and attention from the reader. Another factor could be the unfamiliar separation of Position and Argument (i.e. idea and justification) in IBIS.

But we suspect that there is a related but more subtle issue here: that traditional linear text provides a continuous, unwinding thread of context as ideas are proposed and discussed -- a context which the writer is directly, if unconsciously, constructing to guide the reader to the salient points and away from the irrelevant and distracting ones. Indeed, a good writer anticipates the questions and confusions that the reader may encounter, and carefully crafts the text to prevent these problems.

The hypertext (or at least gIBIS) author, however, is being encouraged to make his or her points discrete, and to separate them from their context. Indeed, we have observed the problem that the gIBIS writer, being in a hurry to capture a design issue and its analysis, sometimes writes only the bare minimum necessary to record the essence of the issue, positions, and arguments (presumably with the intention of returning later to "clean up" the network and make their postings more readable). Even the careful author, however, is in danger of not anticipating all the various routes by which a reader may reach a given node, and so may fail to sufficiently develop the context necessary to make the node's contents clear, if not compelling.

Several techniques may be useful in ameliorating this problem. The notion of a "path," described by Bush [BUS45] and Trigg [TRI86], may provide a sufficient linearization that readers of the network can gleen a useful context from segments of a network. Also, we are experimenting with higher level constructs that aggregate a set of nodes. The new IPA node type described above combines the display of all of the nodes of an IBIS subtree (the Issue, its Positions, and their Arguments) into a single node, and allows additional IPA-specific text to be appended as well. This will linearize the discussions of individual issues and reduce the sense of fragmentation one sometimes has when reading a gIBIS network, but it is probably not sufficient to create or restore the context in which those nodes were created. Finally, part of the context that the writer has in mind is the relative importance of the various points being presented, and we are investigating ways of incorporating a simple importance metric directly into gIBIS nodes. As a methodological experiment, gIBIS users are currently experimenting with providing one of the three keywords "HI IMPORTANCE", "MED IMPORTANCE", or "LO IMPORTANCE" in each node they create. This measure could be used to guide the reader to the most salient points first (see also [LOW85]), as well as to control the level of clutter in the browser display.

Annotative or "Meta" discussions.

It is a commonplace of human conversations to "go meta" and make a comment on the *process* (as opposed to the content) of the discussion, for example, "But that isn't the issue here." Similarly, in IBIS discussions there is sometimes a need for a meta-discussion when a participant in an issue group feels that someone has poorly or inaccurately used the IBIS structure to present their ideas. For example, if B feels that the content of A's Issue node is in fact two Issues and a Position about one of the Issues, B needs some way to express this, and in fact to initiate a discussion about this "meta-issue" with A within the context of the issue group.

In fact, it has been noted that there are three levels of description for collaborative work: *substantive* (the content of the work), *annotative* (comments about substance), and *procedural* (comments about procedures and conventions for use of the medium) [TRI86]. In an IBIS framework, all three levels can theoretically be treated as Issues and their argumentation. For example, in the case of B's disagreement with A, B could post an Issue, connected by a "questions" link to A's Issue, asking "Isn't this really 2 Issues and a Position?" While this is a perfectly valid move in the IBIS rhetorical framework, it has drawbacks. This Issue is by its nature meta-substantive, although it is unclear whether it is annotative or procedural. But by placing it in the network B creates an Issue that adds complexity to the browser display without illuminating the substance of the problem being discussed, and initiates a discussion that may well lead to a change in the network, after which the meta-discussion will have only historical interest.

There are several ways of resolving this problem. One is to have special meta-level Issue, Position, and Argument nodes, so that these meta-discussions could be distinguished from the substantive ones. Or we could provide a mechanism by which any node could be labelled as "only of historical interest;" such nodes could be archived, or at least have their display suppressed so that they would not normally be visible. Thirdly, we could provide each node with its own "meta-layer," so that discussions about the match of the node's contents to its IBIS type would be tracked in this specialized part of the node, which again would only be displayed upon request. At the moment we are experimenting with a simple version of this third option: any user may append a "meta-line" at the end of the body of any node, and may then begin an annotative or procedural discussion in that part of the node by entering his comments and signing them. The author of the node might append a response at the end of the node, or might simply revise the network to correct the structural error.

Macro-level organization of the browser space.

One of the "hot issues" in hypertext research is the problem of the effective use of a graphical browser to navigate in networks that have more than a few dozen nodes. This is linked to the more general problem of disorientation [CON87], but bears particularly on the visual and spatial aspects of disorientation in a large data space. The gIBIS browser ran into these difficulties as well, of course, since the problem is largely independent of implementation. (Note that the global view mechanism described above, which shows a highly reduced view of the entire network, was added after the data for this paper was gathered. Thus while this feature has enjoyed very high user satisfaction and acceptance, our observations here are based on use of the gIBIS tool with only the small scale browser.)

In its current form the gIBIS browser must share the screen space with the node viewing and control panel windows, and so cannot occupy more than about half of the screen This provides the browser enough room to show no more than 40 to 50 nodes at one time. While this may sound like a lot of nodes, recall that the browser only displays a very brief one or two word label for each node in the browser -- to get any detail about a node requires mousing it and reading its contents in the node viewing window.

For some users this made 40-50 nodes the largest network that they wanted to try to work in. Two users, however, developed a way to partially overcome the spatial disorientation problem. These users divided their networks into regions that were meaningful in the terms of the problem they were working on. Nodes were classified according to broad semantic features, and these features were also identified with regions of the browser canvas. For example, one user created a node, labelled "LAYOUT", in which she placed a map of her network:

For navigation this group is laid out roughly as

Notes

Group purpose

Hierarchy
Issues

Meeting Critical Requirements
(Introduction)

Criticality Levels

One user's node containing the browser layout map

This technique has several advantages. Surprisingly, users reported that the effort of coming up with a layout revealed aspects of their problem that were not obvious beforehand. But the map also organized their work within gIBIS, by easing the problem of deciding where to place new issues, and by providing a natural basis for finding nodes whose location *and* keyword information had been forgotten. This is an aid to navigation of large networks that the authors had never considered, and which we will pursue supporting directly in the tool. This experience has reminded us of the value of having "real users" testing out new tools.

Coping with change in an evolving network.

Any database has to have mechanisms for managing changes to the data it contains. Often this is at best a versioning scheme which allows older versions of the data to be marked and archived. In an application like gIBIS, however, the issue of change is of unusual importance, because the very nature of an "issue base" is that it is a vehicle for an evolving discussion in which older material may be accurate and highly important, inaccurate and only of historical interest, or anything in between. For example, the original form in which an Issue was framed may have been biased toward a particular Position, or may have contained a presupposition that was later made explicit and rejected. How should this "outdated" form of the Issue be handled?

In some cases the Issue and its discussion subnet may be isolated and simply wrong, in which case it will be easy to decide to archive that subnet and delete it. But more often there will be parts of the subnet that are wrong, misleading, or irrelevant, and others which are still quite relevant or important to the network, and which are directly linked to network regions where discussion is quite active. How can these partially invalid discussion segments be prevented from "poisoning" the network?

The answer seems to have two parts. One is that we need mechanisms for systematically indicating the age and relevance of network material, such as displaying older nodes as yellowed or frayed -- unless of course they have been recently visited and updated. Like the mechanisms suggested above for importance, salience, and confidence, age and relevance would be somewhat subjective measures that could only be partially automated. The other mechanism for managing change is completely human -- as issue networks grow in size and importance, it will become increasingly important for organizations to have people whose job is to maintain the currency and hygiene of the issue base.

CONCLUSIONS.

We have described the IBIS method, the gIBIS tool, and some preliminary observations about the use of the tool. Our experiments with gIBIS are informing our theory about the structure of design decisions and design rationale, and are providing us with important insights about the design of the Design Journal, a hypertext-based environment for system engineering which we will continue to design, prototype, and test in the next few years. More importantly, our experiences suggest that the computer is indeed a powerful medium for collaboration and debate among members of a team, but that the integration of computers into the fine detail of real work is attended by some severe breakdowns. Some of the breakdowns are due to inadequate interfaces, others to inappropriate underlying representations, and still others to insufficiently rich models of work practices and methods. Our experience with gIBIS suggests that we are just at the beginning of a long but exciting path, which will culminate when we have succeeded in making such tools as effective and transparent in structuring communication as the telephone has grown to be in simply transmitting it.

REFERENCES.

[BEN86] Benbasat, I., Dexter, A. and Todd, P., "An experimental program investigating color-enhanced and graphical information presentation: An integration of the findings". *Commun. ACM*(29)11. 1986.

[BRO82] Brown, J.S., "Notes Concerning Desired Functionality, Issues and Philosophy for an AuthoringLand". *Xerox PARC CIS Working Paper*, 1982.

[BUS45] Bush, Vannevar, "As We May Think," *Atlantic Monthly*, No. 176, pp. 101-108, July 1945.

[CON87] Conklin, J. "Hypertext: a Survey and Introduction". *I.E.E.E. Computer*, Vol. 20, No. 9, September, 1987.

[DEH72] Dehlinger, H. and Protzen, J. "Debate and argumentation in planning: An inquiry into appropriate rules and procedures". *Working paper #178. Institut fur Grundlagen der Planung I.A.* University of Stuttgart.

[DIC86] Dickson, G.W., DeSanctis, G. and McBride, D.J., "Understanding the effectiveness of computer graphics for decision support: A cumulative experimental approach". *Commun. ACM*(29)1. 1986.

[EVE86] Eveland, J. and Bikson, T. "Evolving electronic communication networks: an empirical assessment." *Proc. CSCW'86: MCC/ACM conference on computer-supported cooperative work.* 1986.

[HAL87] Halasz, F., Moran, T. and Trigg, R. "NoteCards in a nutshell." *Proc. ACM CHI'87: Human factors in computing systems.* 1987.

[HER85] Hershey, W. "Idea Processors." *BYTE*. June, 1985.

[HOR86] Horton, M. and Adams, R (Center for Seismic Studies, Arlington, Va.). "How to read the network news." Distributed by Mr. Adams quarterly over the USENET news network.

[LOW85] Lowe, David G. "Cooperative structuring of information: the representation of reasoning and debate." *Int. J. Man-Machine Studies*, Vol. 23. 1985.

[MAL86] Malone, T., Grant, K., Lai, K., Rao, R. and Rosenblitt, D. "Semi-structured messages are surprisingly useful for computer-supported cooperation." *Proc. CSCW'86: MCC/ACM conference on computer-supported cooperative work.* 1986.

[NOR86] Norman, D.A. and Draper, S. W. (ed.), *User centered system design*. Lawrence Erlbaum Associates, Publishers. 1986.

[REI85] Reiss, S. "PECAN: Program development systems that support multiple views." *IEEE Trans. Software Eng.*(SE-11)3. 1985.

[RIT70] Rittel, H. and Kunz, W. "Issues as elements of information systems." *Working paper #131. Institut fur Grundlagen der Planung I.A.* University of Stuttgart.

[SMI86] Smith, John B., Stephen F. Weiss, Gordon J. Ferguson, Jay D. Bolter, Marcy Lansman, and David V. Beard, 'WE: A Writing Environment for Professionals." *Technical Report 86–025, Department of Computer Science, University of North Carolina at Chapel Hill,* 1986.

[TRI83] Trigg, R. H. "A Network-based Approach to Text Handling for the Online Scientific Community." PhD. Thesis, University of Maryland, (University Microfilms #8429934), 1983.

[TRI86] Trigg, Randall, Lucy Suchman, and Frank Halasz, "Supporting Collaboration in NoteCards," *Proceedings of CSCW '86: the Conference on Computer-Supported Cooperative Work,* MCC/STP, Austin, Texas, December, 1986.

[WIN86] Winograd, T. and Flores, F. *Understanding computers and cognition: A new foundation for design*. Ablex Publishing Corp. 1986.