TAKING DESIGN SERIOUSLY: EXPLORING TECHNIQUES USEFUL IN HCI DESIGN

JOHN KARAT
TOM DAYTON

A workshop was held in Seattle on April 1st in conjunction with the CHI'90 conference. The call for participation in the workshop asked for position papers from individuals that described techniques found useful in some way in carrying out HCI design. For the purpose of the workshop, an intentionally broad view of what was considered as a technique was used (e.g., position papers that addressed design process, task-analysis methodologies, and skills necessary for design, were all considered as addressing techniques). In this report we present an overview of the workshop. Additional reports of subgroup discussions, focused on some of the major themes that emerged during the day, are presented in papers by Braudes, Hix and Casaday, Carter, and Notess.

The title of the workshop was "Taking design seriously: Exploring techniques useful in HCI design." The workshop was intended to provide a forum in which researchers and practitioners could present and discuss a range of techniques that they felt provided useful information to the design process. Of the 35 position papers submitted for the workshop, 23 were invited to participate (21 were actually represented at Seattle). The workshop actually extended beyond the one day of face-to-face discussion, and included distribution of position papers to participants in advance of the workshop, and an electronic mail discussion of the meeting plans and agenda.

Basically the workshop was divided into three phases of activity. For the first part of the day everyone had 5 minutes to summarize what they felt were important issues for HCI design. This was not necessarily a summary of the material contained in the position papers, since everyone was assumed to have read them. It provided introductions of people to the group, and the beginning of an effort to identify key topics for additional discussion. After the introductions, there was a brief discussion of how to divide the participants into discussion groups. This resulted in forming four subgroups: Group HCI Design - Problems and Prospects, Design Tools and Craft, Interface Design Decisions and Representations, and Design Methodologies. While we had some notions of how the topics differed (e.g., that design methodologies includes higher level techniques than representation methods does), we did not attempt to provide detailed definitions of the topic areas before breaking for subgroup discussions.

The workshop broke into four groups for the next three hours with the understanding that on reforming, each subgroup would present a 20 minute summary of its discussion (summaries from the subgroups are contained in the additional reports in this issue). The summaries were presented with a chance for discussion. The workshop closed with an open discussion and plans for follow-up activities, including expansion of position papers for a book on the workshop.

Position Papers

The workshop position papers provided an interesting look at the wide range of topics and approaches that must be considered in HCI design. The community is concerned with far more than evaluation methodologies. Below, we provide very brief summaries of the position papers (the workshop participants have been asked to elaborate the papers into chapters for a book to be published early next year). For this report we group the papers into five headings: group factors in design, integrating methodologies, methods for task analysis, new design paradigms, and design skills and craft. The grouping here is really just one way to view the relationships between the papers, and does not necessarily reflect the discussion group breakout; some participants elected to participate in discussions unrelated to their papers.

Group Factors in Design

A number of the participants focused on social factors in design, particularly facilitating group interactions. John Karat from IBM Research ("Supporting collaboration in the design team to meet HCI objectives") presented a summary of work being done in collaboration with John Bennett. Bennett and Karat have focused on developing a perspective on the social side of the system design and development process. They point out that it is possible to have valuable and potentially effective technological tools and still miss the target for achieving systems that provide effective HCI access to function. In design meetings they see the need to foster parallel and complementary discussions of 1) available technology and of 2) ideas on how it can be used during design.

George Casaday from DEC ("Methodological support for the human interface design meeting") also discussed the role of group activities. His paper described a framework for collaborative work on human interface design in the context of a design meeting. The framework creates a structure for conversation and a flexible discipline for thinking; it is much influenced by current practice in software engineering, particularly in object oriented analysis.

Mark Notess from Hewlett-Packard ("From madness to method: Making the most of human factors input during user interface design"), described experience with methods for improving the relationship between human factors professionals and other members of a development team. Recommendations based on experience covered ownership of design (separating the roles of evaluators and designers was seen as necessary), including designers as test observers, involving human factors specialists in early design discussions as external consultants, developing hitlists of issues for focus, and adoption of test methods that work with immature software.

Integrating Methodologies

Several of the participants addressed attempts to integrate user-centered techniques with software engineering methodologies. The claims were that existing methodologies give insufficient guidance to designers for addressing usability issues. Bob Braudes from Georgetown and George Washington Universities ("Integrating conceptual modelling into software engineering") presented one of several frameworks in which user-centered design topics are combined with software engineering methodologies. His system (ConMod) represents an approach for incorporating the notion of conceptual models from user interface practice into a software engineering design practice. Conceptual specifications of a model are generated and used to ensure that the design team and members of the end-user community agree on the system being developed. This specification may also be used as a basis for the prototyping and implementation effort.

Other integrated methodologies were presented and discussed. James Carter from the University of Saskatchewan ("Analyzing structured task analyses leads to better HCI designs") described an attempt to integrate user-centered design with software engineering methodologies. His multi-oriented task analysis (MOST) methodology grew from an elaboration of the ten task analysis questions of Rubenstein and Hersh (1984). A MOST task analysis includes an analysis of: the various types of *users* and the structure of their relations to each other; the existing *tasks* and their structures (and tasks not currently done that should be a part of the application); the logical content and structure of the *data*; the existing *tools*; and any *constraints* on the design of a new system.

Gregory James from the University of California, Irvine ("Applying software process modelling to user interface design") discussed the problem of the inability of software engineering models to handle user interface issues. James seeks the solution to the problem in a marriage of software engineering process modelling with the user-centered advances of HCI. Process models can provide the rigor and structure for integrating the human computer interaction advances. Together these two disciplines can aid developers in the design and construction of user interfaces.

Chris Rouff from the University of Southern California presented work done in collaboration with Ellis Horowitz ("A methodology and system for graphically specifying and rapid prototyping user interfaces"). Rouff described a methodology and supporting system currently under development for graphically specifying and rapidly prototyping multiwindow user interfaces. An interface is specified

by drawing the components, indicating the flow of control between the components, and providing program semantics to be executed. As the designer draws the interface, the system builds and maintains a modified statechart representation of the interface. The methodology consists of the steps the designer takes to specify the interface.

Bernard Catterall from the HUSAT Research Institute ("The HUFIT Planning, Analysis and Specification (PAS) Toolset - easing product designers effectively along the road to user-centered design") discussed experiences with a tool set in HCI design. The HUFIT PAS Toolset was developed to provide design teams having little or no previous expertise in human factors, with a set of high-level tools for the effective input of user and task information into the planning, analysis, and specification phases of the information technology product life cycle. The five components of the toolset are user mapping, user and task characteristics, usability specification for evaluation, user requirements summary, and functionality matrix. Catterall noted that the continued demand for training seminars on the tool set beyond the end of the project indicates the growing demand for tools of this nature.

Susan Harker, also from the HUSAT Research Institute ("Human factors in the design of information technology systems: A UK and European perspective") provided an additional view of European projects in HCI. A number of projects done as part of research and development programs sponsored by national bodies and the European Community have contributed to the development of our understanding of designers and design processes, and offered the opportunity to explore various aspects of design support. The studies that Harker addressed included studies of the design of special purpose software (bespoke products) and studies of the design of generic or off-the-shelf products. A number of issues that caused difficulty in effectively bringing a user-centered view into the process were identified, and efforts are underway to introduce new methods and tools into design.

Methods for Task Analysis

Some of the position papers focused on the analysis and representation of the tasks to be carried out with the system under design. Allan MacLean from Rank Xerox EuroPARC ("Design Rationale: Developing a framework for software design") also addressed details of a general design framework. Work that he presented has been carried out in collaboration with Victoria Bellotti, Tom Moran, and Richard Young. MacLean described "Design Rationale" as a semi-formal notation used to represent the design space around an artifact being produced, and suggested that it is an appropriate design output. This space includes an explicit representation of reasons for choosing among alternative options. The main concepts currently used for the representation are "Questions" which highlight key issues in the design, "Options" which are effectively answers to the questions and "Criteria" which are the reasons that argue for of against the possible options.

Peter Polson attended to present work jointly done with Clayton Lewis, John Reiman and Cathleen Wharton at the University of Colorado ("Cognitive walkthroughs: A method for theory-based design of user interfaces"). This work represents an attempt to bridge a gap between cognitive psychology theory and HCI design practice. All sides agree that it is difficult to apply current theoretical models within the constraints of real-world development projects. Polson et al. derived a cognitive walkthrough procedure for systematically evaluating features of an interface in the context of a theory of exploratory learning. Consideration of the walkthrough procedure sheds light on the consistency with which such a procedure can be applied as well as on the accuracy of the results.

Rex Hartson, Deborah Hix, and Antonio Siochi, from Virginia Tech ("Support for user-centered design from the behavioral view") presented a possible notation (User Action Notation or UAN) for capturing behavioral information for software engineering. Since software engineering traditionally deals with algorithms, data structures, data abstraction, and program correctness, there is reason to believe it can help with the mechanics and structuring of the interface development process. However, since software engineering does not deal explicitly with users, it would be a surprising coincidence if it also provided guidance for a user-centered focus. The UAN was developed to address this issue.

John McGrew from Pacific Bell ("Tools for task analysis: Graphs and matrices") addressed experience with a different form of task analysis. McGrew stresses the use of representational forms (graphs and matrices) that are understood by system developers to describe users' tasks. Methods derived from graph and matrix theory facilitate transfer of knowledge from text and outline into a form more useful to system developers.

New Design Paradigms

Some of the papers called for new approaches to design, involving either different models of how to do design, or different techniques to employ. Andy Cohill from the College of Architecture at Virginia Polytechnic ("Information Architecture: A new approach to software development") described an architecture that seeks to account for not only the

technical aspects of information systems development, but also the organizational and environmental aspects of development. Recognizing that systems design is a process that can be solved by more than one development method will be a major step, but what is also needed is a new kind of project manager -- the information architect -- who understands the multiple dimensions of development and can guide that process in a rapidly changing environment.

Meredith Bricken from the University of Washington ("Virtual world design") brought the workshop a view of designing systems for a new type of environment. Bricken addressed virtual world design with consideration of a number of topics. These included differences from traditional interface design, different models of virtual reality (system, user, and design models), questions of what makes effective, useful virtual worlds, and interface technology and software tools we need to develop for virtual world interaction.

Thomas Lanning from GTE ("Let the users design") stressed the importance of including potential system users in design. The role that he advocated is more than users' pass/fail validations in an iterative design process. Value was seen in including the users in all the design phases that influence their perceived value of the end product - such as concept generation, requirements definition, specification definition, and detailed design. It is likely that this topic would have received greater attention had there not been a separate workshop on it taking place at the same time.

Lawrence Miller from the Aerospace Corporation represented work done in collaboration with Steven Lewis and April Gillam ("A model-based framework for designing interactive computer systems"). From experiences with developing complex aerospace systems, Miller et al. argued that design of interactive computer systems must move from the marvelous hand crafted efforts of the first Xerox Altos and Stars, to an automated process of very good, but not necessarily optimal, interfaces. Further, all decisions about the interaction - the nature of windows, the size, shape, and format of menus, the selection of appropriate output media and modes, and their coordination (the integration of various input and command forms) - must be stated clearly and obviously to designers, with the interface having predictable behavior, and allowing reasonable default behavior when more detailed methods are not needed. They see a route to this through the development of knowledge-based user interface management systems.

Design Skills and Craft

Several papers presented discussions of the importance of developing skills to be used in design, rather than providing details of some specified technique. David Wroblewski from MCC ("Interface design considered as a craft") advocated that building effective human-computer interfaces is a craft, performed by skilled practitioners engaged in a detailed and extensive inquiry into the particulars of a task domain. Supporting the design of good interfaces requires understanding and supporting craftspeople at their craft. Wroblewski focused on important implications of such a craft view. First, craftspeople do not rigorously separate design from implementation. Therefore, their tools must support design in the practice context, rather than in a prior, separate phase. Second, craftspeople use their practice both to create solutions and to evolve their tools, blurring the distinction between tools and materials. Therefore, design tools must themselves be modifiable, else an important result of practice is lost.

Patricia Craig from Microsoft ("Graphic design in software development") discussed the design team's need to include individuals with particular skills. Craig emphasized the role of graphic designers as individuals whose objective is to create products that communicate as well as look pleasing. The composition of the design team was also mentioned as an important factor in several other position papers (specifically those focused on group factors), though not in as much detail as in Craig's paper.

Harold Miller-Jacobs from The Analytic Sciences Corporation ("Rapid prototyping: How valuable in computer systems design?") addressed various aspects of the value of rapid prototyping in design. In addition to the obvious benefits of prototypes when compared to text specifications for clearly conveying interactive aspects of HCI, Miller-Jacobs stressed the focusing value of prototypes. Rapid prototyping ensures that everyone is addressing the same system, because there is a representation of it on the screen in front of them.

Gary Klein ("The use of video technology for the fast prototyping of artificially intelligent software") extended the notion of rapid prototyping to include the use of other technology (in this case video) as a means of delivering the prototype. Klein provided a case history and description of the use of scripted enactments of interactions with a system. He described improvements to design which were attributed to 1) the structure provided by the process of developing a detailed screenplay; 2) the ability to visually illustrate complex technical concepts; 3) the ability to inexpensively illustrate expensive technology, and, 4) the ease of distribution of this material on videotape.

John Tang from Xerox PARC ("Applying video-based interaction analysis methods to study users and guide

the design of new technology") also discussed the use of video in design by presenting a case study in the development of a graphics system for cooperative work. Tang stressed that designing technology to meet users' needs requires involving social scientists who are skilled at analyzing human activity in concert with designers who are skilled at translating those analyses into design prototypes. Intensively analyzing users for innovative insights into the design of new technology, and informally analyzing for evolutionary improvements in iterative design, has been leading to a better understanding of collaborative activity and prototype tools to support that activity.

Subgroup Presentations and Discussion

We devoted the last third of the day to reports from the four subgroups accompanied by discussion from the whole group, and to a general discussion. The subgroups--Design Methodologies, Design Tools and Craft, Interface Design Decisions and Representations, and Group HCI Design-- summarized their meetings in their papers in this issue of the *Bulletin.* In this overview we synopsize the oral presentations and the resulting whole-group discussions. The book of the workshop papers also will contain an overview chapter that will describe the discussion in more detail.

Design Methodologies

The members were Bob Braudes, Rex Hartson, Greg James, Larry Miller, Hal Miller-Jacobs, Peter Polson, and Chris Rouff. The discussion within the group focused on techniques for analyzing the tasks to be carried out by users of systems. In summarizing the subgroup discussion, Polson represented task analysis as a series of representations, progressing from a collection of objects and the actions taken on them, down to the physical requirements of the action sequences, the cognitive operations behind these sequences, and finally the feedback. Braudes mentioned that it is also necessary to consider task interrelationships to determine completeness and consistency of a conceptual model. It was clear from the discussion that the appropriate granularity for identifying the task to be analyzed, could vary during the design process.

Polson claimed that once we have specified the action sequences (in some form, using some technique), we can deal with various learning and performance issues. For example, Polson's use of cognitive walkthroughs has focused on learning, whereas the GOMS model's real strength is in performance. Hartson commented that User Action Notation has also been used to address performance (e.g., cognitive loading).

Given that there are various techniques for analysis (the discussion did not seek to define a best technique) and various ways to view tasks, can we provide practical guidance? Miller talked about the need for a representation that is uniform across all the entities and relations of interest in the design of interactive systems, so the practice isn't swamped by theoretical concerns. He suggested that we have accurate low level cognitive theory that tells us whether a type of interaction will work and where errors might occur, but that we need to take the theories Polson, Hartson, and others have been working on, and provide them in a framework that user interface engineers can use.

Design Tools and Craft

This group was composed of Meredith Bricken, Jim Carter, Andy Cohill, Patricia Craig, Tom Lanning, Allan MacLean, and Dave Wroblewski. The discussion began with an attempt to classify differences between engineering design and "real" (artistic) design, and with the assertion that the craft of HCI design lies between these two extremes. Wroblewski commented that the group struggled a lot with the absence of a standard nomenclature for these issues, and pointed to literature that discusses design (in all arenas, not just in HCI) as discovery. He advocated thinking about methods as aids to discovery--means of revealing constraints, and of opening up the design space by breaking down existing constraints. Maybe current tools can't do the creative things that differentiate good design from mere automated layout, but perhaps tools can be built to catalyze creativity.

Polson commented that much of the work on low level cognitive modeling has the same motivation in the HCI domain, as finite element analysis and other standard engineering methods have in the structural domain (e.g., buildings and bridges). One difference is that structural architects use a lot more of their resources in evaluation than we do in software design. Software is interesting in that you can construct an artifact, look at it, go back and tear it up, and try again. But with the building next door you wouldn't do that. The tools should be as abstract and theoretically driven as possible, because (1) we can do it computationally, which can ultimately be more cost effective, and (2) the theoretical structures give us a set of categories to cumulate our design experience.

Cohill gave what he labeled a minority view: Tools make it easier to manipulate complex designs, but they are not part of design; people design in their heads. McGrew and Wroblewski responded that tools are needed to scale up design activity from a small team to a large one. Some designs are too large to fit in one person's head, so each designer must explicate their thinking for communication to the others. Cohill agreed, but emphasized that it's dangerous to think that we'd necessarily have better design if we had better tools and methods. The tool can become the

driving force of the design, the quantitative evaluation can become the sole definition of design quality, and thus the design can become limited by the bounds of the tool. MacLean and Braudes disagreed, saying that tools and other methods, especially evaluative ones and even quantitative evaluations, can give insights into design. Cohill accepted their point, but drew the distinction that evaluation and design are separate activities, which we often confound. They are one entity only in the sense that the design process is an iterative cycle of the two separate activities. McGrew objected that we cannot eliminate the dangers of relying too much on tools. In many real-world industrial environments the tools are part of the official design process. Often, knowledge of the tools is the only formal training given to people who are assigned the task of design.

Lanning said that iteration and evaluation are abstract, general techniques that are the kind of meta-level tool we need. We already have many tools specific to narrow applications or domains, which were built by people doing the craft of design to solve an immediate problem. For example, a technique for laying out iconic interfaces isn't useful for an audio interface.

We need generic methods both to help people do their job and to stimulate creativity. Rouff clarified the point by saying that it might be useful to distinguish between the activity or process of design and the artifact being designed. He felt most people would agree that tools and methodologies to help represent, manipulate, and evaluate the design artifact are beneficial, but that there are certain aspects of the design *process* to which it would be difficult to apply tools, and perhaps that's where many hard research questions arise.

Interface Design Decisions and Representations

The members of this subgroup were George Casaday, Tom Dayton, Deborah Hix, John Karat, John McGrew, and Antonio Siochi. Hix stated the group's goal as discovering how to capture design decisions. The group decided that a key obstacle to design is the absence of a widely accepted terminology, structure, method, or process for organizing the development of the human-computer interface, in particular for communicating our thoughts about design or development issues. Three notations were proposed: McGrew's graph/matrix technique, cognitive walkthroughs, and user action notation. The group addressed three sub-questions: What categories or foci are involved in design decisions? What broad categories of things do you have to do in design? What approaches do *we* use, and how do they map onto the various categories? The group used a diagram from Casaday's position paper which presented six foci for HCI design: user mental activities, user physical activities, context for design, functional design, physical design, and implementation policies (for details, see the Hix and Casaday report in this issue).

MacLean commented that one way to choose a notation is to pick one that can handle all the foci of the model; user action notation, for example, covers three. Cohill asked whether a notation for mental activities was feasible--a notation that covered all the nodes. Karat said that the snowflake model is intended not to be all inclusive, but to help put tools and methods into broader context. It may be that different techniques are more appropriate for the different foci. Carter said similar models exist, such as the MOST methodology in his position paper.

Lanning asked where the entertainment value (i.e., aesthetics, polish, fun level, user preference) of the interface goes in this snowflake model of design issues. McGrew said it could be folded into the *functional design* focus. Lanning suggested that *implementation policies* might comprise forty to fifty percent of design, instead of the single vertex it is accorded in the snowflake model. McGrew assented that implementation is important and that it is really part of all the other vertices, but he thought implementation should continue to have its own vertex to emphasize its importance.

Miller thought it ironic that user physical activities occupy little of the model, whereas they get the money and glamour in the real world--the gee whiz stuff such as the data glove. Casaday explained that the model gives physical actions short shrift because in the real world, there are no choices--designers are stuck with a mouse and a bit-mapped screen.

Group HCI Design

This group was Bernard Catterall, Susan Harker, Gary Klein, Mark Notess, and John Tang. Notess explained that they first listed the characteristics of multidisciplinary user interface design groups and the tasks the groups set for themselves: For instance, design groups exist in an organizational context that changes often, and membership is interdisciplinary. These characteristics raise issues such as how to communicate intuitions and visions to implementers who just want to know what to do. The group then came up with some solutions: Prototyping and scenarios are good for all the members of an interdisciplinary design team, because people don't need much training to relate to such concrete, narrative, rich notations. Another solution is to have a decision matrix that tracks all the different constraints and their weighting, and then looks at the decisions that were made, and why. He mentioned that there's been some work on a spreadsheet model

for doing just that. The subgroup also concluded that software engineers should be trained in general design, not just in things like algorithm design. Finally mentioned was the need to bring the tools to the designers who need them, instead of waiting for the people to come to the tools.

Lanning responded to the group's enthusiasm for prototypes as the communications medium, by pointing out that whoever controls that medium has a political stranglehold on the design activity. Suppose a graphics designer wants to use some medium to show an idea, but the medium is controlled by a software organization that thinks the idea is trivial? Bricken suggested that this could be overcome by placing the medium under control of a committee representing all the users of the medium. Hartson suggested that instead the tools be made usable by anyone--for example, a Hypercard stack could be fed into anyone's computer.

Harker said that the subgroup found a group processes reason for iteration's importance, thus supporting the other subgroups' emphases on iteration: Without iteration, each member of an interdisciplinary team just throws in their contribution and sees it disappear. With iteration, they get repeated chances to compare views across disciplines.

General Discussion

Polson started by saying that if we're going to make any real progress in the field, it's necessary to bring some order out of the chaotic, large amount of extant work on task analysis, user representations, and so on. He mentioned that workers at the Fraunhofer Institute in Germany had, as of last summer, uncovered 30 or 40 different methods of task analysis! Software engineers have every right to be suspicious of methods we suggest, Polson remarked, because we can't give a convincing (to them) argument for any one method over the plethora of others, and because many methods have existed for years but weren't taught in the engineers' undergraduate training.

Another problem Polson mentioned was that these task analysis methods are not user friendly. He asked the workshop to think about how long it took any of us to really master a professionally relevant notation, like mathematics or logic or a programming language. He asserted it's a task at least as complicated as learning to read. Notess suggested that a combination of English, graphics, and prototyping is better than inventing a new notation that *no* one already knows, even if that makes some non-English speakers slightly disadvantaged. Karat added that looking at a prototype--sitting down in front of the system to see if you like it--beats being presented with a long analysis with some well based theory and quantitative predictions, such as a GOMS analysis. Tang threw in an interesting analogy from his group: Conveying design by prototypes or scenarios is more like learning from apprenticeships than from formal textbooks. He agreed that natural language and prototypes are easier to understand than new, formal notations are.

Hix agreed with Polson that we need success stories to entice people to use new methods, but asked, how are we to get them? Polson responded that he is trying to do that by exporting the cognitive walkthrough method. So far he has discovered that it cannot easily be applied to large tasks or to many subjects; an interface that allows the user to do 500 tasks is too onerous to analyze. Karat noted that Hix's study, in which 80% of the users of User Action Notation could follow it exactly, is an example of the sort of case study that we need. He mentioned that this sort of positive information hasn't been carried into the literature very well.

Notess pointed to the bright side: User testing finally has become successful now that it has been refined, made less expensive and time consuming, and more adapted to particular situations. Polson emphasized this point for interfaces that carry high costs of failure, such as the current generation of automated aircraft.

Braudes thought that designs can't be tested by either showing users prototypes alone or showing them design documents alone, because there are some things in the documentation that are not going to show up in the prototype no matter how good the prototype is. Braudes advocated use of some form of structured English that might be easy for users to learn. Klein agreed, and added that we must be clear about who the audience is: Formal notations may be fine for communicating among researchers, but prototyping is needed for getting users' comments. Braudes disagreed, saying that showing users a pretty interface is not going to communicate whether the design does certain tasks. Klein persisted that we know from experience that users have trouble learning new notations.

Carter jumped in with the assertion that the set of records in a MOST analysis provides a structured task analysis record, but informally and in English. The methodology provides ways to further structure the results if desired. Harker asked if the design solution generated from that analysis can be reflected back to the user. Carter said yes, because much of the solution's documentation can be done in English sentences. Catterall believed both forms are needed, but was convinced it is difficult to make the formal notations user friendly. For instance, some academics have objected to his representation because it *is* user friendly. McGrew thought that users must be trained some, because there is no notation that everyone can use. He's been able to train field technicians and

contractors in little time.

Siochi said that when you produce the description of the interface using English, you typically produce a book, and that we all know users faithfully read their user manuals! Notess brought up the example of the OSF Motif style guide. Hix contended the Motif guide was not very satisfactory, and suggested a more concise presentation such as User Action Notation. Notess and others agreed that more precision is needed, and said OSF is trying to do that. But Hartson objected that there is a limit to the precision of natural language. He went through the novice's Macintosh manual, and discovered that some of the instructions are precise enough only for some contexts. For example, it says that clicking the mouse does a selection, which implies that selection happens when you push the button down and let it up, but that's incorrect--selection happens when the button goes down. Though most people don't care about the difference, designers and implementers do. Carter rejoined that Hartson had just explained in natural language what really goes on, so Carter didn't believe natural language is quite so bad for description--people just use it sloppily. He didn't think artificial language, with its problems of translation and information loss, is the panacea either, so he favored precise English.

Hartson objected that there are many style guide users who will say it's a losing battle to go around mopping up English sloppiness, and Hix contended that getting people to write precise English is very difficult. Hartson observed that this argument was exhausted in the database and AI communities some time ago, and that it ended favoring artificial language. McGrew contended that precise natural language is almost impossible to read (any mathematics text is an example); some redundancy and ambiguity are necessary.

McGrew argued that we can't create a universal notation, so we must make one that is just natural enough for the notation's users to be able to learn it in the training time we can afford to give (we *must* train them some). Karat distinguished between (1) designing systems for a particular end user population: We will design some systems for experts in some domain, from whom we can require much background knowledge. We will design other systems for people who have never seen this system before, who should be able to walk up and correctly guess how to use it; and (2) designing systems for designers to use. We ought to attend to whether the design techniques we're advocating are really useful for designers, not for users of the interface.

Lanning shifted gears by declaring that *many* design techniques are necessary. Maybe at one time, he said, the methods of the software engineer were good enough, but then we decided we needed experimental and cognitive psychology. Now we finally admit that we need graphic designers, and people who understand the organizational environment. The days of giving the interface to someone with a single background are over, because a good job requires a large range of techniques. He concluded that, unfortunately, there is not yet a recipe for putting together the perfect multidisciplinary design team. Karat added that he hopes we're moving away from assuming there is a single formula for success. He observed that for a while, there seemed to be a search for a single Holy Grail of design techniques. Now there is an attempt to create some framework in which a range of things can be found, so we can more intelligently compare what the techniques cover.

What is design, and how should we do it? Karat concluded the discussion by summarizing the workshop. He stated that the ideas presented ranged from including people with certain skills in the process, to cognitive walkthroughs and user action notation. As techniques move from theory to practice, the focus becomes the costs and benefits of the techniques. The costs for bringing any technique into practice include learning; methods that require PhDs and ten years experience, unless they produce the HCI equivalents of Beethovens, are unlikely to get used. This is where case studies and success stories are needed. If you've got a technique that you're fond of, people will latch onto it more from successful case studies than from just another paper on the next generation of a familiar technique. While the workshop did not develop a clear description of HCI design, it did offer participants a chance to hear the case for a number of techniques that were claimed to be useful.