Some Experiences With Cooperative Interactive Storyboard Prototyping

Kim Halskov Madsen

Information Systems and Software Engineering
George Mason University
Fairfax, Virginia 22043-4444
halskov@daimi.aau.dk

Peter Aiken

Hypermedia Technologies Laboratory
George Mason University
Fairfax, Virginia 22043-4444
703/993-1640
paiken@gmuvax2.gmu.edu

Abstract

Certain types of products appear to be continually associated with poor interfaces. Especially for these product types, the need for active end-user participation in development activities is seen increasingly as critical to quality interface development. Cooperative Interactive Storyboard Prototyping (CISP) is a development approach facilitating active end-user participation in design activities that help to address these apparently difficult design problems. Key features of CISP include: 1) reducing the time and effort required to produce iterations of prototype interfaces; 2) support for interactive prototype use, evaluation and modification; and 3) creation and use of modifiable, domain specific building blocks.

Keywords

Design Process, Participatory Design, Design Support Tool, Design Methodology, Rapid Prototyping, Group Work, CSCW

Introduction

The interfaces of computer systems embedded in certain types of consumer electronic products are frequently targets of criticism. Perhaps the most frequently cited examples of poor user interfaces are those associated with video cassette recorders (VCRs). To cite a popular reference, the April 29, 1991 issue of Business Week reported that only 3% of total TV viewing time went to shows that were recorded by the users. Further surveys report a high percentage of VCR clocks flashing off and on, indicating they aren't set. Two of the obvious consequences of these poorly developed human computer interfaces are: 1) a considerable portion of the population is unable to benefit from the primary functionality of these products; and 2) the loss of potential recording tape sales represented by the flashing clocks.

In *PDC'92: Proceedings of the Participatory Design Conference*. M.J. Muller, S. Kuhn, and J.A. Meskill (Eds.). Cambridge MA US, 6-7 November 1992. Computer Professionals for Social Responsibility, P.O. Box 717, Palo Alto CA 94302-0717 US, cpsr@csli.stanford.edu.

The need for active end-user participation in development activities has been acknowledged and is reflected as increasing interest in professional gatherings such as the Computer Supported Cooperative Work and Participatory Design conferences. Prototyping has contributed to the success of end-user design activities [7]. Both theoretical arguments and empirical evidence indicate a strong cause and effect relationship between development approaches permitting realistic conditions for prototype evaluation and successful interface development efforts. Better solutions are achieved when the user is better prepared to participate in development processes [10], [11].

Storyboard prototyping is a variation on the general 'plan to throw one away in order to get it right' school of software development promoted by Brooks and others [1], [3], [8], [13]. As in film production, the use of storyboards in the development of computer systems is a way to 'sketch out' the future system early in the development process. In an effort to verify the requirements, the developer uses nonfunctional mockups, a technology dating at least to the 1930's, to illustrate a task driven view of the proposed system for the user. The concept of *iteration as a discovery process* is the key to prototyping. Each successive iteration brings the prototype one step closer to correctly representing the user needs.

Delay associated with production of the next version of the storyboard is a source of frustration for storyboard developers. Too much time between user review sessions leads to loss of cognitive momentum and can introduce errors to and perpetuate omissions in the development process. The following quote from Steve Andriole's first edition of *Storyboard Prototyping* succinctly states the current iterative production approach. "The screens were developed on an Apple Macintosh. The board was demonstrated for comment and changes made." [1, p. 73].

Inspired by Scandinavian research into cooperative design [11], the thrust of the Cooperative Interactive Storyboarding Prototyping (CISP) approach is to involve users more actively in the prototype interface development. CISP empowers users with tools and techniques encouraging them to interactively contribute to real-time, storyboard use, evaluation and modification. Crucial here is the concept of

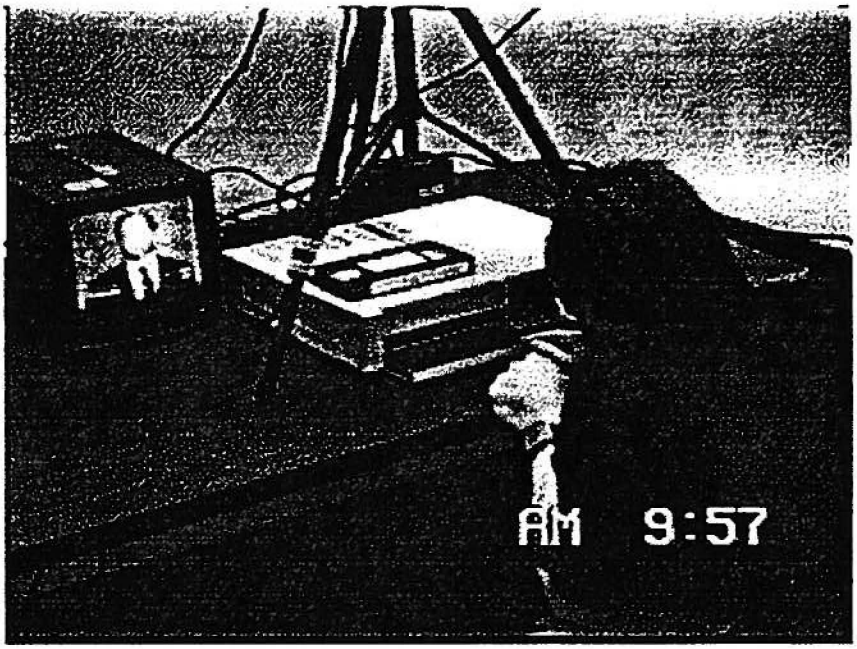

Figure 1. Video Recording Setup and View from Scene Camera.

the role of the user changing from *reviewer* to *co-developer*. By drawing the user into the production process itself, CISP seeks to reduce the delays typically associated with the production of the 'next iteration' of the storyboard. In addition to reducing iteration delay, CISP also provides situations where users can evaluate the storyboard prototypes under realistic circumstances and modify them in real time.

While the effort described is very much 'work-in-progress,' we have had some experience using CISP, having used it to develop a VCR interface and evaluated the efforts of a small group of Masters-level students, studying Human-Computer Interaction, as they used the interface to complete certain VCR specific tasks. Preliminary results, what Brooks labels *observations* [3], are promising. When applied to prototyping, the increased, collective, cognitive momentum seems to favorably effect the quality of the resulting development efforts. The next section of this paper contains our motivation for developing the approach and a description of the CISP-Tool. It is followed by a brief description of related research approaches. A report of our observations and experiences is followed by a set of preliminary conclusions.

THE CISP-TOOL

A large part of the impetus for this investigation came shortly after we excitedly unpacked a new piece of multimedia equipment for the Hypermedia Technologies Laboratory. We anticipated the unit, the NEC PC-VCR S-VHS video cassette recorder with an RS-232 interface, would be an important tool - augmenting our efforts in the lab to apply hypermedia technologies to the analysis phase of decision making and problem solving. Unfortunately, while the unit was technically functional, correctly interfacing with the lab's Macintosh™ computer network, the user-computer interface was no better and in some ways worse than any of the hundreds of other available video cassette recorders, lacking even on-screen programming.

Deciding that anything we did would be an improvement over the current interface, we began analyzing user interaction with the existing interface. As part of our analysis, we attempted to use scene and overhead cameras as shown in Figures 1 and 2. Finding the 'talking out loud protocol supplemented with video images' approach unsuitable for eliciting user comments, we began to develop the CISP-Tool with an eye towards creating a solution permitting a general approach to these difficult problems. VCRs have limited interaction modes; most user actions consist of pushing buttons. Limited interaction modes made it easy to develop a storyboard for this prototype. As a HyperCard extension, the CISP-Tool is not limited to situations where user interaction consists of just pushing buttons, but permits the use of external commands and functions which allow developers to create fully functional interfaces such as the one created for the PC-VCR.

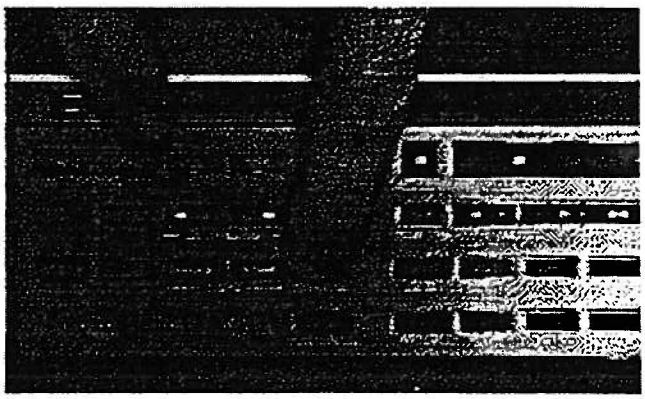

Figure 2. Upside Down View from the Overhead Camera.

The next section of the paper describes how CISP is used to achieve two specific participatory design goals: 1) interface development using domain specific building blocks; and 2) support for interface use, evaluation and modification by users.

Interface Development Using Domain Specific Building Blocks

The more users are able to work with familiar objects the better they will be able to relate to the development process. CISP supports collaborative interface development by permitting the user to combine building blocks made up of domain specific objects. Figure 3 shows a sample building block, switches used to move between system states such as "power on/off," "s-vhs on/off," and "record speed fast/slow." The lower part of the illustration represents the VCR control panel and the upper part represents the VCR display panel. As the switch is the turned on and off, the display changes accordingly. Typically, this type of building block is created using multiple button and text fields. CISP permits developers to create this type of composite domain specific object. The switch object can be duplicated using a single copy/paste operation and, if necessary, modified afterwards. It can be dragged as a unit using keyboard adjuncts. In addition, our observations show that users can perform the above operations after a five minute introduction to the tool.

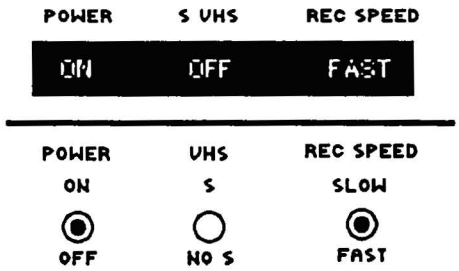

Figure 3. CISP Building Blocks.

The lower part of Figure 4 represents the VCR control panel while the upper part represents the VCR display panel. This type of building block was also initially created from a collection of buttons and fields. Additional building blocks of this type can be created with copy/paste and subsequent labeling operations using a dialog box. In the example, the day of the week may be set directly by pushing the specific day button. The display changes accordingly. This particular feature offers the possibility of using a single menu selection to change the panel and display style during a design session so the user can more effectively evaluate the choices. Figure 4 also shows another set of domain specific building blocks. The figure shows a composite object type that has proven to be useful for selecting between and displaying a number of states, for example the day of the week or time of day.

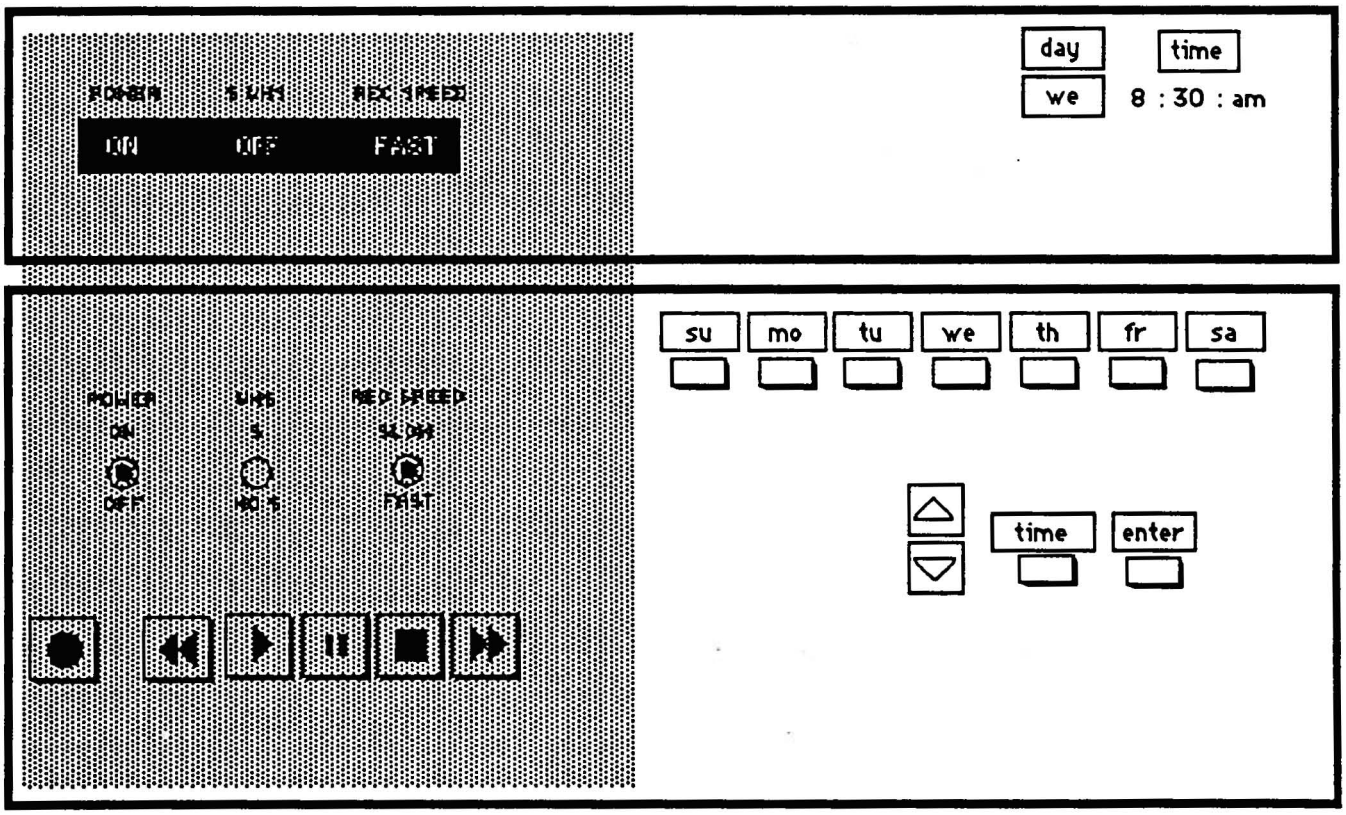

Figure 4. An Alternate Means Of Representing System States.

Figure 5 illustrates an alternative where the day of the week can be set by first clicking the "day" button and subsequently clicking the "up arrow" or the "down arrow" buttons instead of a display showing each day. An additional concept illustrated in the preceding two pictures is use of the "gray shade." Users can reduce clutter by determining how much, if any, of the display panel is visible. The gray shade emulates the use of plastic shades used on VCRs to reduce interface complexity and permit developers to quickly modify the level of interface complexity. In this way, CISP encourages rapid generation and efficient evaluation of a large number of completed, or partially completed, potential design alternatives.

Support For Interface Use, Evaluation and Modification By Users

Current prototyping tools do not provide much support for system use and evaluation. Hypercard's inherent ability to mimic other interfaces make it an ideal base for adding application specific functions via externals enabling the creation of evolutionary prototypes [9]. This allows the user to interact with and evaluate a close surrogate of the system under development in order to quickly 'get a feel' for it. CISP is capable of capturing and reproducing user actions as they interact with the storyboard. At the same time it eliminates one of the most glaring defects in our video capture process - identifying which button the user actually pushed during evaluation sessions. (Button selections were not clear even when user comments were added to the videotape.) By recording and 'transcribing' actions and permitting 'slow motion' replay of the user interaction CISP can provide users and developers the opportunity to play back and evaluate the *actual* rather than the *hypothesized* user-system interaction.

RELATED APPROACHES

There were three primary inspirations to the development of CISP. Each is discussed below.

Trillium

Trillium is a computer-based environment for designing interfaces for machines such as copiers and printers [12]. We adopted a key Trillium philosophy: reducing the impact of the use-evaluate-modify cycle by offering developers use of an interactive interface construction kit. Trillium is an industrial design environment while CISP is in the experimental stage. CISP expands the Trillium concept in two directions: first, it seeks to reduce the amount of programming required by substituting direct manipulation techniques like clicking, dragging, and copy/paste. Second, Trillium is a tool for software developers. We suggest an expanded role for this kind of tool: to be used by end-users working in conjunction with developers as the requirements for the product are being developed.

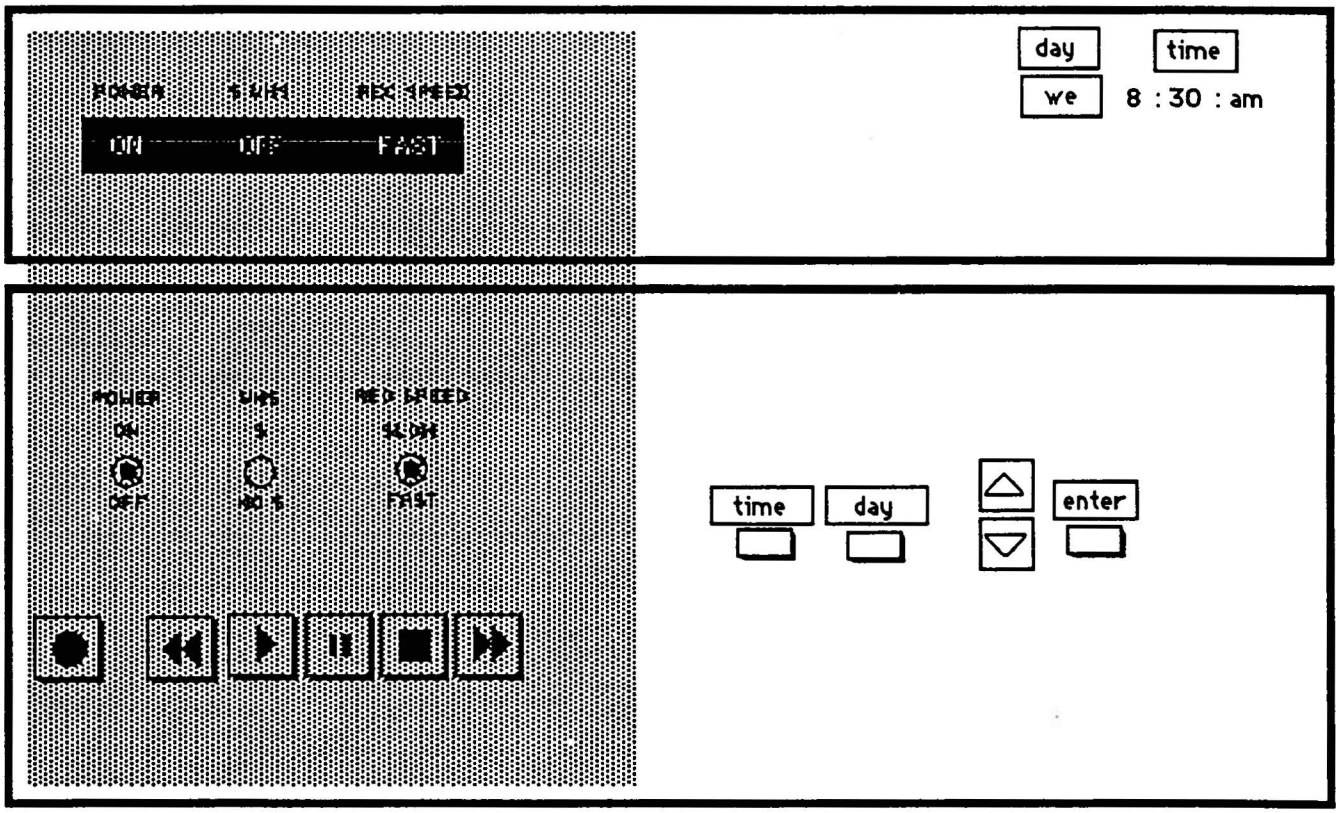

Figure 5. An Alternate Means Of Representing System States.

The Scandinavian Approach

In the Scandinavian countries, a long standing tradition of focusing on end-user's needs and situation requirements emphasizes ideals such as "quality at work" and "workplace democracy" [2]. Research results of the 1980's included such concepts as: "creating a design situation with similarity with the future use situation," "taking practice seriously," and "from human factors to human actors," (see [10] and [11]). Inspired by research such as the UTOPIA project [4] and the COOP project [5], one of the promising paths currently being investigated has end-users actively taking part in design by applying a cooperative prototyping approach. According to Bødker and Grønbæk, a major obstacle is the developer's limited ability to respond smoothly to user ideas or requests for prototype changes during design sessions [6]. Joint, cooperative development of the prototype permits users, as domain experts/lay developers, and professional developers to each contribute their knowledge to prototype development tasks. Building on cooperative development aspects of the Scandinavian approach, CISP seeks to further explore the potential resulting from combined user/developer abilities to manipulate computerized domain specific building blocks during prototyping activities.

Pictive

Also inspired by the Scandinavian approach, PICTIVE--Plastic Interface for Collaborative Technology Initiatives through Video Exploration--is a participatory design technique combining the use of 'low tech' objects with video recording technology to increase end-user participation in system design; improve the designers' ability to collect information about the use of the proposed system; and improve the sense of "design ownership" of the final product [14]. Major design components are literally made out of plastic and are thus seen by the users as malleable and adaptable. End-users prepare and participate in job and task scenarios where they use the plastic components to evaluate aspects of the system development. CISP expands two aspects of PICTIVE: first, offering *computerized building blocks* to increase active user participation, and second, offering the possibility for users to evaluate *use* in context of the system being built.

THE CISP TECHNIQUE

Besides facilitated iteration, the CISP approach to interface development involves five specific techniques that function to complement existing prototyping strategies. Each is examined below.

1. Realistic Prototype "Look And Feel"

As stated above, all of the user actions can be recorded, stored, and played back for later discussion between users and developers. This created a more realistic evaluation situation without the need for "think out loud" techniques. Instead, more realistic evaluation sessions permit more accurate assessments to be made of user-prototype interaction. Figure 6 shows the storyboard prototype for the NEC PC-VCR created using CISP. The upper half of the storyboard represents the display panel and the lower half represents the two panels with the various buttons, switches, etc. Not shown in the illustration is a third section of the storyboard listing the target of each button selection and providing additional space for user comments as described in the previous section. (For more detail see our chapter in [13].) Since we chose not to simulate the TV monitor, we asked users to perform tasks where it was not essential. User interaction with the storyboard was evaluated as users attempted specific tasks. Replaying each user's actions to them during subsequent analysis permitted the developer to ask questions such as: "I notice that when you were trying to set the clock you clicked twice on the timer button - was there some confusion there?" (An almost immediate CISP enhancement would involve user ability to add voice annotations to the storyboard.) The results of this analysis helped us to better understand the actual effectiveness of the proposed interface design solutions.

2. Interactive Modification

Because the composite domain specific building block can easily be copied and moved around as a single object, CISP enables users and developers to interactively and collaboratively modify the storyboard in real time. This is accomplished by dragging controls and displays to different locations (responding to comments such as "How about if we move that switch over there?"), duplicating existing screen objects ("Can you make this function react like that one?"), and creating new objects ("This function really shouldn't be lumped together with those controls!").

3. Generating and Comparing Design Alternatives

An essential part of CISP is to generate and compare several completed, or partially completed, potential development solutions, analyzing the trade-offs made in each potential solution. Consider an information display-oriented interface such as a monitoring/reporting system for a vehicle. Various types of user evaluation can be conducted to determine the effectiveness of alternative display modes during user evaluation sessions. A single user can interact with a whole series of development alternatives and then 'pick and choose' the screen display elements producing the "best" information display. Another use of this facility could be the development of a "creativity audit trail." An automated means of tracking the way in which new solutions are created without having to interrupt the creative process.

4. Creation of Families of Systems

CISP facilitates the creation of families of systems where all systems share several core components but differ slightly from each other in other respects (consider real-time command and control systems such as those being created for the NASA Space Station Freedom Project). This is accomplished by constructing the storyboard of common composite objects and introducing slight variations. Another example could be a real estate multiple listing system which needs to be customized for each individual county because of particular local reporting requirements. Systems for counties needing specific groups of information could also be prototyped using this 'theme and variations' approach with each prototype starting with the core screen elements and then adding group 'X' objects in response to type X requirements, etc.

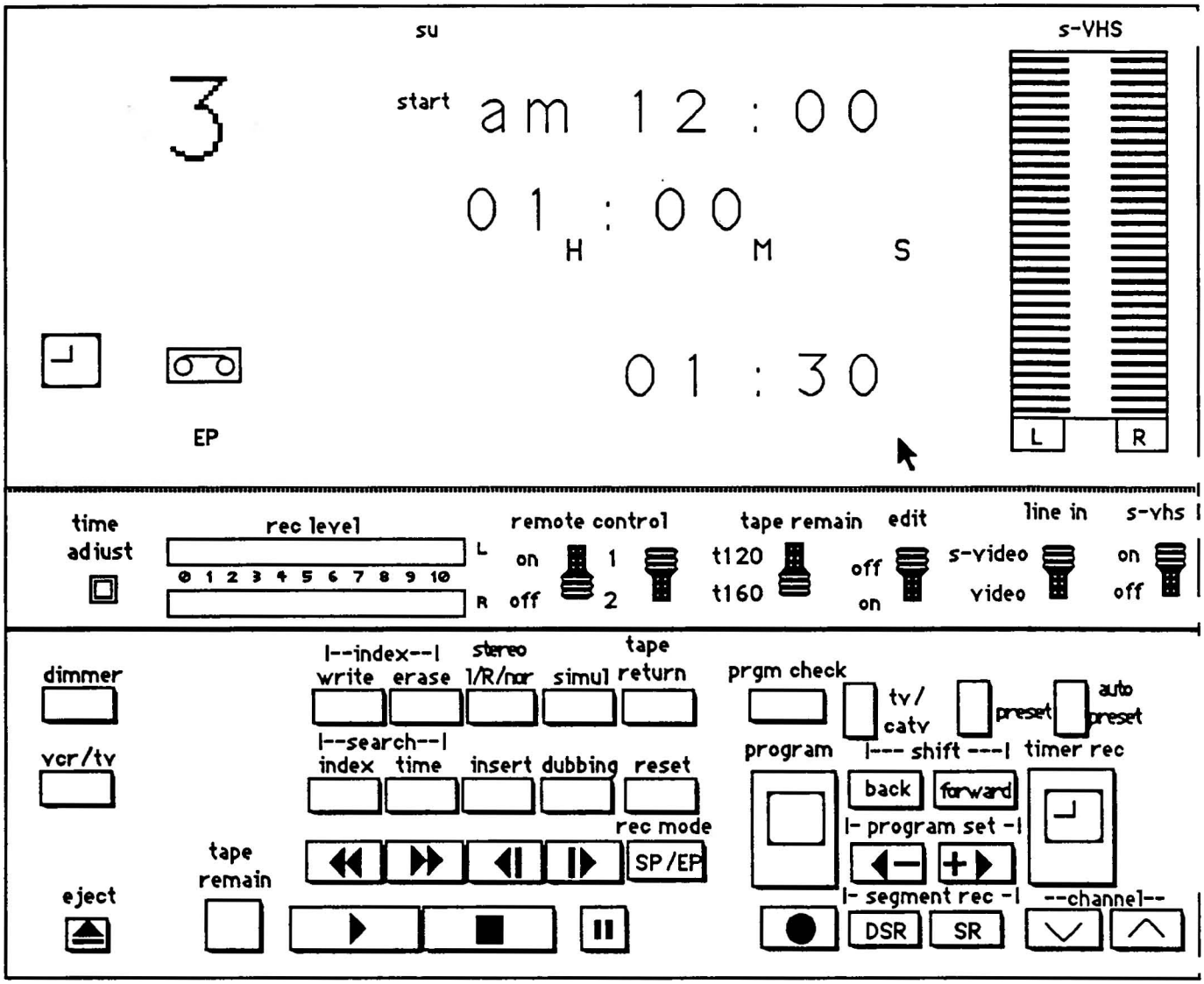

Figure 6. NEC PC-VCR Interface Layout.

5. Realistic Interface/Prototypes

By building on the existing HyperCard/SuperCard strengths CISP users are able to interact with realistic prototypes and get a feel for the system being developed. This includes the ability to use interface components such as sliders, switches, and other types of specialized controls. CISP-Tool can be used to create and evaluate specific user interfaces responding to conditions such as low light or presentation of highly complex information. This approach to prototyping is not unique - Figure 7 illustrates a realistic prototype interface for a Hewlett-Packard 12-C calculator created using a 'calculator construction set' and placed on a freeware disc.

INITIAL RESULTS/EXPERIENCES

In order to evaluate the effectiveness of CISP, we repeated the series of exercises we had conducted using the 'talking out loud protocol supplemented with video images' approach referenced above. After evaluating tasks including several variations of taping programs and controlling the tape, we settled on two tasks. Volunteers were asked: 1) to set the VCR clock and 2) to set the VCR to record a segment from Headline News. To each individual, we explained the two tasks and showed them where the operating manual was located. Time permitted us to evaluate about 10 individuals after we completed the debugging process. We left them alone while they attempted to complete the tasks. Most completed the two tasks and about half used the manual supplied with the PC-VCR to aid in task completion.

As the user clicked on buttons attempting to get the VCR to perform as desired, CISP recorded the name of each button selected by the user and upon command, could produce a HyperCard stack capable of physically reproducing each mouse click. An additional option could be set to reproduce the results of each individual click.

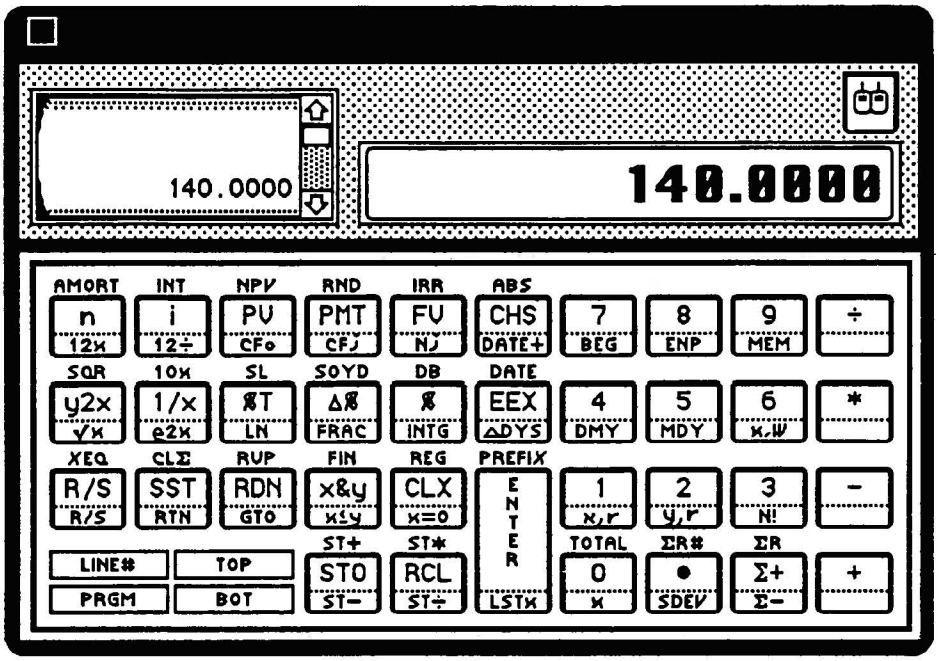

Figure 7. Another Realistic Interface Prototype.

To analyze user interaction, we used CISP capabilities to 'play back' the solution executed by the user. We asked the users to 'walk us through' the various keystrokes representing their path through the interface. The combination of the button names presented in context with the appropriate interface response seemed to prompt the memory of many users who were able to describe what they were considering as they tried to complete the tasks. Their comments were recorded on a comment field provided below the interface shown in Figure 6 (the comment field is not shown in the exhibit).

For the final third of each protocol, the users were shown how to use the CISP-Tool features permitting them to modify the interface. Then they were invited to make any changes they felt like making to the interface. With assistance from the authors, users rearranged button clusters, added graphics, eliminated comment fields, and made a number of other modifications to the original NEC interface to the PC-VCR.

Analysis of these experiences has permitted us to identify some promise and problems associated with CISP.

Promise

Though more effort is needed to make the domain specific building blocks, nonetheless a significant work-load savings was achieved by using CISP to create prototype interfaces of how an improved system might look. The effort reduction seems to be particularly applicable to the development of families of systems. For instance, building the switch shown in Figure 3 from scratch requires numerous switches between the HyperCard tools and a number of other actions to make each element separately. Our environment has reduced this task to a single copy/paste operation. Similarly, the ability to move composite objects without the HyperCard grouping tool is a noticeable improvement. Features like these make it significantly easier to build and modify multiple development alternatives. Users and developers particularly liked the ability to replay the session for subsequent evaluation. A typical reaction from a user was "(without the replay facility) I wouldn't have been able to give as thorough comments."

Problems

Building our tool in HyperCard has provided us with a useful collection of interface components. But it was difficult to implement some data structures and missing object oriented features made it harder to create and modify domain specific building blocks. Aggregation of primitive objects like buttons and field into composite objects like switches has been handled by either naming conventions or use of the ID numbers automatically assigned to objects by the HyperCard environment. Though easy to handle in small scale, the approach could become unwieldy if care is not taken during the scaling up process. Another problem we have encountered is HyperCard performance. In a complex prototype such as Figure 6, performance was less than desirable on the top of the line '030-based CPUs. We believe some re-coding could speed things up but there is no substitute for faster hardware.

CONCLUSIONS

We have two primary motivations for increasing the effectiveness of prototyping efforts: 1) to obtain more active/interactive user participation in the use, evaluation and redevelopment of storyboards, and 2) technological tool development to help shorten the use-redevelop loop used in many storyboarding efforts. In the ideal scenario, users and developers will be able to use storyboarding tools such as

we have presented in this paper to interactively change storyboard elements and immediately try out their changes. We are continuing our efforts to enable users and developers to be able to modify the storyboard while minimizing the amount of programming required.

Acknowledgments

We would like to thank Jonathan Grudin, Kaj Grønbæk, Jennifer Papp, Philip Sage, Kristine Thomsen, and Randy Trigg for many useful comments on earlier drafts. We would like to thank the participants from the SWSE 632 (User Interface Development) class who helped us to debug, participated and gave us feedback on this project. The paper was written during Kim Halskov Madsen's stay as a visiting professor at the Department of Information Systems and Software Engineering, George Mason University, during the academic year 1990-1991. The stay was financially supported by "Knud Højgaards Fond," "Christian og Ottilia Brorsons Rejselegat," "Aarhus Universitets Forskningsfond", The Danish Research Council (grant number 5.26.18.29-1) all Denmark and by George Mason University. Kim Halskov Madsen is currently at The Department of Information and Media Science, Aarhus University, Niels Juelsgade 84, DK 8200 Århus N, Denmark. Email: halskov@imv.aau.dk. Peter Aiken's research was supported in part by the Virginia Center for Innovative Technology.

References

1. Andriole, S. J. *Storyboard Prototyping: A New Approach to User Requirements* Analysis, (First edition) QED Information Sciences, Wellesley, MA, 1989.
2. Bjerknes, G., Ehn, P. and Kyng, M. *Computers and Democracy - a Scandinavian challenge.* Aldershot, UK: Avebury, 1987.
3. Brooks, F. "Grasping Reality Through Illusion - Interactive Graphics Serving Science." In *Proceedings: ACM/SIGCHI Conference on Human Factors in Computing Systems* May 1988, pp. 1-10.
4. Bødker, S., Ehn, P., Kammersgaard, J., Kyng, M. and Sundblad, Y. "A Utopian experience: On Design of Powerful Computer-Based Tolls for graphical workers" in [2].
5. Bødker, S., Knudsen, J., Kyng, M. and Madsen, K. "Computer Support for Cooperative Design." In *Proceedings for the Conference on Computer Supported Cooperative Work* (Portland Oregon, September 26-29, 1988). ACM New York 1988, 1981, pp. 377-394.
6. Bødker, S., and Grønbæk, K. Cooperative Prototyping: Users and Designers in Mutual Activity. *International Journal of Man-Machine Studies*, 34, 1991.
7. Connor J., and Shafer, L. *Structured Rapid Prototyping*, Prentice Hall, 1989
8. Curtis, G. and Vertelney, L. *Storyboards and Sketch Prototypes for Rapid Interface Visualization* CHI-90 Tutorial 1990.
9. Davis, A. *Software Requirements: Analysis and Specification*, Prentice-Hall, 1990.
10. Ehn, P. *Work-oriented design of computer artifacts.* Hillsdale NJ: Lawrence Earlbaum, 1989.
11. Greenbaum, J. and Kyng, M. (editors) *Design at Work: Cooperative Design of Computer Systems.* Hillsdale NJ: Lawrence Earlbaum, 1991.
12. Henderson, A. "The Trillium User Interface Design Environment." In M. Mantei and P. Orbeton, (eds) *Human Factors in Computing Systems*, SIGCHI '86 Proceedings, April 13-17, pp. 221-227.
13. Madsen, K. and Aiken, P. "Cooperative Interactive Storyboard Prototyping." In S. J. Andriole (editor) *Storyboard Prototyping: A New Approach to User Requirements Analysis*, (Second edition) 1991.
14. Muller, M. PICTIVE - "Exploration in Participatory Design." In [15] pp. 225-231, 1991.
15. Robertson, S. Olson, S., and Olson, J. (Eds.). *Reaching Through Technology. Proceedings of CHI '91.* New Orleans: Addison-Wesley Publishing Company, 1991.