INTERACTING with Sketched Interface Designs: An Evaluation Study

Beryl Plimmer
Department of Computer Science
University of Auckland
b.plimmer@auckland.ac.nz

Mark Apperley
Department of Computer Science
University of Waikato
m.apperley@cs.waikato.ac.nz

ABSTRACT
Digital hand-drawn sketches provide a new and unique way of interacting with a prototype user interface design while it is still rendered as a sketch. The successful use of prototypes and scenarios for exploring design ideas is well documented. Hand-sketched designs have also been found preferable to formal diagrams during early design. The study reported here shows that interacting with digital sketches adds an exciting new dimension to the interface design process, we found that people do more revisions and more accurate revisions with digital sketches.

Author Keywords
sketch tools, prototyping, scenario-based design.

ACM Classification Keywords
D2.2, H5.2, H5.3, I3.6

INTRODUCTION
Traditionally designers hand-sketch their early design ideas because this is a fast and unconstrained technique. Computer design environments are now available for most design disciplines; for example, CAD for engineers and architects, form builders for user interface designers, and publishing tools for graphic designers. Most of these tools adopt an element-based approach to design, where the user selects a widget or tool from a toolkit and place predefined elements on the drawing surface. Although these computer tools offer superior support for tasks such as editing, designers continue to hand-draw their early design, and must make a conscious decision and effort to commit a design to the computer-based system. A number of computer-based sketch tools have been developed to provide computer support for informal sketches.

We have developed such a sketch tool, Freeform, for designing user interface forms, and have evaluated it against a traditional environment and a widget-based form designer. This evaluation suggests that computer-sketch tools are a viable alterative to traditional tools and also offer new ways to explore the design-space during early design.

CHI 2004, April 24–29, 2004, Vienna, Austria.
ACM 1-58113-703-6/04/0004.

BACKGROUND
Many designers reject the use of current computer drawing tools for creating and exploring early design ideas. To render an idea onto the canvas with the current generation of computer tools, typically the user must select a widget or tool from a toolbox and then place, size and position the widget on the canvas. This is in contrast with traditional environments, such as pen and paper or whiteboard, where the user selects a tool (for example, a pen or an eraser) and then renders their ideas freehand onto the canvas. Designers' rejection of widget-based computer design tools for the early creative phase of design is supported by studies by Goel [5] and Black [2], which show that widget based design tools interfere with the creative process.

The disadvantage of widget selection and placement is two-fold. First, the designer is required to commit to a specific widget type, when they may at this stage prefer to leave the widget as an ambiguous space-holder. Designers find themselves being distracted by self-talk such as 'should this be a radio button or a checkbox, why do I need to decide now?' Secondly, the formal diagrams that are the product of computer design tools presuppose tidy, aligned, regularly sized elements. To create a tidy design takes time and cognitive effort, which distracts the designer from the creative process, and a tidy design implies, perhaps incorrectly, that the design is polished, committed and complete. Wong[13] and Wagner[12] note that sharing informal, hand-drawn designs with associates elicits more appropriate comment on the 'big picture' than formal designs, when comment tends to focus detail. Goldschmidt [6] similarly claims that the ability to interact with hand-sketches is valuable to the designer.

Yet, despite the rejection of computer-design tools during early design, they are universally used later in the design process, when well-defined diagrams are required and the computer support for editing, storage and sharing of the documents is invaluable.

Many software user interfaces are designed by software engineers and programmers, skilled in the construction and use of computer systems but less experienced in design. They see the benefits of computer design tools without recognizing these tools' disadvantages [2]. Sketch-based computer design tools can bridge this gap by providing a design-friendly computer-supported environment and add a useful new dimension to the design process.

A number of experimental sketch tools have been developed for a range of disciplines. For example, architecture [11], engineering [10], software modeling [3, 4] and user interface design [1, 7, 8]. Most sketch tools comprise a main drawing space where the user can hand-draw ideas, a storyboard or overview where separate smaller designs are shown in miniature and the user can establish relationships between parts, and, in the case of user interface designs, a run mode where the designer can interact with the sketched design. Different sketch tools have provided different levels of behavior in run mode; navigation is common [1, 7, 8], while others have operational widgets, for example scroll bars [7] and media clip players [1].

Sketch-based design tools are still in their infancy and we are continuing to learn both how to construct better tools and how they can best support the design process.

We do know that hand-drawn designs are better for eliciting comment from others [13]. Rettig [9] suggests that interacting with paper prototypes is a powerful way to explore both interface and behavioral requirements. However interacting with a static sketch is more difficult than interacting with a computer-based prototype.

In a previous study, we compared Freeform as a design tool against a 'normal' whiteboard [8] to create a computer form interface. The designs created in Freeform were independently assessed to be slightly better than those created on the whiteboard, which we believe is a positive outcome as other studies that compared traditional tools to widget-based tools have found the traditional tools better [5], [2]. The participants in this study also: enjoyed Freeform more than the whiteboard; where motivated to learn more programming; where more prepared to complete the problem; and found checking the scenarios easier on Freeform than on the whiteboard.

We observed during the study that the subjects using Freeform made many more changes after interactively checking the design in the Freeform run mode, in comparison with the number of changes made by subjects using the static whiteboard. We hypothesize that the ability to check interactively prompted the students to focus on the user interaction and to think specifically about the behavioral requirements of the problem. A typical difference we noticed between the static and interactive checking was the number of lines required for address data. Most groups started with a single line for the address. After checking, three of the four groups using Freeform had space for multiple address lines, while only one of four groups who designed on the whiteboard provided multiple address lines in their designs. One of these groups that failed to provide enough space for the address on the whiteboard actually wrote part on the address into the box, remarked that it did not fit, but then moved on without making a change!

We concluded that interactive checking of designs was more successful than static checking. To determine the role of the informal nature of the design artifact in this effect, we undertook another study that compared interact review of designs rendered as (i) sketches, and (ii) as formal diagrams.

STUDY

For this study we created interface designs for two applications, and then rendered each of these in two forms; a hand drawn implementation using Freeform, and a formal diagram created with the Visual Basic (VB) form designer (Figure 1). The first application was a form for a book catalogue, and the second was a credit card application form.

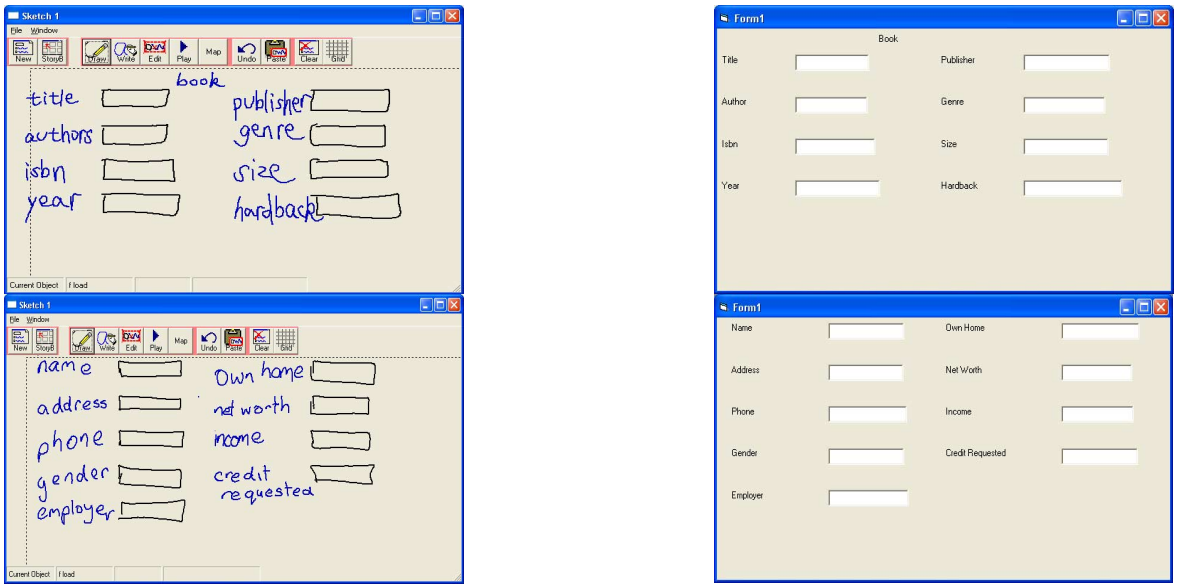

Figure 1. Application designs rendered as both sketch and formal diagram.

Small groups of subjects (two or three people) were asked to review one of the designs for each application; one of the designs they were given was a sketch, the other was a formal diagram. The study was balanced by alternating the problems, rendering and order, so that some groups first checked the sketch design while other groups checked the formal design first, and so that some checked the sketch design of the book application while other groups checked the book application's formal design. All the subjects were drawn from a second year university programming course. Six groups participated in the study.

For each application they were supplied with the design, running in either Freeform or the VB form design, together with a brief problem statement and two scenarios. All the subjects were familiar with VB but not with Freeform; they were shown how to use the Freeform sketch-space and run mode.

Subjects were asked to check the design against the application description and use the scenarios provided to fill-in the form. To check the sketch, they used the Freeform run mode and then modified the sketch in the Freeform sketch-space. To check the formal diagram, they used the VB IDE to run the program (interface only) and then modified the diagram using the VB form designer.

We specifically specified the applications so there would be no common elements (e.g. 'name' does not appear on both), however, there were comparable elements. For example, each application included a mutually exclusive option pair, which would usually be represented by radio buttons; the book catalog included the specification of the book binding as hardback or paperback, while the credit application included applicant gender. Each application also included a selection from a set that would usually be represented by a dropdown list; the book catalog suggested the genre classification of the book could be selected from a small set, while the credit application specified applicant income to be described in a set of specified ranges.

FINDINGS

There were two significant differences between the work carried out on designs in the two environments; the number of changes made to the designs and the time spent on different types of activity.

Of the six groups, five groups made more changes in Freeform, regardless of the application or the order of the exercise. The other group made the same number of changes on both designs; however, when this group did the VB exercise, they decided to hand-sketch the design before changing it on the computer. The mean number of changes made to the formal design was 6.5 and to the informal design 8.67. This gives a 95% confidence interval of mean difference 1.03-4.96 (p 0.01), and given the small number of elements on the designs (8 and 9), this is a large difference. Nearly all the changes were improvements, the few changes that were incorrect were to the formal diagram, for example the formal diagram in Figure 2 shows the book binding as check boxes where they should be radio buttons.

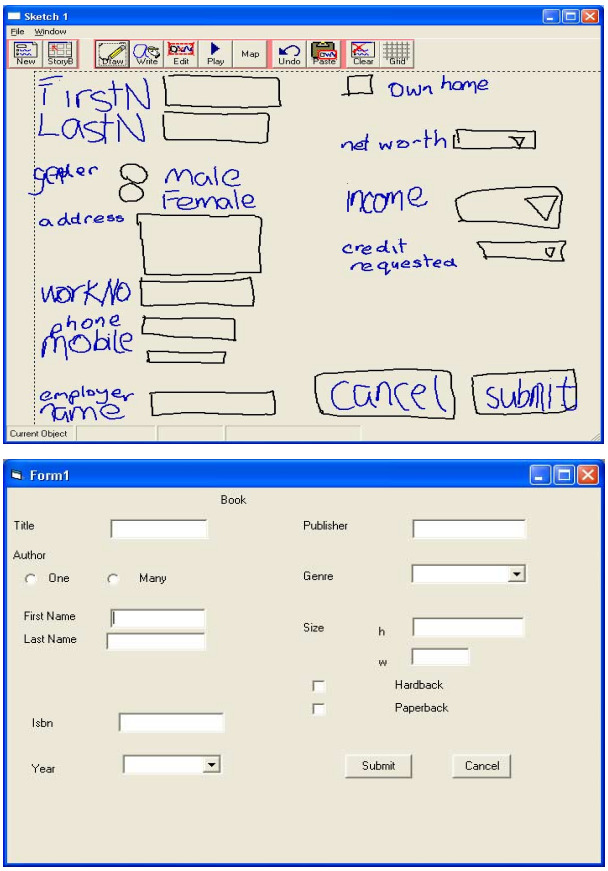

Figure 2. Designs reviewed by one group.

There were some clear differences between the changes that were made to the sketches and formal diagrams. We will illustrate this using the comparable elements described above. The book binding and gender would, in most cases, best be expressed as mutually exclusive radio buttons. On the informal sketch all six groups changed these elements to radio buttons; on the formal design four groups used radio buttons, one group used checkboxes, while the other group did not change the design. With the book genre and credit application income, which would usually both be dropdown lists, there was a similar pattern; with the informal designs all groups changed the elements to a dropdown list, while with the formal designs, four groups changed to dropdown lists, but the other two groups made no change. There was no relationship between the groups and the changes; each 'error' with the radio buttons and lists was made by a different group.

The other notable difference was the speed with which changes could be made. Although the subjects were already familiar with VB, and received only 5 minutes training on Freeform, it took them longer to complete the VB exercise (mean difference 4.4 minutes) with no group completing the VB exercise quicker than the Freeform exercise. We noticed that when using VB they spent a disproportionate effort on keeping the design tidy – aligning and sizing the controls.

This was in contrast to Freeform, where the focus was on the application and the design task. After each group completed both exercises we informally discussed the experience. The subjects were surprised that the VB task had taken longer, and also that they had made more changes in Freeform. Figure 2 shows the completed designs by one group; they made eight changes to the sketch and six to formal diagram.

DISCUSSION

Computer tools, from word processors to CAD suites, are superior to traditional tools in regard to their support of editing, storage and sharing of documents. However, they do not provide the natural, unrestricted environment that is a necessity for creative design. Sketch-based design tools will continue to develop as better pen-input hardware becomes available, recognition engines are refined and we become more skilled at creating a natural user experience.

Computer-based sketches offer more compelling interaction opportunities than their tradition counterparts. The study reported here suggests that the ability to interactively exploring a sketched design is more valuable than interacting with either traditional prototypes or formal computer prototypes. Our earlier study [8] suggested that checking a design on a whiteboard was less successful than interacting with a digital sketch. This study shows that interactively checking a digital sketch is more successful than interactively checking a formal design. More exploration and more changes will lead to better designs.

Simple navigation behavior is essential to explore a multi-form/page interface design; more extensive behavior support requires accurate recognition. In Freeform, we deliberately delay recognition until the user wishes to convert a sketched design to a formal diagram on the assumption that over emphasizing recognition will interfere with the design process. While it is possible to add quite extensive behavior to a recognized sketch, for example sets of radio buttons can interact so that when one is selected the others are unselected, what affect behavioral functionality will have on the design process is unknown.

CONCLUSIONS & FUTURE WORK

Computer-based sketch tools are a design-friendly alternative to widget-based tools. In the near future we are likely to have computer-supported design tools that cover the continuum from first renderings of ambiguous design ideas to completed detailed designs, and which provide seamless transition during the design process. Exploring designs while they are still rendered as a sketch will add a new dimension to the design process by allowing the designer to explore the behavior of the design while the design ideas are still fluid.

More studies are required to ascertain the usefulness of behavioral support during interactive checking. It may be found that different levels of behavior are required at different stages of design. It is not possible to predict whether adding intelligence to the sketch will enhance the design process by making the interaction more realistic, or stifle it by imposing artificial restrictions.

REFERENCES

1. Bailey, B.P. and J.A. Konstan. Are Informal Tools Better? Comparing DEMAIS, Pencil and Paper, and Authorware for Early Multimedia Design. *Proc CHI 2003*. p. 313-320.
2. Black, A., Visible planning on paper and on screen: The impact of working medium on decision-making by novice graphic designers. *Behaviour and information technology*, 1990. 9(4): p. 283-296.
3. Chen, Q., J. Grundy, and J. Hosking. An E-whiteboard application to support early design-stage sketching of UML diagrams. *Proc HCC*. 2003. IEEE: p. 219-226.
4. Damm, C.H., K.M. Hansen, and M. Thomsen. Tool support for cooperative object-oriented design: Gesture based modelling on and electronic whiteboard. *ProcChi 2000*. p. 518-525.
5. Goel, V., *Sketches of thought*. 1995, The MIT Press.
6. Goldschmidt, G., Serial sketching: Visual problem solving in design, The backtalk of self-generated sketches. *Cybernetics and Systems*, 1992. 23: p. 191-219.
7. Landay, J. and B. Myers, Sketching Interfaces: Toward more human interface design. *Computer*, 2001. 34(3): p. 56-64.
8. Plimmer, B.E. and M. Apperley. Software for Students to Sketch Interface Designs. *Proc Interact*. 2003. p. 73-80.
9. Rettig, M., Prototyping for tiny fingers. *Communications of the ACM*, 1994. 37(4): p. 21-27.
10. Stahovich, T.F., The engineering sketch. *IEEE Intelligent Systems,* 1998 13 (3): p. 17-19.
11. Trinder, M. The computer's role in sketch design: A transparent sketching medium, *Proc Computers and Building, CAAD futures 99*. p. 227-244.
12. Wagner, A., Prototyping: A day in the life of an interface designer, *in The art of human-computer interface design*, B. Laurel, Editor. 1990, Addison-Wesley: Reading MA. p. 79-84.
13. Wong, Y.Y. Rough and ready prototypes: Lessons from graphic design. *Proc Human Factors in Computing Systems CHI '92*. 1992. Monterey: p. 83-84.