FreeForm: An Informal Environment for Interface Prototyping

Beryl Plimmer
Manukau Institute of Technology
Private Bag 94006
Manukau City, New Zealand
+64 9 968 8000
Beryl.Plimmer@manukau.ac.nz

Mark Apperley & Masood Masoodian
Department of Computer Science
University of Waikato
Private Bag 3105, Hamilton
+64 7 838 4528
{M.Apperley, M.Masoodian}@cs.waikato.ac.nz

ABSTRACT
Best practice in interface design suggests that hand-drawn sketches are preferable at the early stages of the design process. This paper describes the FreeForm software which supports informal sketched interface design by acting as a Visual Basic Add-In. The software utilises a digital whiteboard and pen input to support sketching and "running" of an informal prototype.

Keywords
Informal interfaces, prototyping, digital whiteboards

INTRODUCTION
Current programming IDEs do not support the informal higher-level design which is typically done using pen and paper or whiteboards[7]. The aim of this research is to create an informal environment for designing forms, using low-cost large interactive whiteboards. The software described in this paper utilises a whiteboard digitiser and a back projected large screen to provide a pen-based computerised interaction environment (see Figure 1).

RELATED WORK
Sketching is the preferred preliminary capture process for designers because it provides a quick and easy way to externalise design ideas. While sketching a designer can leave portions of the sketch vague or partially defined. In contrast computer design tools typically require the user to select from a predefined set of widgets which are placed and sized in the design space. Goel [3] demonstrated that current computer drawing tools impeded design. Many others have reported that designers, regardless of their particular field, prefer to hand-draw early designs [2, 4, 5].

This work draws on the Xerox Parc research on shared whiteboards [6] for meetings and sketch design tools. Most notable of the sketch design tools are: Landay's Silk and Denim [5], the CASE Tool Knight developed by Damm *et al.* [2] and a tool for multimedia applications' design by Bailey et al. [1].

FREEFORM
This software has been developed as a Visual Basic (VB) Add-In and is intended primarily for the design of VB forms. There are five major parts to the software; the sketch space, storyboard, run mode, recognition engine and conversion of the sketch to a VB form. The following sections will briefly describe each of these.

Figure 1: A low-cost pen-based large interactive display

Sketch Space
In the sketch space the user can draw a form design. The sketch space emulates a whiteboard and as such allows users to draw and write on the surface. Normal whiteboard functionality is supplemented with conventional software functionality such as resize, move, delete, copy, paste and undo. The users can create multiple sketches each depicting a different form.

Storyboard
The storyboard shows miniature views of all the form sketches. Links can be drawn between the forms that will show in run mode (see below) to allow navigation between the forms.

Run Mode
Run mode allows the users to work through use-case scenarios. As with the sketch space the user may write and draw on the screen. In run mode the sketch is shown but can not be altered. Hotspots created by the storyboard links are also displayed. Touching a hotspot will take the user to the linked form.

Recognition Engine
Recognition is required to convert the sketches to VB forms. The software recognises both shapes and characters.

Shape Recognition

Shapes are recognised in two stages. The basic pen strokes are recognised using Rubine's [8] algorithm and then the relationships between shapes are analysed using rule based techniques. The shape library and rules for combining shapes are exposed to the user. Figure 2 shows typical mapping, the first rectangle contains a triangle so is recognised as a dropdown list, the second contains circles and squiggles so the rectangle is recognised at a frame and the circles and squiggles interpreted as radio buttons. The last rectangle is empty so it becomes an edit box and the word becomes a label.

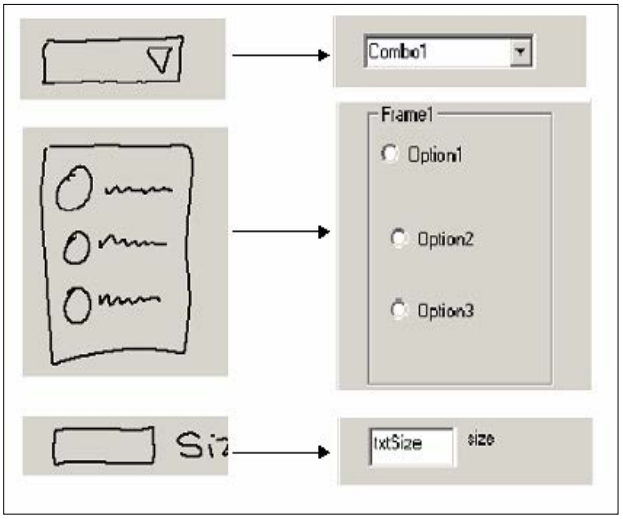

Figure 2: Typical recognition of shapes to controls

Character Recognition

Word recognition is a three stage process. The pen strokes are recognised using a modified Rubine's algorithm (extra features were added). Most letters are expected to be formed by a single pen stroke, however for the letters 'i' and 'j' if a dot is present it will be combined with the stoke beneath it and 'f', 't' and 'x' are recognised by combining the two intersecting stokes. After this stage each letter is represented as a list of probable letters in descending order of likelihood. The resulting collection of letters is then matched against a vocabulary list. If there is no near-match then the "word" is displayed as the most likely letter in each place.

After recognition, the system allows the user to alter the decisions that the software has made, or add words to the vocabulary.

Conversion to formal environment

The recognised sketches may be converted to VB forms. VB controls are created by the software by mapping the attributes of the recognised sketch to the VB control attributes. How these attributes are mapped is exposed to the user and can be altered. The software also beautifies the form by aligning controls and standardising sizes. Users can also control much of this process by setting the grid size and standard units of height and/or width of controls.

EVALUATION

A usability study has been conducted on a single-form version of the FreeForm software. The users made positive comments about the software. They also requested that various functionality such as character recognition, form beautifying and a run mode to be added to the software. Their "wish list" has been used as a guide to the development of the current version. Usability tests have also been planed to evaluate the existing version.

CONCLUSIONS

This paper has described the FreeForm software which provides an informal environment for form design. FreeForm combines the informality of low-fidelity tools such as whiteboards with the functional support of computer software. After the next usability tests we plan to convert the software to the .Net™ framework which will provide a platform to dynamically create both windows and web forms.

REFERENCES

1. Bailey, B.P., J.A. Konstan, and J.V. Carlis. Demais: Designing Multimedia Applications with Interactive Storyboards. in *ACM Multimedia*. 2001, p. 241-250.
2. Damm, C.H., K.M. Hansen, and M. Thomsen. Tools Support for Cooperative Object-Oriented Design: Gesture Based Modelling on and Electronic Whiteboard. in *Chi 2000*. 2000: ACM, p. 518-525.
3. Goel, V., Sketches of Thought. 1995, Cambridge, Massachusetts: The MIT Press.
4. Gross, M. and E.Y.-L. Do. Ambiguous Intentions: A Paper-Like Interface for Creative Design. in *UIST '96*. 1996. Seattle Washington: ACM, p. 183-192.
5. Landay, J.: Informal User Interfaces for Natural Human-Computer Interaction, in *IEEE Intelligent Systems* (1998) p. 14-16.
6. Moran, T.P., P. Chiu, and W. van Melle. Pen-Based Interaction Techniques for Organizing Material on an Electronic Whiteboard. in *10th Annual Symposium on User Interface Software and Technology*. 1997. Banff, Canada: ACM SIGSOFT, p. 45-54.
7. Plimmer, B.E. and M. Apperley. Computer-Aided Sketching to Capture Preliminary Design. in *Australian User Interface Conference*. 2002. Melbourne, p 9-12.
8. Rubine, D. Specifying Gestures by Example. in *Proceedings of Siggraph '91*. 1991: ACM, p 329-337.