

Sandbox: An Encapsulated Exploratory Environment

Bo Lu and Michael B. Twidale

Graduate School of Library and Information Science
University of Illinois at Urbana-Champaign
Champaign, IL 61820
{bolu, twidale}@uiuc.edu

ABSTRACT

In this paper, we describe the conceptual idea of the Document Sandbox: an environment where users can make exploratory changes to a document safely and without concern of irreversibly altering the original, and which facilitates a user's search for the right feature for a task. The aim of this tool is to allow the user freedom and framework to systematically find the right feature for a situation by trial-and-error in a safe, easily and infinitely recoverable environment. In addition, the Document Sandbox can take on a Computer-Supported Collaborative Work role in Over the Shoulder Learning situations, where a well-presented record of what has already been tried will save time and facilitate communication with the colleague who is called in to help. The Document Sandbox is currently in the conceptual stages, and not yet implemented.

KEYWORDS: Search, manual search, pessimistic design, user help, computer-supported collaborative work.

MOTIVATION: what about when the GUI isn't 'intuitive'?

We have undertaken various workplace studies of collaborative help giving [7,8]. In many cases we have seen users take advantage of the visibility of a graphical user interface to look for inspiration for steps that might fulfil or at least get them closer to their goal. As well as just looking at the interface, they often 'trawl' through drop-down and side menus looking for inspiration, as well as rapidly opening and closing dialogue boxes that look plausible for the same reason. Although users do sometimes guess the action or actions that will do what they want (a goal of good design), this is not always the case, and there is little support for this situation. We realized that manually searching an application's UI is done fairly often, and that there should be a way for the UI to assist that search.

Today's GUIs aspire to be intuitive, or "guessable." Complex interfaces with many options will inevitably be confusing at times. While this may work much of the time, when GUIs fail, users must resort to trying to find the right feature on their own, but are left with a UI that does not

facilitate manual search and problem-solving. Current systems like automated help and wizards attempt to sidestep the problem. But they, just like GUIs themselves, do not always work, and more importantly do not provide a backup solution for when they don't work. This paper explores some ideas for what we might term 'pessimistic design': designing on the assumption that despite our best efforts, users will sometimes struggle with the interface, and so need a safe, supportive environment for their own explorations.

With the knowledge that often users will resort to the "let me find it on my own" mentality, we put forth the Document Sandbox with the idea that this is the least we can do is help. The Document Sandbox is currently in the preliminary design conceptualization stage, and has yet to be implemented.

User interaction as Search

The Document Sandbox operates on the same conceptual idea as search, but instead of searching for a particular web page on the Internet or a file on the network, the user is searching for a feature in an application. Like any other search, the ideal is that the search returns a single result - the one the user wanted. However, more realistically the user gets either only a few results, which do not include what was wanted, or the user is inundated with far too many results, which then must be narrowed by searching again within the results or by manual browsing. Current implementations of user help systems already incorporate this "Search for a feature" idea. However, there can be problems when the user and the application designers use different words for the same task or feature. The Sandbox comes in when the user needs to browse through some possible options or narrow a set of search results down to the one(s) he or she wants. In its most basic incarnation, Document Sandbox is much like the search results page of any web-based search engine. There, results are presented in linear order, and the user then goes through a semi-systematic self-narrowing of the results to find the few he or she wants. Exploration in web-space is practically risk free compared to exploring a conventional application, since the user can experiment without fear of breaking something, with the ultimate undo (the browser back button) as a fallback.

Users “winging it”

Users’ searches of an application’s interface are often error-prone and frustrating. The usual search starts with the user conceptualizing what exactly he or she is trying to do, and then searching the pull-down menus for a feature that appears plausible, trying each reasonable-looking entry in turn. However, current GUIs do not have an external representation of the search process. Without this representation, many users try to keep track of what they’ve tried and what they haven’t mentally, which is error-prone. Moving back and forth between several levels of dialogue boxes, users tend to forget what they have and haven’t tried, and thus end up missing entire sections of the feature tree or wasting time re-trying options – only to realize as soon as the dialogue box comes up that it looks familiar and they’ve “been here before.” Users might consider keeping track using paper and pencil, but that requires a large initial outlay of effort and foresight, whereas people usually just try to wing it, hoping they’ll stumble across the right feature. (Indeed, although we have observed many cases of the mousing-search, we have not seen such record keeping.)

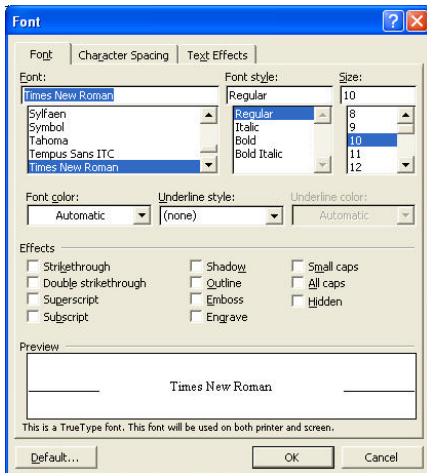


Figure 1: The Font dialogue box from Microsoft Word 2000. This particular dialogue box has 3 tabs, and the first tab itself contains three regions.

The second problem is that most applications organize similar features in a tree-like pattern, which although extremely useful for certain kinds of information discovery, can hinder manual searching. Drop down menu choices launch dialogue boxes that can themselves launch several more. Usually, features are grouped into frames or groups even within a single dialogue box, further increasing the cognitive depth of the tree and inhibiting the trial and error method of search (Figure 1). When one or more features on a dialogue box fail expectations, users are liable to mentally mark the entire dialogue box as *searched*, and may not explore the options available in dialogue boxes that launch off the current one (i.e. when “leaf” features fail to perform correctly, users oftentimes move up the tree without trying all the leaf features at that level, or going further down the tree).

The need for support may only be perceived post-hoc

Users oftentimes are in the quick-fix mode of thought, and will attempt to “wing it,” selecting a few likely candidates

hoping one of their first tries works. Thus, any software implementation of Document Sandbox should have a constantly-on background cache of the user’s last several menu choices, so that when users give up on the “let’s wing it” method, what they’ve already tried as been logged by the computer and put into the Sandbox framework, facilitating the transition from informal search to more formalized search.

Sandbox: a place for playing around

Drawing on research in microworlds and scaffolding [4,5,6,9], we wish to create an environment that more explicitly supports exploration and experimentation. Skilled users of software can create their own microworlds by use of various techniques (such as making multiple backups, creating simpler cases of their document to experiment on, and carefully evaluating the effects of individual changes). These still can be rather memory intensive for the user, creating the possibility of slips and consequent confusion. Social microworlds can also be created, by calling over a colleague for help [7,8]. Even if the colleague does not know the solution, working together on a problem can be productive, with two pairs of eyes to watch out for potentially promising actions to pursue or potentially catastrophic irreversible steps to be wary about.

The overall design aim for the Sandbox is to design features to explicitly support search and experimentation. We presume the existence of a reasonably sophisticated undo feature [1,2], and develop an environment that can exploit such a feature and complement it with support for remembering what has been tried and considering what might be tried next.

An example task

There is a setting in Microsoft Word that allows users to, by default, view their documents in white text on a blue background. A user who may never have changed this setting might be inspired to try it after seeing a colleague’s distinctive white-on-blue computer screen. Figure 2 is our conceptual design of what Sandbox would look like while a user is using it to find this ‘white-on-blue’ feature.

Uses in Automated User Help Systems

We foresee that the Document Sandbox idea finding greatest use as part of an Automated Help System UI. Before a user seeks out the automated help system, he or she is likely to try to “wing it” first, mousing over buttons and menus in a quick manual search. When a user finally runs a query to help, the results would be displayed in the Document Sandbox environment, making it in many ways the results page of an online search engine. To help users manually narrow the results, Sandbox would bring together the fallback feature of undo along with what the user already tried in the “winging it” section of search, before he or she ran the automated help feature.

Document Sandbox: Design and Layout

We envision the Sandbox to be implemented via three distinct parts, each housed in its own delineated region in

the Sandbox UI (See Figure 2). The “Trial List” region will cover most of the left side of the Sandbox window, with the “Feature” frame and the “Document Preview and Navigation” frame on the right.

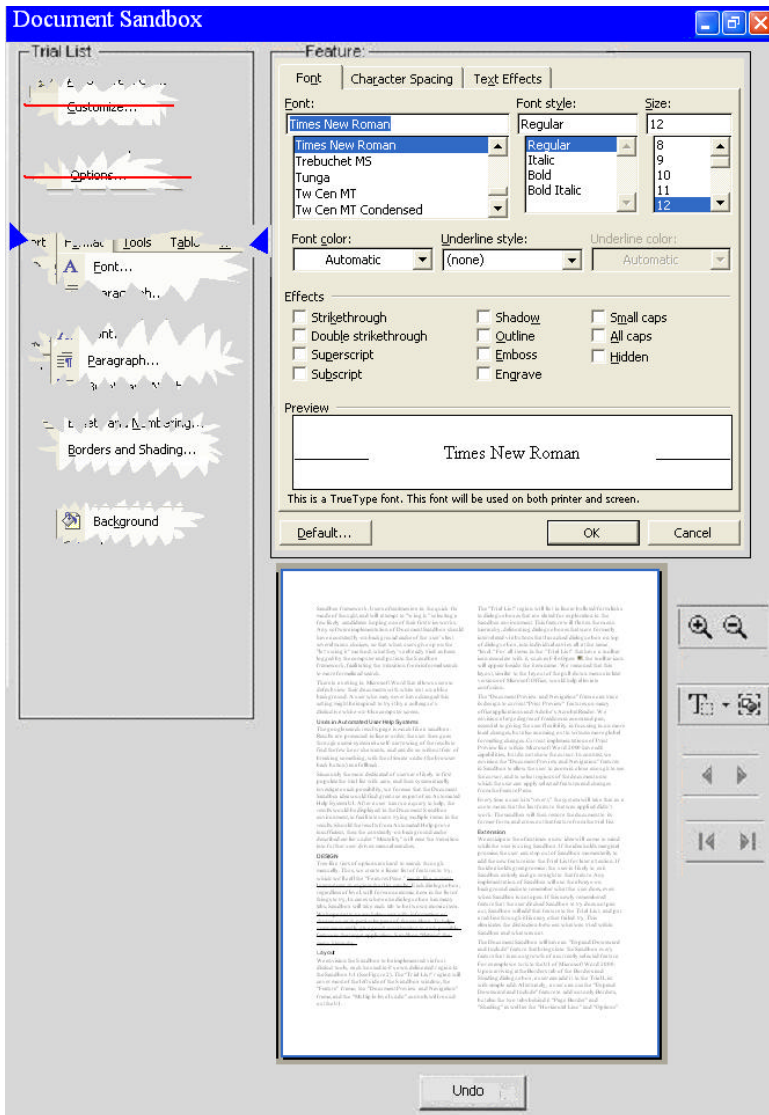


Figure 2: The Document Sandbox UI

The “Trial List” region will list, in linear bulleted form, links to parts of the UI that are slated for exploration. The List itself will be populated most often by the search results of automated help systems, but can also be populated manually by the user using drag-and-drop. The Trial List feature will flatten the menu hierarchy, changing the tree-design of dialogue boxes that were formerly interrelated via buttons that launched dialogue box on top of dialogue box, into individual entries all at the same “level.” Graphically, items to be tried will be represented in the Trial List by ragged-edge (see later for a discussion of this) focused screen grabs of the control that launches the item, be it a button with text label, toolbar button, or icon. Items in the Trial List will have one of three possible states. The top

section of the list will house items that have been tried and marked “failed” with a red line through them (Figure 2). Interspersed among the items that have already been tried may be a few items marked “alternate,” which the user feels aren’t ideal, but might be a substitute should a better feature not be found. Finally, as the blue “cursors” move down the list, the items below the cursor are in the “waiting to be tried” state.

The “Feature Frame” will simply be a place to display the dialogue box from the native UI of the target application. Here, users can select features or options to apply and be able to see their global and local effects on the document through the Document Preview and Navigation frame described below.

Current “Print Preview” features inspired the “Document Preview and Navigation” frame. We envision a large degree of freedom in zoom and pan, essential to giving the user flexibility in focusing in on more local changes, but also zooming out to witness more global formatting changes. Current implementations of Print Preview like Microsoft Word 2000 have edit capabilities, but do not show the cursor. In contrast, we envision the “Document Preview and Navigation” features in Sandbox to allow the user to zoom in close enough to see the cursor. This enables users to zoom in and apply a feature to a specific section of the document.

Since much research has been done in the area of well-designed undo systems [1,2], for the purposes of the Sandbox design, we will assume that there exists a well-implemented global undo. Every time a user hits “undo,” the system will take that as a cue to mean that the last feature that was applied didn’t work. The sandbox will then restore the document to its former form, and cross out that feature from the trial list.

Pausing the Document Sandbox

We anticipate the oftentimes a new idea will come to mind while the user is using Sandbox. If the idea holds some promise, the user can step out of Sandbox momentarily to add it into the Trial List for later attention. If the idea holds great promise, the user is likely to exit Sandbox entirely and go straight to that feature. Any implementation of Sandbox will use the always-on background cache to remember what the user does, even when Sandbox is not open. If the new idea does not work out (as evidenced by the user quickly hitting Undo after applying that feature), Sandbox will add that feature to the Trial List, and put a red line through it like any other failed try. This eliminates the distinction between what was tried within Sandbox and what was not.

Screen size and resolution concerns: The Sandbox UI, which we envision to be a dialogue box itself, will need a relatively high resolution display to function. For example,

on a personal computer running Microsoft Word 2000 on 800x600 resolution, the options dialogue box covers roughly 80% of the screen vertically a slightly more than 50% of the screen horizontally. It is clear from Figure 2 that Sandbox needs a screen large enough so that any dialogue box within the target application covers at most 40% of the vertical space and 50% of the horizontal space of the screen, to allow room for the rest of the Sandbox UI components.

Interfaces to Interfaces

The Sandbox idea is about creating a safe environment for exploring an application by interacting with it via its interface. It is based on a notion of 'pessimistic design' – that we cannot design an interface (or even a help system or a wizard) that will be guaranteed to enable a user to interact successfully with the application all the time. Therefore we need to facilitate learning by exploration, employing techniques from research on scaffolding and microworlds. One challenge is to provide an externalization of interaction with the system, effectively creating another interface to help the user think about the 'real' interface. This solution could easily add to rather than reduce the problem. One aspect of this is our exploration of a representation of 'ripped out' interface elements in the Trial List region as a way of distinguishing between thinking about trials with elements of the interface and the real buttons and menu items. This is in part inspired by another metaphor; that of the lab book, where a scientist makes notes on his or her evolving thinking and actions and may include pasted-in results such as output, or graphs. We believe that this is an issue that deserves greater scrutiny in future development.

Extensibility and Future Work

The Sandbox idea is extendible into situations beyond merely handling documents. Sandbox could give assistance when users are searching for an option or setting inside an application, by allowing the user to search the application for that option and watching in a preview window what happens when a particular option is invoked [3].

Our next stage of research will focus on increasing the robustness of the Sandbox design to handle tasks that span across several applications, tasks that require several independent steps of interaction, and tasks that require both. For these situations, it is no longer sufficient to search the feature-list of an application, but to search across applications, and even search for applications themselves.

In addition to supporting single user interaction, we also intend to explore the potential of the Document Sandbox in collaborative situations, such as where a user initially tries to solve a problem themselves, but gets stuck and then needs to ask a colleague or technical support person for help, or two people try and work together to solve a problem with getting an application to have the desired

effect. A well-presented record of what has already been tried and might be tried next could save time and facilitate more effective and efficient communication.

Acknowledgements

We'd like to acknowledge the inspiration provided by current off-the-shelf software, in particular, our conceptual drawings of the Document Sandbox incorporated features borrowed from Microsoft Word 2000 and Adobe Acrobat Reader 4.0. This work is supported by the National Science Foundation under Grant No. 0081112.

REFERENCES

1. Berlage, Thomas. A Selective Undo Mechanism for Graphical User Interfaces Based On Command Objects. *ACM Transactions on Computer-Human Interaction* 1, 3 (1994), pp. 269-294.
2. Greenberg, S. and Cockburn A. (1999). Getting Back to Back: Alternate Behaviors for a Web Browser's Back Button. *Proceedings of the 5th Annual Human Factors and the Web Conference*, Held at NIST, Gaithersburg, Maryland, USA, June 3rd.
3. Mynatt, Elizabeth D., Terry, Michael. Side Views: Persisten, On-Demand Previews for Open-Ended Tasks. In *Proceedings of the 15th Annual ACM Symposium on User Interface Software and Technology* (October 27-30, Paris, France), ACM, NY, 2002, pp. 71-80
4. Papert, Seymour (1987): *Microworlds: Transforming Education*. In R. W. Lawler, and M. Yazdani (eds.): *AI and Education: learning environments and tutoring systems, Vol. 1*, Norwood, NJ: Ablex, pp. 79-94.
5. Quintana, C., Reiser, B., Davis, E.A., Krajcik, J., Golan, R., Kyza, E., Edelson, D., & Soloway, E. (2002). "Evolving a Scaffolding Design Framework for Designing Educational Software". *Proceedings, ICLS 2002: International Conference of the Learning Sciences*, Seattle, WA.
6. Rogoff, B Barbara (1990): *Apprenticeship in Thinking: Cognitive Development in Social Context*. New York: Oxford University Press.
7. Twidale, M.B. Over the shoulder learning: supporting brief informal learning embedded in the work context. *Technical Report ISRN UIUCLIS--1999/2+CSCW*.
8. Twidale, Michael B. and Nichols, David M. "Using Studies of Collaborative Activity in Physical Environments to Inform the Design of Digital Libraries." citeseer.nj.nec.com/twidale98using.html
9. Vygotsky, Lev (1986): *Thought and Language*. Cambridge, MA: MIT Press