

Extending Direct Manipulation in a Text Editor

David Novick, Francisco Romero, Edgar Rene Saenz and Armando Sandoval

Department of Computer Science
The University of Texas at El Paso
El Paso, TX 79968-0518
+1 915-747-5480

novick@cs.utep.edu

ABSTRACT

This paper describes the implementation of a prototype text editor that incorporates conversation-like features through the direct-manipulation modality. In this way, traditional direct-manipulation interaction techniques such as direct reference via pointing can be extended to include techniques more commonly associated with human conversation, such as negotiation of reference. The paper illustrates the use of the prototype with an extended example, and discusses research issues raised by the implementation.

Categories and Subject Descriptors

H.5.2 [Information Interfaces and Presentation]: User Interfaces – *Interaction styles (e.g., commands, menus, forms, direct manipulation)*.

General Terms

Documentation, Human Factors

Keywords

Direct manipulation

1. INTRODUCTION

The craft of creating electronic documents relies largely on software and interfaces such as Word™ and FrameMaker™ that embody now-familiar interaction techniques based on principles of direct manipulation as articulated by Shneiderman (1983):

- Continuous representation of the objects of interest
- Physical actions on objects vs. complex syntax
- Fast, incremental and reversible operations with an immediately apparent effect on the objects of interest
- Layered or spiral approach to learning

These principles are typically embodied in “WIMP” interfaces (windows-icons-menus-pointers). Word-processors are likely the archetypal application for WIMP interfaces. But as Beaudoin-Lafon (2000) observed, WIMP interfaces do not fully realize the potential of direct manipulation. There are other, arguably more

powerful, techniques that extend the usefulness and usability of direct-manipulation interfaces beyond WIMP. Novick (2001) further extended this idea by arguing that direct-manipulation techniques could also be used to implement aspects of interaction that are normally regarded as conversational. That is, the traditional direct-manipulation interaction techniques such as direct reference via pointing can be extended to include techniques more commonly associated with human conversation, such as negotiation of reference.

This paper reports the first implementation of a text editor that demonstrates this approach of extended direct manipulation (EDM). The prototype editor enables the author-user to define new kinds of referential relations, using direct-manipulation techniques instead of verbal language. The paper presents the theoretical background for the approach, illustrates the use of the prototype with an extended example, and discusses research issues raised by the implementation.

2. RELATED WORK

The field of providing support to authors is broad, to say the least. This paper looks in particular at extending the author’s interaction with a word processor through new uses of a familiar modality of interaction, direct manipulation. We will also briefly review approaches to reducing repetitive work in word processors, which address some of the problems of document preparation similar to those addressed by the extended direct-manipulation prototype. Finally, we will outline the interaction theory underlying the prototype’s design.

2.1 Direct-Manipulation “versus” Conversation

Direct-manipulation interfaces, typically implemented in the WIMP style, are good at enabling users to express direct reference through pointing. This deictic interaction lets users express things along the lines of “that sentence” or “this style.” Beaudoin-Lafon (2000) demonstrated an extension of direct-manipulation through *interaction instruments* that, for example, made text searches in a document more useful. In this case, the interaction instrument was a tool that let the user observe all instances of a search term in a window simultaneously. This went beyond WIMP but (a) still constituted a way of referring to items directly and (b) was a tool provided by a designer rather than created by the user him or herself.

In contrast with direct-manipulation’s ease in expressing deixis, verbal conversation faces difficulties. Even if one says “that couch”, one usually still has to point physically to be understood. Nevertheless, conversation has its uses. One use, which is difficult to execute non-verbally, is developing new relations among

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SIGDOC’02, October 20-23, 2002, Toronto, Ontario, Canada.
Copyright 2002 ACM 1-58113-543-2/02/0010...\$5.00.

referents and expressions. When they interact, people in conversations *create* new means of referring to things. In other words, referring is a collaborative process (Clark and Wilkes-Gibbs, 1986). This isn't possible in a typical WIMP interface because (a) such interfaces aren't collaborative and (b) the set of ways of referring to things is fixed by the design of the interface. In a broader sense, what conversants do—and what users of word processors cannot do—is collaboratively define and label new kinds of relations.

It has been shown that it is possible, in theory, to enable definitions of new kinds of relations in a direct-manipulation interface (Novick, 2001). For example, an author might want to create an abstract by selecting the first sentence of each paragraph in a document; the author could do so by defining a new “first-sentence-of-a-paragraph” relation and applying to the document. In other words, the issue is not direct-manipulation “versus” conversation; because conversation-like functions of defining new relations can be, in theory, expressed in the direct-manipulation modality. The open issue, then, is whether construction of an actual interface based on the EDM approach is feasible. As will be shown below, such an interface is, in fact, possible.

2.2 Other Approaches to Reducing Repetitiveness

Other researchers have taken a variety of approaches to saving authors' time and effort when faced with repetitive tasks like selecting multiple text fragments. Extending the taxonomy of Miller and Myers (2001), authors could use tools such as:

- Search and replace
- Keyboard macros
- Custom programs (e.g., in Perl or Emacs Lisp)
- Simultaneous editing of multiple text regions
- Programming by demonstration

Simultaneous editing of multiple text regions, developed by Miller and Myers (2001), involves the user's designating a set of regions to edit and then, by editing one of these regions, causing all of the regions to be edited identically. The deeper advantages and disadvantages of this technique are beyond the scope of this paper. The main point, though, is that this approach does help authors reduce repetitive work but it neither extends direct-manipulation interaction beyond WIMP nor enables authors to define *new* relations among objects being processed.

Programming by demonstration (PBD) involves the author's use of direct manipulation of objects in the interface (i.e., text, in a word processor), while the program observes and learns the interaction. In some cases, the user explicitly tells the system when to begin observing (e.g., Paynter & Witten, 2000). In other cases, the the system observes all the time, and offers assistance when it concludes that it might be of help (e.g., Cypher, 1991). In either case, though, the principal interaction is between the user and the application interface; the PBD functions as an observer of this interaction. Also, the feedback from the PBD is large-grained. That is, feedback normally comes at the end of a period of observation when the PBD concludes that it has grasped the demonstrated action.

A key difference, then, between PBD and the quasi-conversational approach taken here is that in EDM the user interacts directly with the system to negotiate new relations. This helps to avoid

problems of understanding that are inherent in the passive conversational role that observers take (see Schober & Clark, 1989). Direct interaction, as opposed to observation, involves finer-grained feedback that enables conversants to resolve ambiguities and avoid misunderstandings more effectively. Two parties interacting can reach a shared understanding; it's the point of the interaction. Reaching a shared understanding is much more difficult when one of the parties is simply an observer.

PBD interfaces, though, are more advanced than our EDM prototype with respect to action. That is, the point of PBD interfaces is to learn and repeat actions on interface objects. The EDM prototype as currently designed enables the author to define new relations but relies on the author to take any actions based on the new relations.

2.3 Theories of Meaning and Action in Human-Computer Interaction

The EDM approach grows out of two strands of research, both involving the relationship between meaning and action in interactions.

The first research strand centers on the notion of *grounding* in conversation. Grounding is the process by which two participants in a conversation know that they have reached mutual understanding of the elements that each has contributed to the conversation. Clark and his colleagues (Clark and Marshall, 1981; Clark and Wilkes-Gibbs, 1986) reported that, in human-human conversations, the conversants formed mutual beliefs about referents, and that this mutuality extended to the way in which expressions about these referents were generated and understood. Other work (e.g., Lambert and Carberry, 1992) had explored negotiation subdialogues about domain tasks. In particular, Clark et al. (1986) observed a kind of negotiation about the means through which concepts in the conversation were grounded. The conversants were able to create, in effect, new relationships that applied to things within the domain of their conversation. This led to use referring expressions that were shorter and more direct. The negotiated new meanings were based, of course, on previously known meanings that were—or could be assumed to be—shared between the conversants.

The second research strand centers on different means of taking action in human-computer interaction. Suchman (1987) demonstrated that a user's action in an interface is situated. That is, the user's task and other aspects of context provide guidance on what action to take next, as opposed to planning actions in advance. This idea of *situated actions*, which dealt principally with actions in the interface, was later extended to *situated acts*, which operate entirely at the level of domain actions (Novick & Perez-Quinones, 1998). One of the implications of the theory of situated acts is that actions in an interface can be abstracted to acts in the corresponding domain task, and then “respecialized” into a different set of interface actions. The nature of the new set of interface actions would depend on the modalities into which the acts were respecialized.

So creating relations in the direct-manipulation modality involves “negotiating” a new relation based on prior shared knowledge, using negotiation interactions that are respecialized for the direct-manipulation modality. In other words, the problem of creating new relations in a direct-manipulation interface consists of

solving the more abstract problem of creating new relations independent of modality and then specializing the solution into the direct-manipulation interface, if possible. Thus a “conversational” function like negotiation of the definition of a relation could extend a direct-manipulation interface by recognizing, at the more abstract, situated-acts level, that two things are needed to do create the new relation: (1) a means to perform ordered selection and (2) a common set of concepts for expressing relations. The EDM prototype interface presented in this paper is an implementation of this approach.

2.4 Spoken-Language Editing

Before leaving our review of related work, we want to make clear the distinction between (a) our proposed extension of the direct-manipulation modality with conversational functions and (b) using the speech modality to edit text.

In the case of EDM, the idea is to gain the power of some aspects of conversation while retaining the referential and expressive precision of direct manipulation. In the case of editing text by speaking to the system, performance is limited by referential ambiguity and by errors in speech recognition. Empirical comparison of methods of correcting text errors indicates that editing text is more effective in the direct-manipulation modality than in spoken-language modality (Suhm, 1997).

Review of spoken-language interfaces for text editing suggests that these speech interfaces are basically spoken-language implementations of WIMP interfaces. So, ironically enough, these interfaces suffer the well-known limitations of the spoken modality for document-editing tasks while not offering functions, such as defining new relations, that makes spoken conversation powerful.

3. THE PROTOTYPE

We now turn to the implementation of a working prototype based on the concepts of extended direct-manipulation presented in (Novick, 2001). We will present an extended example of interaction with the prototype.

3.1 Features of the Interface

The EDM prototype was implemented in Java. It is a simple text editor, with the basic functionality required to demonstrate the feasibility of EDM. Figure 1 presents a view of the prototype, with a document being edited in the text pane. The *file* and *edit* menus, and the *open*, *save*, *cut*, *copy* and *paste* icons, provide the usual functions familiar from WIMP word processors. The interface objects that enable the creation of new relations are

- The “Set Object 1” button. To define a relation, the user will need to select more than one object, and the order of selection may play a part in the semantics of the defined relation. Selections as Object 1 are highlighted in the text frame in the color displayed in the button, in this case a shade of pink.
- The “Undo Selection” button. The system will provide feedback, which we discuss below, about the object selected. Should the user learn that he or she ended up selecting something that they did not intend, then the user can click the “Undo Selection” button to undo the selection and try again.

- The “Reset” button returns the editor to a state where no objects have been set and no relations defined.
- The “Objects” pane. The system provides feedback to the user about the object selected. The feedback includes (a) the selected text and (b) the system’s characterization of the text in terms of concepts that the system and the users should mutually understand. Currently the prototype supports the concepts of character, word, sentence and paragraph.
- The “Relations” pane. In the view shown in Figure 1, the “Objects” pane is visible. The user can click the “Relations” tab to make the “Relations” pane visible. In this pane, the system expresses its understanding of the relation the user has defined. The user can highlight relations in the pane, and apply the relation to the text pane

With these features of the EDM prototype editor, the author can now use selection by direct manipulation to define new relations.

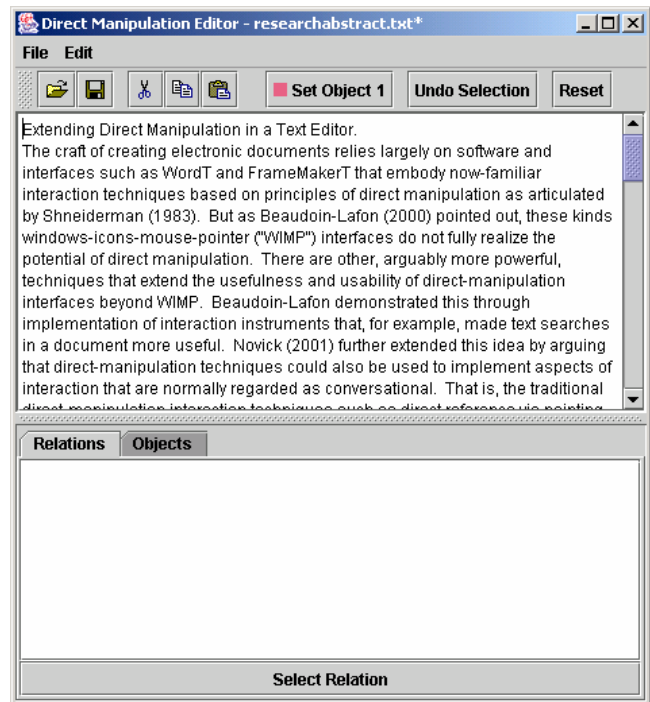


Figure 1. EDM Prototype Editor

3.2 Using the Prototype

As indicated in our introduction to the features of the prototype’s interface, the EDM editor has built-in primitive concepts of character, word, sentence and paragraph. The system uses simple syntactic rules about letters, spaces, returns, and punctuation to determine the type of a selected object. The system also has concepts of ordinality and of whether one object contains another. Using the direct manipulation modality to perform ordered selection on objects in the text window, the author can combine these concepts to define new relations, such as *second sentence of a paragraph*, or *last character of a word*.

Figures 2 to 4 show the use of prototype editor to define the relation *first word of a sentence*. This relation might be useful if,

for example, the author wished to change the font of the first word of every selected sentence to bold face.

From the state of the editor as presented in Figure 1, the author uses the cursor to highlight the first word of any sentence in the text window. In our example, the author highlights the word *Beaudoin-Lafon*. The author then clicks the “Set Object 1” button, which produces the state of the editor shown in Figure 2. At this point, the word *Beaudoin-Lafon* is highlighted in the color of the “Set Object 1” button. In the interface, this color is a dark pink; in the paper’s black-and-white figures, this highlighting prints as a medium gray. In the “Objects” pane at the bottom of the window, the editor reports that Object 1 has been set to “*Beaudoin-Lafon*” and that the object is a word. At the same time, the set-object button has changed label to “Set Object 2” and changed color to a medium blue. The dark-pink selection of the word in the text pane, the message in the “Objects” pane, and the change of button label all serve indicate that the system has selected “*Beaudoin-Lafon*” as Object 1 and that it is ready for the author to select Object 2.

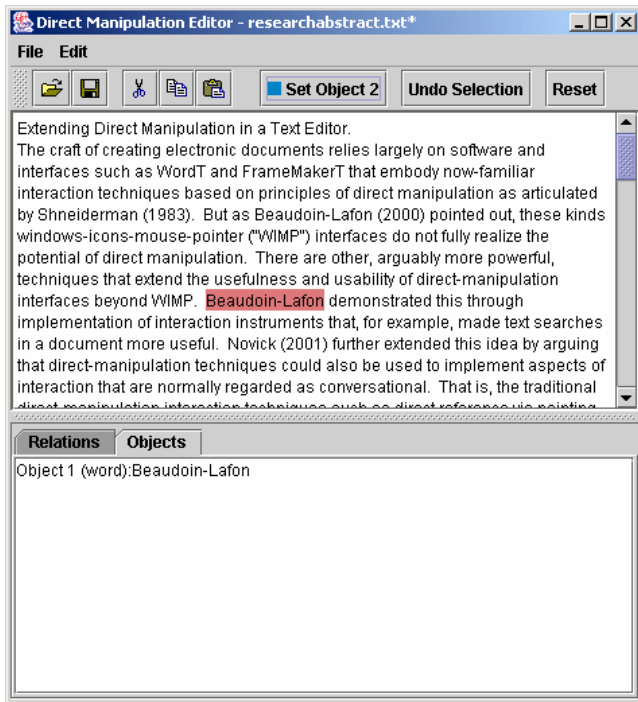


Figure 2. Object 1 selected

Using the same cursor that was used to select Object 1, the author now highlights the sentence for which “*Beaudoin-Lafon*” serves as the opening word. The author then clicks the “Set Object 2” button, which produces the result shown in Figure 3. Object 1, “*Beaudoin-Lafon*” remains highlighted in dark pink. Object 2, the entire sentence, is highlighted in medium blue, matching the color of the “Set Object 2 button.” In the paper’s black-and-white figures, this highlighting prints as a light gray. And in the Objects pane, the editor reports that Object 2 is “*Beaudoin-Lafon demonstrated this through implementation of interaction instruments that, for example, made text searches in a document more useful,*” and indicates that this object is a sentence. The set-object button returns to “Set Object 1,” indicating that the editor

is ready to define another relation or perform a normal editing operation such as cut, copy or paste.

Figure 4 shows the same state of the editor as in Figure 3, except that the author has now clicked on the “Relations” tab of the bottom pane. This causes the editor to show the Relations pane, which in this case contains the relation intended by the author, “*First word of a sentence.*” By highlighting this relation and clicking on the “Select Relation” button, the author can apply the *first word of a sentence* relation to the entire document or to selected text.

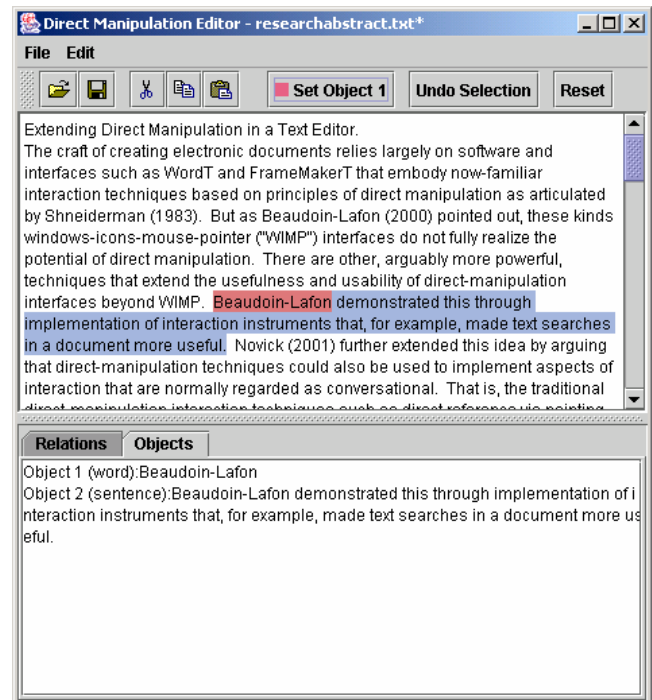


Figure 3. Object 2 selected

This example is representative of a class of relations involving parts of text and ordinality. Other similar instantiations of parts of text and specific instantiations of parts of text. For example, an author wanting to select all of the e-mail addresses in a document could, using techniques similar to those in Figures 2 to 4, create and apply the relation “word containing the character ‘@’”.

4. DISCUSSION

The EDM prototype editor is subject to a number of evident limitations. For one thing, the editing environment is fairly primitive, because the point of the prototype was to demonstrate the feasibility of the approach rather than to build a feature-laden application.

One possible criticism of the EDM prototype might be that, for relations like *first word of a sentence*, a developer could easily build in such relations without having the overhead of the EDM. In other words, why bother with defining new relations when the editor could have these already available, something like pre-defined styles for paragraphs. While the examples or relations presented in this paper may seem simple, though, they represent whole classes of possible relations. So while it would be possible

to build the “first word of a sentence” relation into the interface, this approach would not provide a general solution to this sort of problem, as the author might want to use some other relation later on. Even with the limited set of primitives provided by the prototype editor, an author could define relations such as *sentence containing the word “Texas”* or *word containing the letter “Q.”* So building in every possible relation in advance would be impractical and would likely lead to reference confusion with respect to relations. So the solution must lie in providing means for users and systems to negotiate new relations, as the prototype demonstrates to be possible.

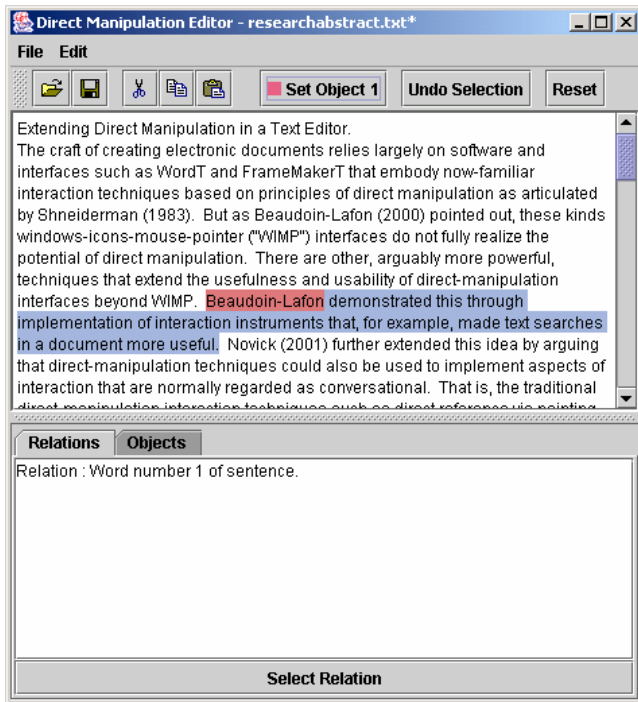


Figure 4. Relation defined and ready to apply

Another criticism of the EDM prototype might be that, even if the relations are not predefined, the prototype somehow “cheats” by relying on predefined primitives such kinds of text objects (word, sentence) and ordinals. In fact, such mutually understood building blocks are the essential foundation of all conversational negotiation of new relations. Just as the people conversing in Clark and Wilkes-Gibbs’s (1986) study used mutually known English words to define new ways of referring, so too does the EDM prototype rely on the fact that the primitive concepts are mutually known by the system and the author.

4.1 Insights Provided by the Implementation

The process of implementation, and the use of the prototype made possible by the implementation, enabled us both to refine our theory of extended direct manipulation and to propose different kinds of relations. From a theoretical standpoint, the implementation has helped us to see that the EDM approach, in effect, enables users to define their own interaction instruments, rather than having to depend on predefined system tools.

The implementation also led to more practical insights. First, in its original conceptualization, the prototype required two cursors. It

turned out that one cursor, with the alternating “Set Object” button, was sufficient for providing ordered selection. Second, the implementation process also enabled us to develop new kinds of relations and primitives. An example of this would be the case of an author selecting all sentences containing the word “implementation”.

4.2 Open Issues

A possible criticism of the EDM prototype, and the EDM approach in general, is that its utility may be limited to the examples we discuss in this paper. This may be, in fact, a serious problem. The widely accepted rationale for the failure of PBD systems like Eager (Cypher, 1991) to be adopted is that such systems worked well on their canonical examples but did not have much utility beyond that. We expect that the EDM approach will have broader application because (a) it applies to more than just repetitive tasks, (b) it does not depend on a learning algorithm, and (c) new relations are routinely developed across a wide range of human conversations. Nevertheless, this is an important open issue that is, in part, the subject of our current research.

We are beginning usability and other, similar measures, to determine if the EDM approach is not only feasible but actually helpful.

The implementation effort also raised a set of additional issues that we believe are worth considering. These issues include:

- How should the interface handle ambiguous specifications of relations? For example, did the author mean to define the relation *first word of a sentence or sentence that contains the word “Beaudoin-Lafon”*?
- How does the author express more relations—such as *sentence with the words “foo” and “bar”* or *words with two F's in them*? This would appear to require more than two set-object states or an unpalatable dialogue-box approach.
- How does the author express relations where an example of the relation is not in view?
- Does the order of selection have meaning? That is, are there natural, mutually understood semantics for ordered selection? For example, does selecting a word and then a *sentence intend first word of a sentence or sentence containing <word>*?
- Would it be possible to reuse and reincorporate previously defined relations in newly defined relations?
- What other kinds of relations can be created with the EDM approach?
- What other kinds of conversational functions, in addition to defining new relations, could be provided through direct manipulation?

We will be looking at these questions through continued refinement of the prototype EDM editor, through preliminary usability studies, through application of EDM principles to applications beyond text editing, and possibly through discourse analysis of simulations of computing systems.

5. ACKNOWLEDGMENTS

This work was supported by National Science Foundation Grant No. 0080940. Gabriel Sotelo helped with the implementation of the prototype. Karen Ward, Robert Demolombe, Laurent

Chaudron and the SIGDOC reviewers provided helpful comments.

6. REFERENCES

- [1] Beaudoin-Lafon, M. (2000). Instrumental interaction: An interaction model for designing Post-WIMP user interfaces. *Proceedings of CHI 2000*, The Hague, The Netherlands, April 2000, 446-453.
- [2] Clark and Marshall, 1981] Herbert Clark and Catherine Marshall. Definite reference and mutual knowledge. In A K. Joshi, B. L. Webber, I. A. Sag (Eds.), *Elements of discourse understanding*, pages 10-63. Cambridge University Press New York, 1981.
- [3] Clark, H., and Wilkes-Gibbs, D. (1986). Referring as a collaborative process. *Cognition*. 22, 1-39.
- [4] Cypher, A. (1991). Eager: Programming repetitive tasks by demonstration. *Proceedings of CHI 91*, April, 1991, New Orleans, LA, 33-39.
- [5] [Lambert and Carberry, 1992]. Lynn Lambert and Sandra Carberry. Modeling negotiation subdialogues. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, Newark, DE. Association for Computational Linguistics, 193-200.
- [6] Miller, R. C., and Myers, B. A. Interactive simultaneous editing of multiple text regions, *Proceedings of the 2001 USENIX Annual Technical Conference*, June 2001.
- [7] Novick, D., (2001). "Conversational" Dialogues in Direct-Manipulation Interfaces, *Proceedings of the 2nd IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, Seattle, WA, August 5, 2001.
- [8] [Novick and Perez-Quiñones, 1998]. David Novick and Perez-Quiñones. Cooperating with computers: Abstracting from action to situated acts, *Proceedings of the Ninth European Conference on Cognitive Ergonomics (ECCE-9)*, pages 49-54. Limerick, IR, August, 1998. European Association for Cognitive Ergonomics.
- [9] Paynter G. W. and Witten I. H. (2000) Developing a practical programming by demonstration tool, *Proc. Australian Conference on Computer-Human Interaction (OzCHI)*, pp. 307-314, Sydney, Australia.
- [10] Schober, M. F., and Clark, H. H. (1989). Understanding by addressees and overhearers. *Cognitive Psychology*, 21, 211-232.
- [11] Shneiderman, B. (1983). Direct manipulation: A step beyond programming languages. *IEEE Computer*, 16(8):57-69, 1983.
- [12] [Suchman, 1987] Lucy Suchman. *Plans and situated actions*. Cambridge University Press, Cambridge, 1987.
- [13] Suhm. B. Empirical evaluation of interactive multimodal error correction, *IEEE Workshop on Automatic Speech Recognition and Understanding*, , Santa Barbara (CA), 1997, 583-590