

# Combining the Analysis of Spatial Layout and Text to Support Design Exploration

J. Michael Moore & Frank M. Shipman III

Department of Computer Science, Texas A&M University

College Station, TX 77843-3112 USA

1 979 862 3216

{michael, shipman}@csdl.tamu.edu

## Abstract

The Design Exploration (DE) approach allows a large number of probable end users to communicate with software developers by creating mockups of user interfaces and augmenting the partial designs (e.g. windows and widgets) produced with textual descriptions and explanations. Software developers collect and analyze these annotated partial designs to gain an understanding of the user, their activity, and the domain of concern. This paper presents the DE Analyzer and its techniques for automatically analyzing the collection of annotated partial designs. The DE Analyzer uses a combination of textual and spatial layout analysis algorithms to assist software developers' navigation and exploration of the information collected using the DE Builder.

## 1. Introduction

When creating interactive software, designers must understand the domain, work practices, and user expectations before determining formal requirements or generating initial design mock-ups. Gathering this information from users and other stakeholders during initial concept development and the early stages of software design can greatly decrease the cost of integration. Design Exploration (DE) is one approach for collecting that information.

DE provides a communication medium to elicit

more information than surveys while maintaining a low-overhead per participating user. While not meant to generate the rich information provided via participatory design or other methods requiring face-to-face meetings, it can be used to validate and enhance feedback from such a process. Addressing this middle ground broadens the number of users that can contribute to software design.

This approach retains the remote and asynchronous communication of surveys while making authoring of feedback easier by providing users alternatives to textual communication for expressing their suggestions and tacit understanding of the domain [7]. Situating this communication in reference to an artifact facilitates "design by doing" [1]. Some have noted that this integration of spatial and linguistic information is required for effective communication [3]. So, communication is facilitated from users to designers by combining visual and textual modes of expression.

The DE process (Figure 1) allows users to communicate requirements information through the construction of graphical user interface mock-ups augmented with textual argumentation, i.e. descriptions and explanations [5][6]. These user constructed annotated partial designs are then collected and asynchronously examined by software developers.

The volume of expressions produced by users can become quite large making manual analysis of each user's windows, widgets and annotations cumbersome. Clearly, analysis assistance is vital for this method to

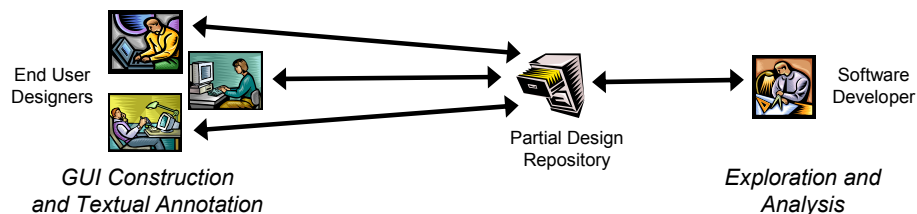


Figure 1: Design Exploration process

be successful. Automatically processing user expressions using a combination of spatial and textual techniques helps software developers manage the analysis of user expressions.

Next we describe the tools we have created to support DE. Then we discuss the automated processing of user expression and finish with conclusions.

## 2. The DE Builder

The DE Builder allows potential users and stakeholders to create annotated partial designs and collects them for analysis by software developers. While we could have modified an IDE (integrated development environment), the complexity of such an interface would negate our goal of providing a simple environment to users. We did not incorporate sketching [4] since most users do not have sketching equipment and sketching is difficult using a mouse.

Users can rapidly create and mock-up multiple windows from a set of widgets and describe the function of the windows and widgets via text as they deem desirable. The widget set includes specific widgets, e.g. Radio Buttons, as well as a "generic widget" that can represent interface design elements, e.g. Drawing Area, that are not in the available widget set. This combination supports the expression of interface designs that are easy to develop using standard widgets and those that are not. Also, this supports expression when users have a specific design in mind and when they do not

## 3. The DE Analyzer

The Analyzer is a tool to browse and understand the communication generated through the DE Builder. Its interface is based on the structure used in many IDEs, tools familiar to most software developers.

Browsing and navigating within the collection of annotated partial designs is supported by a combination of spatial layout and textual analysis techniques (presented in detail below). The DE Analyzer includes a term dictionary and text-based search. Spatial analysis is used to identify substructures in the users' partial designs such as lists of radio or menu buttons.

The analyzer provides several perspectives on the set of partial designs. These include a term-based view, a hierarchical view of windows, a view based on spatial parsing, and a view showing the results of hierarchic agglomerative clustering of design elements based on the textual similarity of their content.

## 4. Understanding User Expression

Communication through design creates information that is expressed both textually and graphically through the layout of windows and widgets. Providing the designer with appropriate browsing opportunities and search results depends on identifying similarity between design elements. Analysis techniques must be tailored to the unique information space generated by the DE process. How does the complex structure of interface designs translate into documents for textual analysis? How do the terms related to the task of building user interfaces affect textual analysis of domain concepts? Finally, what is the best way to cope with the sparse text used to annotate and describe design elements?

The DE Analyzer addresses these concerns by blending spatial and textual analysis. A spatial parser [8] identifies design substructures composed of groups of widgets. These substructures are treated as additional documents for textual analysis. These together enable the DE Analyzer to apply similarity metrics and clustering to the design substructures.

### 4.1. Spatial Parsing

The layout of widgets in a window results in structural information that is apparent to people but not always to machines. Requiring people to explicitly provide this information adds a burden to users in a process that is meant to be quick and informal. Spatial parsing allows computational systems to infer structural information from spatial organization.

Figure 2 shows a window from one user's partial design that contains multiple distinct substructures. One substructure is the set of navigation buttons at the bottom right-hand corner. Each list of radio buttons also represents a substructure. These lists are perceptually a single sub-group, even though they are composed of multiple individual radio buttons. Each list represents different aspects of the domain. Failing

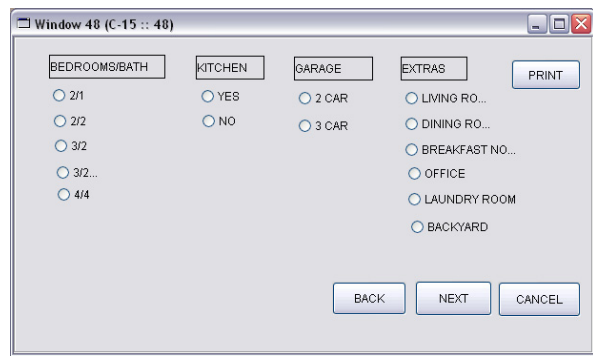
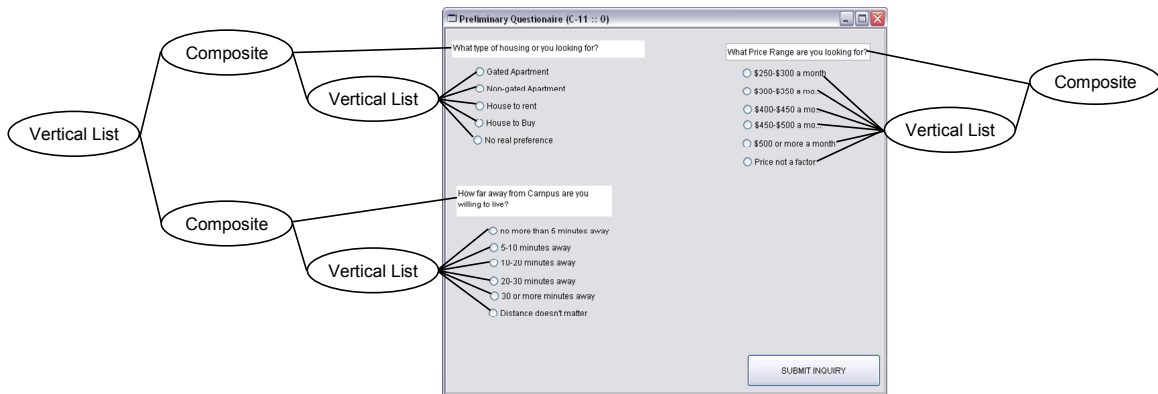


Figure 2: Partial design with distinct areas



**Figure 3: Tree representation of spatial parse of a user designed window**

to consider these groups as distinct units could miss finer grained connections. In this case, using the whole window would dilute the impact of each list's terms when searching for similarity across user designs.

Spatial parsers are designed to recognize the structures in object layouts that people perceive [8]. As such, they are a special class of pattern recognition or vision system that makes use of the discrete nature of the objects, their well-defined locations, and their explicit or implicit type. Most work on spatial parsing has been in the context of spatial hypermedia systems [2]. Spatial parsers look for patterns in the spatial layout of objects to identify orderings of objects such as lists and stacks. The DE Analyzer uses the spatial parser from VIKI and VKB [8].

Figure 3 illustrates a hierarchical spatial parse of one window from a user's design. Determining how to break the parse tree into meaningful units depends on the domain and task. For the DE Analyzer, combining composites actually lessens the effectiveness of parsing as the groups are meant to provide access to relatively focused domain concepts. The items in each list of radio buttons are related to each other, but the only relationship shared by the different lists is that they are aspects of the larger housing search process. Currently, we limit analysis to spatial groupings that contain only leaves, i.e. only widgets and not groupings that include composites. So in Figure 3, three spatial groups of radio buttons are identified for use as documents (i.e. the three vertical lists).

## 4.2. Textual Analysis

"Documents" must be identified for calculating textual similarity. The three document types are 1) an individual widget consisting of the widget's descriptive text and any text defining properties of the widget e.g. the label on a button, 2) the aggregation of a window and the widgets it contains, and 3) design substructures identified through spatial parsing.

Figure 4 shows a simple user created window and how it is broken up into eight documents. Document 1 is the user entered text for the window excluding any of the widgets it contains. Documents two through six consist of the user text for each of the individual widgets. Document 7 is a document derived by aggregating the text from each of the four widgets in the list (documents three through six). Finally, document 8 aggregates the text from the window and all of the widgets contained in the window.

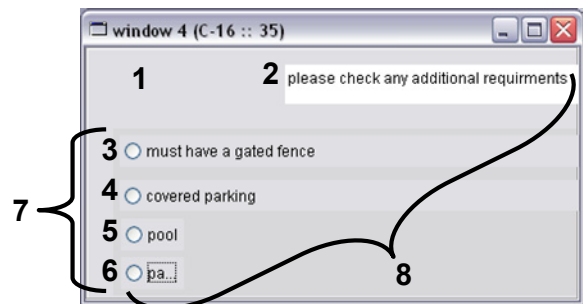
### 4.2.1. Term Vectors

We currently use the standard vector space model to represent and compare documents, e.g. units of text [9]. In addition to standard stop words, the DE process uses 99 additional stop words that stem from using the language of the GUI. Examples of these are "button" and "click." These terms appear more frequently because they describe user interface elements and actions, but do not deal directly with domain concepts.

### 4.2.2. Document Similarity

Similarity is primarily based on text using the vector space model [9]. Here similarity is assessed by taking the cosine between the term vectors for each document.

For comparison, the term vectors for all three types



**Figure 4: Document identification**

of documents are analyzed as a set. A list of similar items can be extracted for a particular term vector, e.g. from a group or element found while exploring the space. Similar documents can be individual widgets and sets of widgets identified through spatial parsing.

### 4.2.3. Term Dictionary

To calculate similarity between documents and to provide term-based access to the set of partial designs, the DE Analyzer maintains a dictionary of all partial design terms including references to the windows and widgets (i.e. documents) that contain those terms. The interface to the term dictionary shows the term, the windows and widgets where it appears. When a term is selected, it is highlighted where it appears.

Terms may be ordered alphabetically or by frequency. Terms with a low frequency can indicate unique concepts that designers may want to explore further. Terms with a high frequency across users indicate common domain concepts.

### 4.3. Clustering

The DE Analyzer uses hierarchical agglomerative clustering where term vector cosine similarity is the distance metric. Clustering is performed on three sets of documents: 1) individual widgets and windows (e.g. documents 1 through 6 in Figure 4). 2) aggregation of entire windows (e.g. document 8), and 3) design elements that can hold multiple items i.e. windows (e.g. document 8), groupings identified by the spatial parser (e.g. document 7), and list and combo box widgets (if they contain more than one sub-item).

Clustering presents possible relationships between groups of design elements and acts as a starting point for examining similarity between users' partial designs. Moreover, overlapping concepts may get assigned to different clusters.

## 5. Conclusion

Combining spatial layout and textual analysis gives software developers easier access to relations that would otherwise be unavailable. Participants in a small formative study found that the ability to search and navigate the collection of partial designs valuable but had mixed reactions to the similarity-based navigation and the clusters provided. One participant indicated that the clustering helped navigation through the partial designs while another thought that the clusters made no sense. Regardless, participants saw the value of the process and the need for a tool to assist with

analysis, especially as the number of partial designs gets larger.

The DE process generates a uniquely structured information space of user interface design elements. Combining spatial and textual analysis techniques helps software developers navigate and analyze this space.

The spatial parser identifies substructures to use in addition to windows and widgets for textual analysis that is tailored for the DE process. This combination of spatial and textual analysis provides options for exploration through clustering, similarity search, and term browsing. The DE process is a viable alternative for collecting design information and promises to improve as the automated analysis techniques address issues uncovered in our formative study.

## 6. Acknowledgements

This work supported in part by NSF grant 04-38887.

## 7. References

- [1] Ehn, P. 1988. Playing the language-games of design and use-on skill and participation. In *Proc. of the ACM SIGOIS and IEEECS TC-OA 1988 Conf. on Office Information Systems*. Palo Alto, CA, Mar. 142-157.
- [2] Francisco-Revilla, L. Shipman, F. 2005. Parsing and Interpreting Ambiguous Structures in Spatial Hypermedia. In *Proc. of the 16<sup>th</sup> ACM Conf. on Hypertext and Hypermedia*, Sept. 107-116.
- [3] Glenberg, A.M. McDaniel, M.A. 1992. Mental models, pictures, and text: Integration of spatial and verbal information. *Memory and Cognition* 20, 5, 458-460.
- [4] Landay, J.A. Myers, B.A. 2001. Sketching interfaces: Toward more human interface design. *Computer* 34, 3, 56-64.
- [5] Moore, J.M. 2003. Communicating requirements using end-user GUI constructions with argumentation. In *Proc. 18th IEEE International Conf. on Automated Software Engineering*. Montreal, Canada, Oct. 360-363.
- [6] Moore, J.M. Shipman, F.S. 2000. A comparison of questionnaire-based and GUI-based requirements gathering. In *Proc. of the 15th IEEE Int. Conf. on Automated Software Engineering*, Sept. 35-43.
- [7] Polanyi, M. (1966). *The Tacit Dimension*. Garden City, NY: Doubleday.
- [8] Shipman, F.M., Marshall, C.C., Moran, T.P. 1995. Finding and using implicit structure in human organized spatial layouts of information. In *Proc. of the SIGCHI Conf. on Human Factors in Comp. Systems*. 346-353.
- [9] Salton, G., Wong, A., Yang, C.S. 1975. A Vector Space Model for Automatic Indexing. In *Comm. of the ACM*, vol. 18, nr. 11, pages 613-620.