CHAPTER 8

A SYSTEMATIC APPROACH TO USER INTERFACE DESIGN FOR A HYPERTEXT FRAMEWORK [*]

Abstract

A number of navigational tools exist for hypertext systems. Authoring guidelines have also been proposed for the organization of information in hypertext systems. However, there has been no systematic and comprehensive approach towards the design of user interfaces for hypertext systems. This paper is an attempt to apply a set of user interface design guidelines to a hypertext framework based on a cognitive model. This framework had classified nodes and links into various semantic types. We believe that such a classification is of great importance in developing an appropriate design metaphor/user interface for a hypertext system. A systematic approach to user interface design would also reduce functional opacity and system opacity.

Keywords: Hypertext, User Interface Design, Cognitive Models, Guidelines.

1. Introduction

The issue of incorporating semantics into a hypertext network has been addressed before [7,25]. Rao and Turoff had proposed a general framework for hypertext functionality based on Guilford's Structure of the Intellect Model [20]. They had observed that as hypertext databases grow in size, they suffer from a lack of coherence due to ambiguity in meanings assigned to nodes and links. Hence, they proposed a framework which classified nodes into six different semantic types and links into twelve different types. Such a comprehensive framework would help designers develop better design metaphors and implementation models for hypertext systems. A first step in this direction is to develop an appropriate user interface which would reduce functional and system opacities.

A number of tools have been developed for hypertext navigation since Conklin addressed the issues of disorientation and cognitive overhead [8]. These include graphical browsers, overview diagrams, web views, paths, trails, guided tours and tabletops, history lists, timestamps, footprints, bookmarks, backtracks, queries, embedded menus, fisheye views and roam and zoom techniques [3,7,11,13,17,24,28,30]. Authoring guidelines have also been proposed for the organization of information in a hypertext system [23]. However, there has been no systematic and comprehensive approach towards the design of user interfaces for hypertext systems. This paper attempts to explore the application of a set of user interface design guidelines to the aforementioned hypertext framework.

2. Principles of User Interface Design

Designing modern user interfaces requires an in−depth understanding of various cognitive and mental models. One of the most widely used cognitive task analysis models in the area of user interface design is the GOMS (Goals, Operators, Methods, and Selection Rules) model [5]. It states that users employ certain established methods or procedures to achieve a pre−defined set of goals using operators and selection rules. The KLM (Keystroke−Level Model) attempts to predict time measures for error−free performance of tasks based on the sum of time taken for keystroking, pointing, homing, drawing, thinking, and waiting for the system to respond [21]. Both these models deal with error−free operation and do not address uncharted anomalies of operations when there are multiple goals and constraints [10].

The GOMS model and KLM models are suitable for tasks with well−defined structure such as text−editing or other linear tasks. However, they are not suitable for tasks with high cognitive variability such as navigating through a hypertext network. Hence, alternative models or approaches are required to develop user interfaces for hypertext systems. A first step in this direction is to develop a good user interface in order to reduce disorientation and cognitive overhead. This requires an understanding of the organizational setting, the targeted task domain, the typical user population, and the desired outcomes of hypertext navigation.

Various user interface design techniques and usability principles have been proposed by researchers and practitioners [1,15,18,19,21,26]. We have chosen the following set of guidelines [27] to be applied to the hypertext framework since they were found to be more systematic and comprehensive than other similar guidelines:

1. Identify the metaphor.

2. Identify all objects that make up the system.

3. Identify all actions/functions that can be performed on these objects. Separate them into generic actions, explicit actions, and control functions.

4. Identify modifiers/filters that select subsets of objects.

5. Identify strategic choices which allow the user to accomplish a specific task.

6. Identify lateral classifications – objects and actions that relate to each other.

7. Identify the formats of objects, parts of objects, menus etc.

8. Identify lists of objects.

9. Identify reactive choices that can be performed on a list or a set of objects.

10. Identify processes or functions that share information.

11. Identify all user interaction states.

12. Identify necessary help throughout the system.

13. Identify all error conditions.

14. Identify the screen layout – workspace, control area, status area, message area etc.

3. User Interface Design Guidelines for a Hypertext Framework

This section describes the user interface guidelines applied to the hypertext framework proposed by Rao and Turoff. The classification of nodes and links into various types makes it easier to design an appropriate user interface for a hypertext system.

3.1. Identify the metaphor

Carroll and Thomas have established that people develop new cognitive structures by metaphorically extending old ones [6]. Users of a new computer system can master it if they can metaphorically extend it to some real world objects or entities. A good metaphor not only helps the user, but also provides a rigid framework within which the hypertext author or designer must work to maintain consistency. Choosing an appropriate metaphor would also reduce both functional opacity (mismatch between the framework and the metaphor) and system opacity (mismatch between the metaphor and the implementation model) [20]. Hypertext has been compared to electronic encyclopedia, notecards, journeys, browsing, windows, paths, guided tours, travel holiday, and survey–type maps [9,12,14]. The travel metaphor serves as an extremely powerful aid to hypertext navigation [2]. At the same time, metaphors should not become too restrictive.

Hammond and Allinson say that "the system should improve upon the metaphor, not be bounded by it." [12].

The metaphor suggested by this framework is "the general cognitive model of how individuals think about complex problems." [26]. In order to understand hypertext, designers must understand writing and reading models. According to the Cognitive Framework for Written Communication, writing is a combination of three activities: exploring, organizing, and encoding [22]. Writing is the transformation of a network of related concepts (retrieved from long−term memory or external sources) into an outline or a hierarchy which is later encoded into a linear sequence of words, sentences, paragraphs, sections, chapters, and illustrations. Reading is the execution of the above three processes in the reverse order. That is, a linear sequence of text is transformed into a hierarchy which is later integrated into a network in long−term memory. Thus, both reading and writing processes are based on the non−linear nature of thinking, a natural process in human beings. Human cognition is essentially organized as a semantic network in which concepts are linked together by associations. Hypertext systems should try to exploit this basic nature of cognition.

3.2. Identify all objects that constitute the system

In the hypertext framework based on the Structure of the Intellect model, both nodes and links can be considered as objects. Nodes are classified into six semantic types − detail, collection, proposition, summary, issue, and observation. Links are of two major types − Convergent and Divergent. Convergent links can be further classified into specification, membership, association, path, alternative, and inference links. Divergent links can be categorized into elaboration, opposition, speculation, branch, lateral, and extrapolation links. The emphasis of this framework is the association of semantics to nodes and links. Organizing nodes and links semantically helps manage the hypertext network and its sub−networks better. It would also help reduce ambiguity, disorientation, and cognitive overhead.

A collection node can be considered to be a composite node that can be comprised of details, propositions, summaries, issues, observations, and collections. A hypertext template can be considered as a collection node since it is defined as a set of pre−linked documents that can be duplicated [4]. The usage of a template will greatly speed up the process of an average user’s understanding of the model or the metaphor. Templates automate the process of creating hypertext collections by creating skeletons of documents and linking them. Without a template, a hypertext author will have to start constructing the hypertext collection of ideas from the beginning. Many applications such as collaborative writing, collaborative hypertext, teaching aids etc., will greatly benefit from the concept of a collection node or hypertext template.

3.3. Identify all actions that can be performed on the objects

Actions and functions are the operations that can be performed on these 6 node types and 12 link types. Actions can be generic, explicit, or control−like.

3.3.1 Generic Actions

The ability to create webs of information from nodes and links between nodes is an inherent property of hypertext. Facilities to add, modify, and delete nodes and links can be considered as generic actions. That is, though these actions have a common meaning, their execution depends on the kind of object they act on. For example, deletion of a proposition node must delete all associative and speculative links to and from it. Similarly, deletion of an elaborative link must delete the footnote (a detail node) related to another detail node. Thus, the delete operation must behave differently based on whether the object is a node or a link. Similarly, a search operation can behave differently for different kinds of objects. Generic actions can also include the ability to create, edit, duplicate, or delete templates. Duplication of a template involves creating empty documents or nodes and the links or link sets associated with them.

3.3.2 Explicit Actions

Explicit actions are those which require an explicit qualifier or a modifier. The list operation is an explicit action since it requires the type of an object or other information. The result of listing all detail nodes can be a list in short form with object names or identifiers. A slightly informative version can be a preview of object names with all links to and from them and all nodes connected to them. A more complete explicit function can be the listing of nodes with their contents and all other nodes connected to them. The index operation can be provided by associative links in this framework. Explicit actions on a template can include the ability to add contents to empty documents, listing templates and their constituent documents and links, looking at an overview of the template, accessing a template by its type ("get a copy of the planning template") etc.

3.3.3 Control Functions

Control functions provide the ability to zoom into a subset of the hypertext network, forward and backward navigation through the web, provide overview maps, roam and zoom techniques, paths, trails etc. Control actions on a template can include displaying an overview of the template, zooming into specific link sets or webs (or "specific subgraphs") and looking at the contents of documents.

3.4. Identify all modifiers that select subsets of objects

The framework identifies all nodes and links by their semantic types. This type information can become a modifier or qualifier to the specific object. For example, a search can be performed on all nodes of type "detail". Similarly, we can also request for all links of type "specification". Type information and keyword information can be combined to narrow down the search criteria. For example, there can be a query to retrieve all nodes of type "detail" containing the keyword "hypertext". Modifiers for templates can include the type of a template, the author’s name, creation date etc.

3.5. Identify all strategic choices

Strategic choices include user interaction with the system in order to accomplish a specific task. These can be treated as landmarks [17]. Strategic choices might include overview requests (displays of summary nodes), structural and content query facilities, navigation mechanisms, editing tools, display options, audit trail mechanisms, linearization techniques, and backtracking facilities. The inherent nature of hypertext is that there is really no need for strategic choices. Any node or link in the entire network can be a strategic choice. In a true hypertext system each node should be self sufficient and complete. Also, the set of strategic choices need not be the same every time the user interacts with the system. For example, the user can directly access required nodes through query mechanisms. This feature must be available even without traversing the network. Hence, a query facility can be considered as a strategic choice. Strategic choices can also include an overview diagram of the template (its contents and links), the ability to find out from which master template a duplicate was created, the ability to edit the master template etc.

3.6. Identify lateral classifications

Hypertext provides the inherent capability of creating lateral classifications. The ability to create a lateral link to another version of a node or to an annotation (a detail or observation node) can be considered as lateral classification.

3.7. Identify the formats of objects and parts of objects

Though this section is more applicable only when designing the actual interface for a particular application, the designer must decide on the formats for nodes and links. Formats should also be defined for overview diagrams, indexing mechanisms, query facilities, and results of a query. A template can be considered as a pre−formatted collection of webs that can be directly used by the author.

3.8. Identify lists of objects

This is related to identifying explicit actions and modifiers. It is necessary to identify the kinds of lists that might be produced − lists of nodes or links or templates classified by semantic type, lists of nodes last visited, lists of nodes or links last modified etc. Lists can also include user created annotations.

3.9. Identify reactive choices that can be performed on a list or a set of objects

The ability to directly interact with (or manipulate) a node or a link in an overview diagram and view information contained within a node (or traverse to the next node) can be considered as a reactive choice. Previewing a node and the links to and from it (which can be highlighted showing a subset of the network) is another reactive choice. The ability to mark a node which has been visited using timestamp or footprint mechanisms is another reactive choice. Other reactive choices include marking a text item, creating elaboration links, and selecting from a list of objects. The ability to mark and annotate will help the user to store and manipulate retrieved information in order to integrate with other work. Reactive choices can include the ability to directly manipulate the contents of documents within a template such as editing, deleting, creating new links etc.

3.10. Identify shared processes

The ability to create objects (whether nodes or links) or list them is common across the system irrespective of the kind of object. Hence, these can be shared processes reacting differently based on the type of object being acted upon (similar to the object−oriented concept of operator overloading). The shared process of creation can be extended to create templates.

3.11. Identify all user interaction states

The designer must identify all possible interaction states, navigation paths, list requests, queries, backtracks etc. The user should always be informed about the current interaction state so as to minimize disorientation. The system should also allow the user to "peek" at a destination (before reaching it) by highlighting all the nodes and links connected to the specific node. The system can also inform the user about the node and link types in the user's language. For example, if a detail node has an elaboration link to a footnote (another detail node), then the user should be informed through a feedback mechanism − "See associated footnote for further explanation of this concept". All user interactions can be stored in a history file so that the user can revisit the previously visited nodes the next time he or she brings up the system. Guided tours can be considered as a set of pre−defined interaction states or trails. Restoring previous interaction states to the way they were originally seen is very important for backtracking mechanisms [16].

3.12. Identify necessary help

Context−sensitive help is very essential in this kind of an environment to reduce disorientation and cognitive overhead. In this context, all the six node types and twelve link types must be well defined with examples so that users understand the exact classification of information. Help can be provided in terms of examples of hypertext collections created out of templates. Such templates can also be used for personal notes or annotations. Help can also be in the form of feedback on interaction states, reactive choices available at a particular state, navigational cues etc.

3.13. Identify all error conditions

All possible error conditions should be identified − it is better to prevent errors than "handle" them. There should not be menu choices or options that are not applicable in a particular interaction state. Error messages should be precise and constructive so that the user is informed of the exact cause of the problem and the steps to be taken to recover from it [15].

3.14. Identify the screen layout

Although screen design is application dependent, appropriate graphic design principles and usability principles must be used in designing the actual screen layout. The menubar should contain strategic choices as menu elements that are consistent across different components of the system and across different interaction states. The workspace or canvas should contain overview diagrams and displays of information contained within a node. The feedback area should be used to keep the user informed about interactions, status of queries, updates etc. The control area should provide facilities to zoom, pan etc.

4. Discussion

Rao and Turoff had claimed that all current hypertext systems fall under their hypertext framework and their semantic morphology can be extended to all future systems [20,26]. Since the user interface guidelines have been applied to such a general framework, we believe that these guidelines encompass user interface designs for all current and proposed hypertext systems. The next step is to determine the effectiveness of this approach by developing a user interface prototype for the discussed hypertext framework and conducting usability experiments.

5. Conclusion

Very little work has been done towards a systematic and comprehensive approach to the design of a user interface for a hypertext system using traditional and modern user interface design principles. This paper proposed the application of a set of user interface design guidelines to a hypertext framework developed earlier. It has been suggested that the understanding of how people use their cognitive skills to handle information will greatly enhance the design and evaluation of hypertext systems [29]. We believe that a user interface design based on the above systematic approach combined with the semantic richness of nodes and links would greatly reduce ambiguity, disorientation, and cognitive overhead. We need such an approach in order to integrate hypertext applications into our daily work in an increasingly collaborative environment.

References

1. Apple. Human Interface Guidelines: The Apple Desktop Interface, Addison−Wesley Publishing Company, Inc., 1987.

2. Baird, Patricia and Percival, Mark. Glasgow Online: Database Development using Apple’s HyperCard, Chapter 5, Hypertext: Theory Into Practice, Edited by McAleese, Ray., Ablex Publishing Corporation, New Jersey, 1989.

3. Beard, David & Walker, John. Navigational Techniques to Improve the Display of Large Two−Dimensional Spaces, University of North Carolina at Chapel Hill Technical Report TR87−031, November 1987.

4. Catlin, Karen S., Garrett, Nancy L., and Launhardt, Julie A. Hypermedia Templates: An Author’s Tool. Proceedings of Hypertext’91, ACM Press, 1991.

5. Card, S.K., Moran, T.P., and Newell, A. The Psychology of Human−Computer Interaction, Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1983.

6. Carroll, John M., and Thomas, John C. Metaphor and the Cognitive Representation of Computing Systems, IEEE Transactions on Systems, Man, and Cybernetics, March − April 1982.

7. Collier, George H. Thoth−II : Hypertext with Explicit Semantics, Proceedings of Hypertext ’87 Conference, November 1987.

8. Conklin, Jeff. Hypertext : An Introduction and Survey, IEEE Computer, September 1987.

9. Edwards, Deborah M. and Hardman, Lynda. "Lost In Hyperspace": Cognitive Mapping and Navigation in

a Hypertext Environment, Chapter 7, Hypertext: Theory Into Practice, Edited by McAleese, Ray., Ablex Publishing Corporation, New Jersey, 1989.

10. Grant, Simon and Mayes, Terry. Cognitive Task Analysis ?, Chapter 6, Human−Computer Interaction and Complex Systems, Edited by Weir, George R.S., and Alty, James L., Academic Press, California, 1991.

11. Halasz, Frank. Reflections on NoteCards : Seven Issues for the Next Generation of Hypermedia Systems, Communications of the ACM, July 1988.

12. Hammond, Nick and Allinson, Lesley. The Travel Metaphor as Design Principle and Training Aid for Navigating Around Complex Systems, People and Computers III, 1987.

13. Koved, Larry & Shneiderman, Ben. Embedded Menus : Selecting Items in Context, Communications of the ACM, April 1986.

14. McAleese, Ray. Overview and Questions For Readers [on Hypertext], Chapter 1, Hypertext: Theory Into Practice, Edited by McAleese, Ray., Ablex Publishing Corporation, New Jersey, 1989.

15. Molich, Rolf, and Nielsen, Jakob. Improving a Human−Computer Dialogue, Communications of the ACM, March 1990.

16. Nielsen, Jakob. Hypertext/Hypermedia, Academic Press, 1990.

17. Nielsen, Jakob. The Art of Navigating through Hypertext, Communications of the ACM, March 1990.

18. Nielsen, Jakob. Traditional Dialogue Design Applied to Modern User Interfaces, CACM, October 1990.

19. Open Software Foundation. OSF/Motif Style Guide, Prentice Hall, 1991.

20. Rao, Usha & Turoff, Murray. Hypertext Functionality: A Theoretical Framework, International Journal of Human−Computer Interaction, 1990.

21. Shneiderman, Ben. Designing the User Interface, Addison−Wesley Publishing Company, Inc., 1987.

22. Smith, John B., Weiss, Stephen F., and Ferguson, Gordon J. A Hypertext Writing Environment and Its Cognitive Basis, Proceedings of Hypertext ’87, ACM Press, 1987.

23. Thuring, Manfred, Haake, Jorg M., and Hannemann, Jorg. What’s Eliza Doing in the Chinese room ? Incoherent Hyperdocuments and How to Avoid Them, Proceedings of Hypertext ’91, ACM Press, 1991.

24. Trigg, Randall. Guided Tours and Tabletops : Tools for Communicating in a Hypertext Environment, ACM Transactions on Office Information Systems, October 1988.

25. Trigg, Randall & Weiser, M. TEXTNET: A Network Based Approach to Text Handling, ACM Transactions on Office Information Systems, January 1986.

26. Turoff, Murray, Rao, Usha, and Hiltz, Starr Roxanne. Collaborative Hypertext in Computer Mediated Communications, Proceedings of the Twenty Fourth Annual Hawaii International Conference on System Sciences, January 1991.

27. Turoff, Murray. How to do it, Lecture Notes for Design of Interactive Systems (unpublished), NJIT, 1991.

28. Utting, Kenneth & Yankelovich, Nicole. Context and Orientation in Hypermedia Networks, ACM Transactions on Information Systems, January 1989.

29. Wright, Patricia. Cognitive Overheads and Prostheses: Some Issues in Evaluating Hypertexts, Proceedings of Hypertext ’91, ACM Press, 1991.

30. Zellweger, Polle T. Scripted Documents : A Hypermedia Path Mechanism, Proceedings of Hypertext ’89 Conference, November 1989.

[*] This chapter is based on a paper submitted to Workshop on Information Technologies and Systems ’93. An abstract of this chapter has also been accepted as a poster for Hypertext ’93.