Comparing User and Software Information Structures for Compatibility

Thomas Plocher

Honeywell Laboratories
3660 Technology Drive,
Minneapolis, Minnesota 55418, USA
Email: tom.plocher@honeywell.com

Torkil Clemmensen

Department of Informatics,
Copenhagen Business School,
Howitzvej 60, 2.10
DK - 2000 Frederiksberg C
E-mail: tc.inf@cbs.dk

Abstract. Eastern and Western cultures differ quite systematically in how they group objects, functions and concepts into categories [1,2,3]. This has implications for how navigation features, such as menus, links, directories, should be designed in software applications. This is particularly of interest when the application is developed in one culture for use in a second culture. This paper presents this problem and discusses some approaches to comparing user and software information architectures both visually and quantitatively.

Keywords: culture, cognition, information architecture, usability, visualization

1 Introduction

Eastern and Western cultures differ quite systematically in how they group objects, functions and concepts into categories [1,2] and [3]. Earlier, Choong [4] also found some evidence for these cultural differences in category formation when people used an online shopping system. Eastern cultures tend to place objects together that share relationships. Western cultures group objects together based on similar object attributes. As a simple example, assume that we show Eastern and Western people three pictures: a cow, a dog, and a patch of grass, and ask them to group the pictures. Western people will tend to place the cow and dog in the same category because from their physical attributes, they are both animals. Eastern people will tend to group the cow and the grass together because of the relationship, "cows eat grass".

So what does this mean for user interfaces? Categories form the basis for the information architectures underlying user interfaces to software systems. The content and organization of menus in Windows™ interfaces, links in websites, and file directories in most software applications are based on categories. But who's categories? Most likely those of the software designer, and reflecting the particular biases in category formation associated with her culture or personal cognitive style.

What happens when a user from one culture, say, the United States, attempts to use a software application that is based on categories, e.g. an information architecture, developed in another culture, such as India? The result can be poor usability, as the user attempts to locate functions in menus or information organized in an unfamiliar way by an original designer or engineer thinking in a different style.

2 Visualizing and Comparing User and Software Information Structures

A first step in understanding how the user interface is affected by these cultural differences in category formation is to have some useful methods to make these comparisons and describe the similarities and differences. Initially, it would be of great value to the designer and researcher to have tools for visualizing graphically the information structures of user and software and comparing them in a qualitative manner. Eventually, more quantitative methods of comparison can be developed. The current paper reviews some techniques for visualizing information hierarchies and how these might be applied to the problem of comparing user and software information architectures. We also discuss some ideas for metrics that would allow the usability expert to describe those differences more quantitatively. Such metrics can be used to evaluate usability and re-design an information architecture for a software application. They also can be used to design information architectures for cross-cultural research that deliberately and quantifiably vary from some cultural preference. This is how these metrics will be used on the CULTUSAB project [5]. In the future, one can envision how such metrics could also be integral to software applications that can adapt their information architecture and resulting menus, links, etc., automatically, to better match the preferences and cognitive style of the user.

2.1 Desired Functions

In this context, what does it mean to "compare" two information architectures, the user's versus the application's? First of all, we assume that the information architecture underlying a software application and user interface can be described as a branching hierarchy of main categories and subcategories, X levels deep. Assume also that the hierarchy can be derived by analysis for a given software application from its menus, links, or directory structure. A user's preferred hierarchy can be derived by having him perform a "card sort" exercise (6). Each object or function is written on a card and the user's task is to organize them into categories. Once the hierarchies are documented, tools can be applied to compare them.

At a minimum we would like to have a tool that allows us to:

1. Compare the gross or overall form of one structure to another. It would be useful to display two structures, side by side, and look at them from a bird's eye view. Then rotate them in X, Y, Z to identify gross similarities and differences in number of nodes at each level of the hierarchy (width) and number of levels (depth). From this bird's-eye view we could identify branches of a hierarchy that stand out by being being extremely simple or extremely complex. Further, it would be useful to be able to zoom in on the dissimilar portions of the structures and then make more detailed comparisons, looking at the content or meaning of individual nodes.
2. Systematically compare the nodes and branches of one structure to another. The designer would like to compare mother nodes at each level of the hierarchy and look in detail at the children of each specific mother node. The ability to zoom in on a single node and read its description would be valuable.
3. Cross reference between structures. It would be useful to select a node in one structure and instantly be connected to its location in the other structure by means of a line connecting the two or highlighting the two nodes.
4. Estimate the degree of difference between the two structures. Ideally a tool for comparing two information structures would compute some quantitative measure of similarity or difference between the two.

2.2 Available Visualization Tools

Andrews [7] provides a survey of tools for visualizing information hierarchies. One of the classical approaches to this problem is that of Cone Trees [8] and their more recent and enhanced derivatives, Reconfigurable Disc Trees [9]. Cone Trees provide a three-dimensional representation of an information hierarchy. Each node in the hierarchy is represented as the apex of a cone. The "children" or nodes underneath the apical node are arranged in a three-dimensional circle. The height of each cone is scaled so that the entire hierarchy fits on the screen. Each cone is slightly shaded to make it distinct but without occluding other cones in the tree. The tree can be rotated by the user to explore its structure. Also, clicking on a node causes it and all the nodes in its path to rotate to the front and center of the tree in animation. The text describing each node is also displayed when the node has been rotated to the front. Cone Trees are searchable, with the nodes identified by the search highlighted in color.

Unlike textual two-dimensional depictions of information hierarchies, the three-dimensional Cone Tree allows a rather large hierarchy to be displayed and viewed on a screen without scrolling. Cone Trees can accomodate up to 1,000 nodes on a single screen but, beyond that, begin to suffer from visual clutter. To solve this problem and accomodate larger hierarchies, Jeong and Pang [9] developed Reconfigurable Disc Trees (RDTs). An RDT uses discs rather than cones to represent node and children relationships. The children are arrayed around a circle rather than a cone. This improves the ease of visual projection on the screen, reducing occusion between elements of the tree and increasing the number of nodes that can be displayed on a single screen. Figures 1 and 2 show examples of Cone Trees and RDTs.

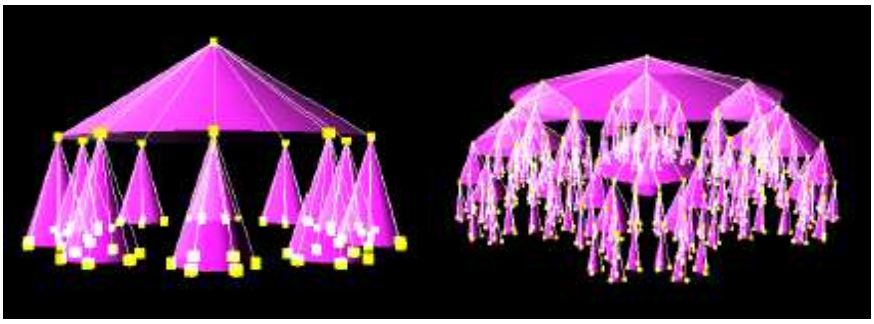

Fig. 1. Cone Trees for simple and complex (516 nodes) information hierarchies. From Jeong and Pang [9].

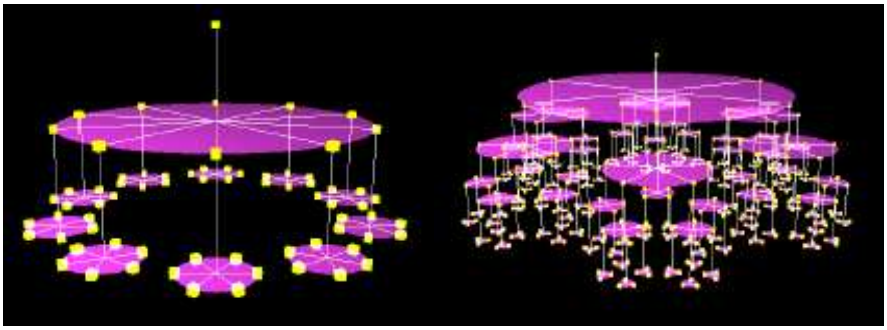

Fig. 2. Reconfigurable Disc Trees for simple and complex (516 nodes) information hierarchies. From Jeong and Pang [9].

Cone Trees and RDTs provide useful tools that can be used in their current form for analyzing user and software information architectures. However, these tools were intended to support analysis of a single information hierarchy, not the systematic comparison of two hierarchies. To support the latter task, these tools need to be modified in several ways. First, the two hierarchies being compared should be displayed side-by-side, on the same screen. Second, it would be highly desireable if Cone Trees and RDTs provided the ability to map a node in one structure to the node with the same content in the second structure. This is analogous to the problem of mapping one database schema or ontology to another. A number of tools are available for that, most notably Microsoft's BizTalk Schema Mapper and its recent enhancements described by Robertson, Czerwinski and Churchill [10]. While the latter clearly map one schema hierarchy to another, node by node, the representation is still two-dimensional and largely textual. The 3D graphical visualization that makes Cone Trees and RDTs so useful for our purposes and so efficient in the use of screen real estate, are missing. These capabilities need to be combined.

3. What Could We Measure?

The visualization tools reviewed by Andrews were intended for viewing a single information hierarchy, not for comparing two such structures. So it is not surprising that none of these hierarchy visualization tools is currently capable of computing measures of dissimilarity between two structures. The ability to do so is critical to this line of research on the CULTUSAB project so that we can systematically manipulate similarity of information structures as an independent variable in our experiments. We are contemplating various ways of measuring similarity.

1. We can simply count the number of nodes at each level of each structure and compute the difference in number of nodes at each level.
2. We can evaluate the cross-reference maps of nodes in one structure to the same nodes in the second structure and compute the differences in where the same nodes are placed within each hierarchy. We could measure differences in depth of placement down through the levels of the hierarchies. For example an item placed at Level 1 in one hierarchy and Level 3 in the second hierarchy would have a difference score of 2. These placement difference scores could be summed or combined in some way to produce a depth of placement score for the entire structure. Structures can also differ in where nodes are placed laterally. That is, nodes can be placed under different high level mother nodes and could be scored accordingly. Dissimilarity of placement of a node could be comprised of both the depth difference score and the lateral difference score.

Of course, such measures are subject to validation with experimental data. A validation experiment is currently in progress.

References

1. Nisbett, R. E., Peng, K., Choi, I., & Norenzayan, A. Culture and systems of thought: Holistic vs. analytic cognition. *Psychological Review*. 108 (2001) 291-310.
2. Nisbett, R.E. *The Geography of Thought: Why We Think the Way We Do*. NY: The Free Press (2003).
3. Choi, I. and Nisbett, R.E. Culture, categorization, and inductive reasoning. *Cognition*. 65 (1997) 15-32.
4. Choong, Y-Y. *Design of computer interfaces for the Chinese population*. PhD Dissertation. Purdue University. (1996).
5. Clemmensen, T. and Plocher, T. The Cultural Usability Project (CULTUSAB): Studies of Cultural Models in Psychological Usability Evaluation Methods. HCI International Conference, Beijing, China. (2007).
6. Dong, J., Martin, S., and Waldo, P. A user input and analysis tool for information architecture. CHI '01 Conference on Human Factors in Computing Systems. (2001) 22-24.

7. Andrews, K. Information Visualization. Tutorial Notes, IICM, Graz University of Technology. (2002) 10-21. http://www.iicm.edu/ivis/
8. Robertson, G.G., Mackinlay, J.D.,and Card, S.K. Cone trees: Animated 3D visualizations of hierarchical information. In: *Proceedings of the ACM SIGSCHI conference on Human Factors in Computing Systems*. New York: ACM Press, (1991) 1890-194.
9. Jeong, C. and Pang, A. Reconfigurable Disc Trees for visualizing large hierarchical information spaces. In: *Proceedings of Information Visualization '98*. (1998) 19-25.
10. Robertson, G.G., Czerwinski, M.P., and Churchill, J.E. Visualization of mappings between schemas. In: *Proceedings of CHI 2005*, Portland Oregon. (2005) 431-439.