METAPHOR IN USER INTERFACE DESIGN: A VIEW FROM THE TRENCHES

William Hudson
Syntagm Ltd
10 Oxford Road
Abingdon, UK, OX14 2DS
william.hudson@syntagm.co.uk

ABSTRACT

The application of metaphor to user interface design is poorly understood and with only a few exceptions, badly executed. Yet metaphor has great potential to help users acquire appropriate mental models of interactive systems. In this paper I examine some of the issues that plague metaphoric user interfaces and suggest solutions that could lead to greater rates of success.

Keywords

Metaphor; user-interface design; mental models; structure-mapping theory

1. INTRODUCTION

The popularity of metaphor in user interface design largely coincided with the development of graphical user interfaces, most notably the desktop metaphor from Xerox PARC [1]. Not surprisingly, this lead to the belief that user interface metaphor was primarily visual in nature. However, the greatest benefit to be gained from metaphor in UI design is from conceptual *structure*, not superficial appearance. It is through similarities in structure – the constituent entities and relationships between them – that we are able to call into use mental models acquired by users in other problem domains.

2. MENTAL MODELS AND ANALOGY

Phillip Johnson-Laird proposed that mental models were fundamental to the way that we think and solve problems [2]. He made the important observation that "their structure is analogous to the structure of the situation that they represent". Metaphor and analogy make use of existing mental models. While different theories of metaphor and analogy abound, Dedre Gentner's structure-mapping theory is one of the most compelling for UI design purposes [3]. In structure mapping, Gentner suggests it is the relationship between entities that carries useful information from the source domain to the target, not similarities between the entities themselves. In her example of Rutherford's analogy between the hydrogen atom and the solar system, it is the relationship between the planets and the sun that provides the mental model useful in the target domain of the hydrogen atom. The physical attributes of the entities are of no concern.

Naturally, we are interested in the surface aspects of a metaphor when it comes to designing a GUI, but this should always take a secondary role to that of structure. In particular, metaphors that offer attractive graphics at the expense of a useful structure or efficient interface are counterproductive, as shown in the following example.

3. STRUCTURE NOT SURFACE

Lotus Organizer (originally developed by British company Threadz) was one of the first popular examples of visual metaphor on the Microsoft Windows platform, with Version 1 shipping in 1991. It had the appearance of a bound personal organizer, as shown in Figure 1.

While the application is superficially very attractive, it is not a particularly helpful use of metaphor:

- the rings shown in the centre of the binder are a poor use of valuable screen space;
- pages like the year planner shown in the figure have to be manually opened by clicking on an "unfold" icon (shown towards the bottom right corner);
- special tools and an unusual process are required to move entries from one page to another;
- the section tabs change side according to where the user is in the organizer.

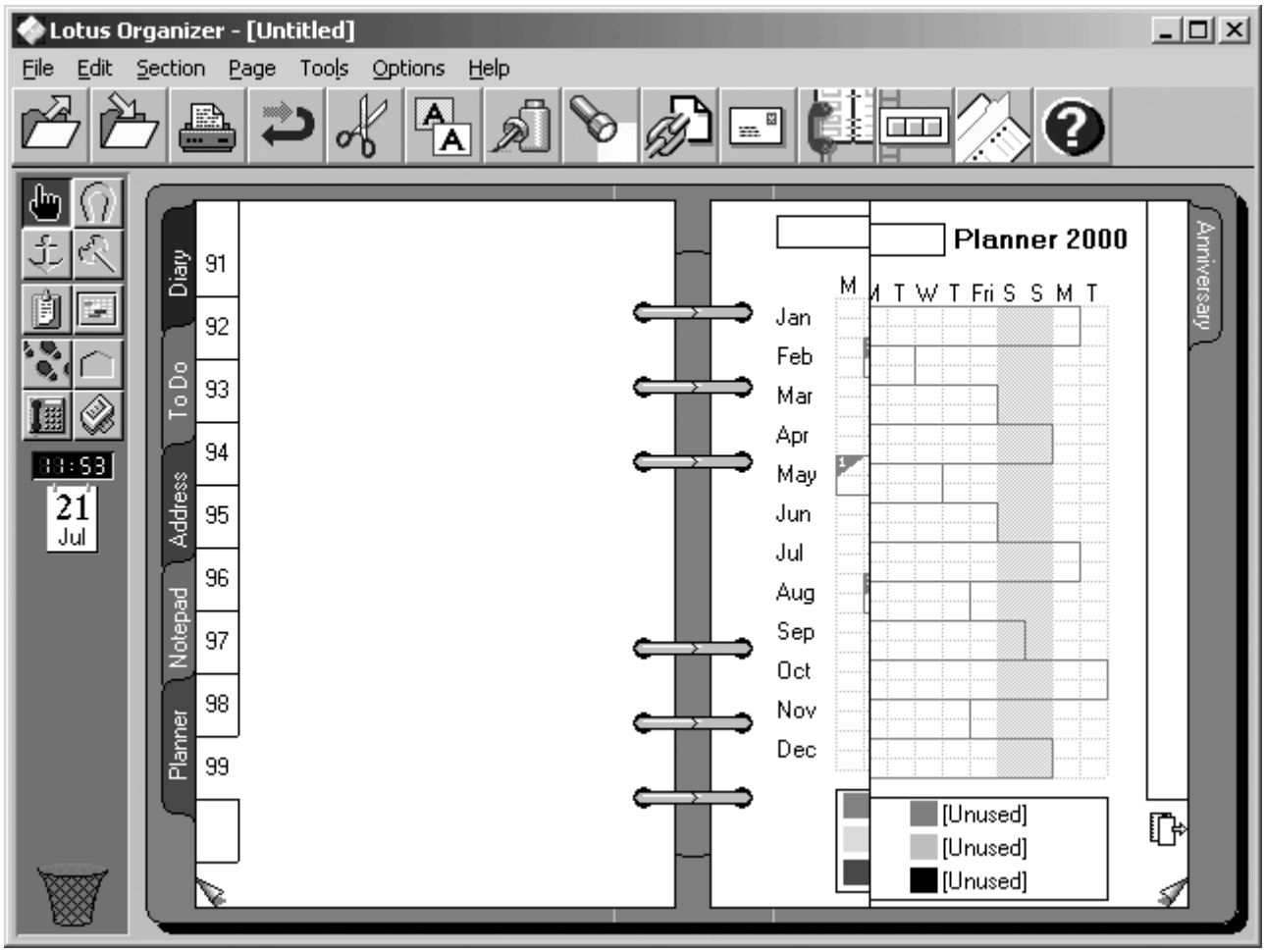

Figure 1, Lotus/Threadz Organizer V1.0

From a structural viewpoint, the most useful aspect of the metaphor is the use of tabs and the knowledge that physical organizers usually contained address, diary and planner sections. These modest benefits do not balance the cost of the metaphor in user interface terms.

4. LITERAL LIMITATIONS

Most of the unhelpful aspects of the Organizer example have their origin in an over-literal implementation of metaphor. If our overriding concern is to produce user interfaces that are "transparent" in the sense of allowing users to focus on the problem domain rather than the interface itself, literal analogies must be avoided. The behaviour of the tabs in Organizer is a case in point. In a physical organizer, the tabs move from one side of the binder to the other because they have to. It adds nothing to users' understanding of the interface in software since the sense of location is largely irrelevant: users simply want to move to the section of interest. This need not be more or less convenient according to the section that is currently open – the tabs could be shown down both sides with no further loss of screen space (careful inspection of Figure 1 will demonstrate this).

In general, the benefits of metaphor should be in providing unifying and familiar concepts. Visual clues of the metaphor being used can be helpful, but not at the cost of interface transparency. In particular, a Goals-Operations-Methods-and-Selections (GOMS) analysis of a user interface design should be no more complex when metaphor is introduced (for the same functionality).

5. METAPHOR MUST SCALE

The application of metaphor can be over-literal in concept as well as presentation. This frequently leads to designs that work well in simple cases – such as those found in the physical world – but do not scale to the more complex situations usually prevalent in software applications. A common example appears in the Microsoft Windows desktop. In the physical world, the desktop offers no means of finding documents quickly, other than a small number that might have been strategically placed by the user. For other documents, office workers are expected to know how to find what they need and to return it to its correct location

when it is no longer required. Yet on the metaphorical desktop life could be much easier. A card-file style index or timeline (similar to the LifeLine approach described in [4]) could automatically keep track of recent documents by type, contact, date and other criteria. Instead, with the exception of a crude recent-documents list, the desktop metaphor offers no automated assistance in common retrieval tasks. (The "find" or "search" functions are not automated in this sense since they require explicit requests.) Users must rely instead on the most-recently-used lists of their favourite applications, which is especially regrettable since the concept of a software application is contrary to the goal of interface transparency. To better deal with complex tasks, a more generous implementation of the desktop metaphor might have included "working folders" whose contents automatically returned to the correct location when the user released them. Another common task could have been addressed by the provision of a stapler on the desktop. It would allow documents to be bound together in a compressed form for storage or distribution (a more suitable metaphor than the "zip" file that has become commonplace on PC platforms and only recently integrated with the Windows desktop).

6. GRAND DESIGNS

A further misunderstanding of metaphors is the assumption that they must be virtual environments, such as the desktop. In truth, relatively few virtual environments exist, certainly for general application. The shopping basket metaphor is perhaps the best example of a successful virtual environment, but it is limited to a fairly narrow range of activities by comparison to the desktop, which attempts to address a plethora of office and computer-related activities. Future development is likely to focus on domain-specific metaphors and those that help users understand relatively isolated problems. For example, many ordinary users have little or no understanding of public key encryption. This is hardly surprising since no physical world security system involves multiple keys with no locks. A metaphor to explain public key encryption would consist of a public *lock* and a private key. Open locks for an email address would be freely distributed. When a message was secured with a lock (encrypted with a public key) it would then be in the closed state and impossible to open without the private key held by the recipient. This is all perfectly natural and well-understood from the use of padlocks and keys in the physical world. The sender's copy of the message would be associated with his or her own lock as the security mechanism, allowing users to understand the role played by their own keys.

7. UNIFYING THEMES

Apart from providing a suitable mental model, a metaphor can also be a source of a unifying theme – a powerful but undervalued tool in improving software usability. In many cases, software applications or web sites evolve to become complex collections of features with little or no attempt at simplification for the benefit of users. The rising popularity of "offset mortgages" provides a useful example. These are combined home loan and savings accounts that allow the borrowed amount, and the associated interest payable, to be reduced by the savings. While they can be quite effective in reducing interest payments for some borrowers, they have the unfortunate side-effect of conflating what might have been several separate accounts, for separate purposes, into one very large overdraft. So whereas customers might have had several current accounts for personal use, joint use, paying bills, household expenses plus a few savings accounts for holidays, further education or a new canal boat, they now have a single account with a single negative balance. Some financial intuitions have recognized this problem by identifying the savings amount separately while others have focussed more on allowing customers to set forecasts and measure monthly expenditure against that. Naturally, these approaches are reflected in the respective online banking facilities offered via the web.

The main difficulty is that the web sites for offset mortgage accounts attempt to solve each issue separately and in a fairly limited way. A more attractive approach would be to make use of a familiar financial metaphor that could also act as a unifying theme. Happily such a metaphor exists. Before the widespread adoption bank accounts, people stored their money in containers of various types: jars, tins, socks, boxes, etc. usually having one container for each purpose. If you wanted to save up for a holiday, you would get a new tin and write "holiday" on it. Each month you would move some money to that tin. This metaphor can easily be applied to offset mortgages. Users could be allowed to create their own tins, jars or socks for their own purposes, with facilities such as

- a monthly transfer amount to/from another container;
- a target or limit for notification (or comparison);
- an interest rate to be calculated on the contents

This solution provides almost limitless flexibility within a simple and metaphorical unifying theme.

8. CONCLUSIONS

Metaphor in user interface design has been a surprisingly controversial topic over the past 20 years or so. As with many other controversies, a contributing factor has been a lack of understanding over what constitutes a good UI metaphor and how it should be implemented. In this short paper I hope I have touched on most of the important issues:

- metaphors are primarily structural, not visual in nature with Gentner's structure-mapping theory providing the theoretical underpinning;

- designers must be careful not to adopt metaphor too literally, especially at the expense of interface transparency;
- virtual environments are not the only useful application of metaphor with many "smaller scale" opportunities waiting to be addressed;
- metaphors have the potential to provide unifying themes that can contribute substantially to simplification.

Perhaps the next 20 years will see a renaissance of effective UI metaphor.

9. REFERENCES

[1] Smith D. C., Irby C., Kimball R., Harslem E., The Star User Interface: An Overview, *Proceedings of the AFIPS 1982 National Computer Conference*, **50**, (1982), 515-528.

[2] Johnson-Laird P. N. (1983). *Mental Models: Towards a Cognitive Science of Language, Inference, and Consciousness*. Cambridge University Press.

[3] Gentner D. (1983). Structure-mapping: A theoretical framework for analogy. *Cognitive Science, 7*, pp 155-170.

[4] Plaisant C., Milash, B., Rose A., Widoff S. and Shneiderman B., LifeLines: visualizing personal histories, *Proceedings of the SIGCHI conference on Human factors in computing systems*, (1996), 221-ff.