Mick Tarel ©1997 Artville, LLC

Metaphor: A Double-Edged Sword

"Those who think that one should use metaphors in design are destined to produce crappy designs."

— Don Norman, in an interview with *WebWord* author John Rhodes.

Is the issue this simple? Web design consultant William Hudson doesn't think so. In this installment of "The Whiteboard," Hudson examines the ins and outs of metaphor as a design tool and suggests guidelines for using it wisely (and without letting it make your designs bad).

Don Norman is not one to steer clear of controversy. In *The Invisible Computer* he writes of his disapproval of metaphor in the design of user interfaces—an opinion that he has repeated in debate on the CHI-WEB e-mail list.

What's all the fuss about? The original issue in CHI-WEB was whether to use a shopping cart metaphor when designing e-commerce sites. Some contributors pointed with glee to examples of shopping sleds and wheelbarrows, while others insisted that a cart is a cart is a cart. Don said that we shouldn't be using any of these things and should just have "lists of items purchased." Naturally, this left quite an air of confusion over the whole subject. To try to iron things out, metaphorically speaking, we need to have a better understanding of some of the issues.

Metaphor attempts to express a new domain (the target) in terms of one that is already understood (the source). This is quite common in language when we discuss *negotiation* as if it were war ("he stood his ground"), use the term *higher* to mean *more*, and view life as a journey (as in "where do you want to be in 2 years?").

WHITEBOARD
COLUMN EDITOR
Elizabeth Buie
Senior Principal Engineer
Computer Sciences
Corporation
15245 Shady Grove Road
Rockville, MD 20850
+1-301-921-3326
fax: +1-301-921-2069
ebuie@csc.com

Some psychologists even argue that thought and language are fundamentally metaphoric. Unfortunately this model of metaphor is complex, requires a good understanding of both the source and target domains, and is specific within culture and language. A more suitable approach to user interface design is the *structure mapping theory of analogy*. The work of Dedre Gentner and her colleagues, this theory still relates target and source domains, but by considering the relationships between the conceptual objects (concepts) within each.

The Controversy

What are the objections to using metaphor in user interface design? The most commonly quoted are that metaphor is helpful only to inexperienced users and that it is overly restrictive. The "desktop" metaphor, which is the basis of most graphical environments found today, is often cited as an example of the failure of metaphorical design. However, I think it deserves a closer look. Figure 1 shows a conceptual model of the source domain: an office of the mid 1970s. The rectangles represent concepts, and the connecting lines represent relationships. I drew the model in the style of a unified modeling language (UML)-class model. In the model, concepts and relationships are read from the labeled end of each line. So, for example, *in-tray receives document and filing cabinet stores folder*. This model (with some minor changes, such as *printer* instead of *copier*) was the basis of the desktop metaphor used on the Xerox Star, the inspiration for the Apple Macintosh, Microsoft Windows, and other graphical environments. The trays, folders, documents, and other concepts were represented with clear icons and descriptive text, allowing inexperienced computer users to rapidly learn the use of a complex and innovative computer system.

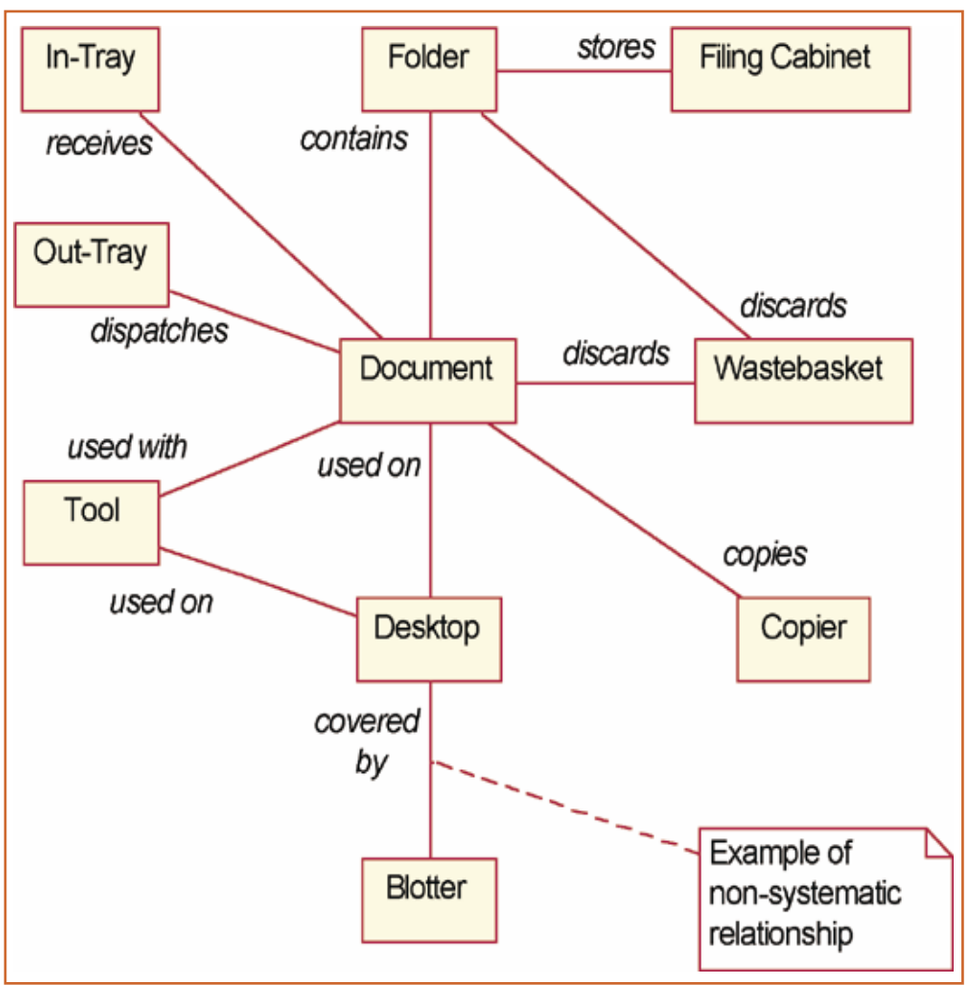

Figure 1. Conceptual model of a mid-1970s office.

Notice, by the way, that Figure 1 includes a relationship that isn't of the same type as the others: *desktop covered by blotter*. Although this relationship was true of many desktops at the time, most of the other relationships are concerned with actions. The relationship between desktop and blotter is superficial and not part of the system of relationships that interest us. This is called a nonsystematic relationship.

But why is the desktop metaphor unhelpful? I think the desktop metaphor on our computer screens has drifted too far from the original

office domain. Microsoft Windows, for example, hides the in-tray and out-tray in an e-mail application, variously called Mail, Inbox, Outlook, or Outlook Express depending on the age and configuration of the system. The *desktop* is confusingly covered by *wallpaper*, and printing documents by dragging them to a printer does not work reliably. The Macintosh does not fare much better: it has the odd feature of allowing users to drag the floppy disk to the wastebasket ("Trash") to eject the floppy. From the office domain it's obvious that the effect of that action (if any) should be the same as discarding a document or folder. So in most cases, the apparent problem with the desktop metaphor is that it does not correspond to the original office domain.

Now we turn to the suggestion that metaphors confine their designers and users. This supposes that by establishing a well-understood system of relationships we are limited to *just* those relationships. Although it is likely that inexperienced users will understand the metaphor literally, this same understanding will give them the confidence to explore and experiment. As long as we do not introduce anomalous relationships such as *wastebasket ejects floppy*, there will still be considerable freedom to innovate. For example, within the desktop metaphor, we could make use of a stapler to group and compress documents (compared with the now common zipper concept that is not part of the office domain; see **Error! Reference source not found.**). We can also use *bridging* concepts to move from one metaphor to another. A screwdriver (Figure 3) could be used as a bridge from the desktop metaphor to the physical hardware. More advanced users might be able to drag the screwdriver to the filing cabinet to perform maintenance operations such as modifying partitions or defragmenting storage. All these metaphorical design features could be used in addition to more direct (but less predictable) mechanisms such as popup menus and shortcut keys.

Some Problems

Having defended the general concept of metaphor, let's be honest about its limitations. The first problem is that designers sometimes use metaphor too literally. For example, in a real office, mailing a document to a customer means that you no longer have a copy of it, unless you have explicitly made a copy beforehand. Enforcing this in a virtual office would be incredibly frustrating to users. Mindless consistency is not an attractive design strategy.

A larger difficulty is that there aren't many useful metaphors for completely new problem domains; the most successful metaphors "virtualize" an existing problem domain. This means adopting not only the concepts and relationships of the domain, but also the activities, as shown for a "virtual store" in Figure 4. Other candidates for virtualization are libraries, clubs, medical services, and automotive sales and service—in short, most things that have well-defined domains.

Figure 2. Which has a more predictable effect on documents?

The Name's the Thing

The debate over terms *shopping sled* versus *shopping cart* may seem trivial at first sight. It is easy to forget that to inexperienced Web shoppers, a shopping cart may be a familiar comfort. Arriving at an e-commerce site that does

Figure 3. Bridging concept from desktop to hardware.

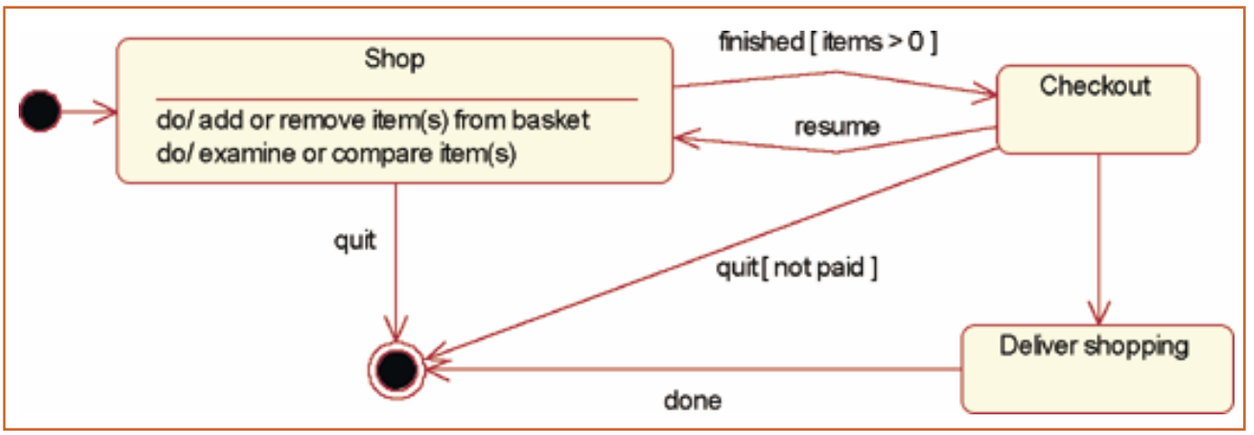

Figure 4. Activity (state) diagram for a virtual store.

not use the shopping cart metaphor may be a learning experience, rather than a shopping one. It could also be a wasted learning experience if most other Web sites use different concepts and activities. The novelty will be beneficial only if it offers a real improvement over familiar solutions and is adopted by the Internet community as a whole.

An Iceberg Model of Metaphor

Many concepts are domain specific. For example, a cash register is almost always associated with shopping and the activity of paying. Other concepts, such as money, are much more general. The image of money in an e-commerce site could be associated with paying, but it might also mean pricing, discounts, or currency. The difference is that using a domain-specific concept such as a cash register invokes the system of hidden relationships to which it belongs. The *cash register* may be the only visible evidence of the metaphor, but the rest of the shopping domain lurks beneath the surface.

This "iceberg" model of metaphor can present problems as well as providing solutions. A number of Web sites have begun using the term *check in*. Some use it as an activity required of visitors to the Web site (i.e., registering or joining), and others ask existing members to *check in*. Which is correct? The term is used almost exclusively in the travel industry to mean registration for a prearranged journey or stay and so is associated with a reservation, as shown in Figure 5. The question of whether the customer is a visitor or a member does not really arise, except that a member might be tenuously considered to have a reservation. It would be best not to use the term at all outside its travel context.

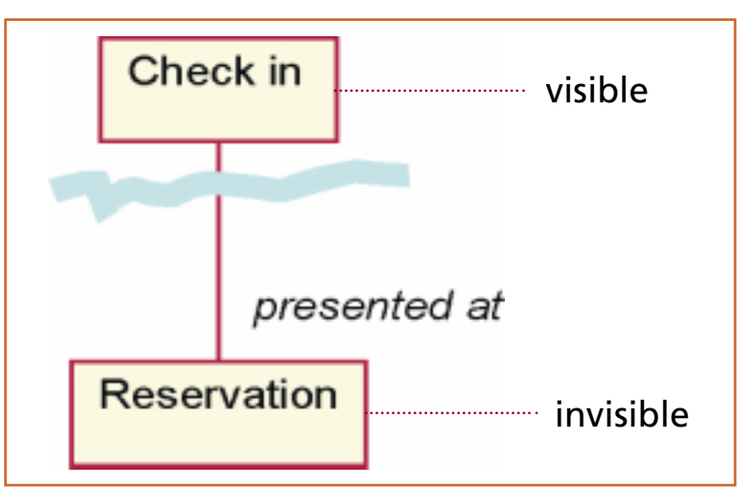

Figure 5. Check in has a reservation hidden below the surface.

Metaphoric Guidelines

My own view is that metaphor is an important tool for user interface design but must be used with care. The following guidelines cover the most important issues.

- **Metaphors operate on systems of relationships, not on individual concepts.** Make sure that the system of relationships is reflected in your user interface, and do not use concepts out of context.
- **Metaphors do not have to be complete, but interfaces need to provide adequate clues to users.** Omitting less

important concepts or changing them slightly may not significantly affect usability, but only testing will tell.

- **Metaphors should not rely on mere appearance.** A concept should not just have the appearance of a shopping basket; it should behave like one too. It needs to have the same or similar relationships in the target domain as it did in the source.
- **Avoid nonsystematic relationships** (such as *desktop covered by blotter* in Figure 1). Most of the important relationships will include how concepts interact with each other, particularly from a user's perspective.
- **Don't let abstract relationships interfere with the metaphor.** For instance, it may be true that a desk and a filing cabinet are both instances of office furniture. However, it's their purpose that is of interest.
- **Choose metaphors that provide concrete images.** The shopping domain is useful because it contains a number of domain-specific concepts that can easily be represented visually: shopping cart (or basket, or trolley), cash register, price tags, discount signs. A catalog metaphor, by comparison, is relatively devoid of distinct concrete images (an order form is not as immediately recognizable as a cart or basket).
- **Try to get as close to the original domain as possible (unless an alternative has obvious advantages).** Mail-order shopping and vending machines are already one step removed from the source shopping domain. Also, some aspects of the vending machine domain are not entirely beneficial to e-commerce: Vending machines usually sell low-value items, most require payment in advance, and not all are reliable.
- **Beware culture-specific metaphors and concepts.** Cart (US) versus trolley (UK) is a minor issue since the image is the same. However, kiss and ride (dropping off a loved one at a public transport facility) does not enjoy widespread recognition.

Further Reading

Gentner, D. and Markman, A. B. Structure mapping in analogy and similarity. *American Psychologist 52* (1997), pp. 45-56.

Ortony, A., ed. *Metaphor and Thought.* Cambridge University Press, Cambridge, United Kingdom, 1993.

Rhodes, J. Thinking beyond usability: An interview with Don Norman, **http://webword.com/interviews/norman.html**, 1999.

The CHI-WEB mailing list is archived at **http://www.acm.org/archives/chi-web.html**. To subscribe to CHI-WEB, complete the form at **http://www.acm.org/sigchi/listserv/request.html**.

William Hudson is a freelance software designer based near Oxford in the United Kingdom. Since he began writing interactive software in the early 1970s, Hudson has gained experience in computing topics ranging from firmware to desktop applications. For the past 10 years he has focused on user interface design, object-oriented design, and human-computer interaction. When Hudson's head is not stuck inside a conceptual model, it is firmly wedged in a mountain bike helmet or diver's mask.
E-mail: WHudson@syntagm.co.uk.