2.4 Interface Metaphors

As mentioned earlier, metaphors are considered to be a central component of a conceptual model. They provide a structure that is similar in some way to aspects of a familiar entity (or entities) but also have their own behaviors and properties. More specifically, an interface metaphor is one that is instantiated in some way as part of the user interface: for example, the desktop metaphor. Another well known one is the search engine. This term was originally coined in the early 1990s to refer to a software tool that indexed and retrieved files remotely from the Internet, using various algorithms to match terms selected by the user. The metaphor invites comparisons between a mechanical engine, which has several parts working, and the everyday action of looking in different places to find something. The functions supported by a search engine also include other features besides those belonging to an engine that searches, such as listing and prioritizing the results of a search. It also does these actions in quite different ways from how a mechanical engine works or how a human being might search a library for books on a given topic. The similarities implied by the use of the term search engine, therefore, are at a general level. They are meant to conjure up the essence of the process of finding relevant information, enabling the user to link these to less familiar aspects of the functionality provided.

ACTIVITY 2.2

Go to a few online stores and see how the interface has been designed to enable the customer to order and pay for an item. How many use the 'add to shopping cart/trolley/basket' followed by the 'checkout' metaphor? Does this make it straightforward and intuitive to make a purchase?

Comment

Making a purchase online is an undertaking with risks and people want to feel they are making the right choice. Designing the interface to have a familiar metaphor (with an icon of a shopping cart/basket – although not a cash till!) makes it easier for people to know what to do at the different stages of making a purchase. Importantly, placing an item in the basket does not commit the customer to purchase it there and then. It also enables them to browse further and select other items – as they might in a physical store. ■

Interface metaphors are intended to provide familiar entities that enable people to readily understand the underlying conceptual model and know what to do at an interface. However, they can also contravene people's expectations about how things should be, such as the recycle bin (trashcan) that used to sit on the desktop. Logically and culturally (i.e. in the real world) it should have been placed under the desk. But users would not have been able to see it because it would be occluded by the desktop surface. So it needed to go on the desktop. Some users find this irksome but most did not find it to be a problem. Once they understood why the bin icon was on the desktop they simply accepted it being there.

Other times, designers can fall into the trap of trying to create a virtual object to resemble a familiar physical object that is itself badly designed. A well-known example is the virtual calculator, which is designed to look and behave like a physical calculator. The interface of some physical calculators, however, have been poorly designed in the first place, based on poor conceptual models, with excessive use of modes, poor labeling of functions, and difficult-to-manipulate key sequences (Mullet and Sano, 1995). The design of the calculator in Figure 2.5(a) has even gone as far as replicating functions needing shift keys (e.g. deg, oct, and hex), which could have been redesigned as dedicated software buttons. Trying to use a virtual calculator that has been designed to emulate a poorly designed physical calculator is much harder than using the physical device itself. A better approach is to think about the kinds of calculations people typically want to do when using their phones or computers. For example, the Mac calculator in Figure 2.5(b) has a much simpler interface that supports a basic set of calculations, which is all that most people need to do (although how many people use the top four memory keys?).

In many cases, new interface metaphors rapidly become integrated into common parlance, as witnessed by the way people talk about them. People surf the net, poke their friends, and leave messages on their wall in the same way they would talk about winning an argument or saving time. As such, the interface metaphors are no longer talked about as familiar terms to describe less familiar computer-based actions; they have become everyday terms in their own right. Moreover, it is hard not to use metaphorical terms when talking about technology use, as they have become so ingrained in the language we use to express ourselves. Just ask yourself or someone else to describe how the Internet works. Then try doing it without using a single metaphor.

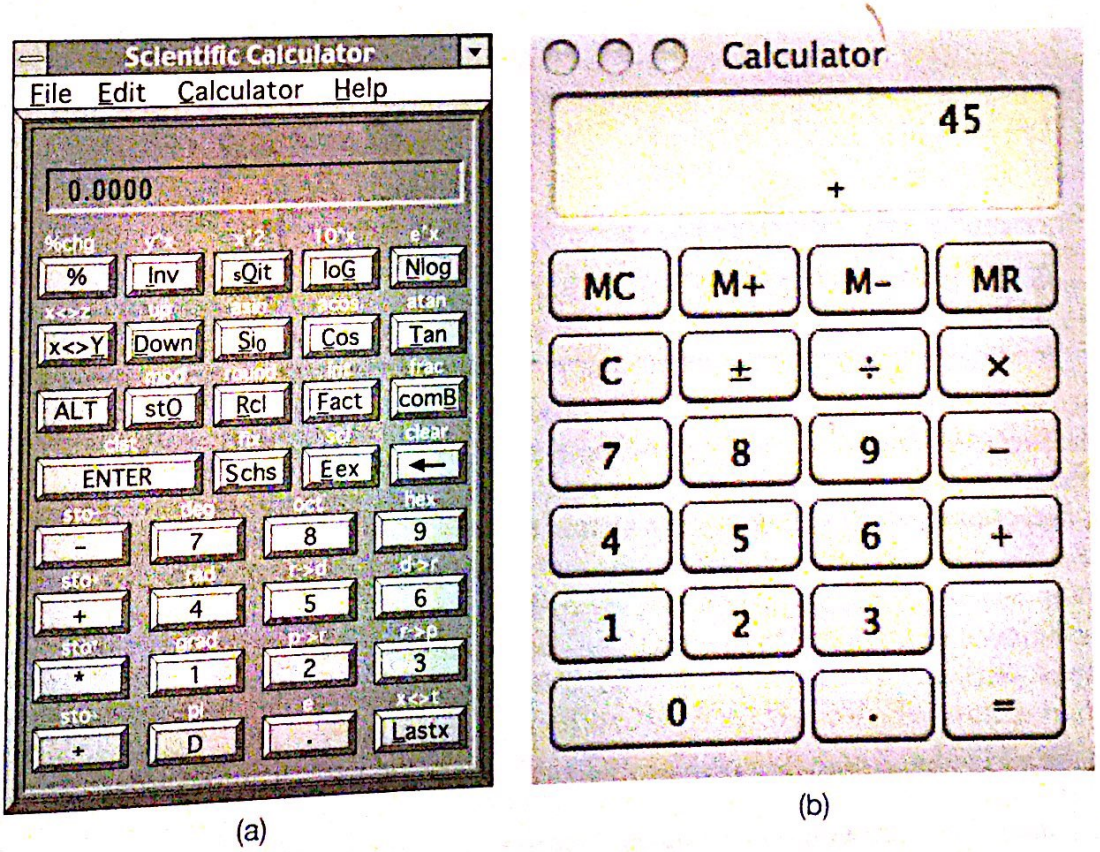


Figure 2.5 Two digital calculators where (a) has been designed too literally and (b) more appropriately for a computer screen

BOX 2.4

Why are metaphors so popular?

People frequently use metaphors and analogies (here we use the terms interchangeably) as a source of inspiration for understanding and explaining to others what they are doing, or trying to do, in terms that are familiar to them. They are an integral part of human language (Lakoff and Johnson, 1980). Metaphors are commonly used to explain something that is unfamiliar or hard to grasp by way of comparison with something that is familiar and easy to grasp. For example, they are commonly employed in education, where teachers use them to introduce something new to students by comparing the new material with something they already understand. An example is the comparison of human evolution with a game. We are all familiar with the properties of a game: there are rules, each player has a goal to win (or lose), there are heuristics to deal with situations where there are no rules, there is the propensity to cheat when the other players are not looking, and so on. By conjuring up these properties, the analogy helps us begin to understand the more difficult concept of evolution – how it happens, what rules govern it, who cheats, and so on.

It is not surprising, therefore, to see how widely metaphors have been used in interaction design to conceptualize abstract, hard to imagine, and difficult to articulate computer-based concepts and interactions in more concrete and familiar terms and as graphical visualizations at the interface. Metaphors and analogies are used in three main ways:

1. As a way of conceptualizing what we are doing (e.g. surfing the web).
2. As a conceptual model instantiated at the interface (e.g. the desktop metaphor).
3. As a way of visualizing an operation (e.g. an icon of a shopping cart into which we place items we wish to purchase on an online shopping site). ■

2.5 Interaction Types

Another way of conceptualizing the design space is in terms of the *interaction types* that will underlie the user experience. Essentially, these are the ways a person interacts with a product or application. We propose that there are four main types: instructing, conversing, manipulating, and exploring. Deciding upon which of these to use, and why, can help designers formulate a conceptual model before committing to a particular interface in which to implement them, e.g. speech-based, gesture-based, touch-based, menu-based, and so on. Note that we are distinguishing here between interaction types (which we discuss in this section) and interface types (which will be discussed in Chapter 6). While cost and other product constraints will often dictate which interface style can be used for a given application, considering the interaction type that will best support a user experience can highlight the potential trade-offs, dilemmas, and pros and cons.

Consider the following problem description: a company has been asked to design a computer-based system that will encourage autistic children to communicate and express themselves better. What type of interaction would be appropriate to use at the interface for

this particular user group? It is known that autistic children find it difficult to express what they are feeling or thinking through talking and are more expressive when using their bodies and limbs. Clearly an interaction style based on talking would not be effective but one that involves the children interacting with a system by moving in a physical and/or digital space would seem a more promising starting point.

Below we describe in more detail each of the four types of interaction. It should be noted that they are not meant to be mutually exclusive (e.g. someone can interact with a system based on different kinds of activities); nor are they meant to be definitive.

1. Instructing – where users issue instructions to a system. This can be done in a number of ways, including: typing in commands, selecting options from menus in a windows environment or on a multitouch screen, speaking aloud commands, gesturing, pressing buttons, or using a combination of function keys.
2. Conversing – where users have a dialog with a system. Users can speak via an interface or type in questions to which the system replies via text or speech output.
3. Manipulating – where users interact with objects in a virtual or physical space by manipulating them (e.g. opening, holding, closing, placing). Users can hone their familiar knowledge of how to interact with objects.
4. Exploring – where users move through a virtual environment or a physical space. Virtual environments include 3D worlds, and augmented and virtual reality systems. They enable users to hone their familiar knowledge of physically moving around. Physical spaces that use sensor-based technologies include smart rooms and ambient environments, also enabling people to capitalize on familiarity.

Besides these core activities of instructing, conversing, manipulating, and exploring, it is possible to describe the specific domain and context-based activities users engage in, such as learning, working, socializing, playing, browsing, writing, problem-solving, decision-making, and information-searching – to name but a few. McCullough (2004) suggests describing them as situated activities, organized by: work (e.g. presenting to groups), home (e.g. resting), in town (e.g. eating), and on the road (e.g. walking). The rationale is to help designers be less ad hoc and more systematic when thinking about the usability of technology-modified places in the environment. Below we illustrate in more detail our four core interaction types and how to design applications for them.

2.5.1 Instructing

This type of interaction describes how users carry out their tasks by telling the system what to do. Examples include giving instructions to a system to perform operations such as tell the time, print a file, and remind the user of an appointment. A diverse range of products has been designed based on this model, including home entertainment systems, consumer electronics, and computers. The way in which the user issues instructions can vary from pressing buttons to typing in strings of characters. Many activities are readily supported by giving instructions.

Operating systems like Unix and Linux have been designed primarily as command-based systems, where users issue instructions at the prompt as a command or set of commands. In Windows and other GUI-based systems, control keys or the selection of menu options via a mouse, touch pad, or touch screen are used. Typically, a wide range of functions are provided from which users have to select when they want to do something to the object on which they

are working. For example, a user writing a report using a word processor will want to format the document, count the number of words typed, and check the spelling. The user instructs the system to do these operations by issuing appropriate commands. Typically, commands are carried out in a sequence, with the system responding appropriately (or not) as instructed.

One of the main benefits of designing an interaction based on issuing instructions is that the interaction is quick and efficient. It is particularly fitting where there is a need to frequently repeat actions performed on multiple objects. Examples include the repetitive actions of saving, deleting, and organizing files.

ACTIVITY 2.3

There are many different kinds of vending machines in the world. Each offers a range of goods, requiring the user initially to part with some money. Figure 2.6 shows photos of two different vending machines, one that provides soft drinks and the other a range of snacks. Both use an instructional mode of interaction. However, the way they do so is quite different.

What instructions must be issued to obtain a soda from the first machine and a bar of chocolate from the second? Why has it been necessary to design a more complex mode of interaction for the second vending machine? What problems can arise with this mode of interaction?

Comment

The first vending machine has been designed using simple instructions. There are a small number of drinks to choose from and each is represented by a large button displaying the

Figure 2.6 Two different types of vending machine

label of each drink. The user simply has to press one button and this should have the effect of returning the selected drink. The second machine is more complex, offering a wider range of snacks. The trade-off for providing more options, however, is that the user can no longer instruct the machine by using a simple one-press action but is required to use a more complex process, involving (i) reading off the code (e.g. C12) under the item chosen, then (ii) keying this into the number pad adjacent to the displayed items, and (iii) checking the price of the selected option and ensuring that the amount of money inserted is the same or greater (depending on whether or not the machine provides change). Problems that can arise from this type of interaction are the customer misreading the code and/or miskeying the code, resulting in the machine not issuing the snack or providing the wrong item.

A better way of designing an interface for a large number of options of variable cost might be to continue to use direct mapping, but use buttons that show miniature versions of the snacks placed in a large matrix (rather than showing actual versions). This would use the available space at the front of the vending machine more economically. The customer would need only to press the button of the object chosen and put in the correct amount of money. There is less chance of error resulting from pressing the wrong code or keys. The trade-off for the vending company, however, is that the machine is less flexible in terms of which snacks it can sell. If a new product line comes out they will also need to replace part of the physical interface to the machine – which would be costly. ■

2.5.2 Conversing

This form of interaction is based on the idea of a person having a conversation with a system, where the system acts as a dialog partner. In particular, the system is designed to respond in a way another human being might when having a conversation. It differs from the activity of instructing insofar as it encompasses a two-way communication process with the system acting like a partner rather than a machine that obeys orders. It has been most commonly used for applications where the user needs to find out specific kinds of information or wants to discuss issues. Examples include advisory systems, help facilities, and search engines.

The kinds of conversation that are currently supported range from simple voice-recognition, menu-driven systems that are interacted with via phones, to more complex natural language-based systems that involve the system parsing and responding to queries typed in by the user. Examples of the former include banking, ticket booking, and train-time inquiries, where the user talks to the system in single-word phrases and numbers – e.g. yes, no, three – in response to prompts from the system. Examples of the latter include search engines and help systems, where the user types in a specific query – e.g. 'how do I change the margin widths?' – to which the system responds by giving various answers.

A main benefit of developing a conceptual model that uses a conversational style of interaction is that it allows people, especially novices, to interact with a system in a way that is familiar to them. For example, the search engine Ask Jeeves for Kids! allows children to ask a question in a way they would when asking their teachers or parents rather than making them reformulate their question in terms of keywords and Boolean logic. Similarly, the

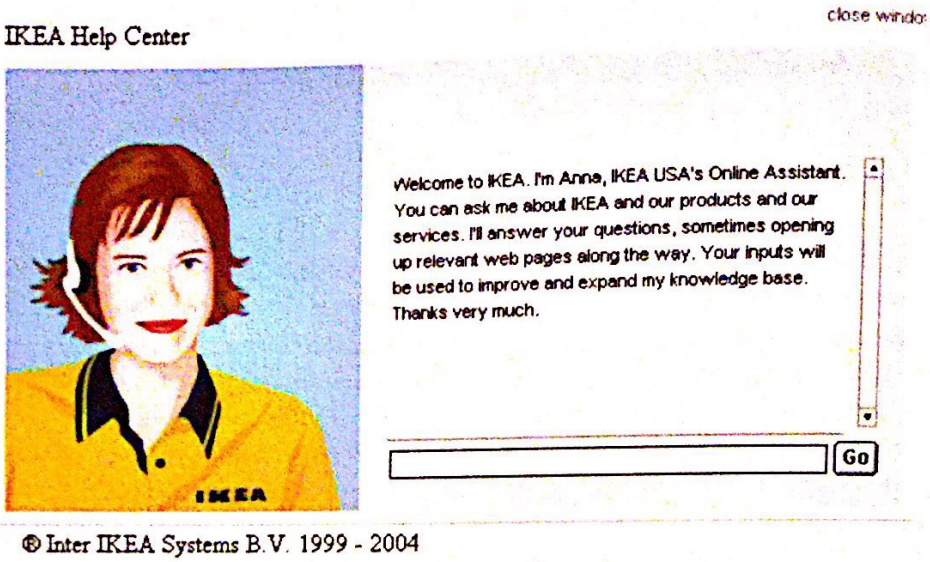


Figure 2.7 An example of an online agent, Anna, designed by Verity for the Ikea furniture store

generation of virtual representatives that have been incorporated into online store websites offer customers quick and direct answers to their product-related queries. An example is Anna, whose appearance was commented upon in the last chapter. She is a semi-cartoon character fronting the Swedish furniture store Ikea's Help Center (www.ikea.com) by directing the user to a part of the store's website in response to his questions typed in at the dialog box (see Figure 2.7). For example, when a user types in 'do you have any kitchen chairs?' Anna replies 'please have a look at the chairs' and a page of chairs is automatically displayed. The system matches keywords in the queries to a database of suitable web pages or answers.

A problem that can arise from using a conversational-based interaction type is that certain kinds of tasks are transformed into cumbersome and one-sided interactions. This is especially true for automated phone-based systems that use auditory menus to advance the interaction. Users have to listen to a voice providing several options, then make a selection, and repeat through further layers of menus before accomplishing their goal, e.g. reaching a real human or paying a bill. Here is the beginning of a dialog between a user who wants to find out about car insurance and an insurance company's reception system:

<user dials an insurance company>

'Welcome to St. Paul's Insurance Company. Press 1 if you are a new customer; 2 if you are an existing customer.'

<user presses 1>

'Thank you for calling St. Paul's Insurance Company. If you require house insurance press 1, car insurance press 2, travel insurance press 3, health insurance press 4, other press 5.'

<user presses 2>

'You have reached the car insurance division. If you require information about fully comprehensive insurance press 1, third-party insurance press 2 . . .'

**"If you'd like to press 1, press 3.
If you'd like to press 3, press 8.
If you'd like to press 8, press 5..."**

2.5.3 Manipulating

This form of interaction involves manipulating objects and capitalizes on users' knowledge of how they do so in the physical world. For example, digital objects can be manipulated by moving, selecting, opening, and closing. Extensions to these actions include zooming in and out, stretching, and shrinking – actions that are not possible with objects in the real world. Human actions can be imitated through the use of physical controllers (e.g. Wii) or gestures made in the air (e.g. Kinect) to control the movements of an on-screen avatar. Physical toys and robots have also been embedded with computation and capability that enables them to act and react in programmable ways depending on whether they are squeezed, touched, sensed, or moved. Tagged physical objects (e.g. balls, bricks, blocks) that are manipulated in a physical world (e.g. placed on a surface) can result in other physical and digital events occurring, such as a lever moving or a sound or animation being played.

A framework that has been highly influential in informing the design of GUI applications is direct manipulation (Shneiderman, 1983). It proposes that digital objects be designed at the interface so that they can be interacted with in ways that are analogous to how physical objects in the physical world are manipulated. In so doing, direct manipulation interfaces are assumed to enable users to feel that they are directly controlling the digital objects represented by the computer. The three core principles are:

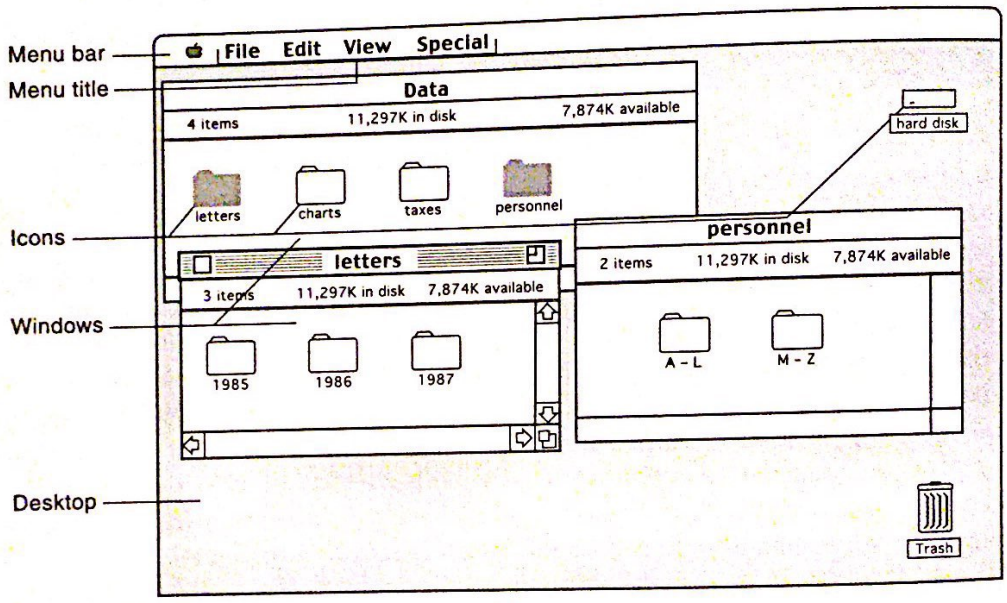
1. continuous representation of the objects and actions of interest;
2. rapid reversible incremental actions with immediate feedback about the object of interest;
3. physical actions and button pressing instead of issuing commands with complex syntax.

According to these principles, an object on the screen remains visible while a user performs physical actions on it and any actions performed on it are immediately visible. For example, a user can move a file by dragging an icon that represents it from one part of the desktop to another. The benefits of direct manipulation include:

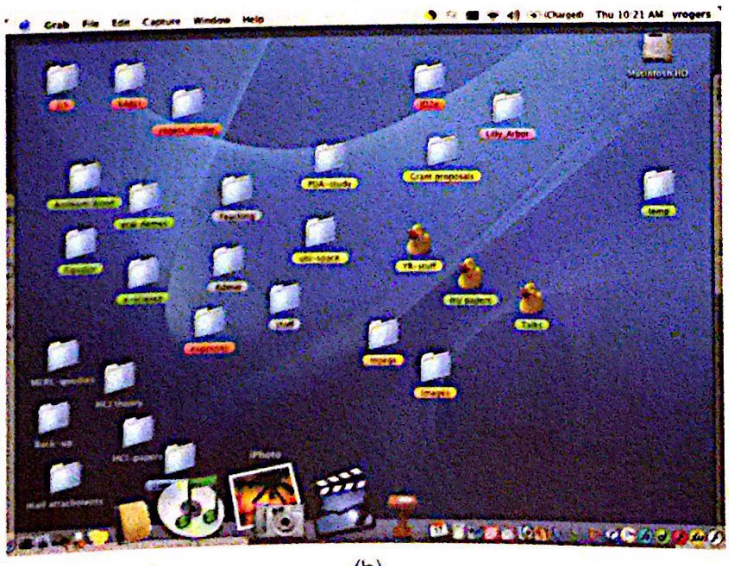
- helping beginners learn basic functionality rapidly;
- enabling experienced users to work rapidly on a wide range of tasks;
- allowing infrequent users to remember how to carry out operations over time;
- preventing the need for error messages, except very rarely;
- showing users immediately how their actions are furthering their goals;
- reducing users' experiences of anxiety;
- helping users gain confidence and mastery and feel in control.

Apple was one of the first computer companies to design an operating environment that used direct manipulation as its central mode of interaction. The highly successful Mac desktops (Figures 2.8a and b) and the more recent iPad display (Figure 2.8c) show the evolution of direct manipulation interfaces over the past 25 years.

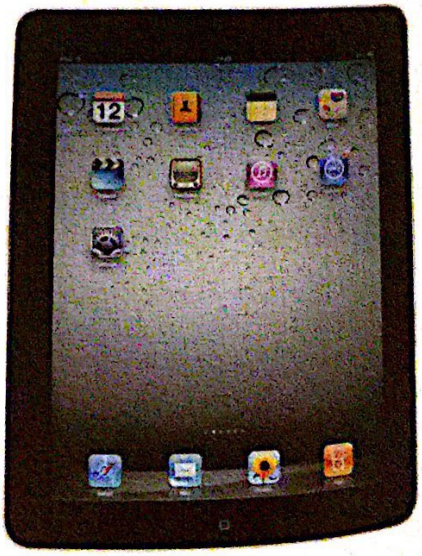
Many applications have been developed based on some form of direct manipulation, including word processors, video games, learning tools, and image editing tools. However, while direct manipulation interfaces provide a very versatile mode of interaction they do have



(a)



(b)



(c)

Figure 2.8 Three screen shots of (a) an original Mac desktop (1987), (b) the Mac OS X (2005), and (c) a first iPad interface (2010). What are the main differences?

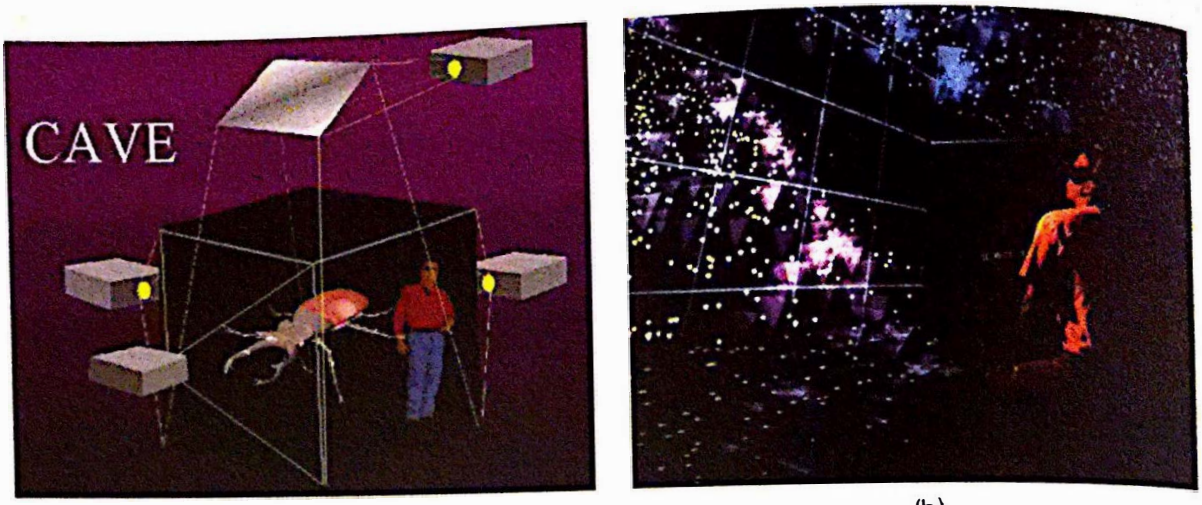
their drawbacks. In particular, not all tasks can be described by objects and not all actions can be undertaken directly. Some tasks are also better achieved through issuing commands. For example, consider how you edit an essay using a word processor. Suppose you had referenced work by Ben Shneiderman but had spelled his name as 'Schneiderman' throughout the essay. How would you correct this error using a direct manipulation interface? You would need to read through your essay and manually select the 'c' in every 'Schneiderman,' highlighting and then deleting it. This would be very tedious and it would be easy to miss one or two. By contrast, this operation is relatively effortless and also likely to be more accurate when using a command-based interaction. All you need to do is instruct the word processor to find every 'Schneiderman' and replace it with 'Shneiderman.' This can be done through selecting a menu option or using a combination of command keys and then typing the changes required into the dialog box that pops up.

2.5.4 Exploring

This mode of interaction involves users moving through virtual or physical environments. For example, users can explore aspects of a virtual 3D environment, such as the interior of a building. Physical environments can also be embedded with sensing technologies that, when they detect the presence of someone or certain body movements, respond by triggering certain digital or physical events. Similar to direct manipulation and direct manipulatives, the fundamental idea is to enable people to explore and interact with an environment, be it physical or digital, by exploiting their knowledge of how they move and navigate through existing spaces.

Many 3D virtual environments have been built that include virtual worlds designed for people to move between various spaces to learn (e.g. virtual universities) and fantasy worlds where people wander around different places to socialize (e.g. virtual parties). One of the best-known examples is Second Life. Numerous virtual landscapes depicting cities, parks, buildings, rooms, and datasets have also been built, both realistic and abstract, that enable users to fly over them and zoom in and out of different parts. Other virtual environments that have been built include worlds that are larger than life, enabling users to move around them, experiencing things that are normally impossible or invisible to the eye (Figure 2.9a); highly realistic representations of architectural designs, allowing clients and customers to imagine how they will use and move through planned buildings and public spaces; and visualizations of complex datasets that scientists can virtually climb inside and experience (Figure 2.9b).

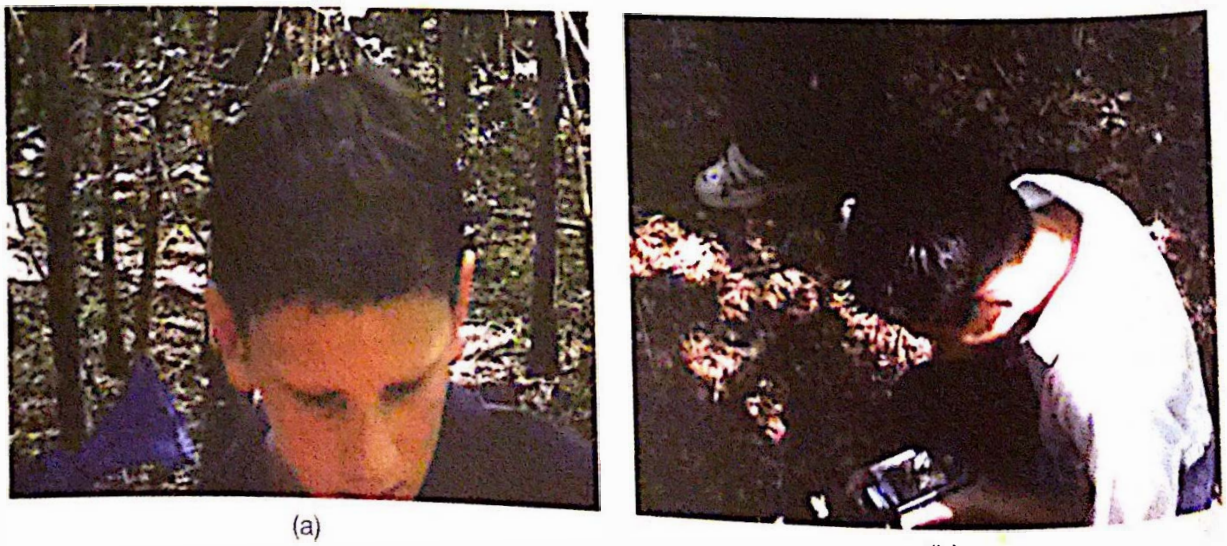
A number of physical environments have been developed using embedded sensor technologies and other location-detection technologies. They are often called context-aware environments: the location and/or presence of people in the vicinity of a sensing device is detected and based on this, the environment decides which digital information to provide on a device (e.g. a nearby coffee bar where friends are meeting) or which action to perform (e.g. changing lights in a room) that is considered relevant or useful to the person at a particular time and place. Many location-based virtual guides have been developed for cell phones, which provide relevant information about restaurants, historical buildings, and other places of interest as the person wanders near them. Physically embedded environments have also been designed to extend how children learn. For example, the Ambient Wood project was designed as an outdoor learning experience where a physical woodland was wired to present various forms of digital information to children, as they moved



(a) (b)

Figure 2.9 (a) A CAVE that enables the user to stand near a huge insect, e.g. a beetle, be swallowed, and end up in its abdomen; and (b) NCSA's CAVE being used by a scientist to move through 3D visualizations of the datasets

around it (Rogers *et al*, 2005). Depending on which part of the woodland they passed by (e.g. a particular kind of tree, a bush, a hole), an image would occasionally pop up on a screen they were carrying, or a sound was played via hidden speakers or heard through a special handheld audio device – the ambient horn (see Figure 2.10). The idea was to provide contextually relevant digital information that would enhance the usual physical experience available to children when exploring an outdoor world.



(a) (b)

Figure 2.10 The Ambient Wood. Children (a) listening to ambient sounds and (b) viewing images that popped up as they walked past certain locations in the wired wood