

Introduction to Interactive Systems

The last part of the twentieth century witnessed an unprecedented growth in computing and information. The resulting explosion of computing innovation has reshaped the modern world, entering homes, business, entertainment and communication. This growth has made computers more accessible and useful. As computing attempts to serve more people in more ways than ever before, the human-computer interface is of critical importance. This book is about the technologies that shape the human-computer interface. The goal is to present not only principles for implementing today's interactive systems but to prepare for future models of interaction.

The growth in computing is best characterized by Moore's Law¹, which says that every 18 months computers will be twice as fast and have twice as much memory with no increase in cost. Gordon Moore first stated his law in 1965 and it has been true since then. It is widely believed that although we will eventually reach the barriers of our known physics, this law can continue to hold for two more decades.

This explosive growth, with no increase in cost, continues to radically change the shape and nature of computing. In the late 1960s and early 1970s computing was characterized by huge machines occupying large specially air-conditioned rooms and served by specially trained personnel. User interaction was primarily via punched cards with cumbersome expensive terminal devices being introduced later. In the 1980s computers began to dominate the desktop and at the turn of the century a person could carry more computing in their shirt pocket than a 1,000 square foot room once held. A simple way to think of Moore's Law is that computing capacity will increase roughly 10 times in 5 years and 100 times in 10 years. Thus the average laptop of the year 2010 will execute approximately 20 billion instructions per second using 20 gigabytes of RAM.

Moore's law is not the only exponential growth in computing. Growth in disk space has exceeded that of computing power. The number of computers on the

Internet has grown even faster. All of these developments bring the power to create, manipulate and communicate information into the hands of ordinary people. It is a central thesis of this book that computing exists to serve the needs of human beings.

The exponential growth in computing stands in stark contrast to the growth in human capacity to perceive, remember and express information. The curve describing the change in human capacity over time is flat. Our personal capabilities are not changing. At first this might be a bit discouraging until one considers how great human capacity really is. The human brain contains several orders of magnitude more memory and computing capacity than the largest computers ever built. When it comes to sophisticated information processing human beings dominate.

So what then are the advantages of computers over people? Computers are more precise. Their memory is exact rather than approximate. Computation produces a crisp rather than a fuzzy result. We may use “fuzzy” reasoning in our programs but the result is always a particular fuzzy value. Computers and their networks assist us in transcending the barriers of time and space. They can perform tasks without us paying attention. Computers can perform the repetitive and boring without getting tired. In many ways computers complement rather than replace the capabilities of human beings. The things computers do for us are frequently those things that as humans we enjoy the least or perform poorly. The human/computing team forms a powerful combination in seeking solutions to problems.

To understand the future of interactive systems, a more interesting curve is the inverse of Moore’s Law. If instead of holding cost constant we hold computing capacity constant and look at the change of price. The cost of a unit of computing is cut in half every 18 months or is 100 times cheaper in 10 years. It is this inverse law that brought us the personal computer and the personal digital assistant (PDA). In the year 2000 a computer that can perform speech recognition cost \$1,000. In the year 2010 speech recognition will be a \$10 part that can be included in virtually any device.

The inverse of Moore’s Law is very important when we consider the future interactive systems. For both human and economic reasons the dominant model for interacting with a computer is seated at a desk with a screen, keyboard and mouse. However, that is not how most people live, work and play. For the recent past the desktop computer is the most economical compromise between the

capabilities of computers and the physical/perceptual capabilities of human beings. When powerful computing costs \$10 the economic balance between humans and computing will shift just as powerfully as when the room-size computers arrived on the desktop.

Computers will be everywhere. Interactive devices will be everywhere. At the end of the twentieth century we witnessed a transformation from each computer served by hundreds of people to each person served by their own computer. This transition will occur again with each person served by hundreds of computers. This will radically change the way in which people interact with technology. In particular, interactive computers must blend into the physical situations in which people live and work. Interacting in a car is very different from interacting on the trading floor of a stock exchange which is different from the needs of a mechanic repairing a computer-controlled engine, which contrasts with a search and rescue worker on an earthquake site. The computer must adapt to people rather than the other way around. In this book we will consider not only the dominant forms of interaction, as they exist today, but also those principles that will carry us into the next decade.

Humans as part of information processing systems

In most human/computer teams the role of the computer is to remember, manipulate, find and communicate information. The handling of information is what a computer does best. Consider the diagram of the Internet shown in figure 1.1. This is the most common view of the Internet with servers, workstations and people on the outside with the network in the middle forming the connective tissue that holds it all together.

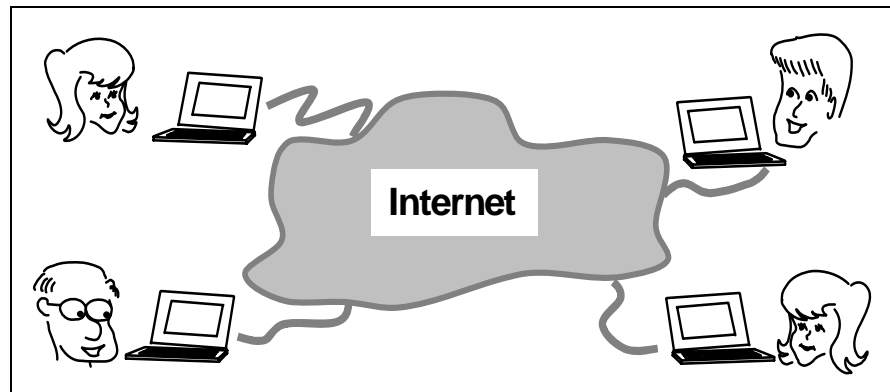


Figure 1.1 - Network-centric view of the Internet

However, if we consider the Internet as a massive information-processing engine, it looks more like figure 1.2. The network and its computers form the conduits that connect people together. This is more reflective of the modern business and social world. When someone decides to buy a laptop, they will access the website of a vendor and decide what configuration they want. They will communicate that decision to some staff member who will check the budget and communicate the order to a purchasing agent. The purchasing agent will recheck the budget and send a purchase order to the vendor. The original client will get anxious about the delay and contact staff, who will contact purchasing, who will contact the vendor. In the mean time the vendor has sent the order to manufacturing, who starts to assemble the computer and gives it to the delivery service. The message comes back to the client that it is being delivered and a tracking number can be used to find the state of the delivery. This entire process is a network of information transactions. From this view the network appears as shown in figure 1.2.

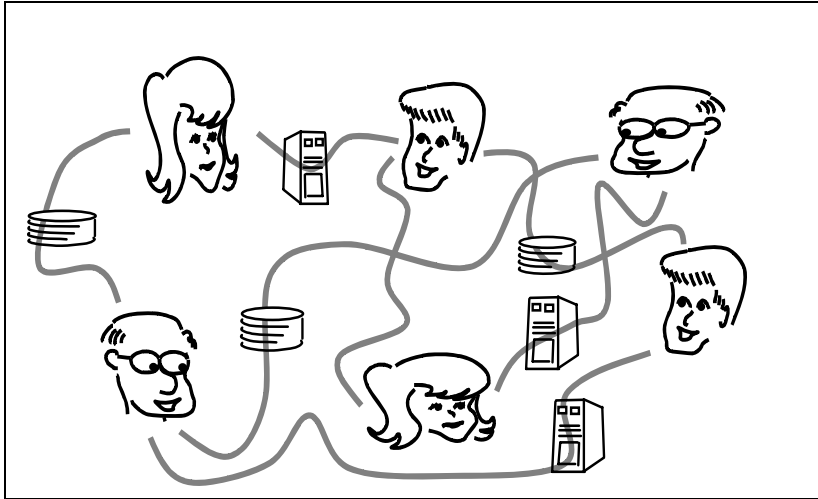


Figure 1.2 - Human-centric view of the Internet

The nodes of the network are human, with computing devices as the connections. We frequently think about the speed of the network in terms of bits transferred. However, in a human-centric network it is the human throughput, not the network that constrains the speed of the process. The human/computer interface is a key bottleneck to these transactions. As we look to the future, increasing the effectiveness of the human/computer interface will do as much or more for information processing as increasing the speed of computers and networks.

This book presents the whole of interactive technology rather than simply the design of interactive software for a screen/keyboard/mouse. The principles are frequently the same among the various modes of interaction. We will focus particularly on the tools and technologies that make up an interactive system. A fundamental thesis of this book is that tools and technology drive what is possible. The economics of software development dictate rapid development time. This forces many products to pursue well-trodden interface designs because those are the tools that come readily to hand. Programmers, when pressed, tend towards the skills that they know. The goal of this book is to provide a solid background in a variety of interactive technologies that cover the whole range of human activity.

Though the coverage of all of human activity sounds like a huge task it is actually limited by the number of ways in which humans can perceive and express information. This book will use the set of human capabilities accompanied by a set of basic information processing tasks as an overarching outline.

Architecture of Interactive Systems

Regardless of the form it may take, all human-computer interaction has an architecture like that shown in figure 1.3. It all begins with a *model* of the information to be manipulated. This information can serve a variety of purposes. It might be relatively static information like a diagram or a document that changes only when the user requests a change. The information may also be a representation of an ongoing process, such as the state of a robot, the stock market or an online auction where independent events constantly change the information. Regardless of the mode of interaction (pictures, speech, pressure on the skin, etc.) the software must present the current state of its information model to the user. This presentation is called the *view*. Not only must the view present the state of the model but also it must update its presentation in response to model changes from the user or some other activity.

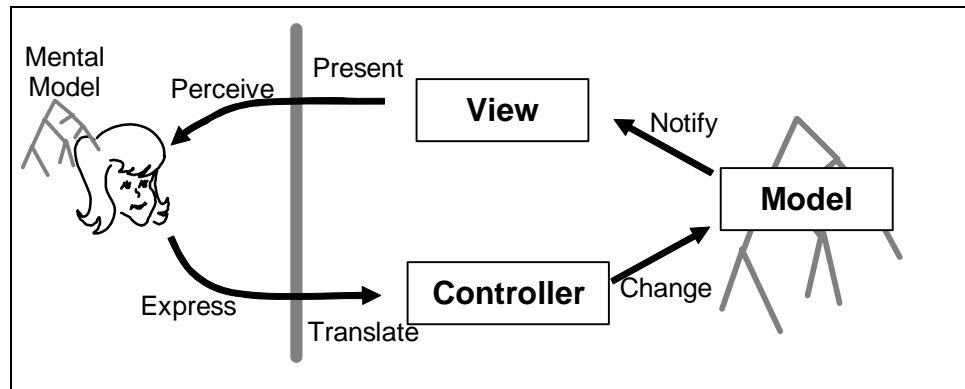


Figure 1.3 – Model of Interaction

When the information is presented, the user must *perceive* that presentation and form a mental model of the application's state. This leads to the problem described by Don Norman as the "Gulf of Evaluation"². A gulf of evaluation occurs when the user's mental model is inconsistent with the actual model of the application. This gulf can arise for a variety of reasons. The user may be

distracted, poorly trained or just not paying attention. On the other hand the presentation may be poor, the form of presentation may be inadequate for the information to be communicated or the model too complex for the problem the user is trying to solve. The user will make decisions and take action based on their mental model of the application's state. If the mental model is incorrect the user will make incorrect decisions on what to do next.

Based on their mental model, a user will formulate a plan of action and *express* some desire to the application. The software must understand what is being expressed and translate that expression into changes to the underlying model. These changes may do a variety of things. In a simple document-editing task the model changes are simply modifications to the document. In a robot control task the change of information may be to modify the direction the robot should travel. In the majority of human-computer interaction the user expresses or modifies some information, which the computer then uses to guide its own future behavior. It is the *controller* that must understand the user's expression. The controller must translate the user's input into model changes. For example, a word processor's controller translates the backspace key into a deletion of a character in the document. Frequently the translation is much more complicated than simple key mapping. For example, speech or handwriting recognition may be part of the input translation.

It is currently impossible to write a controller that can understand all possible forms of human expression and translate them accurately into correct model changes. Human beings themselves cannot do it for all forms of expression with any reliability. Any controller can only understand a limited set of human expressions. Therefore Don Norman's "Gulf of Execution" arises. This is where the human has formulated a plan for a desired change but has difficulty translating that plan into an expression the controller can understand. This can occur either because the human wants to do something that the program cannot do or because the human cannot remember or figure out the necessary expression.

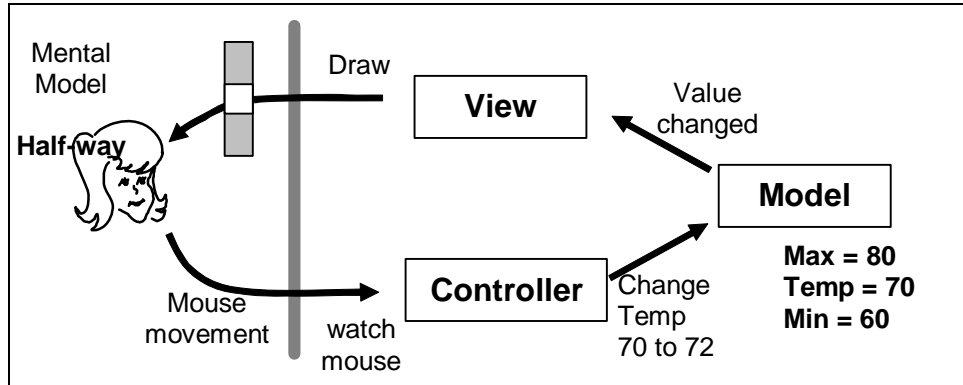


Figure 1.4 – Graphical temperature control

In this discussion we have used the general terms present, perceive, express and translate. The reason for this generality is the wide range of human and computer capabilities that can be used for these activities. We can take as an example a simple temperature control. The model consists of three integers Max, Min and Temp. Figure 1.4 shows a scroll bar on a screen being used to present our temperature control. The view draws a picture of the scroll-bar and the user perceives the thumb of the scroll-bar to be halfway between the max and the min. Note that the user's mental model is not exactly the same as the actual model. If the user feels a little cool and wants to make the room warmer this presentation may be enough. However, the user may be confused about what Max and Min are and may move the scroll-bar too high. This would be a gulf of evaluation problem (does not know Max and Min) that can cause a gulf of execution problem (does not know how far to move the scroll-bar). We could improve this by having the view present the actual values for Max, Min and Temp. However, for other uses of a scroll-bar this additional information may unnecessarily clutter the interface. Resolving such design questions requires thoughtful consideration of the users and their goals.

The user uses the mouse to press down over the thumb, drag the mouse up and then release. The controller receives the mouse-down, mouse-movement and mouse-up events and in cooperation with the view concludes that the user wants to change Temp from 70 to 72. The controller calls the model to make this change and the model notifies the view of the change. The view will then redraw the scroll-bar to reflect this new value. This translate-change-notify-present cycle is fundamental to all interaction.

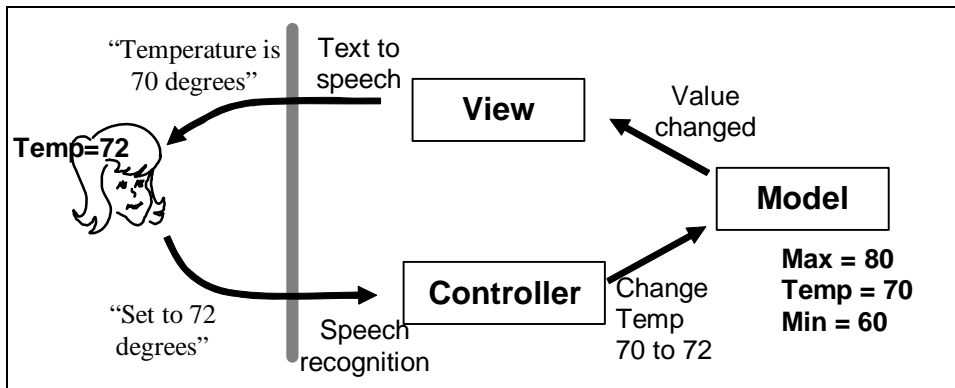


Figure 1.5 – Speech temperature control

We could modify figure 1.4 to the one in figure 1.5. In this case the user has called their home automation system from a cell-phone and the interface now uses speech instead of graphics as its interactive medium. The view is implemented differently because instead of drawing its version of the model it translates it into a sentence, which is passed to a text-to-speech engine, which then speaks the view information to the user. The mental model is different because the user hears the exact temperature. The user expresses herself in a sentence, which a speech recognizer must interpret and the controller must translate into a change to the Temp value. The model notifies the view and the new temperature is spoken to the user. The architecture is the same. Only the mode of presentation and translation has changed.

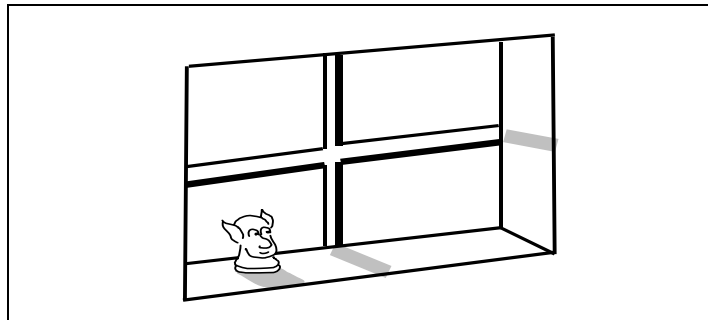


Figure 1.6 - The temperature elf

In the home of the future we could place a little statue of an elf on the kitchen windowsill. A camera mounted in the ceiling would watch the position of the elf (as well as many other things in the kitchen). To make the room warmer, we move the elf to the right and to make it colder we move the elf to the left. The view does nothing because the physical position of the elf is the feedback to the user. The user interacts with the system by moving a physical object in the real world (the elf). The camera system watching the room detects the change of the elf's position. The controller interprets the new position as a change to the model. The only effect is to notify the heater to turn up the temperature. There is no need to notify the view of anything.

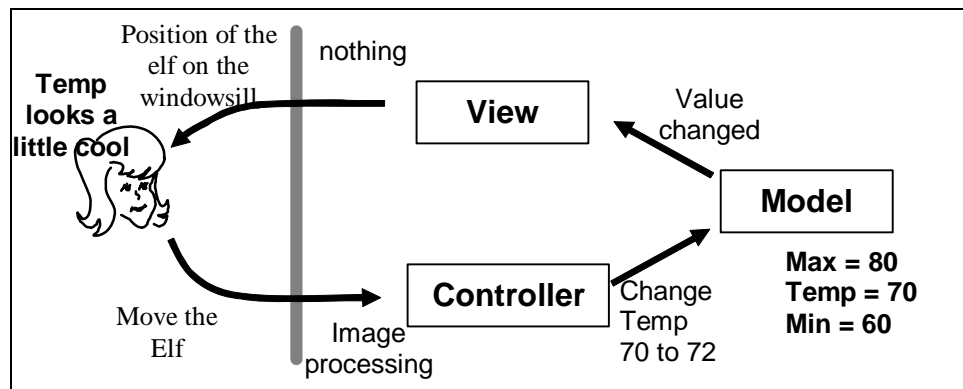


Figure 1.7 – Physical world temperature control

These three examples, scroll-bar, speech and the magic elf, all demonstrate the model/view/controller architecture that pervades interactive computing. They also demonstrate the wide range of ways in which people might interact with their computers.

Forms of Presentation

Now that we have a software architecture around which we can discuss user interfaces we must also consider the human aspects of the problem. Only a brief overview of the human issues is given here. We need to understand how our software can present information to users and how users can express their desires to the computer. A more detailed discussion of how human physiology impacts the user interface will be given in chapter 19.

At first, the task of addressing all forms of human-computer interaction may seem daunting until one considers that there are limitations on human abilities to interact. There are fixed ways that humans perceive and express information. For the purposes of interacting with information only the visual, audio and tactile (touch) senses have sufficient bandwidth and precision to be useful as an interactive medium.

Vision is the highest bandwidth channel for humans to receive information. The information bandwidth of the two eyes exceeds all other senses by at least an order of magnitude. The most important interactive consideration is the spatial resolution of the eye. Human vision has about a 90-degree field of view. However, the resolution is not uniform across that area. Most of our visual acuity is in the macula at the center of the retina. The macula also contains most of the color vision. Our peripheral vision is low resolution and mostly gray scale. We compensate for this by rapid eye movement to focus the macula on the regions of interest. This accounts for the time it takes to visually scan a page. This also accounts for many of the visual design principles that we will discuss in later chapters.

Audio is the second highest bandwidth information channel. Audio bandwidth is much lower than visual and we have much less control over it. With audio we need not pay attention to receive input. Conversely we find audio hard to ignore. Audio is also a transient medium. Unlike vision where the eyes can rapidly go back and review any part of an image that was not understood, there is no audio playback function in the real world. Consequently we must experience audio as it happens. Listening as compared to looking can be slow and tedious.

Speech is a very special form of audio presentation because it uses words and language. Though the audio channel has very low bandwidth, speech conveys a large amount of information by exploiting shared experiences. When someone says the word “puppy” very few bits of audio are transmitted, but the word invokes in the hearer a large store of visual, tactile and emotional information associated with puppies. This is both an advantage and a disadvantage. The advantage is the communication leverage that we gain and the disadvantage is the differences in experience between the speaker and the hearer. The speaker is thinking of warm, soft and cute while the hearer is remembering tooth marks on the furniture and unpleasant wet spots on the carpet.

As humans we do receive information through the sense of touch. The bandwidth, however, is much lower than audio and our precision is very poor.

There are tactile presentation devices but they are relatively expensive and are limited in their ability to communicate a sense of touch for arbitrary objects.

Forms of Expression

To complete the interactive cycle, the user must express their desires to the computer in some fashion. The highest bandwidth human expression is by voice. This can be speech or other sounds. A second form of expression is by pointing or gesturing. This can take place either on a surface such as a screen, on a piece of paper or in the 3D physical world. A rather specialized form of expression is the keyboard. This form is effective only because of widespread training of the workforce. Without the training, typing is a rather unnatural form of expression. With such training, keyboards are currently the most accurate way for humans to communicate words to machines. A last form of human expression is the movement of various body parts or the movement of physical objects. An example of this would be the conductor of an orchestra who communicates tempo and dynamics through her hands and baton.

A key problem with human expression to machines is the accuracy of that expression. There is a mismatch between what humans can communicate and what computers can understand. Speech recognition by computers has error problems. Current computer vision cannot accurately recognize sign language used by the deaf. One of the key reasons that we still use typing as a means of input to computers is that we can trust the computer to accurately receive the message.

There is also a mismatch between the accuracy of expression required by a computer and the accuracy with which people express themselves. Suppose when working on a project a user says “That is such a pain!” In a human-human setting this is entirely appropriate. It communicates feelings of disgust and frustration. Other people rarely attempt to clarify what “That” really is and they do not ask what hurts. In this situation, disambiguating the references is not important to what is being communicated. That is not to say that this phrase has no meaning or importance. Sensing a companion’s frustration will change the way we interact with them in the near future. Computers, in their present state, cannot handle such vague communications. They must either attempt to understand this phrase or ignore it completely. Our computer applications do not carry a “sense of the mood”.

The speed and accuracy of human expression largely depends on training, voluntary muscle control and the ratio of muscle mass to body part being manipulated. Our eye muscles are fast and accurate but most of their motion is semi-voluntary flitting rapidly from point to point, controlled mostly by attention management rather than conscious will. The fingers are very good because they are small compared to the large muscle mass in the forearms that control their movement. The wrist and hand have much larger size than fingers, yet are controlled by muscles of roughly the same size. Interaction through the wrist and hand is slower and less accurate than the fingers. It is easy to do handwriting with a pen (fingers) and almost impossible with a mouse (wrist). This rough analysis of physical dexterity extends to other body parts is sufficient for most of our interactive designs.

Models

The two previous sections discussed ways in which people perceive and express information. However, the design of the user interface does not begin with these surface issues. It begins with the model. The model defines “what” the user interface can accomplish for the user rather than how it is interactively done. In addition to the Gulf of Evaluation and Gulf of Execution, designers of user interface also face the Modeling Gulf. It is not physically possible to model all of reality in a computer since that would require that a computer completely model itself. Every application models only a portion of reality, and in fact it only models an abstraction of that. We always leave a great deal of any application out of our model. What we include and what we leave out are critical issues of user interface design and are of particular importance to programmers as they are the ones that best understand what can be modeled given current technology. The process of creating model designs is called functional design and is discussed in chapter 18.

At this point we need to better understand what a model is and how it relates to the rest of our interactive architecture. The model is the information that the user will interactively manipulate and its supporting mechanisms. In our scrollbar example the model was the minimum, current and maximum values. In figure 1.8 we see a collection of radio buttons. Their model is the currently selected choice.

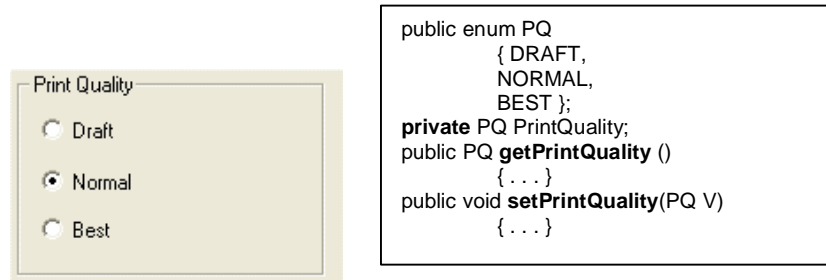


Figure 1.8 – Radio button model

In addition to storing the information to be manipulated, the model must also notify all views of any changes. For this reason we never make model variables public. We always access them through a method interface, as shown in figure 1.8. This method interface defines our model.

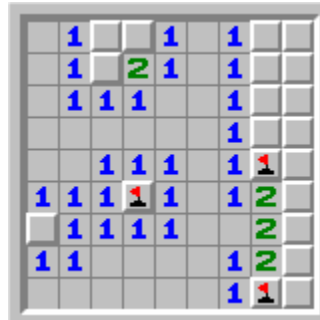
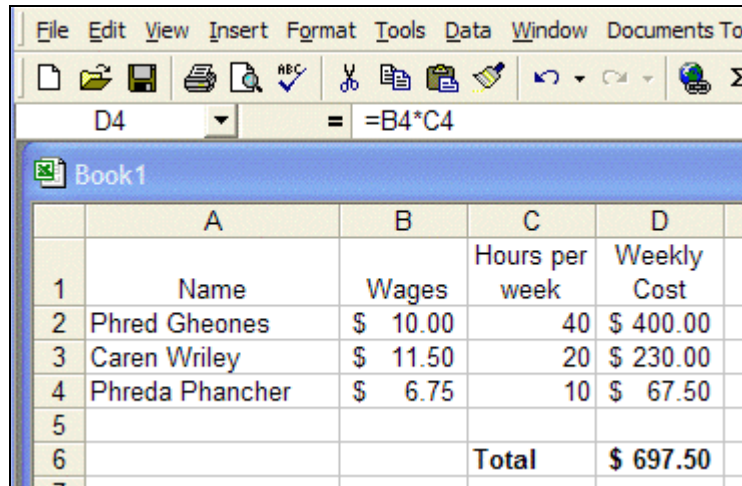


Figure 1.9 - Minesweeper

Figure 1.9 shows the game Minesweeper. The model for this game is an array of cells. Each cell is hidden or showing, has a mine or not, and has a flag or not. In addition there is the number of mines in the field and the number of flags that the user has set. Lastly there is the state of won, lost or still playing. The mines, flags and won/loss values are all read-only because they are computed from the rest of the model and cannot be set directly by the user. The only actions that the controller can take on behalf of the user are to restart the game, set a flag or show a hidden cell. Minesweeper is also a simple example of the model notification problem. When the user opens a cell, many cells may be uncovered. It is the model's responsibility to do this, not the controller and not the view. Only the model knows how to play the game. When a user, through the

controller, requests the opening of a cell, the model must inform the view of all of the cells that have been opened so that the view is correctly updated.

We could easily put a speech-based controller in front of this model that supports commands like “show B-4” to show the cell at row B and column 4. In fact if we had two controllers, one for the mouse and one for speech that shared the same model, the user could play either by clicking or speaking and the rest of the program would not care.



The screenshot shows a spreadsheet application window. At the top is a menu bar with options: File, Edit, View, Insert, Format, Tools, Data, Window, Documents To. Below the menu bar is a toolbar with various icons for file operations and editing. A formula bar shows the active cell D4 with the formula $=B4*C4$. The spreadsheet itself is titled 'Book1' and contains a table with the following data:

	A	B	C	D
	Name	Wages	Hours per week	Weekly Cost
1	Phred Gheones	\$ 10.00	40	\$ 400.00
2	Caren Wriley	\$ 11.50	20	\$ 230.00
3	Phreda Phancher	\$ 6.75	10	\$ 67.50
4				
5				
6			Total	\$ 697.50

Figure 1.10 – Spreadsheet model

Figure 1.10 shows a fragment of a spreadsheet. The model for a spreadsheet is an array of cells (as with minesweeper) but there is much more information about cells, rows and columns. Cells have a textual value but they may also contain a functional expression for that value, a fill color, text color, border information, font style and font size. There are actually many attributes of a spreadsheet cell. In addition, the model allows many of these attributes to be set for entire rows or columns. Rows and columns also have heights and widths, which may be absolute or automatically sized from content.

In addition to the model information shown in the sheet, the menu and tool bar items of the application correspond to methods on the model that provide a variety of functions including saving, opening and printing spreadsheets, cutting and pasting, inserting new functions, sorting, changing the calculation strategy, etc.

Figure 1.10 also shows two additional views of the model beyond the sheet view. Just above the sheet and below the tool bar there is a special view that shows the currently selected cell and a view of the contents (an expression) of the selected cell. When the model changes, all three of these views must be updated to keep the user informed.

Overview

Because the dominant means for humans to receive information is through vision, the graphical user interface is and will remain the primary interactive medium. Consequently the first half of this book will focus on software principles for building graphical user interfaces. The second half will explore interactive technologies for the other human senses and means of expression. This will include speech, gestures, touch and 3D interfaces. All of these provide ways in which people can receive and express information.

Although this book focuses primarily on the technology, a basic knowledge of human perception and expression is required to design effective interaction. The goal is not to provide complete coverage of human capacities and usability engineering, but to provide an introduction so as to not leave technology students ignorant of the human issues.

Because this book is about interactive technology, it relies upon a variety of well known mathematical concepts and computing algorithms. Some readers will already have an extensive background in computer science and mathematics, which will make the software techniques obvious. Other readers will not have this background. The appendix contains much of the necessary mathematics and algorithms. Portions of the appendix are referenced throughout the book to provide pointers to the necessary algorithms. Algorithms in the appendix contain many references to other parts of the appendix for background concepts and algorithms. Following these dependencies should allow readers to readily learn material that they need without getting bogged down in descriptions of familiar concepts.

¹ Moore, Gordon E. "Cramming More Components onto Integrated Circuits" *Electronics*, 38(8), 1965.

² Norman, Donald A. *The Design of Everyday Things*, Doubleday, 1988.