UNIT I
FOUNDATIONS OF HCI

1. Explain in detail about I/O channels?OVERVIEW:

- √ **Vision**
- √ **The Human Eye**
- √ **Visual Perception**
- √ **Perceiving Size And Depth**
- √ **Perceiving Brightness**
- √ **Perceiving Color**
- √ **Hearing**
- √ **Touching**
- √ **Movement**

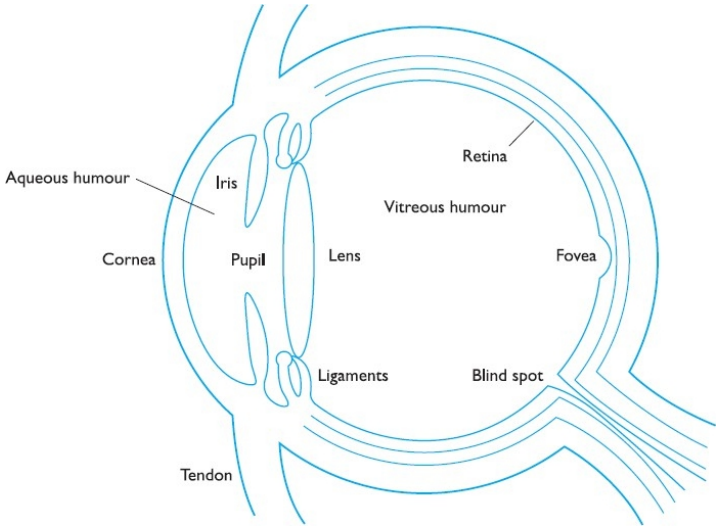
- ❖ A person's interaction with the outside world occurs through information being received and sent: input and output.
- ❖ In an interaction with a computer the user receives information that is output by the computer, and responds by providing input to the computer – the user's output becomes the computer's input and vice versa.
- ❖ Consequently the use of the terms input and output may lead to confusion shall blur the distinction somewhat and concentrate on the channels involved.
- ❖ This blurring is appropriate since, although a particular channel may have a primary role as input or output in the interaction, it is more than likely that it is also used in the other role.
- ❖ For example, sight may be used primarily in receiving information from the computer, but it can also be used to provide information to the computer,

Vision

- ❖ Human vision is a highly complex activity with a range of physical and perceptuallimitations, yet it is the primary source of information for the average person.
- ❖ We can roughly divide visual perception into two stages: the physical reception ofthe stimulus from the outside world, and the processing and interpretation of thatstimulus.
- ❖ On the one hand the physical properties of the eye and the visual systemthe interpretative capabilities of visual processing allow images to be constructed from incomplete information.

The human eye

- ❖ Vision begins with light. The eye is a mechanism for receiving light and transforming it into electrical energy.
- ❖ Light is reflected from objects in the world and their image is focussed upside down on the back of the eye. The receptors in the eye transform it into electrical signals which are passed to the brain.



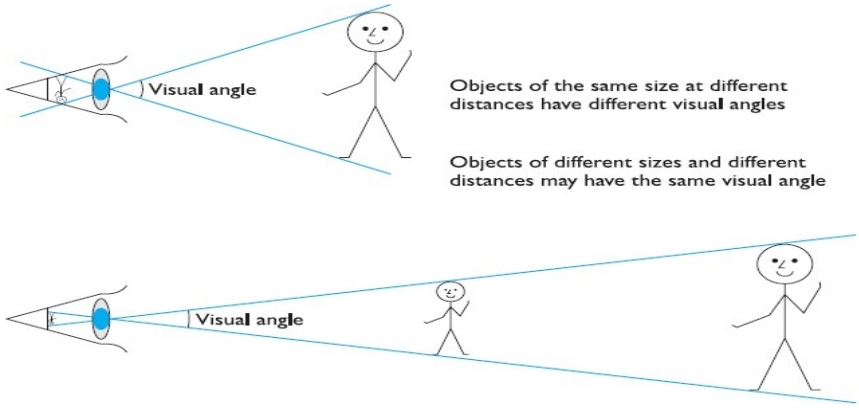
- The eye has a number of important components (see Figure 1.1) which we willlook at in more detail.
- The *cornea* and *lens* at the front of the eye focus the light intoa sharp image on the back of the eye, the *retina*. The retina is light sensitive and contains two types of *photoreceptor*: *rods* and *cones*.
- Rods are highly sensitive to light and therefore allow us to see under a low level of illumination. However, they are unable to resolve fine detail and are subject to light saturation.
- This is the reason for the temporary blindness we get when moving froma darkened room into sunlight: the rods have been active and are saturated by the sudden light.
- The cones do not operate either as they are suppressed by the rods. We are therefore temporarily unable to see at all.

Visual perception

- Understanding the basic construction of the eye goes some way to explaining the physical mechanisms of vision but visual perception is more than this.
- The information received by the visual apparatus must be filtered and passed to processing elements which allow us to recognize coherent scenes, disambiguate relative distances and differentiate color.

Perceiving size and depth

- Imagine you are standing on a hilltop. Beside you on the summit you can see rocks, sheep and a small tree.
- On the hillside is a farmhouse without buildings and farm vehicles. Someone is on the track, walking toward the summit. Below in the valley is a small market town



Perceiving brightness

- A second aspect of visual perception is the perception of *brightness*. Brightness is in fact a subjective reaction to levels of light.
- It is affected by *luminance* which is the amount of light emitted by an object. The luminance of an object is dependent on the amount of light falling on the object's surface and its reflective properties.
- Luminance is a physical characteristic and can be measured using a *photometer*. *Contrast* is related to luminance: it is a function of the luminance of an object and the luminance of its background.

Perceiving color

- A third factor that we need to consider is perception of color. Color is usually regarded as being made up of three components: *hue*, *intensity* and *saturation*.
- Hue is determined by the spectral wavelength of the light. Blues have short wavelengths, greens medium and reds long.
- Approximately 150 different hues can bediscriminated by the average person. Intensity is the brightness of the color, and saturation is the amount of whiteness in the color.

The capabilities and limitations of visual processing

- In considering the way in which we perceive images we have already encountered vsome of the capabilities and limitations of the human visual processing system.
- However, we have concentrated largely on low-level perception. Visual processing involves the transformation and interpretation of a complete image, from the light that is thrown onto the retina. As we have already noted, our expectations affect the way an image is perceived.

Reading

- So far we have concentrated on the perception of images in general. However,the perception and processing of text is a special case that is important to interface design, which invariably requires some textual display. We will therefore end this section by looking at *reading*.
- Adults read approximately 250 words a minute. It is unlikely that words are scanned serially, character by character, since experiments have shown that words canbe recognized as quickly as single characters. Instead, familiar words are recognized using word shape.
- This means that removing the word shape clues (for example, by capitalizing words) is detrimental to reading speed and accuracy.

Hearing

- The sense of hearing is often considered secondary to sight, but we tend to under estimate the amount of information that we receive through our ears. Close your eyes for a moment and listen.
- What sounds can you hear? Where are they coming from?What is making them? As I sit at my desk I can hear cars passing on the road outside,machinery working on a site nearby, the drone of a plane overhead and bird song.
- But I can also tell *where* the sounds are coming from, and estimate how far away they are. So from the sounds I hear I can tell that a car is passing on a particular road nearmy house, and which direction it is traveling in.
- I know that building work is in progress in a particular location, and that a certain type of bird is perched in the treein my garden.

The human ear

- Just as vision begins with light, hearing begins with vibrations in the air or *sound waves*. The ear receives these vibrations and transmits them, through various stages to the auditory nerves.
- The ear comprises three sections, commonly known as the *outer ear*, *middle ear* and *inner ear*.The outer ear is the visible part of the ear. It has two parts: the *pinna*, which isthe structure that is attached to the sides of the head, and the *auditory canal*, along which sound waves are passed to the middle ear. The outer ear serves two purposes.

- First, it protects the sensitivemiddle ear from damage. The auditory canal contains wax which prevents dust, dirt and over-inquisitive insects reaching the middle ear.It also maintains the middle ear at a constant temperature.
- Secondly, the pinna and auditory canal serve to amplify some sounds.

Processing sound

- As we have seen, sound is changes or vibrations in air pressure. It has a number of characteristics which we can differentiate. *Pitch* is the frequency of the sound.
- A low frequency produces a low pitch, a high frequency, a high pitch. *Loudness* is proportional to the amplitude of the sound; the frequency remains constant.
- *Timbre* relates to the type of the sound: sounds may have the same pitch and loudness but be made by different instruments and so vary in timbre. We can also identify a sound's location,
- since the two ears receive slightly different sounds, owing to the time difference between the sound reaching the two ears and the reduction in intensity caused by the sound waves reflecting from the head.

Touch

- The third and last of the senses that we will consider is touch or *haptic perception*. Although this sense is often viewed as less important than sight or hearing, imagine life without it.
- Touch provides us with vital information about our environment. It tells us when we touch something hot or cold, and can therefore act as a warning.
- It also provides us with feedback when we attempt to lift an object, for example.
- Consider the act of picking up a glass of water. If we could only see the glass and not feel when our hand made contact with it or feel its shape, the speed and accuracy of the action would be reduced.
- This is the experience of users of certain *virtual reality* The apparatus of touch differs from that of sight and hearing in that it is not localized. We receive stimuli through the skin.
- The skin contains three types of sensory receptor: *thermoreceptors* respond to heat and cold, *nociceptors* respond to intense pressure, heat and pain, and *mechanoreceptors* respond to pressure.
- It is the last of these that we are concerned with in relation to human–computer interaction.

Movement

- Before leaving this section on the human's input–output channels, we need to consider motor control and how the way we move affects our interaction with computers.
- A simple action such as hitting a button in response to a question involves a number of processing stages.
- The stimulus (of the question) is received through the sensory receptors and transmitted to the brain. The question is processed and a valid response generated.
- The brain then tells the appropriate muscles to respond. Each of these stages takes time, which can be roughly divided into reaction time and movement time.

Movement time = $a + b \log_2(\text{distance}/\text{size} + 1)$

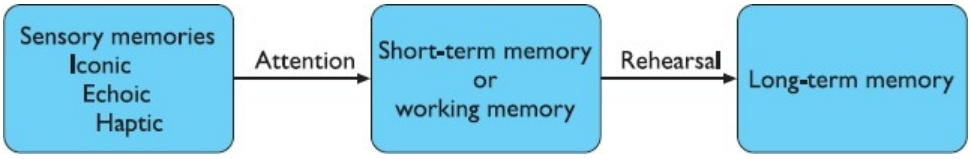
2. Describe in detail about memory?

OVERVIEW:

- √ **Memory**
- √ **Sensory Memory**
- √ **Short-Term Memory**
- √ **Long-Term Memory**
- √ **Long-Term Memory Structure**
- √ **Long-Term Memory Processes**

Memory

- Have you ever played the memory game? The idea is that each player has to recount a list of objects and add one more to the end.
- There are many variations but the objects are all loosely related: 'I went to the market and bought a lemon, some oranges, bacon . . .' or 'I went to the zoo and saw monkeys, and lions, and tigers . . .'
- As the list grows objects are missed out or recalled in the wrong order andso people are eliminated from the game.
- It allows us to repeat actions, to use language, and to use new information received via our senses. It also gives us our sense of identity, by preserving information from our past experiences.
- Memory is structured and the activities that take place within the system.It is generally agreed that there are three types of memory or memory function:*sensory buffers*, *short-term memory* or *working memory*, and *long-term memory*.
- There is some disagreement as to whether these are three separate systems or different functions of the same system. We will not concern ourselves here with the details of this debate, which is discussed in detail by Bad deley [21], but will indicate the evidence used by both sides as we go along.
- For our purposes, it is sufficient to note three separate types of memory.

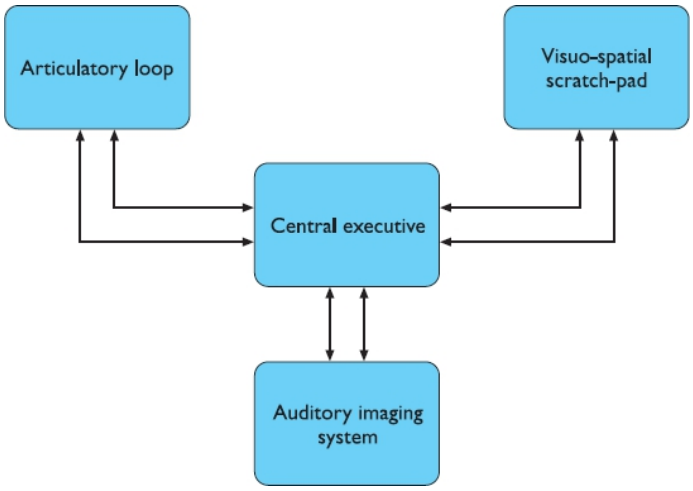


Sensory memory

- The sensory memories act as buffers for stimuli received through the senses. Asensory memory exists for each sensory channel: *iconic memory* for visual stimuli,*echoic memory* for aural stimuli and *haptic memory* for touch.
- These memories are constantly overwritten by new information coming in on these channels.We can demonstrate the existence of iconic memory by moving a finger in front of the eye.
- Can you see it in more than one place at once? This indicates a persistence of the image after the stimulus has been removed. A similar effect is noticed most vividly at firework displays where moving sparklers leave a persistent image. Information remains in iconic memory very briefly, in the order of 0.5 seconds.

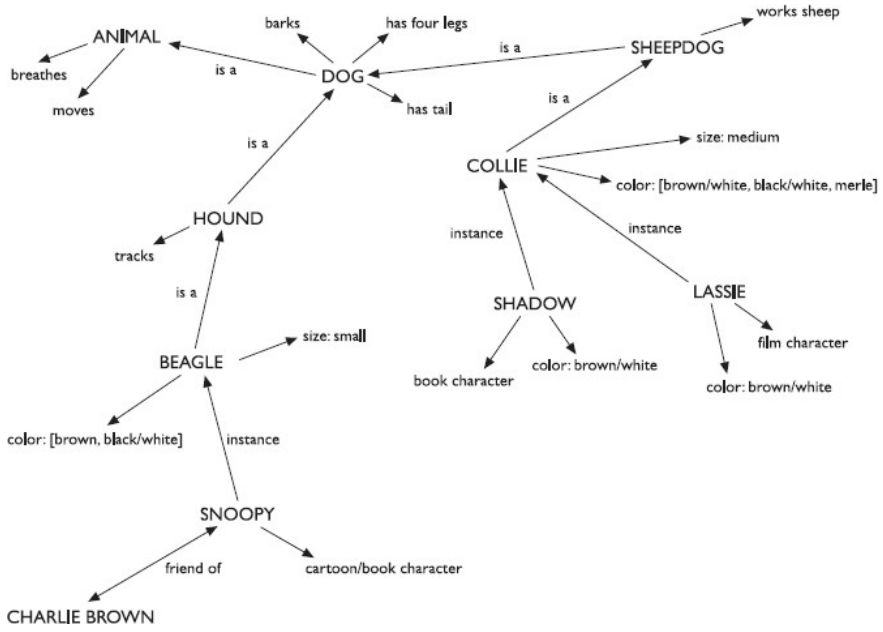
Short-term memory

- Short-term memory or working memory acts as a 'scratch-pad' for temporary recall of information.
- It is used to store information which is only required fleetingly. For example, calculate the multiplication 35 × 6 in your head. The chances are that you will have done this calculation in stages, perhaps 5 × 6 and then 30 × 6 and added the results; or you may have used the fact that 6 = 2 × 3 and calculated 2 × 35 = 70followed by 3 × 70.
- To perform calculations such as this we need to store the intermediate stages for use later. Or consider reading. In order to comprehend this sentence you need to hold in your mind the beginning of the sentence as you readthe rest. Both of these tasks use short-term memory.
- Short-term memory can be accessed rapidly, in the order of 70 ms. However, it also decays rapidly, meaning that information can only be held there temporarily, in the order of 200 ms.



Long-term memory

- If short-term memory is our working memory or 'scratch-pad', long-term memoryis our main resource. Here we store factual information, experiential knowledge,procedural rules of behavior – in fact, everything that we 'know'. It differs from short-term memory in a number of significant ways.
- First, it has a huge, if not unlimited, capacity. Secondly, it has a relatively slow access time of approximately a tenthof a second. Thirdly, forgetting occurs more slowly in long-term memory, if at all.
- These distinctions provide further evidence of a memory structure with several parts.Long-term memory is intended for the long-term storage of information.
- Information is placed there from working memory through rehearsal. Unlike working memory there is little decay:long-term recall after minutes is the same as that after hours or days.



Long-term memory structure

- There are two types of long-term memory: *episodic memory* and *semantic memory*.Episodic memory represents our memory of events and experiences in a serial form.
- It is from this memory that we can reconstruct the actual events that took place at a given point in our lives. Semantic memory, on the other hand, is a structured recordof facts, concepts and skills that we have acquired. The information in semantic memory is derived from that in our episodic memory, such that we can learn new facts or concepts from our experiences.

- A script represents this default or stereotypical information, allowing us to interpret partial descriptions or cues fully. A script comprises a number of elements, which, like slots, can be filled with appropriate information:
  - **Entry conditions** Conditions that must be satisfied for the script to be activated.
  - **Result** Conditions that will be true after the script is terminated.
  - **Props** Objects involved in the events described in the script.
  - **Roles** Actions performed by particular participants.
  - **Scenes** The sequences of events that occur.

- **Tracks** A variation on the general pattern representing an alternative scenario.

Long-term memory processes

- So much for the structure of memory, but what about the processes which it uses?There are three main activities related to long-term memory: storage or rememberingof information, forgetting and information retrieval.
- We shall consider each of these in turn.First, how does information get into long-term memory and how can we improvethis process? Information from short-term memory is stored in long-term memory by rehearsal. The repeated exposure to a stimulus or the rehearsal of a piece of information transfers it into long-term memory.
- This process can be optimized in a number of ways. Ebbinghaus performed numerous experiments on memory, using himself as a subject [117].
- In these experiment she tested his ability to learn and repeat nonsense syllables, comparing his recall minutes, hours and days after the learning process.
- He discovered that the amount learned was directly proportional to the amount of time spent learning.This is known as the *total time hypothesis*.

There are two main theories of forgetting:

- *Decay* and *interference*. The first theory suggests that the information held in long-term memory may eventually be forgotten.
- Ebbinghaus concluded from his experiments with nonsense syllables that information in memory decayed logarithmically ,that is that it was lost rapidly to begin with, and then more slowly.
- *Jost's law*, which follows from this, states that if two memory traces are equally strong at a given time the older one will be more durable.
- The second theory is that information is lost from memory through interference.If we acquire new information it causes the loss of old information. This is termed *retroactive interference*.

3. Explain in detail about Reasoning and problem

solving?OVERVIEW:

- √ **Reasoning**
- √ **Deductive Reasoning**
- √ **Inductive Reasoning**
- √ **Abductive Reasoning**
- √ **Problem Solving**
- √ **Problem Space Theory**
- √ **Errors And Mental Models**

- We have considered how information finds its way into and out of the human system and how it is stored. Finally, we come to look at how it is processed and manipulated.
- This is perhaps the area which is most complex and which separates humans from other information processing systems, both artificial and natural.Although it is clear that animals receive and store information, there is little evidence to suggest that they can use it in quite the same way as humans.

- Similarly, artificial intelligence has produced machines which can see (albeit in a limited way)and store information. But their ability to use that information is limited to small domains.
- Humans, on the other hand, are able to use information to reason and solve problems, and indeed do these activities when the information is partial or unavailable.Human thought is conscious and self-aware: while we may not always be able to identify the processes we use, we can identify the products of these processes, our thoughts.

Reasoning

- *Reasoning* is the process by which we use the knowledge we have to draw conclusionsor infer something new about the domain of interest.
- There are a number of differenttypes of reasoning: *deductive*, *inductive* and *abductive*. We use each of these typesof reasoning in everyday life, but they differ in significant ways.

Deductive reasoning

- Deductive reasoning derives the logically necessary conclusion from the given premises.
- For example,

  If it is Friday then she will go to work
  It is Friday
  Therefore she will go to work.
- It is important to note that this is the *logical* conclusion from the premises; it does not necessarily have to correspond to our notion of truth. So, for example,

  If it is raining then the ground is dry
  It is raining
  Therefore the ground is dry.

Inductive reasoning

- Induction is generalizing from cases we have seen to infer information about caseswe have not seen.
- For example, if every elephant we have ever seen has a trunk, weinfer that all elephants have trunks. Of course, this inference is unreliable and cannotbe proved to be true; it can only be proved to be false.
- We can disprove the inference simply by producing an elephant without a trunk. However, we can never prove ittrue because, no matter how many elephants with trunks we have seen or are known to exist, the next one we see may be trunkless.
- The best that we can do is gather evidenceto support our inductive inference.

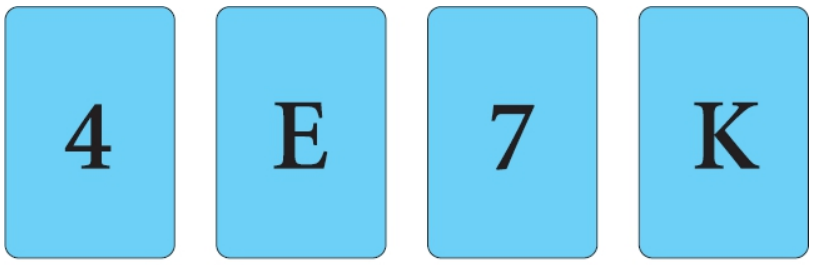


Figure 1.14 Wason's cards

Abductive reasoning

- The third type of reasoning is abduction. Abduction reasons from a fact to the action or state that caused it.
- This is the method we use to derive explanations for the events we observe. For example, suppose we know that Sam always drives too fast when she has been drinking.
- If we see Sam driving too fast we may infer that she has been drinking. Of course, this too is unreliable since there may be another reason why she is driving fast: she may have been called to an emergency, for example.
- In spite of its unreliability, it is clear that people do infer explanations in this way,and hold onto them until they have evidence to support an alternative theory or explanation.

Problem solving

- If reasoning is a means of inferring new information from what is already known,problem solving is the process of finding a solution to an unfamiliar task, using the knowledge we have.
- Human problem solving is characterized by the ability to adapt the information we have to deal with new situations. However, often solutions seem to be original and creative.
- There are a number of different views of how people solve problems.
- The earliest, dating back to the first half of the twentieth century, isthe *Gestalt* view that problem solving involves both reuse of knowledge and insight.
- This has been largely superseded but the questions it was trying to address remain and its influence can be seen in later research

Gestalt theory

- Gestalt psychologists were answering the claim, made by behaviorists, that problemsolving is a matter of reproducing known responses or trial and error.
- This explanation was considered by the Gestalt school to be insufficient to account for human problem-solving behavior. Instead, they claimed, problem solving is both *productive* and *reproductive*.
- Reproductive problem solving draws on previous experience as the behaviorists claimed, but productive problem solving involves insight and restructuring of the problem.
- Indeed, reproductive problem solving could be a hindrance to finding a solution, since a person may 'fixate' on the known aspects of the problem and so be unable to see novel interpretations that might lead to a solution.Gestalt psychologists backed up their claims with experimental evidence.
- Kohler provided evidence of apparent insight being demonstrated by apes, which he observed joining sticks together in order to reach food outside their cages [202].
- However, this was difficult to verify since the apes had once been wild and so could have been using previous knowledge.

Problem space theory

- Newell and Simon proposed that problem solving centers on the problem space.
- The problem space comprises *problem states*, and problem solving involves generating and a goal state and people use the operators to move from the former to the latter.
- Such problem spaces may be huge, and so *heuristics* are employed to select appropriate operators to reach the goal. One such heuristic is *means–ends analysis*.

Analogy in problem solving

- A third element of problem solving is the use of analogy. Here we are interested inhow people solve novel problems.

- One suggestion is that this is done by mapping knowledge relating to a similar known domain to the new problem – called *analogicalmapping*. Similarities betweenthe known domain and the new one are note dand operators from the known domain are transferred to the new one.

Skill acquisition

- The entire problem solving that we have considered so far has concentrated on handling unfamiliar problems.
- However, for much of the time, the problems that we face are not completely new. Instead, we gradually acquire skill in a particular domain area. But how is such skill acquired and what difference does it make to our problem-solving performance?
- We can gain insight into how skilled behavior works,and how skills are acquired, by considering the difference between novice and expert behavior in given domains.
- A commonly studied domain is chess playing. It is particularly suitable since it lends itself easily to representation in terms of problem space theory. The initial stateis the opening board position; the goal state is one player checkmating the other; operators to move states are legal moves of chess.
- It is therefore possible to examine skilled behavior within the context of the problem space theory of problem solving.Each of these differences stems from a better encoding of knowledge in the expert: information structures are fine tuned at a deep level to enable efficient and accurate retrieval.
- But how does this happen? How is skill such as this acquired? One modelof skill acquisition is Anderson's *ACT** model [14]. ACT* identifies three basic levels of skill:

  1. The learner uses general-purpose rules which interpret facts about a problem.
  This is slow and demanding on memory access.
  2. The learner develops rules specific to the task.
  3. The rules are tuned to speed up performance.

- These are best illustrated by example. Imagine you are learning to cook. Initially you may have a general rule to tell you how long a dish needs to be in the oven, and a number of explicit representations of dishes in memory. You can instantiate the rule by retrieving information from memory.

  IF cook[type, ingredients, time]
  THEN
  cook for: time
  cook[casserole, [chicken,carrots,potatoes], 2 hours]
  cook[casserole, [beef,dumplings,carrots], 2 hours]
  cook[cake, [flour,sugar,butter,eggs], 45 mins]
  Gradually your knowledge becomes proceduralized and you have specific rules for
  each case:
  IF type is casserole
  AND ingredients are [chicken,carrots,potatoes]
  THEN
  cook for: 2 hours
  IF type is casserole
  AND ingredients are [beef,dumplings,carrots]
  THEN
  cook for: 2 hours
  IF type is cake

AND ingredients are [flour,sugar,butter,eggs]
THEN
cook for: 45 mins
Finally, you may generalize from these rules to produce general-purpose rules, which
exploit their commonalities:
IF type is casserole
AND ingredients are ANYTHING
THEN
cook for: 2 hours

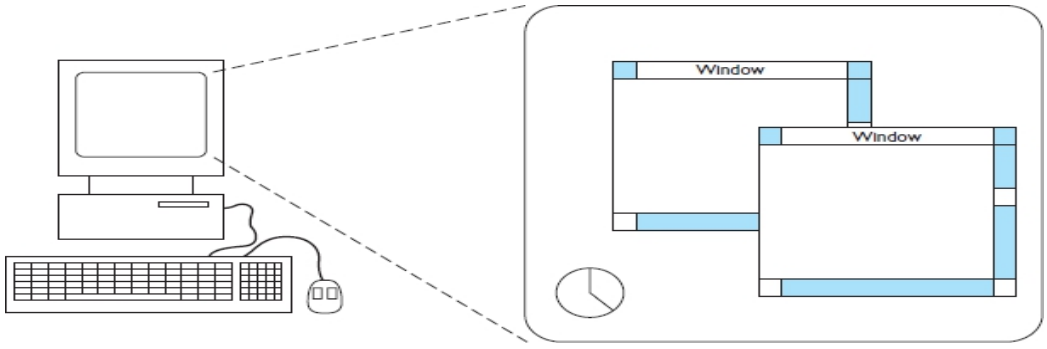
Errors and mental models

- Human capability for interpreting and manipulating information is quite impressive.However, we do make mistakes. Some are trivial, resulting in no more than temporary inconvenience or annoyance.
- Others may be more serious, requiring substantial effort to correct.
- Occasionally an error may have catastrophic effects, aswe see when 'human error' results in a plane crash or nuclear plant leak.Why do we make mistakes and can we avoid them? In order to answer the latter part of the question we must first look at what is going on when we make an error.
- There are several different types of error. As we saw in the last section some errors result from changes in the context of skilled behavior.
- If a pattern of behavior has become automatic and we change some aspect of it, the more familiar pattern may break through and cause an error.

4. Explain in detail about computer system?OVERVIEW:

- √ **Levels of interaction – batch processing**
- √ **Richer interaction – everywhere, everywhen**

- Consider a typical computer setup as shown in Figure 2.1. There is the computer 'box' itself, a keyboard, a mouse and a color screen.
- The screen layout is shown along side it. If we examine the interface, we can see how its various characteristics are related to the devices used. The details of the interface itself, its underlying principles and design, are discussed.
- As we shall see there arevariants on these basic devices. Some of this variation is driven by different hardware configurations: desktop use, laptop computers, PDAs (personal digital assistants).Partly the diversity of devices reflects the fact that there are many different types of comp.



Levels of interaction – batch processing

- In the early days of computing, information was entered into the computer in alarge mass – batch data entry. There was minimal interaction with the machine: the user would simply dump a pile of punched cards onto a reader, press the start button, and then return a few hours later.
- This still continues today although now with pre-prepared electronic files or possibly machine-read forms.
- It is clearly the most appropriate mode for certain kinds of application, for example printing paychecks or entering the results from a questionnaire. With batch processing the interactions take place over hours or days. In contrast the typical desktop computer system has interactions taking seconds or fractions ofa second (or with slow web pages sometimes minutes!).
- The field of Human–Computer Interaction largely grew due to this change in interactive pace. It is easy to assume that faster means better, but some of the paper-based technology discussed in Section 2.7 suggests that sometimes slower paced interaction may be better.

Richer interaction – everywhere, everywhen

- Computers are coming out of the box! Information appliances are putting internet access or dedicated systems onto the fridge, microwave and washing machine: to automate shopping, give you email in your kitchen or simply call for maintenance when needed.
- We carry with us WAP phones and smartcards, have security systems that monitor us and web cams that show our homes to the world

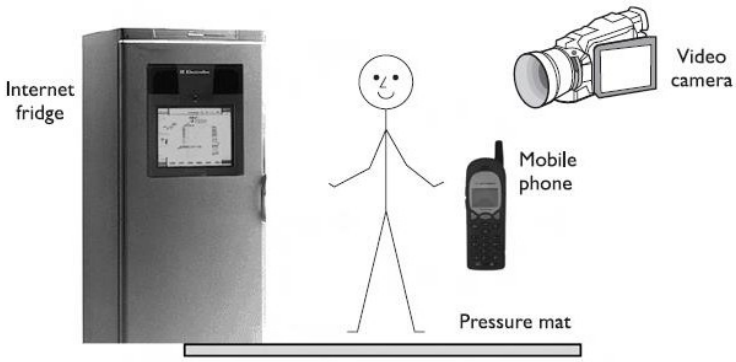


Figure 2.2 A typical computer system? Photo courtesy Electrolux

5. Describe in detail about different types of devices?'

OVERVIEW:

- √ **The alphanumeric keyboard**
- √ **The QWERTY keyboard**
- √ **Ease of learning – alphabetic keyboard**
- √ **Phone pad and T9 entry**
- √ **Chord keyboards**
- √ **Handwriting recognition**
- √ **Speechrecognition**

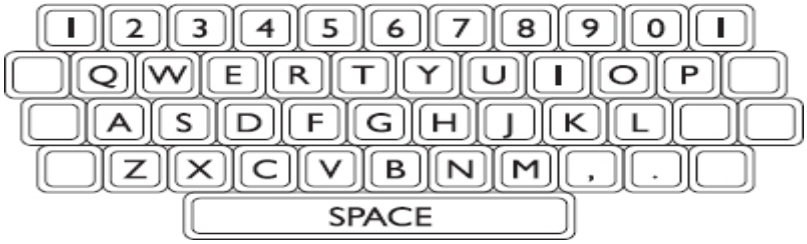
- ❖ Whether writing a book like this, producing an office memo, sending a thank you letter after your birthday, or simply sending an email to a friend, entering text isone of our main activities when using the computer.

The alphanumeric keyboard

- ❖ The keyboard is still one of the most common input devices in use today. It is used for entering textual data and commands.
- ❖ The vast majority of keyboards have a standardized layout, and are known by the first six letters of the top row of alphabetical keys, QWERTY. There are alternative designs which have some advantages over the QWERTY layout, but these have not been able to overcome the vast technological inertia of the QWERTY keyboard.

The QWERTY keyboard

- ❖ The layout of the digits and letters on a QWERTY keyboard is fixed (see Figure 2.3),but non-alphanumeric keys vary between keyboards.
- ❖ For example, there is a difference between key assignments on British and American keyboards (in particular,above the 3 on the UK keyboard is the pound sign £, whilst on the US keyboard there is a dollar sign $).



Ease of learning – alphabetic keyboard

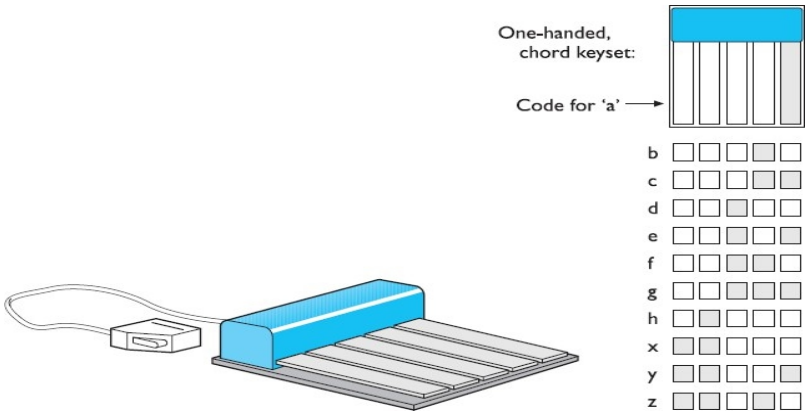
- One of the most obvious layouts to be produced is the alphabetic keyboard, in whichthe letters are arranged alphabetically across the keyboard.
- It might be expected thatsuch a layout would make it quicker for untrained typists to use, but this is not thecase. Studies have shown that this keyboard is not faster for properly trained typists, as we may expect, since there is no inherent advantage to this layout.

Ergonomics of use – DVORAK keyboard and split designs

- The DVORAK keyboard uses a similar layout of keys to the QWERTY system, but assigns the letters to different keys.
- Based upon an analysis of typing, the keyboard is designed to help people reach faster typing speeds. It is biased towards right-handed people, in that 56% of keystrokes are made with the right hand.

Chord keyboards

- Chord keyboards are significantly different from normal alphanumeric keyboards.Only a few keys, four or five, are used (see Figure 2.4) and letters are produced by pressing one or more of the keys at once.
- For example, in the *Microwriter*, the pattern of multiple key presses is chosen to reflect the actual letter shape.



Phone pad and T9 entry

- With mobile phones being used for SMS text messaging (see Chapter 19) and WAP(see Chapter 21), the phone keypad has become an important form of text input.

- Unfortunately a phone only has digits 0–9, not a full alphanumeric keyboard.To overcome this for text input the numeric keys are usually pressed several times a typical mapping of digits to letters.
- For example, the 3 key has‘ def ’ on it. If you press the key once you get a ‘d’, if you press 3 twice you get an ‘e’,if you press it three times you get an ‘f ’. The main number-to-letter mapping is standard,but punctuation and accented letters differ between phones.
- Also there needs to be a way for the phone to distinguish, say, the ‘dd’ from ‘e’. On some phones you need to pause for a short period between successive letters using the same key, for others you press an additional key (e.g. ‘#’).

Handwriting recognition

- Handwriting is a common and familiar activity, and is therefore attractive as a method of text entry.
- If we were able to write as we would when we use paper, butwith the computer taking this form of input and converting it to text, we can see tha it is an intuitive and simple way of interacting with the computer. However, there area number of disadvantages with handwriting recognition.
- Current technology is still fairly inaccurate and so makes a significant number of mistakes in recognizing letters, though it has improved rapidly.

Speech recognition

- Speech recognition is a promising area of text entry, but it has been promising for a number of years and is still only used in very limited situations.
- There is a natural enthusiasm for being able to talk to the machine and have it respond to commands,since this form of interaction is one with which we are very familiar.
- Successful recognition rates of over 97% have been reported, but since this represents one letter in error in approximately every 30, or one spelling mistake every six or so words,this is stoll unacceptable

6. Explain in detail about positioning, pointing and drawing devices?OVERVIEW:

- √ **The Mouse**
- √ **Touchpad**
- √ **Trackball And Thumbwheel**
- √ **Joystick And Keyboard Nipple**
- √ **Touch-Sensitive Screens (Touchscreens)**
- √ **Stylus And Light Pen**
- √ **Eyegaze**
- √ **Cursor Keys And Discrete Positioning**

The mouse

- The mouse has become a major component of the majority of desktop computer systems sold today, and is the little box with the tail connecting it to the machine in our basic computer system picture (Figure 2.1).
- It is a small, palm-sized box housing a weighted ball – as the box is moved over the tabletop, the ball is rolled by the table and so rotates inside the housing.
- This rotation is detected by small rollers that are in contact with the ball, and these adjust the values of potentiometers.
- If you remove the ball occasionally to clear dust you may be able to see these rollers.

Touchpad

- Touchpad are touch-sensitive tablets usually around 2–3 inches (50–75 mm) square.
- They were first used extensively in Apple Powerbook portable computers but are now used in many other notebook computers and can be obtained separately to replace the mouse on the desktop.
- They are operated by stroking a finger over their surface, rather like using a simulated trackball.
- The feel is very different from other input devices, but as with all devices users quickly get used to the action and become proficient.

Trackball and thumbwheel

- The trackball is really just an upside-down mouse!
- A weighted ball faces upwards andis rotated inside a static housing, the motion being detected in the same way as fora mechanical mouse, and the relative motion of the ball moves the cursor.
- Because of this, the trackball requires no additional space in which to operate, and is therefore a very compact device. It is an indirect device, and requires separate buttons for selection.

Joystick and keyboard nipple

- The joystick is an indirect input device, taking up very little space. Consisting of asmall palm-sized box with a stick or shaped grip sticking up from it, the joystick is asimple device with which movements of the stick cause a corresponding movement of the screen cursor. There are two types of joystick: the *absolute* and the *isometric*.
- In the absolute joystick, movement is the important characteristic, since the position of the joystick in the base corresponds to the position of the cursor on the screen. In the isometric joystick, the pressure on the stick corresponds to the velocity ofthe cursor, and when released, the stick returns to its usual upright centered position.
- This type of joystick is also called the velocity-controlled joystick, for obvious reasons.

Touch-sensitive screens (touchscreens)

- Touch screens are another method of allowing the user to point and select objectson the screen, but they are much more direct than the mouse, as they detect thepresence of the user's finger, or a stylus, on the screen itself.
- They work in one ofa number of different ways: by the finger (or stylus) interrupting a matrix of lightbeams, or by capacitance changes on a grid overlaying the screen, or by ultrasonicreflections.

- Because the user indicates exactly which item is required by pointing to it, no mapping is required and therefore this is a direct device.

Stylus and light pen

- For more accurate positioning (and to avoid greasy screens), systems with touchsensitivesurfaces often emply a stylus. Instead of pointing at the screen directly asmall pen-like plastic stick is used to point and draw on the screen.
- This is particularlypopular in PDAs, but they are also being used in some laptop computers.An older technology that is used in the same way is the light pen.
- The pen is connectedto the screen by a cable and, in operation, is held to the screen and detectsa burst of light from the screen phosphor during the display scan. The light pencan therefore address individual pixels and so is much more accurate than the touchscreen.

Digitizing tablet

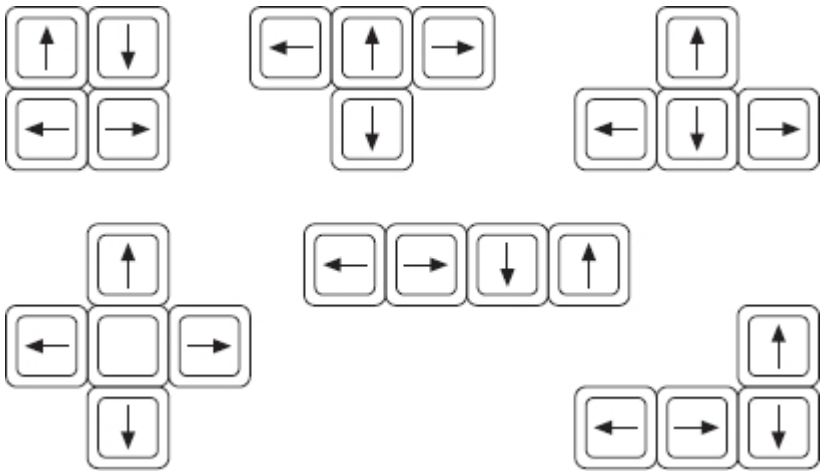
- The digitizing tablet is a more specialized device typically used for freehand drawing,but may also be used as a mouse substitute.
- Some highly accurate tablets, usually using a puck (a mouse-like device), are used in special applications such as digitizing information for maps.

Eyegaze

- Eyegaze systems allow you to control the computer by simply looking at it! Some systemsrequire you to wear special glasses or a small head-mounted box, others arebuilt into the screen or sit as a small box below the screen.
- A low-power laser is shoneinto the eye and is reflected off the retina. The reflection changes as the angle of theeye alters, and by tracking the reflected beam the eyegaze system can determine the direction in which the eye is looking.

Cursor keys and discrete positioning

- All of the devices we have discussed are capable of giving near continuous 2Dpositioning, with varying degrees of accuracy.
- For many applications we are only interested in positioning within a sequential list such as a menu oramongst 2D cellsas in a spreadsheet. Even for moving within text discrete up/down left/right keys cansometimes be preferable to using a mouse.



7. Desribe in detail about display devices?

OVERVIEW:

- √ **Bitmap displays – resolution and color**
- √ **Cathode ray tube**
- √ **Liquid crystal display**
- √ **Large displays and situated displays**
- √ **Digital paper**

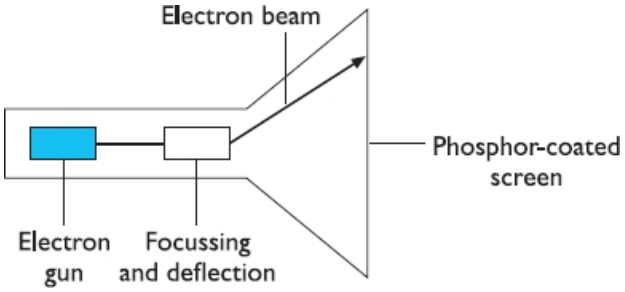
- ❖ The vast majority of interactive computer systems would be unthinkable withoutsome sort of display screen, but many such systems do exist, though usuallyin specialized applications only.
- ❖ Thinking beyond the traditional, systems such ascars, hi-fis and security alarms all have different outputs from those expressible on ascreen, but in the personal computer and workstation market, screens are pervasive.

Bitmap displays – resolution and color

- ❖ Virtually all computer displays are based on some sort of bitmap. That is the display is made of vast numbers of colored dots or pixels in a rectangular grid.
- ❖ These pixelsmay be limited to black and white (for example, the small display on many TVremote controls), in grayscale, or full color.
- ❖ As well as the number of colors that can be displayed at each pixel, the other measurethat is important is the resolution of the screen. Actually the word 'resolution' is usedin a confused (and confusing!) way for screens. There are two numbers to consider:
  - ➢ the ***total number*** of pixels: in standard computer displays this is always in a 4:3ratio, perhaps 1024 pixels across by 768 down, or 1600 × 1200; for PDAs this willbe more in the order of a few hundred pixels in each direction.
  - ➢ the ***density*** of pixels: this is measured in pixels per inch. Unlike printers (seeSection 2.7 below) this density varies little between 72 and 96 pixels per inch.

Cathode ray tube

- ❖ The cathode ray tube is the television-like computer screen still most common as we write this, but rapidly being displaced by flat LCD screens. It works in a similar way to a standard television screen.
- ❖ A stream of electrons is emitted from an electron gun, which is then focussed and directed by magnetic fields. As the beam hits the phosphor-coated screen, the phosphor is excited by the electrons and glows (seeFigure 2.10).
- ❖ The electron beam is scanned from left to right, and then flicked back to rescan the next line, from top to bottom. An alternative approach to producing color on the screen is to use *beam penetration*.A special phosphor glows a different color depending on the intensity of the beam hitting it.
- ❖ The CRT is a cheap display device and has fast enough response times for rapid animation coupled with a high color capability.
- ❖ Note that animation does not necessarily mean little creatures and figures running about on the screen, but refers ina more general sense to the use of motion in displays: moving the cursor, opening windows, indicating processor-intensive calculations, or whatever.



Liquid crystal display

- If you have used a personal organizer or notebook computer, you will have seen the light, flat plastic screens.
- These displays utilize liquid crystal technology and are smaller, lighter and consume far less power than traditional CRTs.
- These are also commonly referred to as flat-panel displays. They have no radiation problems associated with them, and are matrix addressable, which means that individual pixels canbe accessed without the need for scanning.Similar in principle to the digital watch, a thin layer of liquid crystal is sandwiched between two glass plates.
- The top plate is transparent and polarized, whilst the bottom plate is reflective.
- External light passes through the top plate and is polarized,which means that it only oscillates in one direction.

Large displays and situated displays

- Displays are no longer just things you have on your desktop or laptop.
- In Chapter 19we will discuss meeting room environments that often depend on large sharedscreens.
- You may have attended lectures where the slides are projected from a computeronto a large screen. In shops and garages large screen adverts assault us from all sides.
- There are several types of large screen display. Some use gas plasma technologyto create large flat bitmap displays.
- These behave just like a normal screen exceptthey are big and usually have the HDTV (high definition television) wide screenformat which has an aspect ratio of 16:9 instead of the 4:3 on traditional TV andmonitors.

Digital paper

- A new form of ‘display’ that is still in its infancy is the various forms of digital paper.
- These are thin flexible materials that can be written to electronically, just like a computer screen, but which keep their contents even when removed from any electrical supply.
- There are various technologies being investigated for this. One involves the wholesurface being covered with tiny spheres, black one side, white the other. Electronics embedded into the material allow each tiny sphere to be rotated to make it blackor white.
- When the electronic signa is removed the ball stays in its last orientation.
- A different technique has tiny tubes laid side by side. In each tube is light-absorbing liquid and a small reflective sphere.

- The sphere can be made to move to the top surface or away from it making the pixel white or black. Again the sphere stays in its last position once the electronic signal is removed.

8. Explain in detail about devices for virtual reality and 3d interaction?OVERVIEW:

- √ **Positioning In 3D Space**
- √ **Cockpit And Virtual Controls**
- √ **The 3D Mouse**
- √ **Dataglove**
- √ **Virtual Reality Helmets**
- √ **Whole-Body Tracking**
- √ **Seeing In 3D**
- √ **Simulators And VR Caves**

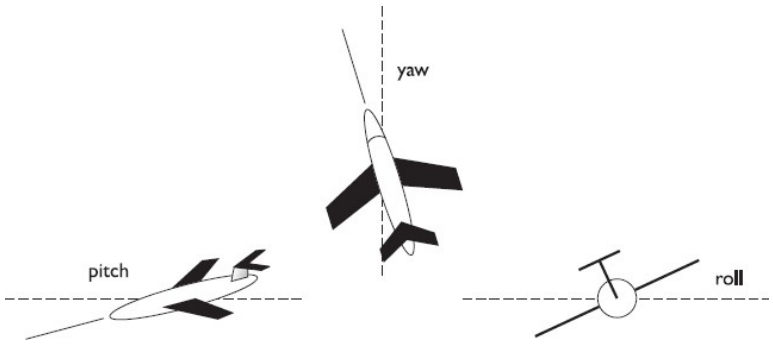
- Virtual reality (VR) systems and various forms of 3D visualization are discussed These require you to navigate and interact in a three-dimensionalspace.
- Sometimes these use the ordinary controls and displays of a desktop computer system, but there are also special devices used both to move and interact with 3D objects and to enable you to see a 3D environment.

Positioning in 3D space

- Virtual reality systems present a 3D virtual world. Users need to navigate through these spaces and manipulate the virtual objects they find there. Navigation is not simply a matter of moving to a particular location, but also of choosing a particular orientation.
- In addition, when you grab an object in real space, you don't simply move it around, but also twist and turn it, for example when opening a door.
- Thus the move from mice to 3D devices usually involvesa change from two degrees offreedom to six degrees of freedom, not just three.

Cockpit and virtual controls

- Helicopter and aircraft pilots already have to navigate in real space. Many arcade games and also more serious applications use controls modeled on an aircraft cockpit to 'fly' through virtual space.
- However, helicopter pilots are very skilled andit takes a lot of practice for users to be able to work easily in such environments.
- In many PC games and *desktop virtual reality* (where the output is shown onan ordinary computer screen), the controls are themselves virtual.
- This may be a simulated form of the cockpit controls or more prosaic up/down left/right buttons.
- The user manipulates these virtual controls using an ordinary mouse (or other 2Ddevice). Note that this means there are two levels of indirection. It is a tribute to the flexibility of the human mind that people can notonly use such systems but also rapidly become proficient.



The 3D mouse

- There are a variety of devices that act as 3D versions of a mouse. Rather than justmoving the mouse on a tabletop, you can pick it up, move it in three dimensions, left/right orientation (called *yaw*) and the amountit is twisted about its own axis (called *roll*) (see Figure 2.12).
- Various sensors are usedto track the mouse position and orientation: magnetic coils, ultrasound or even mechanical joints where the mouse is mounted rather like an angle-poise lamp.

Dataglove

- One of the mainstays of high-end VR systems (see Chapter 20), the dataglove is a 3Dinput device.
- Consisting of a lycra glove with optical fibers laid along the fingers, it detects the joint angles of the fingers and thumb.
- As the fingers are bent, the fiber optic cable bends too; increasing bend causes more light to leak from the fiber, and the reduction in intensity is detected by the glove and related to the degree of bend in the joint. Attached to the top of the glove are two sensors that use ultrasound to determine 3D positional information as well as the angle of roll, that is the degree ofwrist rotation.
- Such rich multi-dimensional input is currently a solution in search of a problem, in that most of the applications in use do not require such a comprehensive form of data input, whilst those that do cannot afford it. However, the availability f cheaper versions of the dataglove will encourage the development of more complex systems that are able to utilize the full power of the dataglove as an input device. There are a number of potential uses for this technology to assist disabled people, but cost remains the limiting factor at present.

Virtual reality helmets

- The helmets or goggles worn in some VR systems have two purposes: (i) they displayWe will discuss the former later when we consider output devices.
- The head trackingis used primarily to feed into the output side. As the user's head moves around theuser ought to see different parts of the scene. However, some systems also use the user's head direction to determine the direction of movement within the space and even which objects to manipulate (rather like the eye gaze systems).

- You can think of this rather like leading a horse in reverse. If you want a horse to go in a particular direction, you use the reins to pull its head in the desired direction and the horse follows its head.

Whole-body tracking

- Some VR systems aim to be immersive, that is to make the users feel as if they are really in the virtual world. In the real world it is possible (although not usually wise)to walk without looking in the direction you are going.
- If you are driving down the road and glance at something on the roadside you do not want the car to do a sudden 90-degree turn! Some VR systems therefore attempt to track different kinds of body movement.
- Some arcade games have a motorbike body on which you can lean into curves. More strangely, small trampolines have been wired up so that the user can control movement in virtual space by putting weight on different parts of the trampoline. The user can literally surf through virtual space.

3D displays

- Just as the 3D images used in VR have led to new forms of input device, they also require more sophisticated outputs.
- Desktop VR is delivered using a standard computer screen and a 3D impression is produced by using effects such as shadows,occlusion (where one object covers another) and perspective. This can be very effective and you can even view 3D images over the world wide web using a VRML(virtual reality markup language) enabled browser.

Seeing in 3D

- Our eyes use many cues to perceive depth in the real world (see also Chapter 1).
- It is in fact quite remarkable as each eye sees only a flattened form of the world, like a photograph. One important effect is *stereoscopic vision* (or simply *stereo vision*).
- Different techniques are then used to ensure that each eye sees the appropriateimage. One method is to have two small screens fitted to a pair of goggles. A different image is then shown to each eye.
- These devices are currently still quite cumber someand the popular image of VR is of a user with head encased in a helmet with something like a pair of inverted binoculars sticking out in front.
- However, smaller and lighter LCDs are now making it possible to reduce the devices towards the size and weight of ordinary spectacles.The ideal would be to be able to look at a special 3D screen and see 3D images just as one does with a hologram – 3D television just like in all the best sci-fi moviesBut there is no good solution to this yet.
- One method is to inscribe the screen withsmall vertical grooves forming hundreds of prisms.
- Each eye then sees only alternate dots on the screen allowing a stereo image at half the normal horizontal resolution.However, these screens have very narrow *viewing angles*, and are not ready yet for family viewing.

VR motion sickness

- We all get annoyed when computers take a long time to change the screen, pop up a window, or play a digital movie.

- However, with VR the effects of poor display performance can be more serious. In real life when we move our head the image our eyes see changes accordingly. VR systems produce the same effect by using sensors in the goggles or helmet and then using the position of the head to determine the right image to show.
- If the system is slow in producing these images a lag develops between the user moving his head and the scene changing. If this delay is more than a hundred milliseconds or so the feeling becomes disorienting.

Simulators and VR caves

- Because of the problems of delivering a full 3D environment via head-mounteddisplays, some virtual reality systems work by putting the user within an environmentwhere the virtual world is displayed upon it.
- The most obvious examples of thisare large flight simulators – you go inside a mock-up of an aircraft cockpit and thescenes you would see through the windows are projected onto the virtual windows.
- In motorbike or skiing simulators in video arcades large screens are positioned to fillthe main part of your visual field.
- You can still look over your shoulder and see your friends, but while you are engaged in the game it surrounds you.

9. Explain in detail about physical controls, sensors and special devices?OVERVIEW:

√ **Special displays**
√ **Sound output**
√ **Touch, feel and smell**
√ **Physical controls**
√ **Environment and bio-sensing**

- As we have discussed, computers are coming out of the box. The mouse keyboard and screen of the traditional computer system are not relevant or possible in applications that now employ computers such as interactive TV, in-car navigation systems or personal entertainment. These devices may have special displays, may use sound, touch and smell as well as visual displays, may have dedicated controls and may sense the environment or your own bio-signs.

Special displays

- Apart from the CRT screen there are a number of visual outputs utilized in complex systems, especially in embedded systems. These can take the form of analog representations of numerical values, such as dials, gauges or lights to signify a certain system state.

- Flashing light-emitting diodes (LEDs) are used on the back of some computers to signify the processor state, whilst gauges and dials are found in systems. Once you start in this mode of thinking, you can contemplate numerous visual outputs that are unrelated to the screen. One visual display that has found a specialized niche is the head-up display that is used in aircraft.
- The pilot is fully occupied looking forward and finds it difficult to look around the cockpit to get information. There are many different things that need to be known, ranging from data from tactical systems to navigational information and aircraft status indicators.

Sound output

- Another mode of output that we should consider is that of auditory signals. Often designed to be used in conjunction with screen displays, auditory outputs are poorly understood: we do not yet know how to utilize sound in a sensible way to achieve maximum effect and information transference.
- We have discussed speech previously,but other sounds such as beeps, bongs, clanks, whistles and whirrs are all used to varying effect. As well as conveying system output, sounds offer an important level of feedback in interactive systems.
- Keyboards can be set to emit a click each time a key is pressed, and this appears to speed up interactive performance. Telephone keypads often sound different tones when the keys are pressed; a noise occurring signifies that the key has been successfully pressed, whilst the actual tone provides some information about the particular key that was pressed.

Touch, feel and smell

- Our other senses are used less in normal computer applications, but you may have played computer games where the joystick or artificial steering wheel vibrated, perhaps swhen a car was about to go off the track.
- In some VR applications, such as theuse in medical domains to 'practice' surgical procedures, the *feel* of an instrument moving through different tissue types is very important.
- The devices used to emulate these procedures have *force feedback*, giving different amounts of resistance depending on the state of the virtual operation. These various forms of force, resistance and texture that influence our physical senses are called *haptic* devices.

Physical controls

- Look at Figure 2.13. In it you can see the controls for a microwave, a washing machine and a personal MiniDisc player.
- See how they each use very different physical devices: the microwave has a flat plastic sheet with soft buttons, the washing machine large switches and knobs, and the MiniDisc has small buttons and an interesting multi-function end.
- A desktop computer system has to serve many functions and so has generic key sand controls that can be used for a variety of purposes. In contrast, these dedicated control panels have been designed for a particular device and for a single use.
- This is why they differ so much. Looking first at the microwave, it has a flat plastic control panel. The buttons onthe panel are pressed and 'give' slightly.
- The choice of the smooth panel is probably partly for visual design – it looks streamlined! However, there are also good practical reasons. The microwave is used in the kitchen whilst cooking, with hands that may be greasy or have food on them.

Environment and bio-sensing

- In a public washroom there are often no controls for the wash basins, you simply put your hands underneath and (hope that) the water flows. Similarly when you open the door of a car, the courtesy light turns on. The washbasin is controlled by a small infrared sensor that is triggered when your hands are in the basin (although it is sometimes hard to find the 'sweet spot' where this happens!).
- The courtesy lights are riggered by a small switch in the car door.Although we are not always conscious of them, there are many sensors in ourenvironment – controlling automatic doors, energy saving lights, etc. and devices monitoring our behavior such as security tags in shops.
- The vision of ubiquitous computing (see Chapters 4 and 20) suggests that our world will be filled with such devices. Certainly the gap between science fiction and day-to-day life is narrow;for example, in the film *Minority Report* (20th Century Fox) iris scanners identify each passer-by to feed them dedicated advertisements, but you can buy just such an iris scanner as a security add-on for your home computer.

10. Discuss in detail about paper: printing and scanning?

OVERVIEW:

- √ **Printing**
- √ **Fonts and page description languages**
- √ **Screen and page**
- √ **Scanners and optical character recognition**

- Some years ago, a recurrent theme of information technology was the *paperless office*.
- In the paperless office, documents would be produced, dispatched, read and filed online.
- The only time electronic information would be committed to paper would be when it went out of the office to ordinary customers, or to other firms who were laggards in this technological race.
- This vision was fuelled by rocketing property prices,and the realization that the floor space for a wastepaper basket could cost thousands in rent each year. Some years on, many traditional paper files are now online, but the desire for the completely paperless office has faded. Offices still have wastepaper baskets,and extra floor space is needed for the special computer tables to house 14-inch color monitors.

Printing

- If anything, computer systems have made it easier to produce paper documents. It is so easy to run off many copies of a letter (or book), in order to get it looking 'justright'.
- Older printers had a fixed set of characters available on a print head. These varied from the traditional line printer to golf-ball and daisy-wheel printers.
- To change a typeface or the size of type meant changing the print head, and was an awkward,and frequently messy, job, but for many years the daisy-wheel printer was the only means of producing high-quality output at an affordable price. However, the drop inthe price of laser printers coupled with the availability of other cheap high-quality printers means that daisy-wheels are fast becoming a rarity.

- All of the popular printing technologies, like screens, build the image on the paper as a series of dots. This enables, in theory, any character set or graphic to be printed, limited only by the resolution of the dots. This resolution is measured in *dots per inch*(dpi).
- Imagine a sheet of graph paper, and building up an image by putting dots at the intersection of each line.
- The number of lines per inch in each direction is the resolution in dpi. For some mechanical printers this is slightly confused: the dots printed may be bigger than the gaps, neighboring print heads may not be able to print simultaneously and may be offset relative to one another (a diamond-shaped rather than rectangular grid).
- These differences do not make too much difference tothe user, but mean that, given two printers at the same nominal resolution, the outputof one looks better than that of the other, because it has managed the physical constraints better.
- The most common types of dot-based printers are dot-matrix printers, ink-jet printers and laser printers. These are listed roughly in order of increasing resolutionand quality, where dot-matrix printers typically have a resolution of 80–120 dpi risingto about 300–600 dpi for ink-jet printers and 600–2400 dpi for laser printers.
- By varying the quantity of ink and quality of paper, ink-jet printers can be used to print photo-quality prints from digital photographs.

Fonts and page description languages

- Some printers can act in a mode whereby any characters sent to them (encoded in ASCII, see Section 2.8.5) are printed, typewriter style, in a single font. Another case,simple in theory, is when you have a bitmap picture and want to print it.
- The dotsto print are sent to the printer, and no further interpretation is needed. However, in practice, it is rarely so simple.Many printed documents are far more complex – they incorporate text in many different fonts and many sizes, often italicized, emboldened and underlined.
- Within the text you will find line drawings, digitized photographs and pictures generated from ‘paint’ packages, including the ubiquitous ‘clip art’. Sometimes the computer does all the work, converting the page image into a bitmap of the right size to be sent to the printer. Alternatively, a description of the page may be sent to the printer.
- At the simplest level, this will include commands to set the print position on the page, and change the font size.
- More sophisticated printers can accept a *page description language*, the most common of which is PostScript. This is a form of programming language for printing.
- It includes some standard programming constructs, but also some special ones: paths for drawing lines and curves, sophisticated character and font handling and scaled bitmaps.
- The idea is that the description of a page is far smaller than the associated bitmap, reducing the time taken to send the page to the printer.
- A bitmap version of an A4 laser printer page at 300 dpi takes 8 Mbytes; to send this down a standard serial printer cable would take 10 minutes! However, a computer in the printer hasto interpret the PostScript program to print the page; this is typically faster than 10minutes, but is still the limiting factor for many print jobs.Text is printed in a font with a particular size and shape. The size of a font ismeasured in points (pt). The point is a printer’s measure and is about 1/72 of an inch.
- The *point size* of the font is related to its height: a 12 point font has aboutsix lines per inch.
- The shape of a font is determined by its *font name*, for example Times Roman, Courier or Helvetica. Times Roman font is similar to the type of many newspapers, such as *The Times*, whereas Courier has a typewritten shape.

Screen and page

- A common requirement of word processors and desktop publishing software is that *what you see is what you get* (see also Chapters 4 and 17), which is often called by its acronym *WYSIWYG* (pronounced whizz-ee-wig).
- This means that the appearance of the document on the screen should be the same as its eventual appearance on the printed page. In so far as this means that, for example, centered text is displayed centered on the screen, this is reasonable.
- However, this should not cloud the fact that screen and paper are very different media.A typical screen resolution is about 72 dpi compared with a laser printer at over600 dpi. Some packages can show magnified versions of the document in order to help in this.
- Most screens use an additive color model using red, green and blue light,whereas printers use a subtractive color model with cyan, magenta, yellow and black inks, so conversions have to be made. In addition, the sizes and aspect ratios are ver different.
- An A4 page is about 11 inches tall by 8 wide (297 □□ 210 mm), whereas a screen is often of similar dimensions, but wider than it is tall.

Scanners and optical character recognition

- Printers take electronic documents and put them on paper – *scanners* reverse this process. They start by turning the image into a bitmap, but with the aid of *optical character recognition* can convert the page right back into text.
- The image to be converted may be printed, but may also be a photograph or hand-drawn picture.
- There are two main kinds of scanner: flat-bed and hand-held. With a flat-bed scanner, the page is placed on a flat glass plate and the whole page is converted into a bitmap.
- A variant of the flat-bed is where sheets to be scanned are pulled through the machine, common in multi-function devices (printer/fax/copier). Many flat-bed scanners allow a small pile of sheets to be placed in a feed tray so that they can all be scanned without user intervention.
- Hand-held scanners are pulled over the image by hand. As the head passes over an area it is read in, yielding a bitmap strip. A roller at the ends ensures that the scanner knows how fast it is being pulled and thus how big the image is. The scanner is typically only 3 or 4 inches(80 or 100 mm) wide and may even be the size of a large pen (mainly used for scanning individual lines of text).
- This means at least two or three strips must be‘glued’ together by software to make a whole page image, quite a difficult process as the strips will overlap and may not be completely parallel to one another, as well as suffering from problems of different brightness and contrast. However,for desktop publishing small images such as photographs are quite common, and as long as one direction is less than the width of the scanner, they can be read in one pass.

12. Explain in detail about the memory and types?

OVERVIEW:

- √ **RAM and short-term memory (STM)**
- √ **Disks and long-term memory (LTM)**
- √ **Understanding speed and capacity**

- √ **Compression**
- √ **Storage format and standards**
- √ **Methods of access**

- ❖ Like human memory, we can think of the computer's memory as operating at different levels, with those that have the faster access typically having less capacity.
- ❖ By analogy with the human memory, we can group these into short-term and long-term memories (STM and LTM), but the analogy is rather weak – the capacity of the computer's STM is a lot more than seven items! The different levels of computer memory are more commonly called primary and secondary storage.

RAM and short-term memory (STM)

- ❖ At the lowest level of computer memory are the registers on the computer chip, butthese have little impact on the user except in so far as they affect the general speed of the computer.
- ❖ Most currently active information is held in silicon-chip *randomaccess memory* (*RAM*). Different forms of RAM differ as to their precise access times, power consumption and characteristics.
- ❖ Typical access times are of the order of10 nanoseconds, that is a hundred-millionth of a second, and information can be accessed at a rate of around 100 Mbytes (million bytes) per second.
- ❖ Typical storagein modern personal computers is between 64 and 256 Mbytes.Most RAM is *volatile*, that is its contents are lost when the power is turned off.However, many computers have small amount of *non-volatile RAM*, which retains its contents, perhaps with the aid of a small battery. This may be used to store setup information in a large computer, but in a pocket organizer will be the whole memory.
- ❖ Non-volatile RAM is more expensive so is only used where necessary, but with many notebook computers using very low-power static RAM, the divide is shrinking.
- ❖ By strict analogy, non-volatile RAM ought to be classed as LTM, but the important thing we want to emphasize is the gulf between STM and LTM in a traditional computer system.

Disks and long-term memory (LTM)

- ❖ For most computer users the LTM consists of *disks*, possibly with small tapes for *backup*.
- ❖ The existence of backups, and appropriate software to generate and retrieve them, is an important area for user security. However, we will deal mainly with those forms of storage that impact the interactive computer user.
- ❖ There are two main kinds of technology used in disks: *magnetic disks* and *opticaldisks*. The most common storage media, floppy disks and hard (or fixed) disks,are coated with magnetic material, like that found on an audio tape, on which the information is stored.
- ❖ Typical capacities of floppy disks lie between 300 kbytes and1.4 Mbytes, but as they are removable, you can have as many as you have room foron your desk. Hard disks may store from under 40 Mbytes to several gigabytes (Gbytes), that is several thousand million bytes. With disks there are two access times to consider, the time taken to find the right track on the disk, and the time to read the track.
- ❖ The former dominates random reads, and is typically of the order of 10 msfor hard disks. The transfer rate once the track is found is then very high, perhapsseveral hundred kilobytes per second.

- Various forms of large removable media are also available, fitting somewhere between floppy disks and removable hard disks, and are especially important for multimedia storage.

Understanding speed and capacity

- So what effect do the various capacities and speeds have on the user? Thinking of our typical personal computer system, we can summarize some typical capacities as in We think first of documents.
- This book is about 320,000 words, or about 2Mbytes, so it would hardly make a dent in 256 Mbytes of RAM. (This size – 2 Mbytes– is unformatted and without illustrations; the actual size of the full data files is an order of magnitude bigger, but still well within the capacity of main memory.) To take a more popular work, the Bible would use about 4.5 Mbytes.
- This would still consume only 2% of main memory, and disappear on a hard disk. However, it might look tight on a smaller PDA.
- This makes the memory look not too bad, so long asyou do not intend to put your entire library online. However, many word processor come with a dictionary and thesaurus, and there is no standard way to use the same one with several products.

	STM small/fast	LTM large/slower
Media:	RAM	Hard disk
Capacity:	256 Mbytes	100 Gbytes
Access time:	10 ns	7 ms
Transfer rate:	100 Mbyte/s	30 Mbyte/s

Compression

- In fact, things are not quite so bad, since *compression* techniques can be used to reduce the amount of storage required for text, bitmaps and video.
- All of these things are highly redundant. Consider text for a moment. In English, we know that if we use the letter ‘q’ then ‘u’ is almost bound to follow.
- At the level of words, some words like ‘the’ and ‘and’ appear frequently in text in general, and for any particular workone can find other common terms (this book mentions ‘user’ and ‘computer’ rather frequently). Similarly, in a bitmap, if one bit is white, there is a good chance the next will be as well.
- Compression algorithms take advantage of this redundancy. For example, *Huffman encoding* gives short codes to frequent words [182], and *run lengthen coding* represents long runs of the same value by length value pairs. Textcan easily be reduced by a factor of five and bitmaps often compress to 1% of their original size.
- For video, in addition to compressing each frame, we can take advantage of thefact that successive frames are often similar.
- We can compute the *difference* between successive frames and then store only this – compressed, of course. More sophisticated algorithms detect when the camera pans and use this information also.
- These differencing methods fail when the scene changes, and so the process periodically has to restart and send a new, complete (but compressed) image. For storage purposes

this is not a problem, but when used for transmission over telephone lines or networks it can mean glitches in the video as the system catches up.

Storage format and standards

- The most common data types stored by interactive programs are text and bitmap images, with increasing use of video and audio, and this subsection looks at the ridiculous range of file storage standards.
- We will consider database retrieval in the next subsection.The basic standard for text storage is the *ASCII* (American standard code for information interchange) character codes, which assign to each standard printable character and several control characters an internationally recognized 7 bit code (decimal values 0–127), which can therefore be stored in an 8 bit byte, or be transmitted as 8 bits including parity.
- Many systems extend the codes to the values 128–255, including line-drawing characters, mathematical symbols and international letters such as 'æ'. There is a 16 bit extension, the UNICODE standard, which has enough room for a much larger range of characters including the Japanese Kanji character set.

Methods of access

- Standard database access is by special key fields with an associated index.
- The userhas to know the key before the system can find the information. A telephone directory is a good example of this. You can find out someone's telephone number if you know their name (the key), but you cannot find the name given the number. This is evident in the interface of many computer systems. So often, when you contact an organization, they can only help you if you give your customer number, or last order number.
- The usability of the system is seriously impaired by a shortsighted reliance on a single key and index. In fact, most database systems will allow multiple keys and indices, allowing you to find a record given partial information. So these problems are avoidable with only slight foresight.
- There are valid reasons for not indexing on too many items. Adding extra indices adds to the size of the database, so one has to balance ease of use against storage cost. However, with ever-increasing disk sizes, this is not a good excuse for all but extreme examples.
- Unfortunately, brought up on lectures about algorithmic efficiency, it is easy for computer scientists to be stingy with storage. Another, more valid, reason for restricting the fields you index is privacy and security. For example, telephone companies will typically hold an online index that, given a telephone number, would return the name and address of the subscriber, but to protect the privacy of their customers, this information is not divulged to the general public.

13. Describe in detail about processing and networks?OVERVIEW:

- √ **Effects Of Finite Processor Speed**
- √ **Limitations On Interactive Performance**
- √ **Computation Bound**
- √ **Storage Channel Bound**
- √ **Graphics Bound**
- √ **Networked Computing**

- Computers that run interactive programs will process in the order of 100 million instructions per second. It sounds a lot and yet, like memory, it can soon be used up.
- Indeed, the first program written by one of the authors (some while ago) 'hung' and all attempts to debug it failed.
- Later calculation showed that the program would have taken more than the known age of the universe to complete! Failures need not be as spectacular as that to render a system unusable. Consider, for example, one drawing system known to the authors.
- To draw a line you press down the mouse button at one end, drag the mouse and then release the mouse button at the other end of the line – but not too quickly.
- You have to press down the button and then actually hold your hand steady for a moment, otherwise the line starts half way! For activities involving the user's hand–eye coordination, delays of even a fraction of a second can be disastrous.

Effects of finite processor speed

- As we can see, speed of processing can seriously affect the user interface. These effects must be taken into account when designing an interactive system.
- There are two sorts of faults due to processing speed: those when it is too slow, and those when it is too fast!
- We saw one example of the former above. This was a *functional fault*, in that the program did the wrong thing.
- The system is supposed to draw lines from where the mouse button is depressed to where it is released.
- However, the program gets it wrong – after realizing the button is down, it does not check the position of the mouse fast enough, and so the user may have moved the mouse before the start position is registered.
- This is a fault at the implementation stage of the system rather\ than of the design. But to be fair, the programmer may not be given the right sort of information from lower levels of system software.
- A second fault due to slow processing is where, in a sense, the program does the right thing, but the feedback is too slow, leading to strange effects at the interface.
- In order to avoid faults of the first kind, the system *buffers* the user input; that is, it\ remembers keypresses and mouse buttons and movement. Unfortunately, this leads to problems of its own.
- One example of this sort of problem is *cursor tracking*, which happens in character-based text editors.
- The user is trying to move backwards on the same line to correct an error, and so presses the cursor-left key. The cursor moves and when it is over the correct position, the user releases the key. Unfortunately, the system is behind in responding to the user, and so has a few more cursor-left keys

Limitations on interactive performance

- There are several factors that can limit the speed of an interactive system:

Computation bound

  - This is rare for an interactive program, but possible, for example when using find/replace in a large document. The system should be designed so that long delays

are not in the middle of interaction and so that the user gets some idea of how the job is progressing.

- For a very long process try to give an indication of duration *before* it starts; and during processing an indication of the stage that the process has reached is helpful.
- This can be achieved by havinga counter or slowly filling bar on the screen that indicates the amount done,or by changing the cursor to indicate that processing is occurring. Many systems notice after they have been computing for some time and then say 'this may take some time: continue (Y/N)?'. Of course, by the time it says this the process maybe nearly finished anyway!

Storage channel bound

- As we discussed in the previous section, the speed of memory access can interfere with interactive performance.
- We discussed one technique, laziness, for reducing this effect. In addition, if there is plenty of raw computation power and the system is held up solely by memory, it is possible to trade off memory against processing speed.
- For example, compressed data take less space to store, and is faster to read in and out, but must be compressed before storage and decompressed when retrieved.
- Thus faster memory access leads to increased processing time. If data is written more often than it is read, one can choose a technique that is expensive to compress but fairly simple to decompress.
- For many interactive systems the ability to browse quickly is very important, but users will accept delays when saving updated information.

Graphics bound

- For many modern interfaces, this is the most common bottleneck.It is easy to underestimate the time taken to perform what appear to be simple interface operations.
- Sometimes clever coding can reduce the time takenby common graphics operations, and there is tremendous variability in performance nbetween programs running on the same hardware. Most computers include a special-purpose *graphics card* to handle many of the most commongraphics operations.
- This is optimized for graphics operations and allows themain processor to do other work such as manipulating documents and other user data.

Networked computing

- Computer systems in use today are much more powerful than they were a few years ago, which means that the standard computer on the desktop is quite capable of high-performance interaction without recourse to outside help.
- However, it is often the case that we use computers not in their standalone mode of operation, but between different parties, provided they are connected into the same network, as wellas allowing the desktop computer to access resources remote from itself.
- Such networks are inherently much more powerful than the individual computers that makeup the network: increased computing power and memory are only part of the story,since the effects of allowing people much more extensive, faster and easier access to information are highly significant to individuals, groups and institutions.Another effect is that the interaction between human and machine becomesan open loop, rather than a closed one.
- Many people may be interacting with themachine at once, and their actions may affect the response to your own.
- Many users accessing a single central machine will slow its response; database updates carried out by one user may mean that the same query by another user at slightly different

times may produce different results. The networked computer system, by the very natureof its dispersal, distribution and multi-user access, has been transformed from a fully predictable, deterministic system, under the total control of the user, into a non deterministic one, with an individual user being unaware of many important things that are happening to the system as a whole.

- Such systems pose a particular problem since ideals of consistency, informative feedback and predictable response are violated.

14. Desribe in detail about models and interaction?OVERVIEW:

- √ **The terms of interaction**
- √ **The interaction framework**
- √ **Assessing overall interaction**

- Interaction involves at least two participants:**the user and the system**. Both are complex, as we have seen, and are very different from each other in the way that they communicate and view the domain and the task.
- The interface must therefore effectively translate between them to allow the interaction to be successful.
- This translation can fail ata number of points and for a number of reasons. The use of models of interaction can help us to understand exactly what is going on in the interaction and identify the likely root of difficulties. They also provide us with a framework to compare different interaction styles and to consider interaction problems.
- We begin by considering the most influential model of interaction, Norman's *execution–evaluation cycle*; then we look at another model which extends the ideas of Norman's cycle. Both of these models describe the interaction in terms of the goals and actions of the user. We will therefore briefly discuss the terminology used and the assumptions inherent in the models, before describing the models themselves.

The terms of interaction

- Traditionally, the purpose of an interactive system is to aid a user in accomplishing *goals* from some application *domain*. (Later in this book we will look at alternative interactions but this model holds for many work-oriented applications.)
- A domain defines an area of expertise and knowledge in some real-world activity. Some examples of domains are graphic design, authoring and process control in a factory.A domain consists of concepts that highlight its important aspects.
- In a graphic design domain, some of the important concepts are geometric shapes, a drawing surface and a drawing utensil. *Tasks* are operations to manipulate the concepts of a domain.
- A *goal* is the desired output from a performed task.
- For example, one task within the graphic design domain is the construction of a specific geometric shape with particular attributes on the drawing surface. A related goal would be to produce a solid red triangle centered on the canvas. An *intention* is a specific action required to meet the goal.
- *Task analysis* involves the identification of the problem space (which we discussed in Chapter 1) for the user of an interactive system in terms of the domain, goals,intentions and tasks.

- We can use our knowledge of tasks and goals to assess the interactive system that is designed to support them.
- We discuss task analysis in detail in Chapter 15. The concepts used in the design of the system and the description of the user are separate, and so we can refer to them as distinct components, called the *System* and the *User*, respectively.
- The *System* and *User* are each described by means of a language that can express concepts relevant in the domain of the application.
- The *System*'s language we will refer to as the *core language* and the *User*'s language we will refer to as the *task language*.
- The core language describes computational attributes of the domain relevant to the *System* state, whereas the task language describes psychological attributes of the domain relevant to the *User* state.
- The system is assumed to be some computerized application, in the context of this book, but the models apply equally to non-computer applications.
- It is also a common assumption that by distinguishing between user and system we are restricted to
- single-user applications. This is not the case. However, the emphasis is on the viewof the interaction from a single user's perspective. From this point of view, other users, such as those in a multi-partyconferencing system, form part of the system.
- The execution–evaluation cycle
- Norman's model of interaction is perhaps the most influential in Human–ComputerInteraction, possibly because of its closeness to our intuitive understanding of theinteraction between human user and computer [265]. The user formulates a plan ofaction, which is then executed at the computer interface. When the plan, or part ofthe plan, has been executed, the user observes the computer interface to evaluate theresult of the executed plan, and to determine further actions.The interactive cycle can be divided into two major phases: execution and evaluation.
- These can then be subdivided into further stages, seven in all. The stages inNorman's model of interaction are as follows:

    1.Establishing the goal.
    2. Forming the intention.
    3. Specifying the action sequence.
    4. Executing the action.
    5. Perceiving the system state.
    6. Interpreting the system state.
    7. Evaluating the system state with respect to the goals and intentions.

- Each stage is, of course, an activity of the user. First the user forms a goal. This is the user's notion of what needs to be done and is framed in terms of the domain, in the task language.
- It is liable to be imprecise and therefore needs to be translated in to the more specific intention, and the actual actions that will reach the goal, before it can be executed by the user.
- The user perceives the new state of the system, after execution of the action sequence, and interprets it in terms of his expectations. If the system state reflects the user's goal then the computer has done what he wanted and
- the interaction has been successful; otherwise the user must formulate a new goal and repeat the cycle.

The interaction framework

- The interaction framework attempts a more realistic description of interaction by including the system explicitly, and breaks it into four main components, as shown in Figure 3.1.
- The nodes represent the four major components in an interactive system– the *System*, the *User*, the *Input* and the *Output*. Each component has its own language.

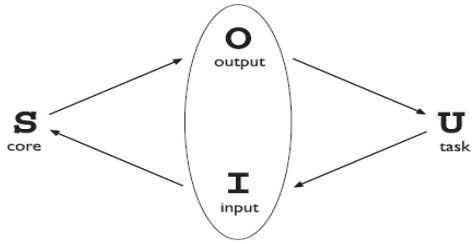
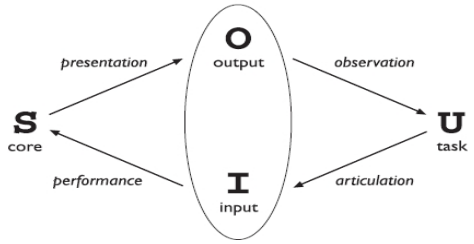


Figure 3.1 The general interaction framework



- In addition to the *User*'s task language and the *System*'s core language,which we have already introduced, there are languages for both the *Input* and *Output* components. *Input* and *Output* together form the *Interface*.
- As the interface sits between the *User* and the *System*, there are four steps in thenter active cycle, each corresponding to a translation from one component to another, as shown by the labeled arcs in Figure 3.2. The *User* begins the interactive cycle with the formulation of a goal and a task to achieve that goal. The only way
- the user can manipulate the machine is through the *Input*, and so the task must be barticulated within the input language.The *User*'s formulation of the desired task to achieve some goal needs to be *articulated*
- in the input language. The tasks are responses of the *User* and they need to be translated to stimuli for the *Input*.
- languagecan reach as many states of the *System* as is possible using the *System* stimuli directly.
- For example, the remote control units for some compact disc players do not allow the user to turn the power off on the player unit; hence the off state of the player cannot be reached using the remote control's input language.

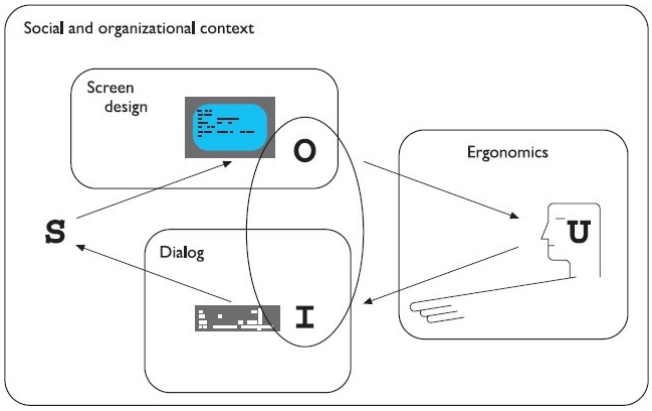
Assessing overall interaction

- The interaction framework is presented as a means to judge the overall usability of an entire interactive system.

- In reality, all of the analysis that is suggested by the framework is dependent on the current task (or set of tasks) in which the *User* is engaged.
- This is not surprising since it is only in attempting to perform a particular task within some domain that we are able to determine if the tools we use are adequate.
- For example, different text editors are better at different things. For= a particular editing task, one can choose the text editor best suited for interaction relative to the task.
- The best editor, if we are forced to choose only one, is the one that best suits the tasks most frequently performed. Therefore, it is not too disappointing that we cannot extend the interaction analysis beyond the scope of a particular task.

15. Explain in detail about framework and hci?

- As well as providing a means of discussing the details of a particular interaction, frameworks provide a basis for discussing other issues that relate to the interaction.
- The ACM SIGCHI Curriculum Development Group presents a framework similar to that presented here, and uses it to place different areas that relate to HCI [9].In Figure 3.3 these aspects are shown as they relate to the interaction framework.
- In particular, the field of *ergonomics* addresses issues on the user side of the interface, covering both input and output, as well as the user's immediate context. Dialog design and interface styles can be placed particularly along the input branch of the framework, addressing both articulation and performance.
- However, dialog is most usually associated with the computer and so is biased to that side of the framework
- Presentation and screen design relates to the output branch of the framework. The entire framework can be placed within a social and organizational context that also affects the interaction.
- Each of these areas has important implications for the design of interactive systems and the performance of the user.



15. Write in detail about ergonomics?

OVERVIEW:

- √ **Arrangement Of Controls And Displays**
- √ **The Physical Environment Of The Interaction**
- √ **Performance**
- √ **Health Issues**
- √ **The Use Of Color**
- √ **Ergonomics And Hci**

- Ergonomics (or human factors) is traditionally the study of the physical characteristics of

the interaction: how the controls are designed, the physical environment in which the interaction takes place, and the layout and physical qualities of the screen.

- A primary focus is on user performance and how the interface enhances or detracts rom this.
- In seeking to evaluate these aspects of the interaction, ergonomics will certainly also touch upon human psychology and system constraints. It is a large and established field, which is closely related to but distinct from HCI, and full coverage would demand a book in its own right.
- Here we consider a few of the issues addressed by ergonomics as an introduction to the field.

Arrangement of controls and displays

- In Chapter 1 we considered perceptual and cognitive issues that affect the way we present information on a screen and provide control mechanisms to the user.
- In addition to these cognitive aspects of design, physical aspects are also important. Sets of controls and parts of the display should be grouped logically to allow rapid access by the user (more on this in Chapter 5). This may not seem so important when we are considering a single user of a spreadsheet on a PC, but it becomes vital when we turn to safety-critical applications such as plant control, aviation and air traffic control. In each of these contexts, users are under pressure and are faced witha huge range of displays and controls.
- Here it is crucial that the physical layout of
- these be appropriate. Indeed, returning to the less critical PC application, inappropriate placement of controls and displays can lead to inefficiency and frustration.
- We have already touched on the importance of grouping controls together logically (and keeping opposing controls separate). The exact organization that this will suggest will depend on the domain and the application, but possible organizations include the following:

  functional controls and displays are organized so that those that are functionally related are placed together;

  sequential controls and displays are organized to reflect the order of their use in a typical interaction (this may be especially appropriate in domains where a particular task sequence is enforced, such as aviation);

  frequency controls and displays are organized according to how frequently they are used, with the most commonly used controls being the most easily accessible.

The physical environment of the interaction

- As well as addressing physical issues in the layout and arrangement of the machine interface, ergonomics is concerned with the design of the work environment itself.
- Where will the system be used? By whom will it be used? Will users be sitting, standing or moving about? Again, this will depend largely on the domain and will be more critical in specific control and operational settings than in general computer use.
- However, the physical environment in which the system is used may influence how well it is accepted and even the health and safety of its users. It should therefore be considered in all design. The first consideration here is the size of the users. Obviously this is going to vary considerably. However, in any system the smallest user should be able to reach all the controls (this may include a user in a wheelchair), and the largest user should not be cramped in the environment.
- In particular, all users should be comfortably able to see critical displays. For long periods of use, the user should be seated for comfort and stability. Seating should provide back support. If required to stand, the user should have room to move around in order to reach all the controls.

Health issues

- Perhaps we do not immediately think of computer use as a hazardous activity but we should bear in mind possible consequences of our designs on the health and safety of users.
- Leaving aside the obvious safety risks of poorly designed safety-critical systems (aircraft crashing, nuclear plant leaks and worse), there are a number of factors that may affect the use of more general computers. Again these are factors in the physical environment that directly affect the quality of the interaction and the user's

performance:

Physical position As we noted in the previous section, users should be able to reach all controls comfortably and see all displays. Users should not be expected to stand for long periods and, if sitting, should be provided with back support. If a particular position for a part of the body is to be adopted for long periods (for example, in typing) support should be provided to allow rest.

Temperature Although most users can adapt to slight changes in temperature without adverse effect, extremes of hot or cold will affect performance and, in excessive cases, health. Experimental studies show that performance deteriorates at high or low temperatures, with users being unable to concentrate efficiently.

Lighting The lighting level will again depend on the work environment. However, adequate lighting should be provided to allow users to see the computer screen without discomfort or eyestrain. The light source should also be positioned to\ avoid glare affecting the display.

Noise Excessive noise can be harmful to health, causing the user pain, and in acute cases, loss of hearing. Noise levels should be maintained at a comfortable level in the work environment. This does not necessarily mean no noise at all. Noise can be a stimulus to users and can provide needed confirmation of system activity.

Time The time users spend using the system should also be controlled. As we saw in the previous chapter, it has been suggested that excessive use of CRT displays can be harmful to users, particularly pregnant women.

The use of color

- In this section we have concentrated on the ergonomics of physical characteristics of systems, including the physical environment in which they are used.
- However, ergonomics has a close relationship to human psychology in that it is also concerned with the perceptual limitations of humans. For example, the use of color in displays is an ergonomics issue.
- As we saw in Chapter 1, the visual system has some limitations with regard to color, including the number of colors that are distinguishable and the relatively low blue acuity. We also saw that a relatively high proportion of the population suffers from a deficiency in color vision The colors used should also correspond to common conventions and user expectations.
- Red, green and yellow are colors frequently associated with stop, go and standby respectively. Therefore, red may be used to indicate emergency and alarms; green, normal activity; and yellow, standby and auxiliary function. These conventions should not be violated without very good cause.
- However, we should remember that color conventions are culturally determined.For example, red is associated with danger and warnings in most western cultures, but in China it symbolizes happiness and good fortune.
- The color of mourning is black in some cultures and white in others. Awareness of the cultural associations of color is particularly important in designing systems and websites

for a global market. We will return to these issues.

Ergonomics and HCI

- Ergonomics is a huge area, which is distinct from HCI but sits alongside it.Its contribution to HCI is in determining constraints on the way we design systems and suggesting detailed and specific guidelines and standards.
- Ergonomic factors are in general well established and understood and are therefore used as the basis for standardizing hardware designs. This issued.

17. Explain in detail about styles?OVERVIEW:

- √ **Menus**
- √ **Natural Language**
- √ **Question/Answer And Query Dialog**
- √ **Form-Fills And Spreadsheets**
- √ **The WIMP Interface**
- √ **Point-And-Click Interfaces**
- √ **Three-Dimensional Interfaces**

- Interaction can be seen as a dialog between the computer and the user. The choice of interface style can have a profound effect on the nature of this dialog.
- Dialog design is discussed in detail in Chapter 16. Here we introduce the most common interface styles and note the different effects these have on the interaction. There are a number of common interface styles including
  - √ command line interface
  - √ menus
  - √ natural language
  - √ question/answer and query dialog
  - √ form-fills and spreadsheets
  - √ WIMP
  - √ point and click
  - √ three-dimensional interfaces.
  - √ Command line interface
- The **command line interface** (Figure 3.7) was the first interactive dialog style to be commonly used and, in spite of the availability of menu-driven interfaces, it is still widely used.
- It provides a means of expressing instructions to the computer directly,using function keys, single characters, abbreviations or whole-word commands.
- In some systems the command line is the only way of communicating with the system,especially for remote access using *telnet*. More commonly today it is supplementary to menu-based interfaces, providing accelerated access to the system's functionality for experienced users.

Menus

- In a menu-driven interface, the set of options available to the user is displayed on the screen, and selected using the mouse, or numeric or alphabetic keys.
- Since the options are visible they are less demanding of the user, relying on recognition rather than recall. However, menu options still need to be meaningful and logically grouped to aid recognition.
- Often menus are hierarchically ordered and the option required is not available at the top layer of the hierarchy.

Natural language

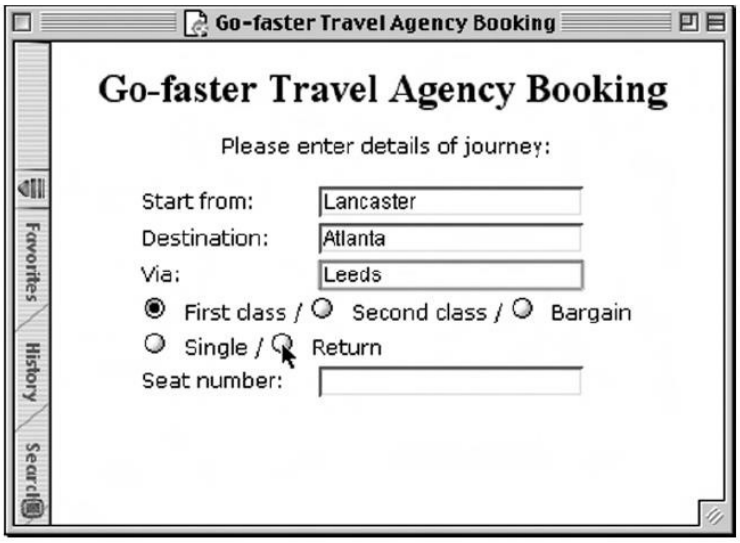
- Perhaps the most attractive means of communicating with computers, at least at first glance, is by natural language.
- Users, unable to remember a command or lost in a hierarchy of menus, may long for the computer that is able to understand instructions expressed in everyday words! Natural language understanding, both of speech and written input, is the subject of much interest and research.
- Unfortunately, however, the ambiguity of natural language makes it very difficult for a machine to understand.
- Language is ambiguous at a number of levels. First, the syntax, or structure, of a phrase may not be clear.
- If we are given the sentence **The boy hit the dog** with the stick we cannot be sure whether the boy is using the stick to hit the dog or whether thedog is holding the stick when it is hit.Even if a sentence's structure is clear, we may find ambiguity in the meaning of the words used.
- For example, the word 'pitch' may refer to a sports field, a throw,a waterproofing substance or even, colloquially, a territory.
- We often rely on the context and our general knowledge to sort out these ambiguities. This information is difficult to provide to the machine. To complicate matters more, the use of pronouns and relative terms adds further ambiguity.

Question/answer and query dialog

- Question and answer dialog is a simple mechanism for providing input to an application in a specific domain.
- The user is asked a series of questions (mainly with yes/no responses, multiple choice, or codes) and so is led through the interaction step by step. An example of this would be web questionnaires.
- These interfaces are easy to learn and use, but are limited in functionality and power.
- As such, they are appropriate for restricted domains (particularly information systems) and for novice or casual users. Query languages, on the other hand, are used to construct queries to retrieve information from a database.
- They use natural-language-style phrases, but in fact require specific syntax, as well as knowledge of the database structure. Queries usually require the user to specify an attribute or attributes for which to searchthe database, as well as the attributes of interest to be displayed.
- This is straight forward where there is a single attribute, but becomes complex when multiple attributes are involved, particularly if the user is interested in attribute A or attributeB, or attribute A and not attribute B, or where values of attributes are to be compared.Most query languages do not provide direct confirmation of what was requested, so that the only validation the user has is the result of the search

Form-fills and spreadsheets

- Form-filling interfaces are used primarily for data entry but can also be useful in data retrieval applications.
- The user is presented with a display resembling a paper form, with slots to fill in (see Figure 3.9). Often the form display is based upon an actual form with which the user is familiar, which makes the interface easier to use. The user works through the form, filling in appropriate values. The data are then entered into the application in the correct place.



The WIMP interface

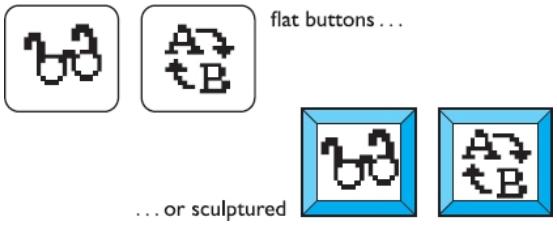
- Currently many common environments for interactive computing are examples of the *WIMP* interface style, often simply called windowing systems. WIMP stands for windows, icons, menus and pointers (sometimes windows, icons, mice and pull-down menus), and is the default interface style for the majority of interactive computer systems in use today, especially in the PC and desktop workstation arena. Examples ofWIMP interfaces include Microsoft Windows for IBM PC compatibles, MacOS for Apple Macintosh compatibles and various X Windows-based systems for UNIX.

Point-and-click interfaces

- In most multimedia systems and in web browsers, virtually all actions take only a single click of the mouse button. You may point at a city on a map and when you click a window opens, showing you tourist information about the city. You may point at a word in some text and when you click you see a definition of the word.
- You may point at are cognizable iconic button and when you click some action is performed.
- This point-and-click interface style is obviously closely related to the WIMP style. It clearly overlaps in the use of buttons, but may also include other WIMP elements.
- However, the philosophy is simpler and more closely tied to ideas of *hypertext*. In addition, the point-and-click style is not tied to mouse-based interfaces, and is also extensively used in touch screen information systems. In this case, it is often combined with a menu-driven interface.

Three-dimensional interfaces

- There is an increasing use of three-dimensional effects in user interfaces. The most obvious example is virtual reality, but VR is only part of a range of 3D techniques\ available to the interface designer.
- The simplest technique is where ordinary WIMP elements, buttons, scroll bars, etc., are given a 3D appearance using shading, giving the appearance of being sculpted out of stone. By unstated convention, such interfaces have a light source at their top right.
- Where used judiciously, the raised areas are easily identifiable and can be used to highlight active areas (Figure 3.12). Unfortunately, some interfaces make indiscriminate use of sculptural effects, on every text area, border and menu, so all sense of differentiation is lost.



18. Describe the elements in detail?OVERVIEW:

- √ **Windows**
- √ **Icons**
- √ **Pointers**
- √ **Menus**
- √ **Buttons**
- √ **Toolbars**

- ❖ We have already noted the four key features of the WIMP interface that give it its name – windows, icons, pointers and menus – and we will now describe these in turn.
- ❖ There are also many additional interaction objects and techniques commonly used in WIMP interfaces, some designed for specific purposes and others more general. We will look at buttons, toolbars, palettes and dialog boxes.

Windows

- ❖ Windows are areas of the screen that behave as if they were independent terminals in their own right.
- ❖ A window can usually contain text or graphics, and can be moved or resized. More than one window can be on a screen at once, allowing separate tasks to be visible at the same time.

- ❖ Users can direct their attention to the different windows as they switch from one thread of work to another. If one window overlaps the other, the back window is partially obscured, and then refreshed when exposed again. Overlapping windows can cause problems by obscuring vital information, so windows may also be *tiled*, when they adjoin but do not overlap each other.
- ❖ Alternatively, windows may be placed in a *cascading* fashion, where each new window is placed slightly to the left and below the previous window. In some systems this *layout policy* is fixed, in others it can be selected by the user.

Icons

- ❖ Windows can be closed and lost for ever, or they can be shrunk to some very reduced

representation.

- A small picture is used to represent a closed window, and this representation is known as an *icon*. By allowing icons, many windows can be available on the screen at the same time, ready to be expanded to their full size by clicking on the icon.

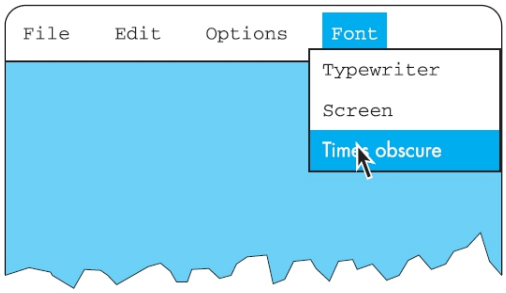
- Shrinking a window to its icon is known as *iconifying* the window. When a user temporarily does not want to follow a particular thread of dialog, he can suspend that dialog by iconifying the window containing the dialog.
- The icon saves space on the screen and serves as a reminder to the user that he can subsequently resume the dialog by opening up the window.

Pointers

- The pointer is an important component of the WIMP interface, since the interaction style required by WIMP relies very much on pointing and selecting things such as and trackballs are other alternatives, as we have previously seen in Chapter 2. The user is presented with a cursor on the screen that is controlled by the input device.

Menus

- The last main feature of windowing systems is the *menu*, an interaction techniquethat is common across many non-windowing systems as well.
- A menu presents a choice of operations or services that can be performed by the system at a given time.



- In Chapter 1, we pointed out that our ability to recall information is inferior to our ability to recognize it from some visual cue. Menus provide information cues in theform of an ordered list of operations that can be scanned. This implies that the names used for the

commands in the menu should be meaningful and informative. The pointing device is used to indicate the desired option.

Buttons

- Buttons are individual and isolated regions within a display that can be selected by the user to invoke specific operations.
- These regions are referred to as buttons because they are purposely made to resemble the push buttons you would find on a control panel. 'Pushing' the button invokes a command, the meaning of which is usually indicated by a textual label or a small icon.
- Buttons can also be used totoggle between two states, displaying status information such as whether the current font is italicized or not in a word processor, or selecting options on a web form.

Toolbars

- Many systems have a collection of small buttons, each with icons, placed at the top or side of the window and offering commonly used functions. The function of this *toolbar* is similar to a menu bar, but as the icons are smaller than the equivalent text more functions can be simultaneously displayed.
- Sometimes the content of the toolbar is fixed, but often users can *customize* it, either changing which functions are made available, or choosing which of several predefined toolbars is displayed.

Palettes

- In many application programs, interaction can enter one of several *modes*.
- The defining characteristic of modes is that the interpretation of actions, such as keystrokes or gestures with the mouse, changes as the mode changes. For example, musing the standard UNIX text editor vi, keystrokes can be interpreted either as operations to insert characters in the document (insert mode) or as operations to perform file manipulation (command mode).
- Problems occur if the user is not aware of the current mode. Palettes are a mechanism for making the set of possible modes and the active mode visible to the user. A palette is usually a collection oficons that are reminiscent of the purpose of the various modes.
- An example in a drawing package would be a collection of icons to indicate the pixel color or pattern that is used to fill in objects, much like an artist's palette for paint

Dialog boxes

- Dialog boxes are information windows used by the system to bring the user's attention to some important information, possibly an error or a warning used to prevent a possible error.
- Alternatively, they are used to invoke a sub dialog between user and system for a very specific task that will normally be embedded within some larger task.
- For example, most interactive applications result in the user creating some file that will have to be named and stored within the filing system. When the user or system wants to save the file, a dialog box can be used to allow the user to name the file and indicate where it is to be located within the filing system. When the save sub dialog is complete, the dialog box will disappear.

19. Explain in detail about interactivity in detail?

- When looking at an interface, it is easy to focus on the visually distinct parts (the buttons, menus, text areas) but the dynamics, the way they react to a user's actions, are less obvious.

- Dialog design, discussed in Chapter 16, is focussed almost entirely on the choice and specification of appropriate sequences of actions and corresponding changes in the interface state.
- However, it is typically not used at a fine level of detail and deliberately ignores the 'semantic' level of an interface: for example, the validation of numeric information in a forms-based system. It is worth remembering that *interactivity* is the defining feature of an *interactive* system. This can be seen in many areas of HCI.
- For example, the recognition rate for *speech recognition* is too low to allow transcription from tape, but in an airline reservation system, so long as the system can reliably recognize *yes* and *no* it can reflect back its understanding of what you said and seek confirmation.
- Speech-based *input* is difficult, speech-based *interaction* easier. Also, in the area of information visualization the most exciting developments are all where users can interact with a visualization in real time, changing parameters and seeing the effect. Interactivity is also crucial in determining the 'feel' of a WIMP environment. All WIMP systems appear to have virtually the same elements: windows, icons, menus, pointers, dialog boxes, buttons, etc.
- However, the precise behavior of these elements differs both within a single environment and between environments. For example, we have already discussed the different behavior of pull-down and fall-down menus.
- These look the same, but fall-down menus are more easily invoked by accident (and not surprisingly the windowing environments that use them have largely fallen into disuse!).
- In fact, menus are a major difference between the MacOS and Microsoft Windows environments: in MacOS you have to keep the mouse depressed throughout menu selection; in Windows you can click on the menu bar and a pull-down menu appears and remains there until an item is selected or it is cancelled. Similarly the detailed behavior of buttons is quite complex.
- In older computer systems, the order of interaction was largely determined by the machine. You did things when the computer was ready.
- In WIMP environments, the user takes the initiative, with many options and often many applications simultaneously available. The exceptions to this are *pre-emptive* parts of the interface, where the system for various reasons wrests the initiative away from the user, perhaps because of a problem or because it needs information in order to continue. The major example of this is *modal dialog boxes*.
- It is often the case that when a dialog box appears the application will not allow you to do anything else until the dialog box has been completed or cancelled. In some cases this may simply block the application, but you can perform tasks in other applications.
- In other cases you can do nothing at all until the dialog box has been completed.
- An especially annoying example is when the dialog box asks a question, perhaps simply for confirmation of an action, but the information you need to answer is hidden by the dialog box
- There are occasions when modal dialog boxes are necessary, for example when a major fault has been detected, or for certain kinds of instructional software.
- However, the general philosophy of modern systems suggests that one should minimize the use of pre-emptive elements, allowing the user maximum flexibility. Interactivity is also critical in dealing with errors. We discussed slips and mistakes earlier in the chapter, and some ways to try to prevent these types of errors.
- The other way to deal with errors is to make sure that the user or the system is able to tell when errors have occurred. If users can *detect* errors then they can correct them.
- So, even if errors occur, the interaction as a whole succeeds. Several of the principles in Chapter 7 deal with issues that relate to this.
- This ability to detect and correct is important both at the small scale of button presses

and keystrokes and also at the large scale.

20. Discuss in detail about paradigms?OVERVIEW:

- √ **Time Sharing**
- √ **Video Display Units**
- √ **Programming Toolkits**
- √ **Personal Computing**
- √ **Window Systems And The Wimp Interface**
- √ **The Metaphor**
- √ **Language Versus Action**
- √ **Hypertext**
- √ **Www**
- √ **Ubiquitous Computing**

Time sharing

- In the 1940s and 1950s, the significant advances in computing consisted of new hardware technologies. Mechanical relays were replaced by vacuum electron tubes.
- Tubes were replaced by transistors, and transistors by integrated chips, all of which meant that the amount of sheer computing power was increasing by orders of magnitude. By the 1960s it was becoming apparent that the explosion of growth in computing power would be wasted if there was not an equivalent explosion of ideas about how to channel that power.
- One of the leading advocates of research into human-centered applications of computer technology was J. C. R. Licklider, who became the director of the Information Processing Techniques Office of the US Department of Defense's Advanced Research Projects Agency (ARPA).
- It was Licklider's goal to finance various research centers across the United States in order to encourage new ideas about how best to apply the burgeoning computing technology.

Video display units

- As early as the mid-1950s researchers were experimenting with the possibility of presenting and manipulating information from a computer in the form of images on a video display unit (VDU).
- These display screens could provide a more suitable medium than a paper printout for presenting vast quantities of strategic information for rapid assimilation. The earliest applications of display screen images were developed in military applications, most notably the Semi-Automatic Ground Environment (SAGE) project of the US Air Force.

Programming toolkits

- Douglas Engelbart's ambition since the early 1950s was to use computer technology as a means of complementing human problem-solving activity.
- Engelbart's idea as agraduate student at the University of California at Berkeley was to use the computer to teach humans. This dream of naïve human users actually learning from a computer was a stark contrast to the prevailing attitude of his contemporaries that computers were a purposely complex technology that only the intellectually privileged\ were capable of manipulating.

Personal computing

- Programming toolkits provide a means for those with substantial computing skills to increase their productivity greatly. But Engelbart's vision was not exclusive to the

computer literate.

- The decade of the 1970s saw the emergence of computing power aimed at the masses, computer literate or not. One of the first demonstrations that the powerful tools of the hacker could be made accessible to the computer novice was a graphics programming language for children called LOGO.
- The inventor, Seymour Papert, wanted to develop a language that was easy for children to use. He and his colleagues from MIT and elsewhere designed a computer-controlled mechanical turtle that dragged a pen along a surface to trace its path.
- A child could quite easily pretend they were 'inside' the turtle and direct it to trace out simple geometric shapes, such as a square or a circle. By typing in English phrases, such as Go forward or Turn left, the child/programmer could teach the turtle to draw more and more complicated figures.

Window systems and the WIMP interface

- With the advent and immense commercial success of personal computing, the emphasis for increasing the usability of computing technology focussed on addressing the single user who engaged in a dialog with the computer in order to complete some work.
- Humans are able to think about more than one thing at a time, and in accomplishing some piece of work, they frequently interrupt their current train of thought to pursue some other related piece of work.
- A personal computer system which forces the user to progress in order through all of the tasks needed to achieve some objective, from beginning to end without any diversions, does not correspond to that standard working pattern.
- If the personal computer is to be an effective dialog partner, it must be as flexible in its ability to 'change the topic' as the human is.

The metaphor

- In developing the LOGO language to teach children, Papert used the metaphor of a turtle dragging its tail in the dirt. Children could quickly identify with the real-world phenomenon and that instant familiarity gave them an understanding of how they could create pictures.
- Metaphors are used quite successfully to teach new concepts in terms of ones which are already understood, as we saw when looking at analogy in Chapter 1.
- It is no surprise that this general teaching mechanism has been successful in introducing computer novices to relatively foreign interaction techniques. We have already seen how metaphors are used to describe the functionality of many interaction widgets, such as windows, menus, buttons and palettes.

Direct manipulation

- In the early 1980s as the price of fast and high-quality graphics hardware was steadily , designers were beginning to see that their products were gaining popularity as their visual content increased.
- As long as the user–system dialog remained largely unidirectional – from user command to system command line prompt – computing was going to stay within the minority population of the hackers who revelled in the challenge of complexity. In a standard command line interface, the only way to get any feedback on the results of previous interaction is to know that you have to ask for it and to know how to ask for it.
- Rapid visual and audio *feedback* on a high-resolution display screen or through a high-quality sound system makes it possible to provide evaluative information for every executed user action. He highlights the following features of a direct manipulation interface:
    - visibility of the objects of interest

- incremental action at the interface with rapid feedback on all actions
- reversibility of all actions, so that users are encouraged to explore without severe enalties
- syntactic correctness of all actions, so that every user action is a legal operation
- replacement of complex command languages with actions to manipulate directly the visible objects (and, hence, the name direct manipulation).

Language versus action

- Whereas it is true that direct manipulation interfaces make some tasks easier to perform correctly, it is equally true that some tasks are more difficult, if not impossible.
- Contrary to popular wisdom, it is not generally true that actions speak louder than words. The image we projected for direct manipulation was of the interface as a replacement for the underlying system as the world of interest to the user.
- Actions performed at the interface replace any need to understand their meaning at any deeper, system level.
- Another image is of the interface as the interlocutor or mediator between the user and the system. The user gives the interface instructions and it is then the responsibility of the interface to see that those instructions are carried out.

Hypertext

- In 1945, Vannevar Bush, then the highest-ranking scientific administrator in the USwar effort, published an article entitled 'As We May Think' in *The Atlantic Monthly*. Bush was in charge of over 6000 scientists who had greatly pushed back the frontiers of scientific knowledge during the Second World War.
- He recognized that a major drawback of these prolific research efforts was that it was becoming increasingly difficult to keep in touch with the growing body of scientific knowledge in the

Computer-supported cooperative work

- Another development in computing in the 1960s was the establishment of the first computer networks which allowed communication between separate machines. Personal computing was all about providing individuals with enough computing power so that they were liberated from dumb terminals which operated on a time-sharing system.
- It is interesting to note that as computer networks became widespread, individuals retained their powerful workstations but now wanted to reconnect themselves to the rest of the workstations in their immediate working environment, and even throughout the world!
- One result of this reconnection was the emergence of collaboration between individuals via the computer – called computer-supported cooperative work, or CSCW.

The world wide web

- Probably the most significant recent development in interactive computing is the world wide web, often referred to as just the web, or WWW. The web is built on top of the internet, and offers an easy to use, predominantly graphical interface to information, hiding the underlying complexities of transmission protocols, addresses and remote access to data.

Ubiquitous computing

- Where does computing happen, and more importantly, where do we as users go to interact with a computer? The past 50 years of interactive computing show that we still think of computers as being confined to a box on a desk or in an office or lab.
- The actual form of the physical interface has been transformed from a noisy teletype terminal to a large, graphical display with a WIMP or natural language interface, but in

all cases the user knows where the computer is and must walk over to it to begin interacting with it.

Sensor-based and context-aware interaction

- The yard-scale, foot-scale and inch-scale computers are all still clearly embodied devices with which we interact, whether or not we consider them 'computers'.
- There are an increasing number of proposed and existing technologies that embed computation even deeper, but unobtrusively, into day-to-day life. Weiser's dream was computers that 'permeate our physical environment so much that we do not notice the computers anymore', and the term ubiquitous computing encompasses a wide range from mobile devices to more pervasive environments.

UNIT II DESIGN & SOFTWARE PROCESS

Interactive Design basics – process – scenarios – navigation – screen design – Iteration and prototyping. HCI in software process – software life cycle – usability engineering – Prototyping in practice – design rationale. Design rules – principles, standards, guidelines, rules. Evaluation Techniques – Universal Design.

1. Interactive Design basics: (Apr/May 2017)

A simple definition of design is:

Achieving goals within constraints.

Goals What is the purpose of the design we are intending to produce? Who is it for? Why do they want it?

Constraints What materials must we use? What standards must we adopt? How much can it cost? How much time do we have to develop it? Are there health and safety issues?

The golden rule of design is

Understand your materials Ie)

i) Understand *computers*

- limitations, capacities, tools, platforms

ii) Understand *people*

- psychological, social aspects, human error.

THE PROCESS OF DESIGN:

Designing is an important phase because based on the requirement the designing will be started.

The simplified view of four main phases plus an iteration loop, focussed on the design of

interaction (Figure 5.1).

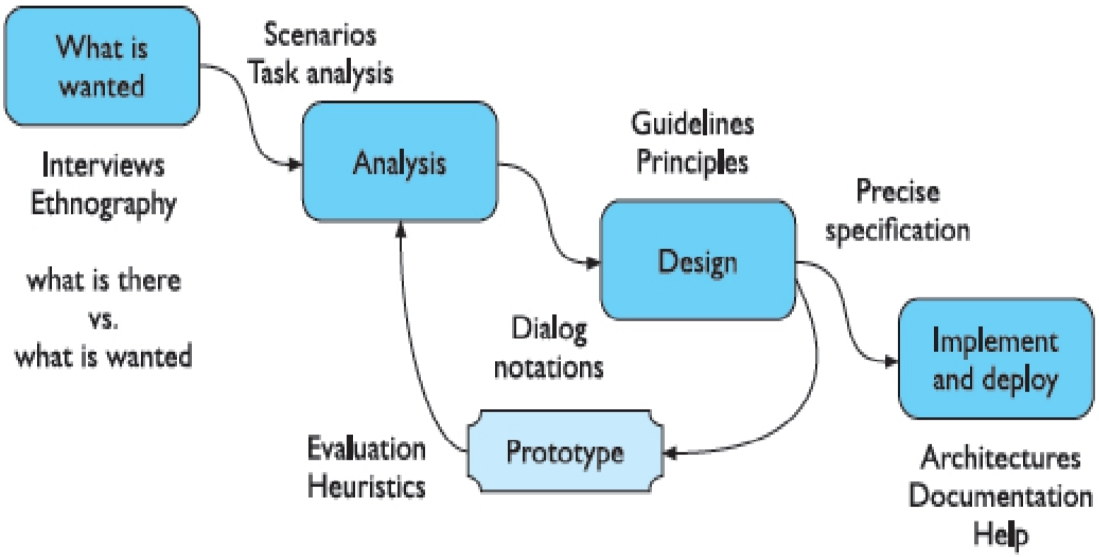


Figure 5.1 Interaction design process

Requirements –The first stage is establishing what exactly is needed. As a precursor to this it is usually necessary to find out what is currently happening?

There are a number of techniques used for this in HCI: interviewing people, videotaping them, looking at the documents and objects that they work with, observing them directly.

Analysis The results of observation and interview need to be ordered in some way to bring out key issues and communicate with later stages of design.

In this analysis Look at scenarios, rich stories of interaction, which can be used in conjunction with a method like task analysis or on their own to record and make vivid actual interaction? These techniques can be used both to represent the situation as it is and also the desired situation.

Design It is a central stage. There are numerous rules, guidelines and design principles is followed here. ie) How to design taking into account many different kinds of user. We need to record our design choices in some way and there are various notations and methods to do this, including those used to record the existing situation.

Some simple notations for designing navigation within a system and some basic heuristics to guide the design of that navigation.. It is at this stage also where input from theoretical work is most helpful, including cognitive models, organizational issues and understanding communication.

Iteration and prototyping: Humans are complex and we cannot expect to get designs right first time. We therefore need to evaluate a design to see how well it is working and where there can be improvements. Some forms of evaluation can be done using the design on paper, but it is hard to get real feedback without trying it out. Most user interface design therefore involves some form of prototyping, producing early versions of systems to try out with real users.

Implementation and deployment Finally, with our design, we need to create it and deploy it. This will involve writing code, perhaps making hardware, writing documentation and manuals – everything that goes into a real system that can be given to others.

SCENARIOS:

Scenarios are stories for design: rich stories of interaction. They are perhaps the simplest design representation, but one of the most flexible and powerful. Some scenarios are quite short: 'The user intends to press the "save" button, but accidentally presses the "quit" button so loses his work'. Others are focused more on describing the situation or context.

> Example of a scenario for the personal movie player. Like the persona it is perhaps more detailed than appears necessary, but the detail helps make the events seem real. The figure shows plain text, but scenarios can be augmented by sketches, simulated screen shots, etc.
>
> These sketches and pictures are called *storyboards* and are similar to the techniques used in film making to envisage plot-lines. Where the design includes physical artifacts the scenarios can be used as a script to act out potential patterns of use. For example, we might imagine a digital Swiss army knife, which has a small LCD screen and uses the toothpick as a stylus.

The knife connects to the internet via a wireless link through your phone and gives interesting tips from other Swiss army knife users. Try getting two together at a party – you will see this would appeal! It sounds like a great design idea – but wait, try acting out the use.

If you have a Swiss army knife, use it, or use something penknifesized if you don't. The tip on the LCD says, 'open the stone remover': a small LED glows near the right blade – you open it. 'Now push the blade into the rubber of the grommet', it says.. Look at the knife in your hand . . . oops, your thumb is covering where the screen would be.

If you add more detail you can get to a blow-by-blow account of the user–system interactions and then ask 'what is the user intending now?'; 'what is the system doing now?'. This can help to verify that the design would make sense to the user and also that proposed implementation architectures would work.

In addition scenarios can be used to:

Communicate with others – other designers, clients or users. It is easy to misunderstand each other whilst discussing abstract ideas. Concrete examples of use are far easier to share.

Validate other models A detailed scenario can be 'played' against various more formal representations such as task models or dialog and navigation models .

Express dynamics Individual screen shots and pictures give you a sense of what a system would look like, but not how it behaves.

In contrast scenarios are linear – they represent a single path amongst all the potential interactions.

This linearity has both positive and negative points:

Time is linear Our lives are linear as we live in time and so we find it easier to understand simple linear narratives. We are natural storytellers and story listeners.

But no alternatives Real interactions have choices, some made by people, some by systems. A simple scenario does not show these alternative paths. In particular, it is easy to miss the unintended things a person may do.

Scenarios are a resource that can be used and reused throughout the design process:

Helping us see what is wanted, suggesting how users will deal with the potential design, checking that proposed implementations will work, and generating test cases for final evaluation.

NAVIGATION DESIGN:

As we stressed, the object of design is not just a computer system or device, but the socio-technical intervention as a whole. However, as design progresses we come to a point where we do need to consider these most tangible outputs of design.

Imagine yourself using a word processor. You will be doing this in some particular social and physical setting, for a purpose. But now we are focussing on the computer system itself. You interact at several levels:

Widgets The appropriate choice of widgets and wording in menus and buttons will help you know how to use them for a particular selection or action.

Screens or windows You need to find things on the screen, understand the logical grouping of buttons.

Table 5.1 Levels of interaction

PC application	Website	Physical device
Widgets	Form elements, tags and links	Buttons, dials, lights, displays
Screen design	Page design	Physical layout
Navigation design	Site structure	Main modes of device
Other apps and operating system	The web, browser, external links	The real world!

Navigation within the application You need to be able to understand what will happen when a button is pressed, to understand where you are in the interaction.

Environment The word processor has to read documents from disk, perhaps some are on remote networks. You swap between applications, perhaps cut and paste

Table 5.1 shows.

There are differences; for example, in the web we have less control of how people enter a site and on a physical device we have the same layout of buttons and displays no matter what the internal state (although we may treat them differently).

We discussed graphical user interface widgets in Chapter 3 and in the next section we will look at details of screen design. In this section we will look mainly at navigation design, that is the main screens or modes within a system and how they interconnect.

We will also briefly consider how this interacts with the wider environment. Just in case you haven't already got the idea, the place to start when considering the structure of an application is to think about actual use:

i) Who is going to use the application?
ii) How do they think about it?
iii) What will they do with it?

This can then drive the second task – thinking about structure. Individual screens or the layout of devices will have their own structure, but this is for the next section.

Here we will consider two main kinds of issue:

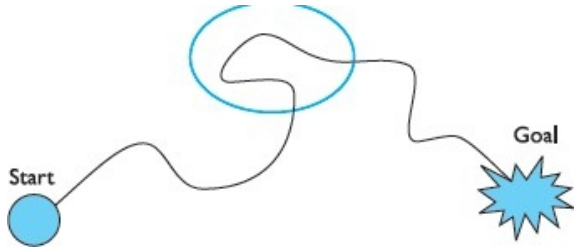
i) local structure
- looking from one screen or page out
ii)global structure
- structure of site, movement between screens.

Local structure:

Much of interaction involves goal-seeking behavior. Users have some idea of what they are after and a partial model of the system. In an ideal world if users had perfect knowledge of what they wanted and how the system worked they could simply take the shortest path to what they want, pressing all the right buttons and links.

However, in a world of partial knowledge users meander through the system. The important thing is not so much that they take the most efficient route, but that at each point in the interaction they can make some assessment of whether they are

getting closer to their (often partially formed) goal.



user enough knowledge of what to do to get closer to their goal. To get you started, here are four things to look for when looking at a single web page, screen or state of a device.

i) knowing where you are
ii) knowing what you can do
iii) knowing where you are going – or what will happen
iv) knowing where you've been – or what you've done.

The screen, web page or device displays should make clear *where you are* in terms of the interaction or state of the system. Some websites show 'bread crumbs' at the top of the screen, the path of titles showing where the page is in the site (Figure 5.5).

It is also important to know *what you can do* – what can be pressed or clicked to go somewhere or do something. Some web pages are particularly bad in that it is unclear which images are pure decoration and which are links to take you somewhere.

On the web the standard underlined links make it clear which text is clickable and which is not. However, in order to improve the appearance of the page many sites change the color of links and may remove the underline too. This is especially confusing if underline is then used as simple emphasis on words that are not links

Figure 5.5 Breadcrumbs. Screen shot frame reprinted by permission from Microsoft Corporation

Chic design is also a problem in physical devices. One of the authors was once in a really high-class hotel and found he could not locate the flush in the toilet. Only after much fumbling did he discover that one of the tiles could be pressed. The 'active' tile was level with the rest of the tiled wall – a very clean design, but not very usable!

In the case of a website or information system this may mean you then have to use some sort of 'back' mechanism to return, but that is all; however, in an application or device the action of clicking the button may already have caused some effect.

If the system has an easy means to undo or reverse actions this is not so bad, but it is better if users do not have to use this 'try it and see' interaction. Where response times are slow this is particularly annoying.

Special care has to be taken if the same command or button press means something different in different contexts. These different contexts that change the interpretation of commands are called *modes*.

Many older text editors would interpret pressing 'x' to mean 'enter me into the text' in a normal typing mode, but 'exit' in a special command mode. In general, modes are less of a problem in windowed systems where the mode is made apparent by the current window . However, physical devices may have minimal displays and may be operated without visual attention.

Finally, if you have just done some major action you also want some sort of confirmation of *what you've done*. If you are faultless and have perfect knowledge, of course you will be sure that you have hit the right key and know exactly what will happen.

In an information system, there is a related but slightly different issue, which is to know *where you have been*. The feeling of disorientation when you do not have sufficient means to know where you are and where you have been has been called 'lost in hyperspace'. Most web browsers offer a history system and also a 'back' button that keeps a list of recently visited pages.

Global structure – hierarchical organization

The overall structure of an application. This is the way the various screens, pages or device states link to one another. One way to organize a system is in some form of hierarchy. This is typically organized along functional boundaries (that is, different kinds of things), but may be organized by roles, user type, or some more esoteric breakdown such as modules in an educational system. The hierarchy links screens, pages or states in logical groupings.

For example,

Figure 5.6 gives a high-level breakdown of some sort of messaging system. This sort of hierarchy can be used purely to help during design, but can also be used to structure the actual system. For example, this may reflect the menu structure of a PC application or the site structure on the web.

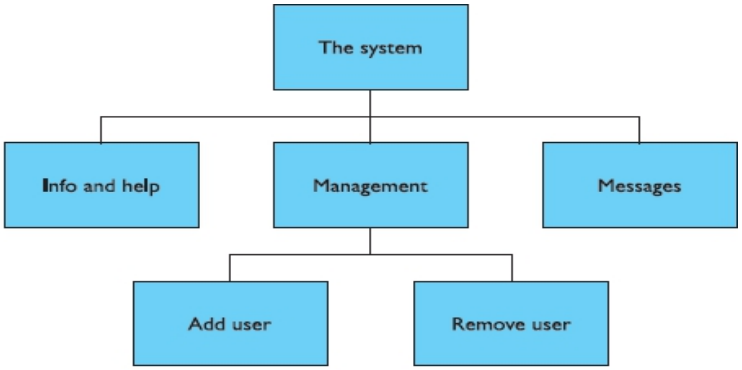


Figure 5.6 Application functional hierarchy

Any sort of information structuring is difficult, but there is evidence that people find hierarchies simpler than most. One of the difficulties with organizing information or system functionality is that different people have different internal structures for their knowledge, and may use different vocabulary.

This is one of the places where a detailed knowledge of the intended users is essential: it is no good creating a hierarchy that the designers understand, but not the users . . . and all too commonly this is exactly what happens.

However much you think you have got the wording and categories right, because there are different users it is inevitable that not everyone will understand it perfectly.

There is also evidence that deep hierarchies are difficult to navigate, so it is better to have broad top-level categories, or to present several levels of menu on one screen or web page. Miller's magic number of 7 □□2 for working memory capacity. It is often misused in this context. Many guidelines suggest that menu breadth, that is the number of choices available at each level in the menu, should be around seven.

However, Miller's result applies only to working memory, not visual search. In fact, optimal breadth can be quite large, perhaps 60 or more items for a web index page if the items are organized in such a way that the eye can easily find the right one [206].

However, here the critical thing is the naturalness of the classification, which itself may depend on the user's purpose. For example, if the user wants to look up information on a particular city, an alphabetical list of all city names would be fast, but for other purposes a list by region would be more appropriate.

Global structure – dialog:

In a pure information system or static website it may be sufficient to have a fully hierarchical structure, perhaps with next/previous links between items in the same group. However, for any system that involves doing things, constantly drilling down from one part of the hierarchy to another is very frustrating.

> Usually there are ways of getting more quickly from place to place. For example, in a stock control system there may be a way of going from a stock item to all orders outstanding on that item and then from an order to the purchase record for the customer who placed the order.
>
> These would each be in a very different part of a hierarchical view of the application, yet directly accessible from one another. As well as these cross-links in hierarchies, when you get down to detailed interactions, such as editing or deleting a record, there is obviously a flow of screens and commands that is not about hierarchy. In HCI the word 'dialog' is used to refer to this pattern of interactions between the user and a system.

Consider the following fragment from a marriage service:

Minister: Do you *name* take this woman . . .
Man: I do
Minister: Do you *name* take this man . . .
Woman: I do
Minister: I now pronounce you man and wife

Notice this describes the general flow of the service, but has blanks for the names of the bride and groom. So it gives the pattern of the interaction between the parties, but is instantiated differently for each service. Human–computer dialog is just the same; there are overall patterns of movement between main states of a device or windows in a PC application, but the details differ each time it is run.

Recall that scenarios gave just one path through the system. To describe a full system we need to take into account different paths through a system and loops where the system returns to the same screen. There are various ways to do this, and in Chapter 16 we will expand on the wedding example and look at several different types of dialog model.

A simple way is to use a network diagram showing the principal states or screens linked together with arrows. This can:

i) Show what leads to what
ii) Show what happens when
iii) Include branches and loops be more task oriented than a hierarchy.

Figure 5.7 shows a network diagram illustrating the main screens for adding or deleting a user from the messaging system in Figure 5.6.

The arrows show the general flow between the states. We can see that from the main screen we can get to either the 'remove user' screen or the 'add user' screen. This is presumably by selecting buttons or links, but the way these are shown we leave to detailed screen design.

We can also see that from the 'add user' screen the system always returns to the main screen, but after the 'remove user' screen there is a further confirmation screen.

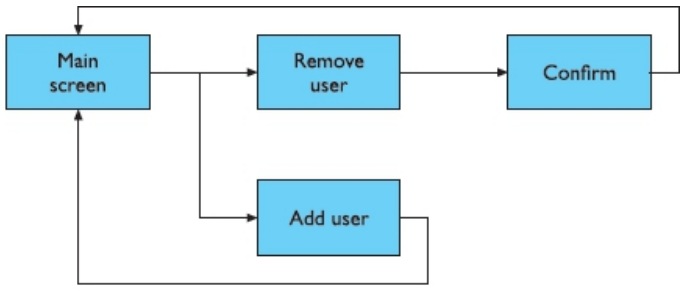


Figure 5.7 Network of screens/states

Wider still :

Donne said 'No man is an Iland, intire of it selfe'. This is also true of the things we design. Each sits amongst other devices and applications and this in turn has to be reflected within our design. This has several implications:

Style issues We should normally conform to platform standards, such as positions for menus on a PC application, to ensure consistency between applications. For example, on our proposed personal movie player we should make use of standard fast-forward, play and pause icons.

Functional issues On a PC application we need to be able to interact with files, read standard formats and be able to handle cut and paste.

Navigation issues We may need to support linkages between applications, for example allowing the embedding of data from one application in another, or, in a mail system, being able to double click an attachment icon and have the right application launched for the attachment.

On the web we have the added difficulty that other sites and applications may include links that bypass our 'home page' and other pages and go direct into the heart of our site or web application. Also, when we link to other sites, we have no control over them or the way their content may change over time.

SCREEN DESIGN AND LAYOUT:

The different elements that make up interactive applications, but not about how we put them together. A single screen image often has to present information clearly and also act as the locus for interacting with the system.

The basic principles at the screen level reflect those in other areas of interaction design:

Ask What is the user doing?

Think What information is required? What comparisons may the user need to make? In what order are things likely to be needed?

Design Form follows function: let the required interactions drive the layout.

Tools for layout:

The number of visual tools available to help us suggest to the user appropriate ways to read and interact with a screen or device.

Figure 5.8 Grouping related items in an order screen

Grouping and structure:

If things logically belong together, then it should normally physically group them together. This may involve multiple levels of structure.

For example, in Figure 5.8 we can see a potential design for an ordering screen. Notice how the details for billing and delivery are grouped together spatially; also note how they are separated from the list of items actually ordered by a line as well as spatially. This reflects the following logical structure:

Order:

Administrative information
Billing details
Delivery details
Order information
Order line 1
Order line 2
. . .

Order of groups and items:

In Figure 5.8 again we can see that the screen seems to naturally suggest reading or filling in the billing details first, followed by the delivery details, followed by the individual order items.

Is this the right order?

what is the natural order for the user? This should normally match the order on screen. For data entry forms or dialog boxes we should also set up the order in which the tab key moves between fields. Occasionally we may also want to force a particular order.

Decoration:

Again looking at Figure 5.8, we can see how the design uses boxes and a separating line to make the grouping clear. Other decorative features like font style, and text or background colors can be used to emphasize groupings.

Figure 5.9 Microwave control panel

Look at the microwave control panel in Figure 5.9. How the buttons differ in using the foreground and background colors (green and gold) so that groups are associated with one another.

How the buttons are laid out to separate them into groups of similar function. *Alignment* Alignment of lists is also very important. For users who read text from left to right, lists of text items should normally be aligned to the left.

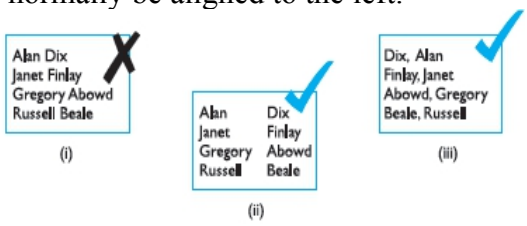


Figure 5.10 Looking up surnames

Numbers, however, should normally be aligned to the right (for integers) or at the decimal point. This is because the shape of the column then gives an indication of magnitude – a sort of minihistogram.

Items like names are particularly difficult. Consider list (i) in Figure 5.10. It is clearly hard to look someone up if you only know their surname. To make it easy, such lists should be laid out in columns as in (ii), or have forename and surname reversed as in (iii).

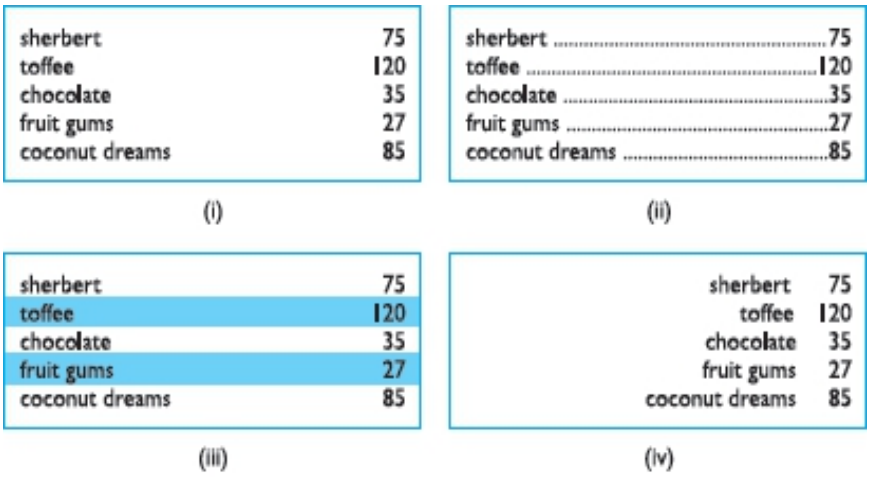


Figure 5.11 Managing multiple columns

The alignment rules, perhaps right aligning some text items as in (iv). This last alternative might be a good solution if you were frequently scanning the numbers and only occasionally scanning the names of items.

White space

In typography the space between the letters is called the counter. In painting this is also important and artists may focus as much on the space between the foreground elements such as figures and buildings as on the elements themselves. Often the shape of the counter is the most important part of the composition of a painting and in calligraphy and typography the balance of a word is determined by giving an even weight to the counters.

Space can be used in several ways. Some of these are shown in Figure 5.12. The colored areas represent continuous areas of text or graphics.

In (i) we can see space used to separate blocks as you often see in gaps between paragraphs or space between sections in a report. Space can also be used to create more complex structures.

In (ii) there are clearly four main areas: ABC, D, E and F. Within one of these are three further areas, A, B and C, which themselves are grouped as A on its own, followed by B and C together.

In Figure 5.12 (iii), we can see space used to highlight. This is a technique used frequently in magazines to highlight a quote or graphic.

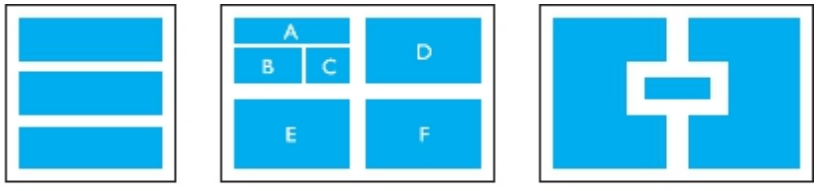


Figure 5.12 Using white space in layout

ITERATION AND PROTOTYPING

All interaction design includes some form of iteration of ideas. This often starts early on with paper designs and storyboards being demonstrated to colleagues and potential users. Later in the design process one might use mockups of physical devices or tools such as Shockwave or Visual Basic to create prototype versions of software.

Any of these prototypes, whether paper-based or running software, can then be evaluated to see whether they are acceptable and where there is room for improvement.

This sort of evaluation, intended to improve designs, is called *formative* evaluation. This is in contrast to *summative* evaluation, which is performed at the end to verify whether the product is good enough. Chapter 9 considers evaluation in detail.

One approach is to get an expert to use a set of guidelines, for example the 'knowing where you are' list above, and look screen by screen to see if there are any violations. The other main approach is to involve real users either in a controlled experimental setting, or 'in the wild' – a real-use environment.

The result of evaluating the system will usually be a list of faults or problems and this is followed by a redesign exercise, which is then prototyped, evaluated . . .
Figure 5.14 shows this process. The end point is when there are no more problems that can economically be fixed.

So iteration and prototyping are the universally accepted 'best practice' approach for interaction design. However, there are some major pitfalls of prototyping, rarely acknowledged in the literature.

Prototyping is an example of what is known as a *hill-climbing* approach. Imagine you are standing somewhere in the open countryside. You walk uphill and keep going uphill as steeply as possible. Eventually you will find yourself at a hill top.

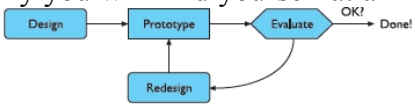


Figure 5.14 Role of prototyping

Figure 5.15 Moving little by little . . . but to where?

This is exactly how iterative prototyping works: you start somewhere, evaluate it to see how to make it better, change it to make it better and then keep on doing this until it can't get any better.

However, if one started somewhere else you might end up at the top of the Matterhorn. Hillclimbing methods always have the potential to leave you somewhere that is the best in the immediate area, but very poor compared with more distant places.

Figure 5.15 shows this schematically: if you start at A you get trapped at the *local maximum* at B, but if you start at C you move up through D to the *global maximum* at E. This problem of

getting trapped at local maxima is also possible with interfaces.

From this we can see that there are two things you need in order for prototyping methods to work:

1. To understand what is wrong and how to improve.
2. A good start point.

The first is obvious; you cannot iterate the design unless you know what must be done to improve it. The second, however, is needed to avoid local maxima.

A good designer might guess a good initial design based on experience and judgment. However, the complexity of interaction design problems means that this insight is hard.

2. THE SOFTWARE LIFE CYCLE:

One of the distinguishing characteristics of any engineering discipline is that it entails the structured application of scientific techniques to the development of some product.

The software life cycle is an attempt to identify the activities that occur in software development.

In the development of a software product, we consider two main parties: the customer who requires the use of the product and the designer who must provide the product.

Activities in the life cycle:

A more detailed description of the life cycle activities is depicted in Figure 6.1. The graphical representation is reminiscent of a waterfall, in which each activity naturally leads into the next.

Requirements specification:

In requirements specification, the designer and customer try to capture a description of *what* the eventual system will be expected to provide.

This is in contrast to determining *how* the system will provide the expected services, which is the concern of later activities. Requirements specification involves eliciting information from the customer about the work environment, or domain, in which the final product will function.

requirements specification begins at the start of product development. Though the requirements are from the customer's perspective, if they are to be met by the

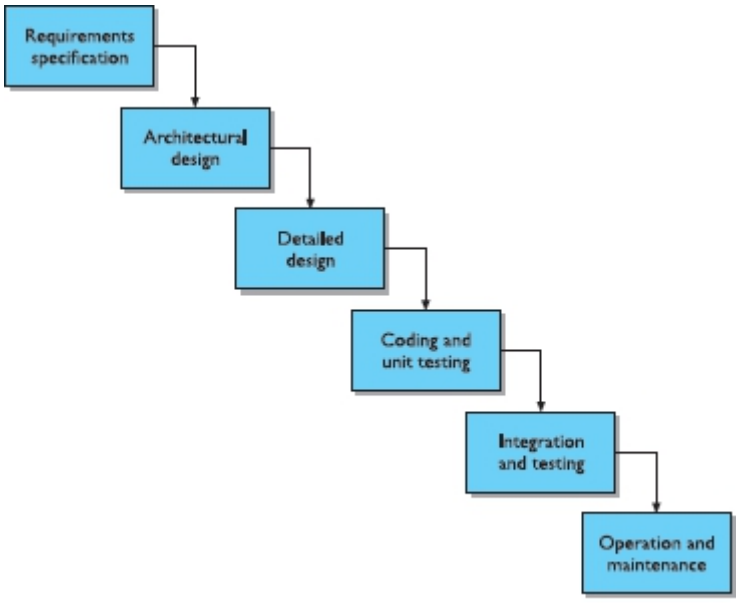


Figure 6.1 The activities in the waterfall model of the software life cycle

software product they must be formulated in a language suitable for implementation. Requirements are usually initially expressed in the native language of the customer.

The executable languages for software are less natural and are more closely related to a mathematical language in which each term in the language has a precise interpretation, or semantics.

Architectural design:

The next activities concentrate on *how* the system provides the services expected from it. The first activity is a high-level decomposition of the system into components that can either be brought in from existing software products or be developed from scratch independently. An architectural design performs this decomposition.

It is not only concerned with the functional decomposition of the system, determining which components provide which services. It must also describe the interdependencies between separate components and the sharing of resources that will arise between components.

There are many structured techniques that are used to assist a designer in deriving an architectural description from information in the requirements specification (such as CORE, MASCOT and HOOD).

The *functional requirements* of the system – the services the system must provide in the work domain – but do not provide an immediate way to capture other *non-functional requirements* – features of the system that are not directly related to the actual services provided but relate to the manner in which those services must be provided.

Detailed design

The architectural design provides a decomposition of the system description that allows for

isolated development of separate components which will later be integrated.

The detailed design is a *refinement* of the component description provided by the architectural design. The behavior implied by the higher-level description must be preserved in the more detailed description.

Thus the language used for the detailed design must allow some analysis of the design in order to assess its properties.

Coding and unit testing:

The detailed design for a component of the system should be in such a form that it is possible to implement it in some executable programming language. After coding, the component can be tested to verify that it performs correctly, according to some test criteria that were determined in earlier activities.

Research on this activity within the life cycle has concentrated on two areas. There is plenty of research that is geared towards the automation of this coding activity directly from a low-level detailed design.

Most of the work in *formal methods* operates under the hypothesis that, in theory, the transformation from the detailed design to the implementation is from one mathematical representation to another and so should be able to be entirely automated.

Integration and testing:

Once enough components have been implemented and individually tested, they must be integrated. Further testing is done to ensure correct behavior and acceptable use of any shared resources.

It is only after acceptance of the integrated system that the product is finally released to the customer. It may also be necessary to certify the final system according to requirements imposed by some outside authority, such as an aircraft certification board.

As of 1993, a European health and safety act requires that all employers provide their staff with usable systems. The international standards authority, ISO, has also produced a standard (ISO 9241) to define the usability of office environment workstations.

Coupled together, the health and safety regulations and ISO 9241 provide impetus for designers to take seriously the HCI implications of their design.

Maintenance:

After product release, all work on the system is considered under the category of maintenance, until such time as a new version of the product demands a total redesign or the product is phased out entirely.

Consequently, the majority of the lifetime of a product is spent in the maintenance activity. Maintenance involves the correction of errors in the system which are discovered after release and the revision of the system services to satisfy requirements that were not realized during previous development.

The maintenance provides feedback to all of the other activities in the life cycle, as shown in

Figure 6.2.

Validation and verification:

In life cycle, the design must be checked to ensure that it both satisfies the high-level requirements agreed with the customer and is also complete and internally consistent.

These checks are referred to as *validation* and *verification*, respectively. Boehm [36a] provides a useful distinction between the two, characterizing validation as designing 'the right thing' and verification as designing 'the thing right'.

Various languages are used throughout design, ranging from informal natural language to very precise and formal mathematical languages. Validation and verification exercises are difficult enough when carried out within one language; they become much more difficult, if not impossible, when attempted between languages.

Verification of a design will most often occur within a single life-cycle activity or between two adjacent activities. For example, in the detailed design of a component

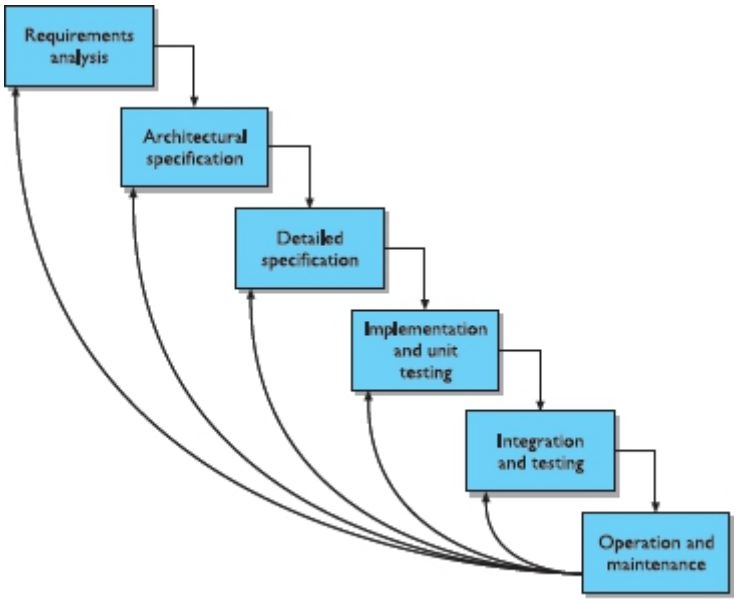


Figure 6.2 Feedback from maintenance activity to other design activities

of a payroll accounting system, the designer will be concerned with the correctness of the algorithm to compute taxes deducted from an employee's gross income.

> The architectural design will have provided a general specification of the information input to this component and the information it should output. The detailed description will introduce more information in refining the general specification.
>
> The detailed design may also have to change the representations for the information and will almost certainly break up a single high-level operation into several low-level operations that can eventually be implemented.
>
> In introducing these changes to information and operations, the designer must show that the refined description is a legal one within its language (internal consistency) and that it describes all of the *specified* behavior of the high-level description (completeness) in a provably correct way (relative consistency).

Validation of a design demonstrates that within the various activities the customer's requirements are satisfied. Validation is a much more subjective exercise than verification, mainly because the disparity between the language of the requirements and the language of the design forbids any objective form of proof.

Validation proofs are much trickier, as they almost always involve a transformation between languages. Furthermore, the origin of customer requirements arises in the inherent ambiguity of the real world and not the mathematical world.

This precludes the possibility of objective proof, rigorous or formal. Instead, there will always be a leap from the informal situations of the real world to any formal and structured development process. We refer to this inevitable disparity as the *formality gap*, depicted in Figure 6.3.

The formality gap means that validation will always rely to some extent on subjective means of proof. We can increase our confidence in the subjective proof by effective use of real-world experts in performing certain validation chores. These experts will not necessarily have design expertise, so they may not understand the design notations used.

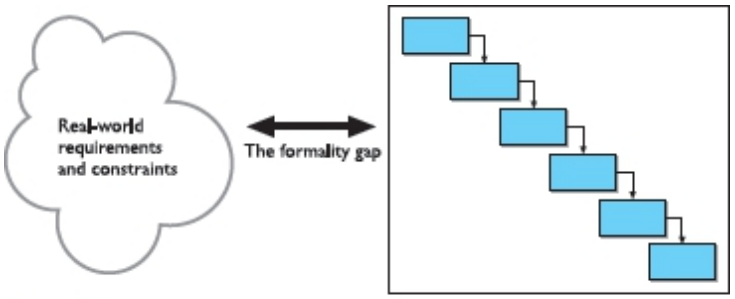


Figure 6.3 The formality gap between the real world and structured design

Therefore, it is important that the design notations narrow the formality gap, making clear the claims that the expert can then validate.

For interactive systems, the expert will have knowledge from a cognitive or psychological domain, so the design specification must be readily interpretable from a psychological perspective in order to validate it against interactive requirements of the system.

Interactive systems and the software life cycle:

In batch-processing systems, then actual design process is *iterative*, work in one design activity affecting work in any other activity both before and after it in the life cycle.

We can represent this iterative relationship as in Figure 6.4, but that does not greatly enhance any understanding of the design process for interactive systems.

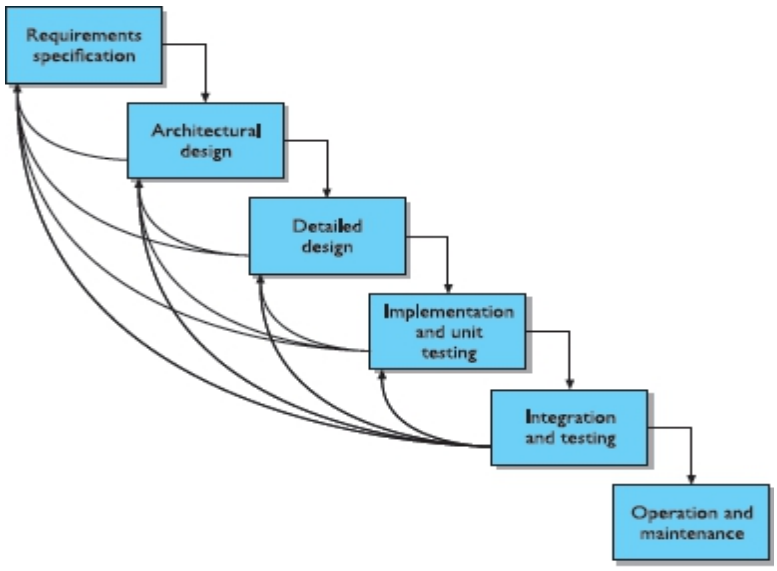


Figure 6.4 Representing iteration in the waterfall model

It depicts how discovery in later activities can be reflected in iterations back to earlier stages. This is an admission that the requirements capture activity is not executed properly.

The more serious claim we are making here is that all of the requirements for an interactive system *cannot* be determined from the start, and there are many convincing arguments to support this position.

The result is that systems must be built and the interaction with users observed and evaluated in order to determine how to make them more usable.

A final point about the traditional software life cycle is that it does not promote the use of notations and techniques that support the user's perspective of the interactive system.

3. USABILITY ENGINEERING:

One approach to user-centered design has been the introduction of explicit *usability engineering* goals into the design process, as suggested by Whiteside and colleagues at IBM and Digital Equipment Corporation and by Nielsen at Bellcore [260, 261].

The emphasis for usability engineering is in knowing exactly what criteria will be used to judge a product for its usability.

The ultimate test of a product's usability is based on measurements of users' experience with it. Therefore, since a user's direct experience with an interactive system is at the physical interface, focus on the actual user interface is understandable.

All simple to derive measurements of activity beyond the physical actions in the world, and so usability engineering is limited in its application.

In relation to the software life cycle, one of the important features of usability engineering is the inclusion of a usability specification, forming part of the requirements specification, that concentrates on features of the user–system interaction which

contribute to the usability of the product.

Various attributes of the system are suggested as gauges for testing the usability. For each attribute, six items are defined to form the usability specification of that attribute. Table 6.1 provides an example of a usability specification for the design of a control panel for a video cassette recorder (VCR), based on the technique presented by Whiteside, Bennett and Holtzblatt [377].

Table 6.1 Sample usability specification for undo with a VCR

Attribute:	Backward recoverability
Measuring concept:	Undo an erroneous programming sequence
Measuring method:	Number of explicit user actions to undo current program
Now level:	No current product allows such an undo
Worst case:	As many actions as it takes to program in mistake
Planned level:	A maximum of two explicit user actions
Best case:	One explicit cancel action

Recoverability refers to the ability to reach a desired goal after recognition of some error in previous interaction. The recovery procedure can be in either a backward or forward sense. Current VCR design has resulted in interactive systems that are notoriously difficult to use; the redesign of a VCR provides a good case study for usability engineering.

In designing a new VCR control panel, the designer wants to take into account how a user might
recover from a mistake he discovers while trying to program the VCR to record some television program in his absence.

One approach that the designer decides to follow is to allow the user the ability to undo the programming sequence, reverting the state of the VCR to what it was before the programming task began.

The backward recoverability attribute is defined in terms of a *measuring concept*, which makes the abstract attribute more concrete by describing it in terms of the actual product. So in this case, we realize backward recoverability as the ability to undo an erroneous programming sequence.

The *measuring method* states how the attribute will be measured, in this case by the number of explicit user actions required to perform the undo, regardless of where the user is in the programming sequence.

The remaining four entries in the usability specification then provide the agreed criteria for judging the success of the product based on the measuring method.

The *now level* indicates the value for the measurement with the existing system, whether it is computer based or not.

The *worst case* value is the lowest acceptable measurement for the task, providing a clear distinction between what will be acceptable and what will be unacceptable in the final product.

The *planned level* is the target for the design and the *best case* is the level which is agreed to be the best possible measurement given the current state of development tools and technology.

In the example, the designers can look at their previous VCR products and those of their competitors to determine a suitable now level. In this case, it is determined that no current model allows an undo which returns the state of the VCR to what it was before the programming task.

For example, if a VCR allows you three separate recording programs, once you begin entering a new program in the number 1 program slot, the VCR forgets the previous contents of that slot and so you cannot recover it unless you remember what it was and then reprogram it.

Table 6.2 Criteria by which measuring method can be determined (adapted from Whiteside, Bennett and Holtzblatt [377], Copyright 1988, reprinted with permission from Elsevier)

1.	Time to complete a task
2.	Per cent of task completed
3.	Per cent of task completed per unit time
4.	Ratio of successes to failures
5.	Time spent in errors
6.	Per cent or number of errors
7.	Per cent or number of competitors better than it
8.	Number of commands used
9.	Frequency of help and documentation use
10.	Per cent of favorable/unfavorable user comments
11.	Number of repetitions of failed commands
12.	Number of runs of successes and of failures
13.	Number of times interface misleads the user
14.	Number of good and bad features recalled by users
15.	Number of available commands not invoked
16.	Number of regressive behaviors
17.	Number of users preferring your system
18.	Number of times users need to work around a problem
19.	Number of times the user is disrupted from a work task
20.	Number of times user loses control of the system
21.	Number of times user expresses frustration or satisfaction

Determining the worst case value depends on a number of things. Usually, it should be no lower than the now level.
The new product should provide some improvement on the current state of affairs, and so it seems that at least some of the usability attributes should provide worst case values that are better than the now level.

The designers figure that they should allow for the user to do a complete restoration of the VCR state in a maximum of two explicit user actions, though they recognize that the best case, at least in terms of the number of explicit actions, would require only one.

Tables 6.2 and 6.3, adapted from Whiteside, Bennett and Holtzblatt [377], provide a list of measurement criteria which can be used to determine the measuring method for a usability attribute and the possible ways to set the worst/best case and planned/ now level targets. Measurements such as those promoted by usability engineering are also called *usability metrics*.

Table 6.3 Possible ways to set measurement levels in a usability specification (adapted from Whiteside, Bennett and Holtzblatt [377]. Copyright 1988, reprinted with permission from Elsevier)

Set levels with respect to information on:
1. an existing system or previous version
2. competitive systems
3. carrying out the task without use of a computer system
4. an absolute scale
5. your own prototype
6. user's own earlier performance
7. each component of a system separately
8. a successive split of the difference between best and worst values observed in user tests

Table 6.4 Examples of usability metrics from ISO 9241

Usability objective	Effectiveness measures	Efficiency measures	Satisfaction measures
Suitability for the task	Percentage of goals achieved	Time to complete a task	Rating scale for satisfaction
Appropriate for trained users	Number of power features used	Relative efficiency compared with an expert user	Rating scale for satisfaction with power features
Learnability	Percentage of functions learned	Time to learn criterion	Rating scale for ease of learning
Error tolerance	Percentage of errors corrected successfully	Time spent on correcting errors	Rating scale for error handling

The ISO standard 9241, described earlier, also recommends the use of usability specifications as a means of requirements specification. Table 6.4 gives examples of usability metrics categorized by their contribution towards the three categories of usability: effectiveness, efficiency and satisfaction.

Problems with usability engineering

The major feature of usability engineering is the assertion of explicit usability metrics early on in the design process which can be used to judge a system once it is delivered. There is a very solid argument which points out that it is only through empirical approaches such as the use of usability metrics that we can reliably build more usable systems.

> The problem with usability metrics is that they rely on measurements of very specific user actions in very specific situations. However, at early stages of design, designers do not have this information.
>
> In setting the acceptable and unacceptable levels for backward recovery, there is an assumption that a button will be available to invoke the undo. In fact, the designer was already making an implicit assumption that the user would be making errors in the programming of the VCR. Why not address the origin of the programming errors, then maybe undo would not be necessary?
>
> We should recognize another inherent limitation for usability engineering, that is it provides a means of satisfying usability specifications and not necessarily usability. The

designer is still forced to understand why a particular usability metric enhances usability for real people.

4. ITERATIVE DESIGN AND PROTOTYPING:

The design can then be modified to correct any false assumptions that were revealed in the testing. This is the essence of *iterative design*, a purposeful design process which tries to overcome the inherent problems of incomplete requirements specification by cycling through several designs, incrementally improving upon the final product with each pass.

On the technical side, iterative design is described by the use of *prototypes*, artifacts that simulate or animate some but not all features of the intended system. There are three main approaches to prototyping:

Throw-away The prototype is built and tested. The design knowledge gained from this exercise is used to build the final product, but the actual prototype is discarded. Figure 6.5 depicts the procedure in using throw-away prototypes to arrive at a final requirements specification in order for the rest of the design process to proceed.

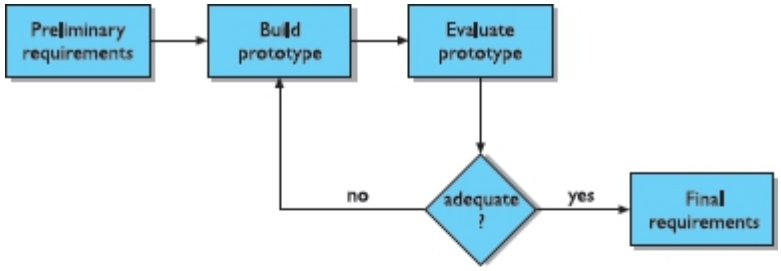


Figure 6.5 Throw-away prototyping within requirements specification

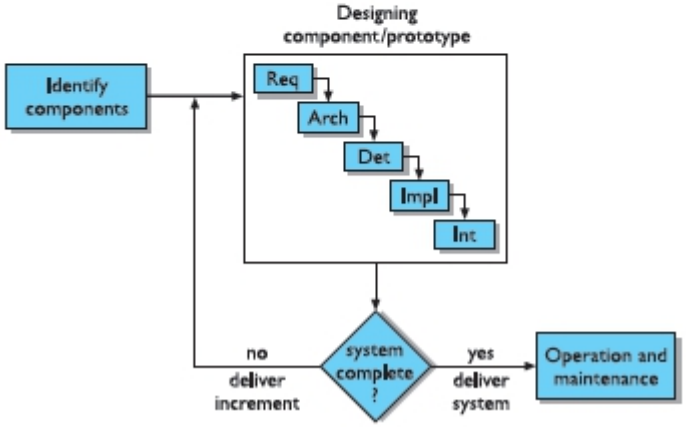


Figure 6.6 Incremental prototyping within the life cycle

Incremental The final product is built as separate components, one at a time. There is one overall design for the final system, but it is partitioned into independent and smaller components. The final product is then released as a series of products, each subsequent release including one more component. This is depicted in Figure 6.6.

Evolutionary Here the prototype is not discarded and serves as the basis for the next iteration of design. In this case, the actual system is seen as evolving from a very limited initial version to its final release, as depicted in Figure 6.7.

Evolutionary prototyping also fits in well with the modifications which must be made to the system that arise during the operation and maintenance activity in the life cycle. Prototypes differ according to the amount of functionality and performance they provide relative to the final product.

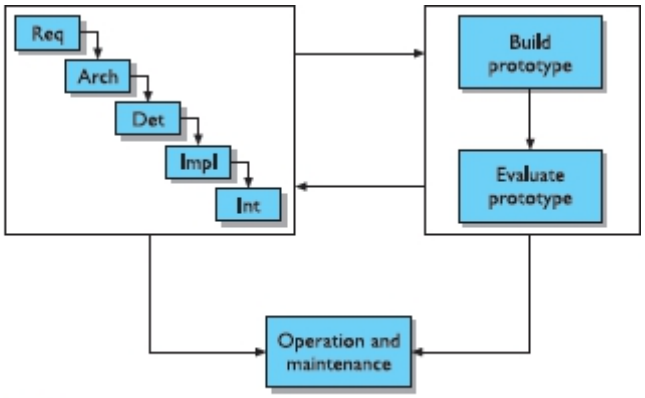


Figure 6.7 Evolutionary prototyping throughout the life cycle

An *animation* of requirements can involve real functionality, or limited functionality to simulate only a small aspect of the interactive behavior for evaluative purposes. At the other extreme, full functionality can be provided at the expense of other performance characteristics, such as speed or error tolerance.

On the management side, there are several potential problems, as pointed out by Sommerville [327]:

Time: Building prototypes takes time and, if it is a throw-away prototype, it can be seen as precious time taken away from the real design task. So the value of prototyping is only appreciated if it is fast, hence the use of the term *rapid prototyping*.

However, rapid development and manipulation of a prototype should not be mistaken for rushed evaluation which might lead to erroneous results and invalidate the only advantage of using a prototype in the first place.

Planning : Most project managers do not have the experience necessary for adequately planning and costing a design process which involves prototyping.

Non-functional features : Often the most important features of a system will be non-functional ones, such as safety and reliability, and these are precisely the kinds of features which are sacrificed in developing a prototype.

For evaluating usability features of a prototype, response time – yet another feature often compromised in a prototype – could be critical to product acceptance. This problem is similar to the technical issue of prototype realism.

Contracts: The design process is often governed by contractual agreements between customer and designer which are affected by many of these managerial and technical issues. Prototypes

and other implementations cannot form the basis for a legal contract, and so an iterative design process will still require documentation which serves as the binding agreement.

There must be an effective way of translating the results derived from prototyping into adequate documentation. A rapid prototyping process might be amenable to quick changes, but that does not also apply to the design process.

Techniques for prototyping

Storyboards:

A prototype is the *storyboard*, which is a graphical depiction of the outward appearance of the intended system, without any accompanying system functionality. Storyboards do not require much in terms of computing power to construct; in fact, they can be mocked up without the aid of any computing resource.

> The origins of storyboards are in the film industry, where a series of panels roughly depicts snapshots from an intended film sequence in order to get the idea across about the eventual scene.
>
> Animation illustrates the dynamic aspects of the intended user–system interaction, which may not be possible with traditional paper-based storyboards. If not animated, storyboards usually include annotations and scripts indicating how the interaction will occur.

Limited functionality simulations

More functionality must be built into the prototype to demonstrate the work that the application will accomplish. Storyboards and animation techniques are not sufficient for this purpose, as they cannot portray adequately the interactive aspects of the system.

To do this, some portion of the functionality must be *simulated* by the design team. Programming support for simulations means a designer can rapidly build graphical and textual interaction objects and attach some behavior to those objects, which mimics the system's functionality.

DESIGN RATIONALE:

In designing any computer system, many decisions are made as the product goes from a set of vague customer requirements to a deliverable entity.

> *Design rationale* is the information that explains why a computer system is the way it is, including its structural or architectural description and its functional or behavioral description.
>
> It is beneficial to have access to the design rationale for several reasons:
>
> An explicit form, a design rationale provides a communication mechanism among the members of a design team so that during later stages of design and/or maintenance it is possible to understand what critical decisions were made, what alternatives were

investigated (and, possibly, in what order) and the reason why one alternative was chosen over the others.

Accumulated knowledge in the form of design rationales for a set of products can be reused to transfer what has worked in one situation to another situation which has similar needs.

The design rationale can capture the context of a design decision in order that a different design team can determine if a similar rationale is appropriate for their product.

The effort required to produce a design rationale forces the designer to deliberate more carefully about design decisions.

The process of deliberation can be assisted by the design rationale technique by suggesting how arguments justifying or discarding a particular design option are formed.

In the area of HCI, design rationale has been particularly important, again for several reasons:

There is usually no single best design alternative. More often, the designer is faced with a set of trade-offs between different alternatives.

For example, a graphical interface may involve a set of actions that the user can invoke by use of the mouse and the designer must decide whether to present each action as a 'button' on the screen, which is always visible, or hide all of the actions in a menu which must be explicitly invoked before an action can be chosen.

The former option maximizes the operation visibility but the latter option takes up less screen space. It would be up to the designer to determine which criterion for evaluating the options was more important and then communicating that information in design rationale.

Even if an optimal solution did exist for a given design decision, the space of alternatives is so vast that it is unlikely a designer would discover it. In this case, it is important that the designer indicates all alternatives that have been investigated.

Then later on it can be determined if she has not considered the best solution or had thought about it and discarded it for some reason. In project management, this kind of accountability for design is good.

The usability of an interactive system is very dependent on the context of its use. The flashiest graphical interface is of no use if the end-user does not have access to a high-quality graphics display or a pointing device. Capturing the context in which a design decision is made will help later when new products are designed.

The classification to describe various design rationale techniques.

The first set of techniques concentrates on providing a historical record of design decisions and is very much tailored for use during actual design discussions. These techniques are referred to as process-oriented design rationale because they are meant to be integrated in the actual design process itself.

The next category is not so concerned with historical or process-oriented information but rather with the structure of the space of all design alternatives, which can be reconstructed by post hoc consideration of the design activity. The structure-oriented approach does not capture historical information.

The final category of design rationale concentrates on capturing the claims about the psychology of the user that are implied by an interactive system and the tasks that are performed on them.

A design rationale for a complex system can be very large and the exploration of the design space changes over time. The kind of information stored in a given design rationale will affect how that vast amount of information can be effectively managed and browsed.

Process-oriented design rationale

Much of the work on design rationale is based on Rittel's *issue-based information system*, or *IBIS*, a style for representing design and planning dialog developed in the 1970s [308].

In IBIS (pronounced 'ibbiss'), a hierarchical structure to a design rationale is created. A root *issue* is identified which represents the main problem or question that the argument is addressing.

Various *positions* are put forth as potential resolutions for the root issue, and these are depicted as descendants in the IBIS hierarchy directly connected to the root issue. Each position is then supported or refuted by *arguments*, which modify the relationship between issue and position.

The hierarchy grows as secondary issues are raised which modify the root issue in some way. Each of these secondary issues is in turn expanded by positions and arguments, further sub-issues, and so on.

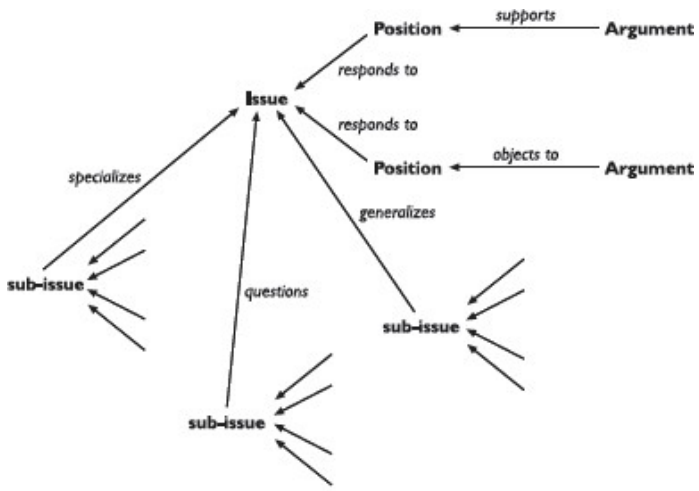


Figure 6.8 The structure of a gIBIS design rationale

A graphical version of IBIS has been defined by Conklin and Yakemovic [77], called *gIBIS* (pronounced 'gibbiss'), which makes the structure of the design rationale more

apparent visually in the form of a directed graph which can be directly edited by the creator of the design rationale.

Figure 6.8 gives a representation of the gIBIS vocabulary. Issues, positions and arguments are nodes in the graph and the connections between them are labeled to clarify the relationship between adjacent nodes. So, for example, an issue can suggest further sub-issues, or a position can respond to an issue or an argument can support a position.

The gIBIS structure can be supported by a hypertext tool to allow a designer to create and browse various parts of the design rationale.

There have been other versions of the IBIS notation, both graphical and textual, besides gIBIS. Most versions retain the distinction between issues, positions and arguments.

The University of Colorado has attempted to link PHI argumentation to computer-aided design (CAD) tools to allow critique of design (in their example, the design of a kitchen) as it occurs.
When the CAD violates some known design rule, the designer is warned and can then browse a PHI argument to see the rationale for the design rule The use of IBIS and any of its descendants is process oriented, as we described above.

It is intended for use during design meetings as a means of recording and structuring the issues deliberated and the decisions made. It is also intended to preserve the order of deliberation and decision making for a particular product, placing less stress on the generalization of design knowledge for use between different products.

Design space analysis:

MacLean and colleagues [222] have proposed a more deliberative approach to design rationale which emphasizes a post hoc structuring of the space of design alternatives that have been considered in a design project. Their approach, embodied in the Questions, Options and Criteria (QOC) notation, is characterized as *design space analysis* (Figure 6.9).

The design space is initially structured by a set of questions representing the major issues of the design. Since design space analysis is structure oriented, it is not so important that the questions recorded are the actual questions asked during design meetings. Rather, these questions represent an agreed characterization of the

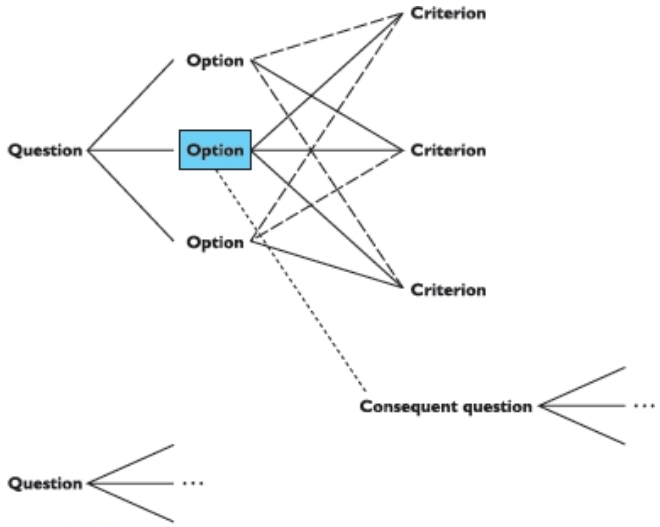


Figure 6.9 The QOC notation

issues raised based on reflection and understanding of the actual design activities. Questions in a design space analysis are therefore similar to issues in IBIS except in the way they are captured. Options provide alternative solutions to the question.

They are assessed according to some criteria in order to determine the most favorable option. In Figure 6.9 an option which is favorably assessed in terms of a criterion is linked with a solid line, whereas negative links have a dashed line.

The most favorable option is boxed in the diagram. The key to an effective design space analysis using the QOC notation is deciding the right questions to use to structure the space and the correct criteria to judge the options.

The initial questions raised must be sufficiently general that they cover a large enough portion of the possible design space, but specific enough that a range of options can be clearly identified.

In the example of the action buttons versus the menu of actions described earlier, we could contextualize the general principle of operation visibility as the criterion that all possible actions are displayed at all times.

It can be very difficult to decide from a design space analysis which option is most favorable. The positive and negative links in the QOC notation do not provide all of the context for a trade-off decision.

There is no provision for indicating, for example, that one criterion is more important than any of the others and the most favorable option must be positively linked.

Another structure-oriented technique, called Decision Representation Language (DRL), developed by Lee and Lai, structures the design space in a similar fashion to QOC, though its language is somewhat larger and it has a formal semantics.

The questions, options and criteria in DRL are given the names: decision problem, alternatives and goals. QOC assessments are represented in DRL by a more complex language for relating goals to alternatives.

The sparse language in QOC used to assess an option relative to a criterion (positive or negative assessment only) is probably insufficient, but there is a trade-off involved in adopting a more complex vocabulary which may prove too difficult to use in practice. The advantage of the formal semantics of DRL is that the design rationale can be used as a computational mechanism to help manage the large volume of information.

For example, DRL can track the dependencies between different decision problems, so that subsequent changes to the design rationale for one decision problem can be automatically propagated to other dependent problems.

Design space analysis directly addresses the claim that no design activity can hope to uncover all design possibilities, so the best we can hope to achieve is to document the small part of the design space that has been investigated.

An advantage of the post hoc technique is that it can abstract away from the particulars of a design meeting and therefore represent the design knowledge in such a way that it can be of use in the design of other products.

The major disadvantage is the increased overhead such an analysis warrants. More time must be taken away from the design activity to do this separate documentation task. When time is scarce, these kinds of overhead costs are the first to be trimmed.

Psychological design rationale:

The final category of design rationale tries to make explicit the psychological claims of usability inherent in any interactive system in order better to suit a product for the tasks users have.

This psychological design rationale has been introduced by Carroll and Rosson, People use computers to accomplish some tasks in their particular work domain, as we have seen before.

When designing a new interactive system, the designers take into account the tasks that users currently perform and any new ones that they may want to perform.

This task identification serves as part of the requirements for the new system, and can be done through empirical observation of how people perform their work currently and presented through informal language or a more formal task analysis language

When the new system is implemented, or becomes an *artifact*, further observation reveals that in addition to the required tasks it was built to support, it also supports users in tasks that the designer never intended.

Once designers understand these new tasks, and the associated problems that arise between them and the previously known tasks, the new task definitions can serve as requirements for future artifacts.

When the first electronic spreadsheet, *VisiCalc*, was marketed in the late 1970s, it was presented simply as an automated means of supporting tabular calculation, a task commonly used in the accounting world.

The purpose of psychological design rationale is to support this natural task– artifact cycle of design activity.

The main emphasis is not to capture the designer's intention in building the artifact. Rather, psychological design rationale aims to make explicit the consequences of a design for the user, given an understanding of what tasks he intends to perform.

The first step in the psychological design rationale is to identify the tasks that the proposed system will address and to characterize those tasks by questions that the user tries to answer in accomplishing them.

DESIGN RULES:

PRINCIPLES TO SUPPORT USABILITY:

The most abstract design rules are general principles, which can be applied to the design of an interactive system in order to promote its usability.

Derivation of principles for interaction has usually arisen out of a need to explain why a paradigm is successful and when it might not be. Principles can provide the repeatability which paradigms in themselves cannot provide. In this section we present a collection of usability principles.

Since it is too bold an objective to produce a comprehensive catalog of such principles, our emphasis will be on structuring the presentation of usability principles in such a way that the catalog can be easily extended as our knowledge increases.

The principles we present are first divided into three main categories:

Learnability – the ease with which new users can begin effective interaction and achieve maximal performance.

Flexibility – the multiplicity of ways in which the user and system exchange information.

Robustness – the level of support provided to the user in determining successful achievement and assessment of goals.

In most cases, we are able to situate these more specific principles within a single category, but we have made explicit those cases when a principle falls into two of the above categories.

Table 7.1 Summary of principles affecting learnability

Principle	Definition	Related principles
Predictability	Support for the user to determine the effect of future action based on past interaction history	Operation visibility
Synthesizability	Support for the user to assess the effect of past operations on the current state	Immediate/eventual honesty
Familiarity	The extent to which a user's knowledge and experience in other real-world or computer-based domains can be applied when interacting with a new system	Guessability, affordance
Generalizability	Support for the user to extend knowledge of specific interaction within and across applications to other similar situations	–
Consistency	Likeness in input–output behavior arising from similar situations or similar task objectives	–

Learnability:

Learnability concerns the features of the interactive system that allow novice users to understand how to use it initially and then how to attain a maximal level of performance.

Table 7.1 contains a summary of the specific principles that support learn ability, which we will describe below.

Predictability

Except when interacting with some video games, a user does not take very well to surprises. Predictability of an interactive system means that the user's knowledge of the interaction history is sufficient to determine the result of his future interaction with it. There are many degrees to which predictability can be satisfied.

The knowledge can be restricted to the presently perceivable information, so that the user need not remember anything other than what is currently observable.

The knowledge requirement can be increased to the limit where the user is actually forced to remember what every previous keystroke was and what every previous screen display contained (and the order of each!) in order to determine the consequences of the next input action.

Predictability of an interactive system is distinguished from deterministic behavior of the computer system alone. Most computer systems are ultimately deterministic machines, so that given the state at any one point in time and the operation which is to be performed at that time, there is only one possible state that can result.

Predictability is a user-centered concept; it is deterministic behavior from the perspective of the user. It is not enough for the behavior of the computer system to be determined completely from its state, as the user must be able to take advantage of the determinism.

Synthesizability

Predictability says nothing about the way the user forms a model of the system's behavior. In building up some sort of predictive model of the system's behavior, it is important for the user to assess the consequences of previous interactions in order to formulate a model of the behavior of the system. Synthesis, therefore, is the ability of the user to assess the effect of past operations on the current state.

When an operation changes some aspect of the internal state, it is important that the change is seen by the user. The principle of *honesty* relates to the ability of the user interface to provide an observable and informative account of such change.

In the best of circumstances, this notification can come *immediately*, requiring no further interaction initiated by the user.

At the very least, the notification should appear *eventually*, after explicit user directives to make the change observable good example of the distinction between immediacy and eventuality can be seen in the comparison between command language interfaces and visual desktop interfaces for a file management system.

. Suddenly, a careless typing error is transformed into unacceptable grammar as the sentence

We will prove the theorem holds as a corollary of the following lemma.

is transformed to

We will prove theorem holds as a corollary of the following lemma.

Familiarity

New users of a system bring with them a wealth of experience across a wide number of application domains. This experience is obtained both through interaction in the real world and through interaction with other computer systems.

For a new user, the familiarity of an interactive system measures the correlation between the user's existing knowledge and the knowledge required for effective interaction.

Generalizability

Users often try to extend their knowledge of specific interaction behavior to situations that are similar but previously unencountered. The generalizability of an interactive system supports this activity, leading to a more complete predictive model of the system for the user.

Generalizability can be seen as a form of consistency. Generalization can occur within a single application or across a variety of applications.

For example, in a graphical drawing package that draws a circle as a constrained form of ellipse, we would want the user to generalize that a square can be drawn as a constrained rectangle.

Consistency:

Consistency relates to the likeness in behavior arising from similar situations or similar task objectives. Consistency is probably the most widely mentioned principle in the literature on user interface design. 'Be consistent!' we are constantly urged.

The user relies on a consistent interface. However, the difficulty of dealing with consistency is that it can take many forms.

Consistency is not a single property of an interactive system that is either satisfied or not satisfied. Instead, consistency must be applied relative to something. Thus we have consistency in command naming, or consistency in command/argument invocation.

Another consequence of consistency having to be defined with respect to some other feature of the interaction is that many other principles can be 'reduced' to qualified instances of consistency.
Hence, familiarity can be considered as consistency with respect to past real-world experience, and generalizability as consistency with respect to experience with the same system or set of applications on the same platform.

Because of this pervasive quality of consistency, it might be argued that consistency should be a separate category of usability principles, on the same level as learnability, flexibility and robustness.

Table 7.2 Summary of principles affecting flexibility

Principle	Definition	Related principles
Dialog initiative	Allowing the user freedom from artificial constraints on the input dialog imposed by the system	System/user pre-emptiveness
Multi-threading	Ability of the system to support user interaction pertaining to more than one task at a time	Concurrent vs. interleaving, multi-modality
Task migratability	The ability to pass control for the execution of a given task so that it becomes either internalized by the user or the system or shared between them	–
Substitutivity	Allowing equivalent values of input and output to be arbitrarily substituted for each other	Representation multiplicity, equal opportunity
Customizability	Modifiability of the user interface by the user or the system	Adaptivity, adaptability

Flexibility:

Flexibility refers to the multiplicity of ways in which the end-user and the system exchange information. We identify several principles that contribute to the flexibility of interaction, and these are summarized in Table 7.2.

Dialog initiative

When considering the interaction between user and system as a dialog between partners it is important to consider which partner has the initiative in the conversation. The system can initiate

all dialog, in which case the user simply responds to requests for information. We call this type of dialog *system pre-emptive*.

For example, a modal dialog box prohibits the user from interacting with the system in any way that does not direct input to the box. Alternatively, the user may be entirely free to initiate any action towards the system, in which case the dialog is *user pre-emptive*.

The system may control the dialog to the extent that it prohibits the user from initiating any other desired communication concerning the current task or some other task the user would like to perform. From the user's perspective, a system-driven interaction hinders flexibility whereas a user-driven interaction favours it.

Multi-threading

A thread of a dialog is a coherent subset of that dialog. In the user–system dialog, we can consider a thread to be that part of the dialog that relates to a given user task.

Multi-threading of the user–system dialog allows for interaction to support more than one task at a time. *Concurrent* multi-threading allows simultaneous communication of information pertaining to separate tasks. *Interleaved* multi-threading permits a temporal overlap between separate tasks, but stipulates that at any given instant the dialog is restricted to a single task.

Multi-modality of a dialog is related to multi-threading. Coutaz has characterized two dimensions of multi-modal systems. First, we can consider how the separate modalities (or channels of communication) are combined to form a single input or output expression. Multiple channels may be available, but any one expression may be restricted to just one channel (keyboard or audio, for example).

Task migratability*:*

Task migratability concerns the transfer of control for execution of tasks between system and user. It should be possible for the user or system to pass the control of a task over to the other or promote the task from a completely internalized one to a shared and cooperative venture.

Hence, a task that is internal to one can become internal to the other or shared between the two partners. Spell-checking a paper is a good example of the need for task migratability.

Robustness:

In a work or task domain, a user is engaged with a computer in order to achieve some set of goals. The robustness of that interaction covers features that support the successful achievement and assessment of the goals. A summary of these principles is presented in Table 7.3.

Table 7.3 Summary of principles affecting robustness

Principle	Definition	Related principles
Observability	Ability of the user to evaluate the internal state of the system from its perceivable representation	Browsability, static/dynamic defaults, reachability, persistence, operation visibility
Recoverability	Ability of the user to take corrective action once an error has been recognized	Reachability, forward/ backward recovery, commensurate effort
Responsiveness	How the user perceives the rate of communication with the system	Stability
Task conformance	The degree to which the system services support all of the tasks the user wishes to perform and in the way that the user understands them	Task completeness, task adequacy

Observability :

Observability allows the user to evaluate the internal state of the system by means of its perceivable representation at the interface.Evaluation allows the user to compare the current observed state with his intention within the task–action plan, possibly leading to a plan revision.

Observability can be discussed through five other principles: browsability, defaults, reachability, persistence and operation visibility. Operation visibility was covered in Section 7.2.1 in relation to predictability.

The remaining four are discussed next.*Browsability* allows the user to explore the current internal state of the system via the limited view provided at the interface. Usually the complexity of the domain does not allow the interface to show all of the relevant domain concepts at once.

Responsiveness:

Responsiveness measures the rate of communication between the system and the user. Response time is generally defined as the duration of time needed by the system to express state changes to the user.

In general, short durations and instantaneous response times are desirable. Instantaneous means that the user perceives system reactions as immediate. But even in situations in which an instantaneous response cannot be obtained, there must be some indication to the user that the system has received the request for action and is working on a response.

As significant as absolute response time is response time *stability*. Response time stability covers the invariance of the duration for identical or similar computational resources.

Task conformance:

Since the purpose of an interactive system is to allow a user to perform various tasks in achieving certain goals within a specific application domain. *Task completeness* addresses the coverage issue and *task adequacy* addresses the user's understanding of the tasks.

> It is not sufficient that the computer system fully implements some set of computational services that were identified at early specification stages. It is essential that the system allows the user to achieve any of the desired tasks in a particular work domain as identified by a task analysis that precedes system specification .
>
> Task completeness refers to the level to which the system services can be mapped onto all of the user tasks. However, it is quite possible that the provision of a new computerbased tool will suggest to a user some tasks that were not even conceivable before the tool. Therefore, it is also desirable that the system services be suitably general so that the user can define new tasks.

STANDARDS

Standards for interactive system design are usually set by national or international bodies to ensure compliance with a set of design rules by a large community.

Standards can apply specifically to either the hardware or the software used to build the interactive system. Smith [324] points out the differing characteristics between hardware and software, which affect the utility of design standards applied to them:

Underlying theory Standards for hardware are based on an understanding of physiology or ergonomics/human factors, the results of which are relatively well known, fixed and readily adaptable to design of the hardware.

On the other hand, software standards are based on theories from psychology or cognitive science, which are less well formed, still evolving and not very easy to interpret in the language of software design. Consequently, standards for hardware can directly relate to a hardware specification and still reflect the underlying theory, whereas software standards would have to be more vaguely worded.

Change Hardware is more difficult and expensive to change than software, which is usually designed to be very flexible. Consequently, requirements changes for hardware do not occur as frequently as for software. Since standards are also relatively stable, they are more suitable for hardware than software.

Historically, for these reasons, a given standards institution, such as the British Standards Institution (BSI) or the International Organization for Standardization (ISO) or a national military agency, has had standards for hardware in place before any for software.

For example, the UK Ministry of Defence has published an Interim Defence Standard 00–25 on *Human Factors for Designers of Equipment*, produced in 12 parts:

Part 1 Introduction
Part 2 Body Size
Part 3 Body Strength and Stamina
Part 4 Workplace Design
Part 5 Stresses and Hazards

Part 6 Vision and Lighting
Part 7 Visual Displays
Part 8 Auditory Information
Part 9 Voice Communication
Part 10 Controls
Part 11 Design for Maintainability
Part 12 Systems

Only the last of these is concerned with the software design process. The international standard ISO 9241, entitled *Ergonomic Requirements for Office Work with Visual Display Terminals (VDT)s*, has 17 parts.

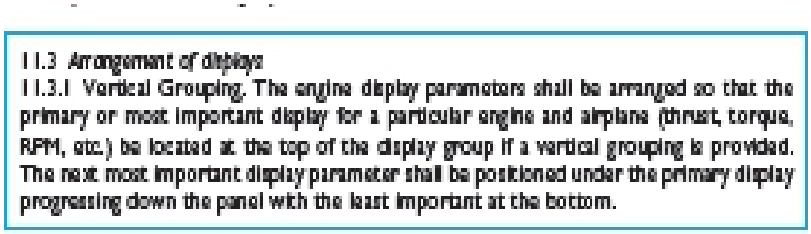
Seven of these are concerned with hardware issues – requirements for visual display, keyboard layout, workstation layout, environment, display with reflections, display colors and non-keyboard input devices.

Seven parts are devoted to software issues –

General dialog principles, menu dialogs, presentation of information, user guidance, command dialogs, direct manipulation dialogs and form-filling dialogs.

However, standards covering software issues are now being produced, for example, the draft standard ISO 14915 covers software ergonomics for multimedia user interfaces.

Figure 7.1 provides examples of the language of standards for displays.



11.3 *Arrangement of displays*
11.3.1 Vertical Grouping. The engine display parameters shall be arranged so that the primary or most important display for a particular engine and airplane (thrust, torque, RPM, etc.) be located at the top of the display group if a vertical grouping is provided. The next most important display parameter shall be positioned under the primary display progressing down the panel with the least important at the bottom.

(a) A typical example of a military standard

5.1 *Subdivision of the display area*
In consideration of a simple, fast and accurate visual acquisition, the display area shall be divided into different sub-areas.
Such a division should be:

- Input area
- Output area
- Area for operational indications (such as status and alarms)

(b) From German standard DIN 66 234 Part 3 (1984), adapted from Smith [324]

5.15.3.2.1 *Standardization*
The content of displays within a system shall be presented in a consistent manner.

(c) From US military standard MIL-STD-1472C, revised (1983), adapted from Smith [324]

Figure 7.1 Sample design standards for displays. Adapted from Smith [324].

In the beginning of that document, the following definition of usability is given:

Usability The effectiveness, efficiency and satisfaction with which specified users achieve specified goals in particular environments.

Effectiveness The accuracy and completeness with which specified users can achieve specified goals in particular environments.

Efficiency The resources expended in relation to the accuracy and completeness of goals achieved.

Satisfaction The comfort and acceptability of the work system to its users and other people affected by its use.

The importance of such a definition in the standard is as a means of describing explicit measurements for usability.

The strength of a standard lies in its ability to force large communities to abide – the so-called authority we have referred to earlier. It should be noted that such authority does not necessarily follow from the publication of a standard by a national or international body.

GUIDELINES (Apr/May 2017)

As a result, the majority of design rules for interactive systems are suggestive and more general guidelines. Our concern in examining the wealth of available guidelines is in determining their applicability to the various stages of design.

> The more abstract the guideline, the more it resembles the principles that which would be most suited to requirements specification. The more specific the guideline, the more suited it is to detailed design.
>
> The guidelines can also be automated to some extent, providing a direct means for translating detailed design specifications into actual implementation. There are a vast amount of published guidelines for interactive system design (they are frequently referred to as guidelines for user interface design).
>
> Several books and technical reports contain huge catalogs of guidelines. A classic example was a very general list compiled by Smith and Mosier in 1986 at the Mitre Corporation and sponsored by the Electronic Systems Division of the US Air Force [325].

The basic categories of the Smith and Mosier guidelines are:

1. Data Entry
2. Data Display
3. Sequence Control
4. User Guidance
5. Data Transmission
6. Data Protection

Each of these categories is further broken down into more specific subcategories which contain the particular guidelines. Figure 7.2 provides an example of the information contained in the Smith and Mosier guidelines.

A striking feature of this compendium of guidelines is the extensive cross-referencing within the catalog, and citation to published work that supports each guideline.

The Mitre Corporation has taken advantage of this structure and implemented the Smith and Mosier guidelines on a hypertext system, which provides rapid traversal of the network of guidelines to investigate the cross-references.

1. **Data Entry**

1.1 *Position Designation*

1.1–1 **Distinctive Cursor**
For position designation on an electronic display, provide a movable cursor with distinctive visual features (shape, blink, etc.).

Exception When position designation involves only selection among displayed alternatives, highlighting selected items might be used instead of a separately displayed cursor.

Comment When choosing a cursor shape, consider the general content of the display. For instance, an underscore cursor would be difficult to see on a display of underscored text, or on a graphical display containing many other lines.

Comment If the cursor is changed to denote different functions (e.g. to signal deletion rather than entry), then each different cursor should be distinguishable from the others.

Comment If multiple cursors are used on the same display (e.g. one for alphanumeric entry and one for line drawing), then each cursor should be distinguishable from the others.

Reference Whitfield, Ball and Bird, 1983

See also 1.1–17 Distinctive multiple cursors
4.0–9 Distinctive cursor

Figure 7.2 Sample guideline from Smith and Mosier [325], courtesy of The MITRE Corporation

Table 7.4 Comparison of dialog styles mentioned in guidelines

Smith and Mosier [325]	Mayhew [230]
Question and answer	Question and answer
Form filling	Fill-in forms
Menu selection	Menus
Function keys	Function keys
Command language	Command language
Query language	–
Natural language	Natural language
Graphic selection	Direct manipulation

A major concern for all of the general guidelines is the subject of *dialog styles*, which in the context of these guidelines pertains to the means by which the user communicates input to the system, including how the system presents the communication device.

Smith and Mosier identify eight different dialog styles and Mayhew identifies seven (Table 7.4 for a comparison). The only real difference is the absence of query languages in

Mayhew's list, but we can consider a query language as a special case of a command language.

Most guidelines are applicable for the implementation of any one of these dialog styles in isolation. It is also important to consider the possibility of mixing dialog styles in one application. In contrasting the action and language paradigms in Chapter 4, we concluded that it is not always the case that one paradigm wins over the other for all tasks in an application and, therefore, an application may want to mix the two paradigms.

This equates to a mixing of dialog styles – a direct manipulation dialog being suitable for the action paradigm and a command language being suitable for the language paradigm. Mayhew provides guidelines and a technique for deciding how to mix dialog styles.

In moving from abstract guidelines to more specific and automated ones, it is necessary to introduce assumptions about the computer platform on which the interactive system is designed.

So, for example, in Apple's *Human Interface Guidelines*: *the Apple Desktop Interface*, there is a clear distinction between the abstract guidelines (or principles), independent of the specific Macintosh hardware and software, and the concrete guidelines, which assume them. The abstract
guidelines provide the so-called philosophy of programming that Apple would like designers to adopt in programming applications for the Macintosh.

The more concrete guidelines are then seen as more concrete manifestations of that philosophy. As an example, one abstract principle espoused in the Apple guidelines is *consistency*:

GOLDEN RULES AND HEURISTICS:

There are many sets of heuristics, but the most well used are Nielsen's ten heuristics, Shneiderman's eight golden rules and Norman's seven principles.

Shneiderman's Eight Golden Rules of Interface Design

Shneiderman's eight golden rules provide a convenient and succinct summary of the key principles of interface design. They are intended to be used during design but can also be applied, like Nielsen's heuristics, to the evaluation of systems.

1. *Strive for consistency* in action sequences, layout, terminology, command use and so on.
2. *Enable frequent users to use shortcuts*, such as abbreviations, special key sequences and macros, to perform regular, familiar actions more quickly.
3. *Offer informative feedback* for every user action, at a level appropriate to the magnitude of the action.
4. *Design dialogs to yield closure* so that the user knows when they have completed a task.
5. *Offer error prevention and simple error handling* so that, ideally, users are prevented from making mistakes and, if they do, they are offered clear and informative instructions to enable them to recover.
6. *Permit easy reversal of actions* in order to relieve anxiety and encourage exploration, since the user knows that he can always return to the previous state.

7. *Support internal locus of control* so that the user is in control of the system, which responds to his actions.
8. *Reduce short-term memory load* by keeping displays simple, consolidating multiple page displays and providing time for learning action sequences.

These rules provide a useful shorthand for the more detailed sets of principles described earlier..

Norman's Seven Principles for Transforming Difficult Tasksinto Simple Ones

Later, in his classic book *The Design of Everyday Things*, he summarizes user-centered design using the following seven principles:

1. *Use both knowledge in the world and knowledge in the head*. People work better when the knowledge they need to do a task is available externally – either explicitly or through the constraints imposed by the environment. But experts also need to be able to internalize regular tasks to increase their efficiency.

2. *Simplify the structure of tasks*. Tasks need to be simple in order to avoid complex problem solving and excessive memory load. There are a number of ways to simplify the structure of tasks.

One is to provide mental aids to help the user keep track of stages in a more complex task. Another is to use technology to provide the user with more information about the task and better feedback.

A third approach is to automate the task or part of it, as long as this does not detract from the user's experience. The final approach to simplification is to change the nature of the task so that it becomes something more simple. In all of this, it is important not to take control away from the user.

3. *Make things visible*: bridge the gulfs of execution and evaluation. The interface should make clear what the system can do and how this is achieved, and should enable the user to see clearly the effect of their actions on the system.

4. *Get the mappings right*. User intentions should map clearly onto system controls. User actions should map clearly onto system events. So it should be clear what does what and by how much. Controls, sliders and dials should reflect the task – so a small movement has a small effect and a large movement a large effect.

5. *Exploit the power of constraints*, both natural and artificial. Constraints are things in the world that make it impossible to do anything but the correct action in the correct way. A simple example is a jigsaw puzzle, where the pieces only fit together in one way. Here the physical constraints of the design guide the user to complete the task.

6. *Design for error*. To err is human, so anticipate the errors the user could make and design recovery into the system.
7. *When all else fails, standardize*. If there are no natural mappings then arbitrary mappings should be standardized so that users only have to learn them once. It is this standardization principle that enables drivers to get into a new car and drive it with very little difficulty – key controls are standardized.

HCI PATTERNS:

Patterns are an approach to capturing and reusing this knowledge – of abstracting the essential details of successful design so that these can be applied again and again in new situations.

Patterns originated in architecture, where they have been used successfully, and they are also used widely in software development to capture solutions to common programming problems. More recently they have been used in interface and web design.

A pattern is an invariant solution to a recurrent problem within a specific context. Patterns address the problems that designers face by providing a 'solution statement'. This is best illustrated by example. Alexander, who initiated the pattern concept, proposes a pattern for house building called 'Light on Two Sides of Every Room'.

The problem being addressed here is that When they have a choice, people will always gravitate to those rooms which have light on two sides, and leave the rooms which are lit only from one side unused and empty.

The specific implementation of the pattern will depend on the circumstance and the designer's creativity. There are many examples of HCI patterns, and the interested reader is referred to pattern collections and languages such as [345, 37, 356] and the Pattern Gallery, which illustrates some of the various forms used in HCI patterns .

A well known example, 'go back to a safe place', adapted from Tidwell's Common Ground collection, is given as an illustration (Figure 7.3).

The pattern also has references to other patterns, indicating both the context in which it can be applied (the top references) and the patterns that may be needed to complete it (the bottom references).

This connects the patterns together into a *language*. Patterns in isolation have limited use, but by traversing the hierarchy, through these references, the user is assisted in generating a complete design.

Patterns and pattern languages are characterized by a number of features, which, taken as a whole, distinguish them from other design rules:

They capture design practice and embody knowledge about successful solutions:

i)They come from practice rather than psychological theory.
ii) They capture the essential common properties of good design: they do not tell the designer *how* to do something but what needs to be done and why.

iii) They represent design knowledge at varying levels, ranging from social and organizational issues through conceptual design to detailed widget design.

iv) They are not neutral but embody values within their rationale. Alexander's language clearly expresses his values about architecture. HCI patterns can express values about what is humane in interface design.

v) The concept of a pattern language is generative and can therefore assist in the development of complete designs.
vi) They are generally intuitive and readable and can therefore be used for communication between all stakeholders.

Patterns are a relatively recent addition to HCI representations, in which there are still many research issues to resolve. For instance, it is not clear how patterns can best be identified or how languages should be structured to reflect the temporal concerns of interaction.

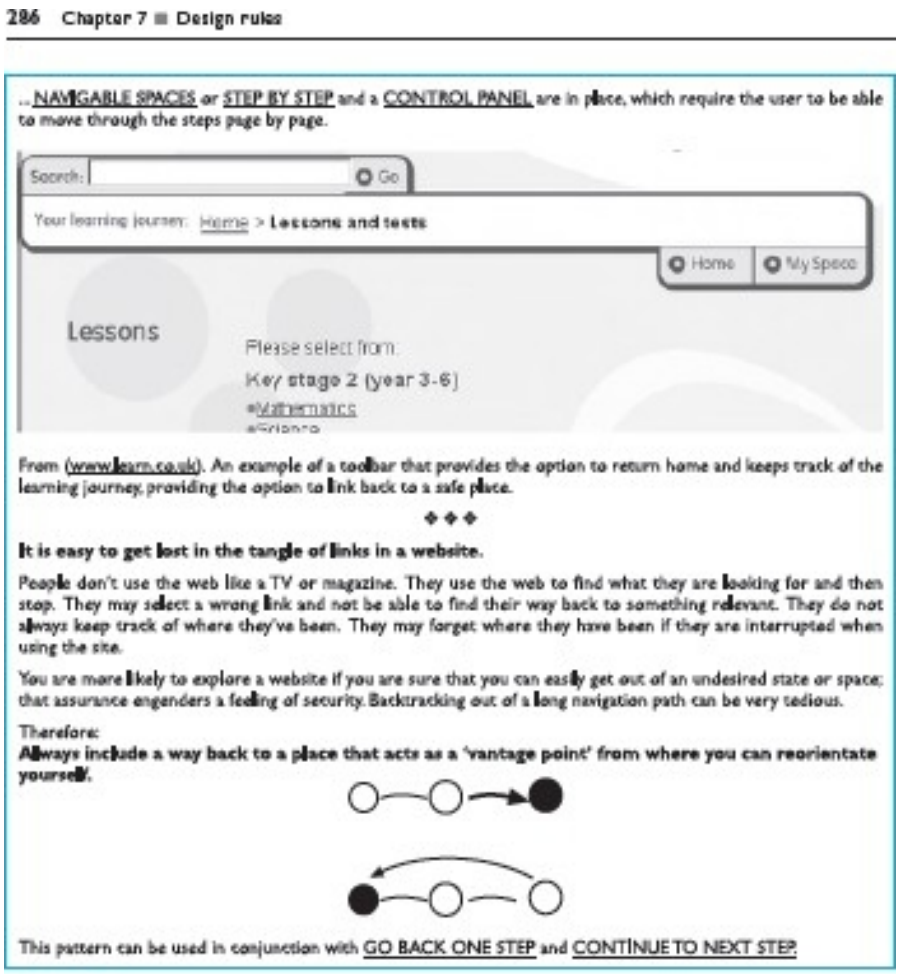


Figure 7.3 An example pattern 'go back to a safe place' adapted from Tidwell's Common Ground collection. Courtesy of Jenifer Tidwell

WHAT IS EVALUATION:

Evaluation should not be thought of as a single phase in the design process (still less as an activity tacked on the end of the process if time permits). Ideally, evaluation should occur throughout the design life cycle, with the results of he evaluation feeding back into modifications to the design.

GOALS OF EVALUATION:

Evaluation has three main goals: to assess the extent and accessibility of the system's functionality, to assess users' experience of the interaction, and to identify any specific problems with the system.

The system's functionality is important in that it must accord with the user's requirements. In other words, the design of the system should enable users to perform their intended tasks more easily. This includes not only making the appropriate functionality available within the system, but making it clearly reachable by the user in terms of the actions that the user needs to take to perform the task.

It also involves matching the use of the system to the user's expectations of the task. For example, if a filing clerk is used to retrieving a customer's file by the postal address the same capability (at least) should be provided in the computerized file system.

Evaluation at this level may also include measuring the user's performance with the system, to assess the effectiveness of the system in supporting the task.

In addition to evaluating the system design in terms of its functional capabilities, it is important to assess the user's experience of the interaction and its impact upon him. This includes considering aspects such as how easy the system is to learn, its usability and the user's satisfaction with it.

The final goal of evaluation is to identify specific problems with the design. These may be aspects of the design which, when used in their intended context, cause unexpected results, or confusion amongst users. This is, of course, related to both the functionality and usability of the design (depending on the cause of the problem).

.

EVALUATION THROUGH EXPERT ANALYSIS:

If the design itself can be evaluated, expensive mistakes can be avoided, since the design can be altered prior to any major resource commitments.

However, it can be expensive to carry out user testing at regular intervals during the design process, and it can be difficult to get an accurate assessment of the experience of interaction from incomplete designs and prototypes.

Consequently, a number of methods have been proposed to evaluate interactive systems through expert analysis. These depend upon the designer, or a human factors expert, taking the design and assessing the impact that it will have upon a typical user.

The basic intention is to identify any areas that are likely to cause difficulties because they violate known cognitive principles, or ignore accepted empirical results. These methods can be used at any stage in the development process from a design specification, through storyboards and prototypes, to full implementations, making them flexible evaluation approaches.

They are also relatively cheap, since they do not require user involvement. However, they do not assess actual use of the system, only whether or not a system upholds accepted usability principles.

Cognitive walkthrough:

Cognitive walkthrough was originally proposed and later revised by Polson and colleagues [294, 376] as an attempt to introduce psychological theory into the informal and subjective walkthrough technique.

The origin of the cognitive walkthrough approach to evaluation is the code walkthrough familiar in software engineering. Walkthroughs require a detailed review of a sequence of actions.

In the code walkthrough, the sequence represents a segment of the program code that is stepped through by the reviewers to check certain characteristics (for example, that coding style is adhered to, conventions for spelling variables versus procedure calls, and to check that system-wide invariants are not violated).

In the cognitive walkthrough, the sequence of actions refers to the steps that an interface will require a user to perform in order to accomplish some known task. The evaluators then 'step through' that action sequence to check it for potential usability problems.

To do a walkthrough (the term walkthrough from now on refers to the cognitive walkthrough, and not to any other kind of walkthrough), you need four things:

1. A specification or prototype of the system. It doesn't have to be complete, but it should be fairly detailed. Details such as the location and wording for a menu can make a big difference.

2. A description of the task the user is to perform on the system. This should be a representative task that most users will want to do.

3. A complete, written list of the actions needed to complete the task with the proposed system.

4. An indication of who the users are and what kind of experience and knowledge the evaluators can assume about them.

For each action, the evaluators try to answer the following four questions for each step in the action sequence.

1. **Is the effect of the action the same as the user's goal at that point?** Each user action will have a specific effect within the system.

Is this effect the same as what the user is trying to achieve at this point? For example, if the effect of the action is to save a document, is 'saving a document' what the user wants to do?

2. **Will users see that the action is available?** Will users see the button or menu item, for example, that is used to produce the action? This is *not* asking whether they will recognize that the button is the one they want.
This is merely asking whether it is visible to them at the time when they will need to use it. Instances where the answer to this question might be 'no' are, for example, where a VCR remote control has a covered panel of buttons or where a menu item is hidden away in a submenu.

3. **Once users have found the correct action, will they know it is the one they need?**
This complements the previous question. It is one thing for a button or menu item to be visible, but will the user recognize that it is the one he is looking for to complete his task? Where the previous question was about the visibility of the action, this one is about whether its meaning and effect is clear.

4. **After the action is taken, will users understand the feedback they get?**

In order to determine if they have accomplished their goal, users need appropriate feedback. It is vital to document the cognitive walkthrough to keep a record of what is good and what needs improvement in the design. It is therefore a good idea to produce some standard evaluation forms for the walkthrough.

This problem report sheet should indicate the system being built (the version, if necessary), the date, the evaluators and a detailed description of the usability problem.

It is also useful to indicate the severity of the problem, that is whether the evaluators think this problem will occur often, and how serious it will be for the users. This information will help the designers to decide priorities for correcting the design, since it is not always possible to fix every problem.

Heuristic evaluation:

A heuristic is a guideline or general principle or rule of thumb that can guide a design decision or be used to critique a decision that has already been made.

Heuristic evaluation, developed by Jakob Nielsen and Rolf Molich, is a method for structuring the critique of a system using a set of relatively simple and general heuristics.

Heuristic evaluation can be performed on a design specification so it is useful for evaluating early design. But it can also be used on prototypes, storyboards and fully functioning systems. It is therefore a flexible, relatively cheap approach. Hence it is often considered a *discount usability* technique.

The general idea behind heuristic evaluation is that several evaluators independently critique a system to come up with potential usability problems. It is important that there be several of these evaluators and that the evaluations be done independently.

Nielsen's experience indicates that between three and five evaluators is sufficient, with five usually resulting in about 75% of the overall usability problems being discovered.

These can be combined into an overall severity rating on a scale of 0–4:

0 = I don't agree that this is a usability problem at all
1 = Cosmetic problem only: need not be fixed unless extra time is available on project
2 = Minor usability problem: fixing this should be given low priority
3 = Major usability problem: important to fix, so should be given high priority
4 = Usability catastrophe: imperative to fix this before product can be released (Nielsen)

Nielsen's ten heuristics are:

1. **Visibility of system status** Always keep users informed about what is going on, through appropriate feedback within reasonable time. For example, if a system operation will take some time, give an indication of how long and how much is complete.

2. **Match between system and the real world** The system should speak the user's language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in natural and logical order.

3. **User control and freedom** Users often choose system functions by mistake and need a clearly marked 'emergency exit' to leave the unwanted state without having to go through an extended dialog. Support undo and redo.

4. **Consistency and standards** Users should not have to wonder whether words, situations or actions mean the same thing in different contexts. Follow platform conventions and accepted standards.

5. **Error prevention** Make it difficult to make errors. Even better than good error messages is a careful design that prevents a problem from occurring in the first place.

6. **Recognition rather than recall** Make objects, actions and options visible. The user should not have to remember information from one part of the dialog to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

7. **Flexibility and efficiency of use** Allow users to tailor frequent actions. Accelerators – unseen by the novice user – may often speed up the interaction for the expert user to such an extent that the system can cater to both inexperienced and experienced users.

8. **Aesthetic and minimalist design** Dialogs should not contain information that is irrelevant or rarely needed. Every extra unit of information in a dialog competes with the relevant units of information and diminishes their relative visibility.

9. **Help users recognize, diagnose and recover from errors** Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

10. **Help and documentation** Few systems can be used with no instructions so it may be necessary to provide help and documentation. Any such information should be easy to search, focussed on the user's task, list concrete steps to be carried out, and not be too large.

Once each evaluator has completed their separate assessment, all of the problems are collected and the mean severity ratings calculated.

Model-based evaluation:

A third expert-based approach is the use of models. Certain cognitive and design models provide a means of combining design specification and evaluation into the same framework.

> For example, the GOMS (goals, operators, methods and selection) model predicts user performance with a particular interface and can be used to filter particular design options.
>
> Similarly, lower-level modeling techniques such as the keystroke-level model provide predictions of the time users will take to perform low-level physical tasks.
>
> Design methodologies, such as design rationale also have a role to play in evaluation at the design stage. Design rationale provides a framework in which design options can be evaluated. By examining the criteria that are associated with each option in the design, and the evidence that is provided to support these criteria, informed judgments can be made in the design.

EVALUATION THROUGH USER PARTICIPATION

However, useful as these techniques are for filtering and refining the design, they are not a replacement for actual usability testing with the people for whom the system is intended: the users.

These include empirical or experimental methods, observational methods, query techniques, and methods that use physiological monitoring, such as eye tracking and measures of heart rate and skin conductance.

User participation in evaluation tends to occur in the later stages of development when there is at least a working prototype of the system in place. This may range from a simulation of the system's interactive capabilities, without its underlying

Styles of evaluation:

Before we consider some of the techniques that are available for evaluation with users, we will distinguish between two distinct evaluation styles: those performed under laboratory conditions and those conducted in the work environment or 'in the field'.

Laboratory studies:

In the first type of evaluation studies, users are taken out of their normal work environment to take part in controlled tests, often in a specialist usability laboratory (although the 'lab' may simply be a quiet room). This approach has a number of benefits and disadvantages.

Field studies:

The second type of evaluation takes the designer or evaluator out into the user's work environment in order to observe the system in action. Again this approach has its pros and cons.

High levels of ambient noise, greater levels of movement and constant interruptions, such as phone calls, all make field observation difficult.

However, the very 'open' nature of the situation means that you will observe interactions between systems and between individuals that would have been missed in a laboratory study.

In particular, controlled experiments can be useful for evaluation of specific interface features, and must normally be conducted under laboratory conditions.

9.4.2 Empirical methods: experimental evaluation

One of the most powerful methods of evaluating a design or an aspect of a design is to use a controlled experiment. This provides empirical evidence to support a particular claim or hypothesis. It can be used to study a wide range of different issues at different levels of detail.

Any experiment has the same basic form. The evaluator chooses a hypothesis to test, which can be determined by measuring some attribute of participant behavior.

A number of experimental conditions are considered which differ only in the values of certain controlled variables. Any changes in the behavioral measures are attributed to the different conditions.

Participants
The choice of participants is vital to the success of any experiment. In evaluation experiments, participants should be chosen to match the expected user population as closely as possible. Ideally, this will involve experimental testing with the actual users but this is not always possible.

Variables
Experiments manipulate and measure variables under controlled conditions, in order to test the hypothesis. There are two main types of variable: those that are 'manipulated' or changed (known as the independent variables) and those that are measured (the dependent variables).

Independent variables are those elements of the experiment that are manipulated to produce different conditions for comparison.

Hypotheses:

A hypothesis is a prediction of the outcome of an experiment. It is framed in terms of the independent and dependent variables, stating that a variation in the independent variable will cause a difference in the dependent variable. The aim of the experiment is to show that this prediction is correct. This is done by disproving the null hypothesis, which states that there is no difference in the dependent variable between the levels of the independent variable.

Experimental design:

The first phase in experimental design then is to choose the hypothesis: to decide exactly what it is you are trying to demonstrate.

The next step is to decide on the *experimental method* that you will use. There are two main methods: *between-subjects* and *within-subjects*. In a between-subjects (or *randomized*) design, each participant is assigned to a different condition.

There are at least two conditions: the experimental condition (in which the variable has been manipulated) and the control, which is identical to the experimental condition except for this manipulation. This control serves to ensure that it is the manipulation that is responsible for any differences that are measured.

The advantage of a between-subjects design is that any learning effect resulting from the user performing in one condition and then the other is controlled: each user performs under only one condition.

The disadvantages are that a greater number of participants are required, and that significant variation between the groups can negate any results.

Statistical measures

The first two rules of statistical analysis are to *look* at the data and to *save* the data. It is easy to carry out statistical tests blindly when a glance at a graph, histogram or table of results would be more instructive.

Variables can be classified as either *discrete variables* or *continuous variables*. A discrete variable can only take a finite number of values or *levels*, for example, a screen color that can be red, green or blue. A continuous variable can take any value (although it may have an upper or lower limit), for example a person's height or the time taken to complete a task.

Observational techniques:

A popular way to gather information about actual use of a system is to observe users interacting with it.

Usually they are asked to complete a set of predetermined tasks, although, if observation is being carried out in their place of work, they may be observed going about their normal duties.

Simple observation is seldom sufficient to determine how well the system meets the users' requirements since it does not always give insight into the their decision processes or attitude.

Consequently users are asked to elaborate their actions by 'thinking aloud'. In this section we consider some of the techniques used to evaluate systems by observing user behavior.

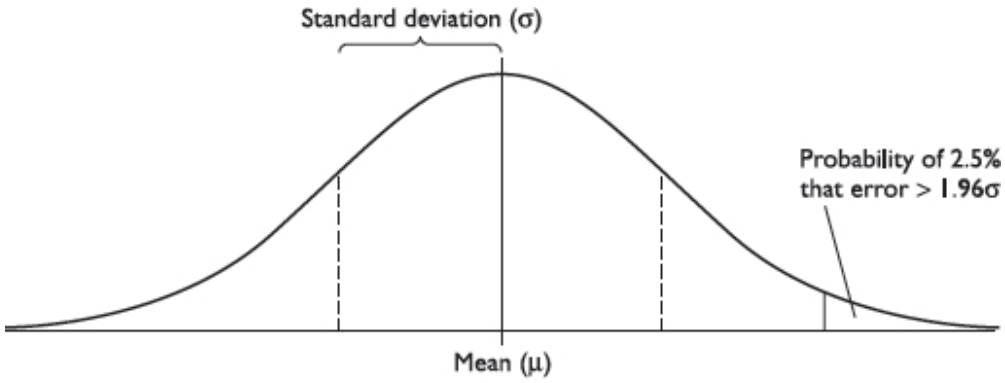


Figure 9.2 Histogram of normally distributed errors

Protocol analysis

Methods for recording user actions include the following:

Paper and pencil This is primitive, but cheap, and allows the analyst to note interpretations and extraneous events as they occur. However, it is hard to get detailed information, as it is limited by the analyst's writing speed.

Coding schemes for frequent activities, developed during preliminary studies, can improve the rate of recording substantially, but can take some time to develop.

Audio recording This is useful if the user is actively 'thinking aloud'. However, it may be difficult to record sufficient information to identify exact actions in later analysis, and it can be difficult to match an audio recording to some other form of protocol (such as a handwritten script).

Video recording This has the advantage that we can see *what* the participant is doing (*as long as* the participant stays within the range of the camera). Choosing suitable camera positions and viewing angles so that you get sufficient detail and yet keep the participant in view is difficult.

Computer logging It is relatively easy to get a system automatically to record user actions at a keystroke level, particularly if this facility has been considered early in the design. It can be more difficult with proprietary software where source code is not available (although some software now provides built-in logging and playback.

Automatic protocol analysis tools

Analyzing protocols, whether video, audio or system logs, is time consuming and tedious by hand. It is made harder if there is more than one stream of data to synchronize. One solution to this problem is to provide automatic analysis tools to support the task.

These offer a means of editing and annotating video, audio and system logs and synchronizing these for detailed analysis.

Post-task walkthroughs

Often data obtained via direct observation lack interpretation. Even where the participant has been encouraged to think aloud through the task, the information may be at the wrong level.

For example, the participant may say 'and now I'm selecting the undo menu', but not tell us what was wrong to make undo necessary. In addition, a think aloud does not include information such as alternative, but not pursued, actions.

A walkthrough attempts to alleviate these problems, by reflecting the participants' actions back to them after the event. The transcript, whether written or recorded, is replayed to the participant who is invited to comment, or is directly questioned by the analyst.

9.4.4 Query techniques

Another set of evaluation techniques relies on asking the user about the interface directly. Query techniques can be useful in eliciting detail of the user's view of a system. They embody the philosophy that states that the best way to find out how a system meets user requirements is to 'ask the user'.

They can be used in evaluation and more widely to collect information about user requirements and tasks. The advantage of such methods is that they get the user's viewpoint directly and may reveal issues that have not been considered by the designer.

Interviews

Interviewing users about their experience with an interactive system provides a direct and structured way of gathering information. Interviews have the advantages that the level of questioning can be varied to suit the context and that the evaluator can probe the user more deeply on interesting issues as they arise.

An interview will usually follow a top-down approach, starting with a general question about a task and progressing to more leading questions (often of the form 'why?' or 'what if ?') to elaborate aspects of the user's response.

Interviews can be effective for high-level evaluation, particularly in eliciting information about user preferences, impressions and attitudes.

They may also reveal problems that have not been anticipated by the designer or that have not occurred under observation. When used in conjunction with observation they are a useful means of clarifying an event (compare the post-task walkthrough).

Questionnaires

An alternative method of querying the user is to administer a questionnaire. This is clearly less flexible than the interview technique, since questions are fixed in advance, and it is likely that the questions will be less probing.

There are a number of styles of question that can be included in the questionnaire.

These include the following:

General These are questions that help to establish the background of the user and his place within the user population. They include questions about age, sex, occupation, place of residence, and so on. They may also include questions on previous experience with computers, which may be phrased as open-ended, multi-choice or scalar questions .

Open-ended These ask the user to provide his own unprompted opinion on a question, for example 'Can you suggest any improvements to the interface? They are useful for gathering general subjective information but are difficult to analyze in any rigorous way, or to compare, and can only be viewed as supplementary.

They are also most likely to be missed out by time-conscious respondents! However, they may identify errors or make suggestions that have not been considered by the designer. A special case of this type is where the user is asked for factual information, for example how many commands were used.

Scalar These ask the user to judge a specific statement on a numeric scale, usually corresponding to a measure of agreement or disagreement with the statement.

For example,
It is easy to recover from mistakes.

Disagree 1 2 3 4 5 Agree

Multi-choice Here the respondent is offered a choice of explicit responses, and may be asked to select only one of these, or as many as apply. For example,

How do you most often get help with the system (tick one)?

- Online manual ❑
- Contextual help system ❑
- Command prompt ❑
- Ask a colleague ❑

Which types of software have you used (tick all that apply)?

- Word processor ❑
- Database ❑
- Spreadsheet ❑
- Expert system ❑
- Online help system ❑
- Compiler ❑

These are particularly useful for gathering information on a user's previous experience. A special case of this type is where the offered choices are 'yes' or 'no'.

Ranked These place an ordering on items in a list and are useful to indicate a user's preferences. For example,

Please rank the usefulness of these methods of issuing a command (1 most useful, 2 next, 0 if not used).

- Menu selection ❑
- Command line ❑
- Control key accelerator ❑

These question types are all useful for different purposes, as we have noted. However, in order to reduce the burden of effort on the respondent, and so encourage a high response rate amongst users, it is best to use closed questions, such as scalar, ranked or multi-choice, as much as possible.

9.4.5 Evaluation through monitoring physiological responses

One of the problems with most evaluation techniques is that we are reliant on observation and the users telling us what they are doing and how they are feeling. What if we were able to measure these things directly? Interest has grown recently in the use of what is sometimes called objective usability testing, ways of monitoring physiological
aspects of computer use.

Potentially this will allow us not only to see more clearly exactly what users do when they interact with computers, but also to measure how they feel. The two areas receiving the most attention to date are eye tracking and physiological measurement.

Eye tracking for usability evaluation

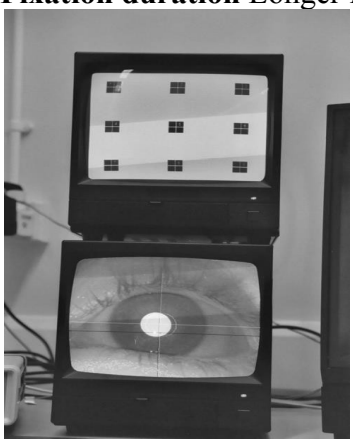
Eye tracking has been possible for many years, but recent improvements in hardware and software have made it more viable as an approach to measuring usability. The original eye trackers required highly invasive procedures where eye caps were attached to the cornea under anaesthetic.

Eye movements are believed to reflect the amount of cognitive processing a display requires and, therefore, how easy or difficult it is to process . So measuring not only where people look, but also their patterns of eye movement, may tell us which areas of a screen they are finding easy or difficult to understand.

Eye movement measurements are based on fixations, where the eye retains a stable position for a period of time, and saccades, where there is rapid ballistic eye movement from one point of interest to another. There are many possible measurements related to usability evaluation including:

Number of fixations The more fixations the less efficient the search strategy.

Fixation duration Longer fixations may indicate difficulty with a display



Scan path indicating areas of interest, search strategy and cognitive load. Moving straight to a target with a short fixation at the target is the optimal scan path but plotting scan paths and fixations can indicate what people look at, how often and for how long.

Eye tracking for usability is still very new and equipment is prohibitively expensive for everyday use. However, it is a promising technique for providing insights into what really attracts the eye in website design and where problem areas are in system use.

Physiological measurements

Emotional response is closely tied to physiological changes. These include changes in heart rate, breathing and skin secretions. Measuring these physiological responses may therefore be useful in determining a user's emotional response to an interface. Could we determine which interaction events really cause a user stress or which promote relaxation?

Physiological measurement involves attaching various probes and sensors to the user. These measure a number of factors.

Heart activity, indicated by blood pressure, volume and pulse. These may respond to stress or anger.
Activity of the sweat glands, indicated by skin resistance or galvanic skin response (GSR).

These are thought to indicate levels of arousal and mental effort.

Electrical activity in muscle, measured by the electromyogram (EMG). These appear to reflect involvement in a task.

Electrical activity in the brain, measured by the electroencephalogram (EEG). These are associated with decision making, attention and motivation.

CHOOSING AN EVALUATION METHOD

9.5.1 Factors distinguishing evaluation techniques

The eight factors that distinguish different evaluation techniques and therefore help us to make an appropriate choice. These are:

The stage in the cycle at which the evaluation is carried out
The style of evaluation
The level of subjectivity or objectivity of the technique
The type of measures provided
The information provided
The immediacy of the response
The level of interference implied
The resources required.

Design vs. implementation

The first factor to affect our choice of evaluation method is the stage in the design process at which evaluation is required. As we saw earlier in this chapter, it is desirable to include evaluation of some sort throughout the design process.
The main distinction between evaluation of a design and evaluation of an implementation is that in the latter case a physical artifact exists.

Laboratory vs. field studies

Laboratory studies allow controlled experimentation and observation while losing something of the naturalness of the user's environment. Field studies retain the latter but do not allow control

over user activity.

Ideally the design process should include both styles of evaluation, probably with laboratory studies dominating the early stages and field studies conducted with the new implementation.

Subjective vs. objective

Evaluation techniques also vary according to their objectivity – some techniques rely heavily on the interpretation of the evaluator, others would provide similar information for anyone correctly carrying out the procedure.
The more subjective techniques, such as cognitive walkthrough or think aloud, rely to a large extent on the knowledge and expertise of the evaluator, who must recognize problems and understand what the user is doing.

Qualitative vs. quantitative measures

The type of measurement provided by the evaluation technique is also an important consideration. There are two main types: *quantitative measurement* and *qualitative measurement*.
The former is usually numeric and can be easily analyzed using statistical techniques. The latter is non-numeric and is therefore more difficult to analyze, but can provide important detail that cannot be determined from numbers.

The type of measure is related to the subjectivity or objectivity of the technique, with subjective techniques tending to provide qualitative measures and objective techniques, quantitative measures.

Information provided

The level of information required from an evaluation may also vary. The information required by an evaluator at any stage of the design process may range from low-level information to enable a design decision to be made (for example, which font is most readable) to higher-level information, such as 'Is the system usable?'.

Some evaluation techniques, such as controlled experiments, are excellent at providing low-level information – an experiment can be designed to measure a particular aspect of the interface. Higher-level information can be gathered using questionnaire and interview techniques, which provide a more general impression of the user's view of the system.

Immediacy of response

Another factor distinguishing evaluation techniques is the immediacy of the response they provide. Some methods, such as think aloud, record the user's behavior at the time of the interaction itself.

Others, such as post-task walkthrough,rely on the user's recollection of events. Such recollection is liable to suffer from bias in recall and reconstruction, with users interpreting events according to their preconceptions.

Intrusiveness

Certain techniques, particularly those that produce immediate measurements, are obvious to the user during the interaction and therefore run the risk of influencing the way the user behaves.

Sensitive activity on the part of the evaluator can help to reduce this but cannot remove it altogether. Most immediate evaluation techniques are intrusive, with the exception of automatic system logging. Unfortunately, this is limited in the information that it can provide and is difficult to interpret.

Resources

The final consideration when selecting an evaluation technique is the availability of resources. Resources to consider include equipment, time, money, participants, expertise of evaluator and context.
Some decisions are forced by resource limitations:

It is not possible to produce a video protocol without access to a video camera (and probably editing facilities as well).

In these circumstances, the evaluator must decide which evaluation tactic will produce the most effective and useful information for the system under consideration. It may be possible to use results from other people's experiments to avoid having to conduct new experiments.

Finally, the context in which evaluation can occur will influence what can be done. For practical reasons it may not be possible to gain access to the intended users of a system.

9.5.2 A classification of evaluation techniques:

Using the factors discussed in the previous section we can classify the evaluation techniques we have considered in this chapter. This allows us to identify the techniques that most closely fit our requirements. Table 9.4 shows the classification for

Table 9.4 Classification of analytic evaluation techniques

	Cognitive walkthrough	Heuristic evaluation	Review based	Model based
Stage	Throughout	Throughout	Design	Design
Style	Laboratory	Laboratory	Laboratory	Laboratory
Objective?	No	No	As source	No
Measure	Qualitative	Qualitative	As source	Qualitative
Information	Low level	High level	As source	Low level
Immediacy	N/A	N/A	As source	N/A
Intrusive?	No	No	No	No
Time	Medium	Low	Low–medium	Medium
Equipment	Low	Low	Low	Low
Expertise	High	Medium	Low	High

Table 9.5 Classification of experimental and query evaluation techniques

	Experiment	Interviews	Questionnaire
Stage	Throughout	Throughout	Throughout
Style	Laboratory	Lab/field	Lab/field
Objective?	Yes	No	No
Measure	Quantitative	Qualitative/ quantitative	Qualitative/ quantitative
Information	Low/high level	High level	High level
Immediacy	Yes	No	No
Intrusive?	Yes	No	No
Time	High	Low	Low
Equipment	Medium	Low	Low
Expertise	Medium	Low	Low

Table 9.6 Classification of observational evaluation techniques

	Think aloud[1]	Protocol analysis[2]	Post-task walkthrough
Stage	Implementation	Implementation	Implementation
Style	Lab/field	Lab/field	Lab/field
Objective?	No	No	No
Measure	Qualitative	Qualitative	Qualitative
Information	High/low level	High/low level	High/low level
Immediacy	Yes	Yes	No
Intrusive?	Yes	Yes[3]	No
Time	High	High	Medium
Equipment	Low	High	Low
Expertise	Medium	High	Medium

1 Assuming a simple paper and pencil record
2 Including video, audio and system recording
3 Except system logs

analytic techniques, Table 9.5 for experimental and query techniques, Table 9.6 for observational and Table 9.7 for monitoring techniques.

The classification is intended as a rough guide only – some of the techniques do not fit easily into such a classification since their use can vary considerably.

Table 9.7 Classification of monitoring evaluation techniques

	Eye tracking	Physiological measurement
Stage	Implementation	Implementation
Style	Lab	Lab
Objective?	Yes	Yes
Measure	Quantitative	Quantitative
Information	Low level	Low level
Immediacy	Yes	Yes
Intrusive?	No[1]	Yes
Time	Medium/high	Medium/high
Equipment	High	High
Expertise	High	High

1 If the equipment is not head mounted

UNIVERSAL DESIGN:

Universal design is the process of designing products so that they can be used by as many people as possible in as many situations as possible

UNIVERSAL DESIGN PRINCIPLES
These principles give us a framework in which to develop universal designs.

Principle one is *equitable use*: The design is useful to people with a range of abilities and appealing to all. No user is excluded or stigmatized.

Wherever possible, access should be the same for all; where identical use is not possible, equivalent use should be supported. Where appropriate, security, privacy and safety provision should be available to all.

Principle two is *flexibility in use*: the design allows for a range of ability and preference, through choice of methods of use and adaptivity to the user's pace, precision and custom.

Principle three is that the system be *simple and intuitive to use*, regardless of the knowledge, experience, language or level of concentration of the user. The design needs to support the user's expectations and accommodate different language and literacy skills.

Principle four is *perceptible information*: the design should provide effective communication of information regardless of the environmental conditions or the user's abilities. Redundancy of presentation is important: information should be represented in different forms or modes (e.g. graphic, verbal, text, touch).
Essential information should be emphasized and differentiated clearly from the peripheral content. Presentation should support the range of devices and techniques used to access information by people with different sensory abilities.

Principle five is *tolerance for error*: minimizing the impact and damage caused by mistakes or unintended behavior. Potentially dangerous situations should be removed or made hard to reach. Potential hazards should be shielded by warnings.

Systems should fail safe from the user's perspective and users should be supported in tasks that require concentration. Principle six is *low physical effort*: systems should be designed to be comfortable to use, minimizing physical effort and fatigue. The physical design of the system should allow the user to maintain a natural posture with reasonable operating effort.

Principle seven requires *size and space for approach and use*: the placement of the system should be such that it can be reached and used by any user regardless of body size, posture or mobility.

Important elements should be on the line of sight for both seated and standing users. All physical components should be comfortably reachable by seated or standing users. Systems should allow for variation in hand size and provide enough room for assistive devices to be used.

These seven principles give us a good starting point in considering universal design.

MULTI-MODAL INTERACTION

Providing access to information through more than one mode of interaction is an important principle of universal design. Such design relies on *multi-modal interaction*.

There are five senses: sight, sound, touch, taste and smell.

Sight is the predominant sense for the majority of people, and most interactive systems consequently use the visual channel as their primary means of presentation, through graphics, text, video and animation.

However, sound is also an important channel, keeping us aware of our surroundings, monitoring people and events around us, reacting to sudden noises, providing clues and cues that switch our attention from one thing to another.

Touch, too, provides important information: tactile feedback forms an intrinsic part of the operation of many common tools – cars, musical instruments, pens, anything that requires holding or moving. It can form a sensuous bond between individuals, communicating a wealth of non-verbal information.

Taste and smell are often less appreciated (until they are absent) but they also provide useful information in daily life: checking if food is bad, detecting early signs of fire, noticing that manure has been spread in a field, pleasure.

The majority of interactive computer systems are predominantly visual in their interactive properties; often WIMP based, they usually make use of only rudimentary sounds while adding more and more visual information to the screen.

The use of multiple sensory channels increases the *bandwidth* of the interaction between the human and the computer, and it also makes human–computer interaction more like the interaction between humans and their everyday environment, perhaps making the use of such systems more natural.

Sound in the interface

Sound is an important contributor to usability. There is experimental evidence to suggest that the addition of audio confirmation of modes, in the form of changes in keyclicks, reduces errors .

Video games offer further evidence, since experts tend to score less well when the sound is turned off than when it is on; they pick up vital clues and information from the sound while concentrating their visual attention on different things.

The dual presentation of information through sound and vision supports universal design, by enabling access for users with visual and hearing impairments respectively.

It also enables information to be accessed in poorly lit or noisy environments. Sound can convey transient information and does not take up screen space, making it potentially useful for mobile applications.

There are two types of sound that we could use: speech and non-speech.

Speech in the interface

Language is rich and complex. We learn speech naturally as children 'by example' – by listening to and mimicking the speech of those around us. This process seems so effortless that we often do not appreciate its complex structures, and it is not until we attempt to learn a new language later in life, or to make explicit the rules of the one we speak, that the difficulties inherent in language understanding become apparent. This complexity makes speech recognition and synthesis by computer very difficult.

Structure of speech If we are fully to appreciate the problems involved with the computer-based recognition and generation of speech, we need first to understand the basic structure of speech. We will use English to illustrate but most other languages have similar issues.

Each phoneme represents a distinct sound, there being 24 consonants and 16 vowel sounds. Language is more than simple sounds, however. Emphasis, stress, pauses and pitch can all be used to alter the meaning and nature of an utterance, a common example being the rise in pitch at the end of a sentence to indicate a question in English. This alteration in tone and quality of the phonemes is termed *prosody* and is used, in addition to the actual words, to convey a great deal of meaning and emotion within a sentence. Prosodic information gives language its richness and texture, but is very difficult to quantify.

Owing to the manner in which sound is produced in the vocal tract, mouth and nose of the speaker, the limitation in response speed means that phonemes sound differently when preceded by different phonemes.

This is termed *co-articulation*, and the resulting differences in sound can be used to construct a set of *allophones*, which represent all the different sounds within the language. Ignoring prosodic information, the concatenation of allophones together should produce intelligible, articulate speech.

However, depending on the analysis of language used, and the regional accent studied, there are between 120 and 130 allophones. These, in turn, can be formed into *morphemes*, which represent the smallest unit of language that has meaning.

They are the basic building blocks of language rather than of speech. Morphemes can be either parts of words or whole words, and they are built into sentences using the rules of grammar of the language.

Even being able to decompose sentences into their basic parts does not mean that we can then understand them: the syntax (structure) only serves as a standard foundation upon which the semantics (meaning) is based.

Speech recognition There have been many attempts at developing speech recognition systems, but, although commercial systems are now commonly and cheaply available, their success is still limited to single-user systems that require considerable training.

The complexity of language is one barrier to success, but there are other, more practical, problems also associated with the automatic recognition of the spoken word.

Background noise can interfere with the input, masking or distorting the information, while speakers can introduce redundant or meaningless noises into the information stream by repeating themselves, pausing or using 'continuation' noises such as 'ummm' and 'errr' to fill in gaps in their usual speech.

.

Speech synthesis Complementary to speech recognition is speech synthesis. The notion of being able to converse naturally with a computer is an appealing one for many users, especially those who do not regard themselves as computer literate, since it reflects their natural, daily medium of expression and communication.

However, there are as many problems in speech synthesis as there are in recognition. The most difficult problem is that we are highly sensitive to variations and intonation in speech, and are therefore intolerant of imperfections in synthesized speech.

Synthesized speech also brings other problems. Being transient, spoken output cannot be reviewed or browsed easily. It is intrusive, requiring either an increase in noise in the office environment or the use of headphones, either of which may be too large a price to pay for the benefits the system may offer.

Modern screen readers read more than simply the text on the screen. They read exactly what they find including icons, menus, punctuation and controls. They also read events, such as dialog boxes opening, so that they can be used with graphical interfaces.

Speech synthesis is also useful as a communication tool to assist people with physical disabilities that affect their speech. Here speech synthesis needs to produce output that is as natural as possible with as little input effort from the user as possible, perhaps using a simple switch.

Human conversation is rapid and complex, making this a significant challenge. Most communication tools of this type use predefined messages, enabling the user to select a message appropriate to the context quickly and easily.

Uninterpreted speech Speech does not have to be recognized by a computer to be useful in the interface. Fixed pre-recorded messages can be used to supplement or replace visual information.

Recordings have natural human prosody and pronunciation, although quality is sometimes low. Segments of speech can be used together to construct messages, for example the announcements in many airports and railway stations.

Recordings of users' speech can also be very useful, especially in collaborative applications, for example many readers will have used voicemail systems. Also, recordings can be attached to other artifacts as *audio annotations* in order to communicate with others or to remind oneself at a later time.

For example, audio annotations can be attached to Microsoft Word documents.

Speech can be played back at up to twice the normal rate without any loss of comprehensibility. This can be used in a telephone help desk where a pre-recorded message asks the enquirer to state his problem.

Non-speech sound

We have considered the use of speech in the interface, but non-speech sounds can offer a number of advantages. As speech is serial, we have to listen to most of a sentence before we understand what is being said. Non-speech sounds can often be assimilated much more quickly.

Speech is language dependent – a speech-based system requires translation for it to be used for another language group.

The meaning of non-speech sounds can be learned regardless of language. Speech requires the user's attention. Non-speech sound can make use of the phenomenon of auditory adaptation: background sounds are ignored unless they change or cease.

However, a disadvantage is that non-speech sounds have to be learned, whereas the meaning of a spoken message is obvious (at least to a user who is familiar with the language used).

Non-speech sound can be used in a number of ways in interactive systems. It is often used to provide transitory information, such as indications of network or system changes, or of errors. It can also be used to provide status information on background processes, since we are able to ignore continuous sounds but still respond to changes in those sounds.

Users of early home computers with their noisy power supplies, and computer operators listening to the chatter of the printer and the spinning of disks and tape drives, both report that they are able to tell what stage a process is at by the characteristic sounds that are made.

Auditory icons *Auditory icons* use natural sounds to represent different types of objects and actions in the interface. The SonicFinder for the Macintosh was developed from these ideas, to enhance the interface through redundancy.

Natural sounds are used because people recognize the source of a sound and its behavior rather than timbre and pitch . For example, a noise will be identified as glass breaking or a hollow pipe being tapped. Such recognition is quite sophisticated:

Earcons

An alternative to using natural sounds is to devise synthetic sounds. *Earcons* use structured combinations of notes, called *motives*, to represent actions and objects (Figure 10.2). These vary according to rhythm, pitch, timbre, scale and volume.

There are two types of combination of earcon. *Compound earcons* combine different motives to build up a specific action, for example combining the motives for 'create' and 'file'. *Family earcons* represent compound earcons of similar types.

As an example, operating system errors and syntax errors would be in the 'error' family. In this way, earcons can be hierarchically structured to represent menus. Earcons are easily grouped and refined owing to their compositional and hierarchical nature, but they require learning to associate with a specific task in the interface since there is an

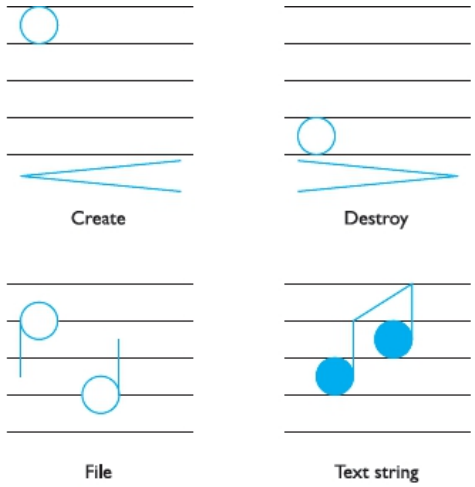


Figure 10.2 Earcons (after Blattner [36], reprinted by permission of Lawrence Erlbaum Associates, Inc.)

arbitrary mapping. Conversely, auditory icons have a semantic relationship with the function that they represent, but can suffer from there being no appropriate sound for some actions.

10.3.2 Touch in the interface

The importance of touch in our interaction with our environment, Touch is the only sense that can be used to both send and receive information. Although it is not yet widely used in interacting with computers, there is a significant research effort in this area and commercial applications are becoming available.

The use of touch in the interface is known as *haptic interaction*. Haptics is a generic term relating to touch, but it can be roughly divided into two areas: cutaneous perception, which is concerned with tactile sensations through the skin; and kinesthetics, which is the perception of movement and position. Both are useful in interaction but they require different technologies.

A number of examples of haptic devices, including some based on vibration against the skin (cutaneous) and others on resistance or force feedback (kinesthethic). They facilitate perception

of properties such as shape, texture, resistance and temperature as well as comparative spatial properties such as size, height and position.

This means haptics can provide information on the character of objects in the interface, as well as more realistic simulations of physical activities, either for entertainment or for training.

Braille displays are made up of a number of cells (typically between 20 and 80), each containing six or eight electronically controlled pins that move up and down to produce braille representations of characters displayed on the screen.

Whereas printed Braille normally has six dots per cell, electronic braille typically has eight pins, with the extratwo representing additional information about that cell, such as cursor position and character case.

Electronic braille displays benefit from two factors: a well-established tactile notation (braille) and a user group with expertise in using this notation.

Braille requires only six or eight pins; a graphical display would require many more, which raises the problem of fitting the necessary number of fast actuators (to move the pins) into a few cubic centimeters. This presents a serious engineering challenge.

The other main type of haptic device is the force feedback device, which provides kinesthetic information back to the user, allowing him to feel resistance, textures, friction and so on. One of the most commonly used examples is the PHANTOM range, from SensAble Technologies.

The PHANTOM provides three-dimensional force feedback, allowing users to touch virtual objects. As well as offering the functionality of the mouse, in addition, the user's movement is monitored by optical sensorson the device, and these, together with models of the virtual objects, are used tocalculate the forces applied back to the user.

10.3.3 Handwriting recognition

Handwriting to be a very natural form of communication. The idea of being able to interpret handwritten input is very appealing, and handwriting appears to offer both textual and graphical input using the same tools.

There are problems associated with the use of handwriting as an input medium, however, and in this section we shall consider these. We will first look at the mechanisms for capturing handwritten information, and then look at the problems of interpreting it.

Technology for handwriting recognition

Free-flowing strokes made with a pen are transformed into a series of coordinates, approximately one every 1/50th of a second (or at the sampling rate of the digitizer).

Rapid movements produce widely spaced dots, in comparison with slow movements: this introduces immediate errors into the information, since the detail of the stroke between dots is lost, as is theDigitizing tablets have been refined by incorporating a thin screen on top to display the information, producing *electronic paper*.

Advances in screen technology mean that such devices are small and portable enough to be realistically useful in handheld organizers such as the Apple Newton. Information written onto the digitizer can simply be redisplayed, or stored and redisplayed for further reference.

However, while this has limited use in itself, systems are most useful when they are able to interpret the strokes received and produce text. It is this recognition that we will look at next.

Recognizing handwriting

The variation between the handwriting of individuals is large (see Figure 10.4); moreover, the handwriting of a single person varies from day to day, and evolves over the years.

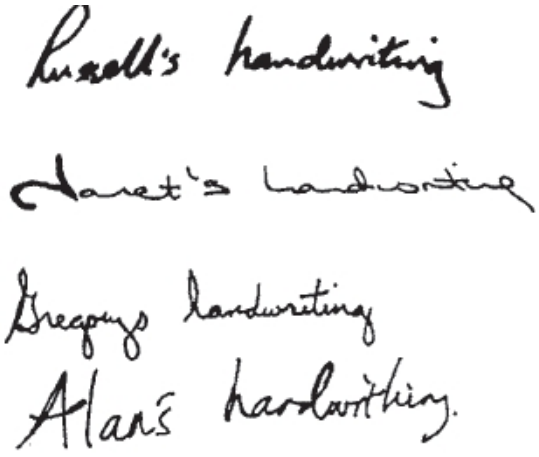


Figure 10.4 Handwriting varies considerably

Gesture recognition:

Gesture is a component of human–computer interaction that has become the subject of attention in multi-modal systems. Being able to control the computer with certain movements of the hand would be advantageous in many situations where there is no possibility of typing, or when other senses are fully occupied.

DESIGNING FOR DIVERSITY

Visual impairment

The sensory impairment that has attracted the most attention from researchers, perhaps because it is potentially also one of the most debilitating as far as interaction is concerned, is visual impairment. The rise in the use of graphical interfaces reduces the possibilities for visually impaired users. In text-based interaction, screen readers using synthesized speech or braille output devices provided complete access to computers: input relied on touch-typing, with these mechanisms providing the output.

However, today the standard interface is graphical. Screen readers and braille output are far more restricted in interpreting the graphical interface, as we saw in Section 10.3.1, meaning that access to computers, and therefore work involving computers, has been reduced rather than expanded for visually impaired people.

There are two key approaches to extending access: the use of sound and the use of touch. We have already considered these in Section 10.3 so we will summarize only

briefly here.
A number of systems use sound to provide access to graphical interfaces for people with visual impairment. In Section 10.3.1 we looked at a range of approaches to the use of sound such as speech, earcons and auditory icons. All of these have been used in interfaces for blind users.

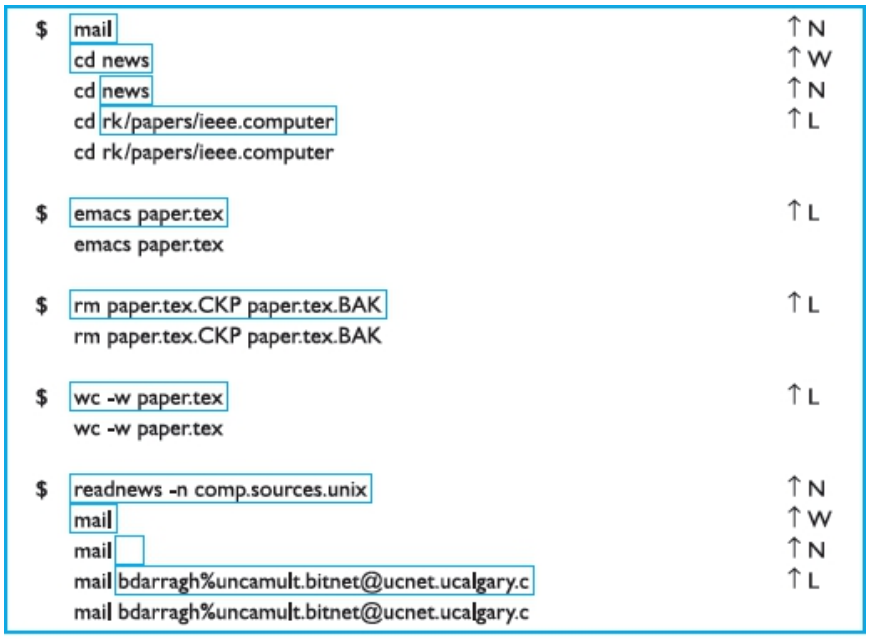
Hearing impairment

Compared with a visual disability where the impact on interacting with a graphical interface is immediately obvious, a hearing impairment may appear to have little impact on the use of an interface. After all, it is the visual not the auditory channel that is predominantly used. To an extent this is true, and computer technology can actually enhance communication opportunities for people with hearing loss. Email and instant messaging are great levellers and can be used equally by hearing and deaf users alike.
Gesture recognition has also been proposed to enable translation of signing to speech or text, again to improve communication particularly with non-signers. However, the increase in multimedia and the use of sound in interfaces has, ironically, created some access difficulties for people with hearing problems. Many multimedia presentations contain auditory narrative. If this is not supplemented by textual captions, this information is lost to deaf users. Captioning audio content, where there is not already a graphical or textual version, also has the advantage of making audio files easier and more efficient to index and search, which in turn enhances the experience of all users – a sure sign of good universal design!

Physical impairment

Users with physical disabilities vary in the amount of control and movement that they have over their hands, but many find the precision required in mouse control difficult. Speech input and output is an option for those without speech difficulties.

An alternative is the eyegaze system (Chapter 2), which tracks eye movements to control the cursor, or a keyboard driver that can be attached to the user's head. If the user is unable to control head movement, gesture and movement tracking can be used to allow the user control. If the user has limited use of a keyboard, a predictive system, such as the Reactive keyboard [157], can help, by anticipating the commands that are being typed and offering them for execution. This can cut the typing requirement considerably. Predictions are based on what the user has typed in the current session or a previous one. The predictions therefore anticipate within the context in which the user is currently working (for example, operating system commands, programming text or free text). Figure 10.7 shows an interaction using the Reactive keyboard.



User's dialog with the Reactive keyboard.
Only the last line in each group is actually executed.

Key	*Description*
↑C (control-C)	Accept the next predicted character
↑W	Accept the next predicted word
↑L	Accept the whole predicted line
↑N	Show the next alternative prediction
↑P	Show the previous alternative prediction

Reactive keyboard commands

Figure 10.7 An interaction using the Reactive keyboard. Source: Courtesy of Saul Greenberg

Speech impairment

For users with speech and hearing impairments, multimedia systems provide a number of tools for communication, including synthetic speech (see Section 10.3.1) and text-based communication and conferencing systems (see Chapter 19). Textual communication is slow, which can lower the effectiveness of the communication. Predictive algorithms have been used to anticipate the words used and fill them in, to reduce the amount of typing required. Conventions can help to provide context, which is lost from face-to-face communication, for example the 'smilie' :-), to indicate a joke. Facilities to allow turn-taking protocols to be established also help natural communication [256]. Speech synthesis also needs to be rapid to reflect natural conversational pace, so responses can be pre-programmed and selected using a single switch.

Dyslexia

Users with cognitive disabilities such as dyslexia can find textual information

difficult. In severe cases, speech input and output can alleviate the need to read and write and allow more accurate input and output. In cases where the problem is less severe, spelling correction facilities can help users. However, these need to be designed carefully: often conventional spelling correction programs are useless for dyslexic users since the programs do not recognize their idiosyncratic word construction methods. As well as simple transpositions of characters, dyslexic users may spell phonetically, and correction programs must be able to deal with these errors. Consistent navigation structure and clear signposting cues are also important to people with dyslexia. Color coding information can help in some cases and provision of graphical information to support textual can make the meaning of text easier to grasp.

Autism

Autism affects a person's ability to communicate and interact with people around them and to make sense of their environment. This manifests itself in a range of ways but is characterized by the *triad of impairments*:

1. Social interaction – problems in relating to others in a meaningful way or responding appropriately to social situations.
2. Communication – problems in understanding verbal and textual language including the use of gestures and expressions.
3. Imagination – problems with rigidity of thought processes, which may lead to repetitive behavior and inflexibility.

How might universal design of technology assist people with autism? There are two main areas of interest: communication and education.

Communication and social interaction are major areas of difficulty for people with autism. Computers, on the other hand, are often motivating, perhaps because they are relatively consistent, predictable and impersonal in their responses. The user is in control. Computer-mediated communication and virtual environments have been suggested as possible ways of enabling people with autism to communicate more easily with others, by giving the user control over the situation. Some people with autism have difficulties with language and may be helped by graphical representations of information and graphical input to produce text and speech. Again this is supported by providing redundancy in the design.

Computers may also have a role to play in education of children with autism, particularly by enabling them to experience (through virtual environments and games) social situations and learn appropriate responses. This can again provide a secure and consistent environment where the child is in control of his own learning. The use of computers to support people with autism in this way is still a new research area and it is likely that new software and tools will develop in the next few years.

10.4.2 Designing for different age groups

We have considered how people differ along a range of sensory, physical and cognitive abilities. However, there are other areas of diversity that impact upon the way we design interfaces. One of these is age. In particular, older people and children have specific needs when it comes to interactive technology.

Older people

The proportion of older people in the population is growing steadily. Contrary to popular stereotyping, there is no evidence that older people are averse to using new technologies, so this group represents a major and growing market for interactive

applications. People are living longer, have more leisure time and disposable income, and older people have increased independence. These factors have all led to an increase in older users.

But the requirements of the older population may differ significantly from other population groups, and will vary considerably within the population group. The proportion of disabilities increases with age: more than half of people over 65 have some kind of disability. Just as in younger people with disabilities, technology can provide support for failing vision, hearing, speech and mobility. New communication tools, such as email and instant messaging, can provide social interaction in cases where lack of mobility or speech difficulties reduce face-to-face possibilities. Mobile technologies can be used to provide memory aids where there is age-related memory loss.

Some older users, while not averse to using technology, may lack familiarity with it and fear learning. They may find the terminology used in manuals and training books difficult to follow and alien (words like ‘monitor’ and ‘boot’, for example, may have a completely different meaning to an older person than a young person). Interests and concerns may also be different from younger users.

1. Explain in detail about cognitive model? Overview:

- √ **Goal And Task Hierarchies**
- √ **Selection**
- √ **Cognitive Complexity Theory**
- √ **Problems And Extensions Of Goal Hierarchies**

INTRODUCTION

- ❖ The techniques and models in this chapter all claim to have some representation of users as they interact with an interface; that is, they model some aspect of the user's understanding, knowledge, intentions or processing.

GOAL AND TASK HIERARCHIES

- ❖ Many models make use of a model of mental processing in which the user achieves goals by solving subgoals in a divide-and-conquer fashion. We will consider two models, *GOMS* and *CCT*, where this is a central feature. However, we will see similar features in other models, such as *TAG* (Section 12.3.2) and when we consider task analysis techniques (Chapter 15).
- ❖ Imagine we want to produce a report on sales of introductory HCI textbooks.
- ❖ To achieve this goal we divide it into several subgoals, say gathering the data together, producing the tables and histograms, and writing the descriptive material. Concentrating on the data gathering, we decide to split this into further subgoals: find the names of all introductory HCI textbooks and then search the book sales database for these books. Similarly, each of the other subgoals is divided up into further subgoals, until some level of detail is found at which we decide to stop.
- ❖ We thus end up with a hierarchy of goals and subgoals. The example can be laid out to expose this structure:

```
produce report
gather data
. find book names
. . do keywords search of names database
<<further subgoals>>
. . sift through names and abstracts by hand
<<further subgoals>>
. search sales database
<<further  subgoals>>
layout tables and histograms
<<further subgoals>>
write description
<<further subgoals>>
```

```
GOAL: ICONIZE-WINDOW
. [select GOAL: USE-CLOSE-METHOD
. . MOVE-MOUSE-TO-WINDOW-HEADER
. . POP-UP-MENU
. . CLICK-OVER-CLOSE-OPTION
GOAL: USE-L7-METHOD
. . PRESS-L7-KEY]
```

The dots are used to indicate the hierarchical level of goals.

Selection

- From the above snippet we see the use of the word select where the choice of methods arises. GOMS does not leave this as a random choice, but attempts to predict which methods will be used.
- This typically depends both on the particular user and on the state of the system and details about the goals. For instance, a user, Sam, never uses the L7-METHOD, except for one game, 'blocks', where the mouse needs to be used in the game until the very moment the key is pressed. GOMS captures this in a selection rule for Sam:

  User Sam:
  Rule 1: Use the CLOSE-METHOD unless another rule applies.
  Rule 2: If the application is 'blocks' use the L7-METHOD.
- The goal hierarchies described in a GOMS analysis are almost wholly below the level of the unit task defined earlier. A typical GOMS analysis would therefore consist of a single high-level goal, which is then decomposed into a sequence of unit tasks, all of which can be further decomposed down to the level of basic operators:

  GOAL: EDIT-MANUSCRIPT
  . GOAL: EDIT-UNIT-TASK *repeat until no more unit tasks*
- The goal decomposition between the overall task and the unit tasks would involve detailed understanding of the user's problem-solving strategies and of the application domain. These are side-stepped entirely by the method as originally proposed

Cognitive complexity theory

- Cognitive complexity theory, introduced by Kieras and Polson [199], begins with the basic premises of goal decomposition from GOMS and enriches the model to provide more predictive power. CCT has two parallel descriptions: one of the user's goals and the other of the computer system (called the *device* in CCT).

- This is a reasonably frequent typing error and so we assume that we have developed good procedures to perform the task. We consider a fragment of the associated CCT

  production rules.
  (SELECT-INSERT-SPACE
  IF (AND (TEST-GOAL perform unit task)
  (TEST-TEXT task is insert space)
  (NOT (TEST-GOAL insert space))
  (NOT (TEST-NOTE executing insert space)))

  THEN ((ADD-GOAL insert space)
  (ADD-NOTE executing insert space)
  (LOOK-TEXT task is at %LINE %COL)))
  (INSERT-SPACE-DONE
  IF (AND (TEST-GOAL perform unit task)
  (TEST-NOTE executing insert space)

Problems and extensions of goal hierarchies

- The formation of a goal hierarchy is largely a post hoc technique and runs a very real risk of being defined by the computer dialog rather than the user. One way to rectify this is to

produce a goal structure based on pre-existing manual procedures and thus obtain a natural hierarchy [201].

- To be fair, GOMS defines its domain to be that of expert use, and thus the goal structures that are important are those which users develop out of their use of the system. However, such a natural hierarchy may be particularly useful as part of a CCT analysis, representing a very early state of knowledge.

2. Explain in detail about linguistic models?OVERVIEW:

- √ **Linguistic Models**
- √ **Bnf**
- √ **Task–Action Grammar**
- √ **The Challenge Of Display-Based Systems**

LINGUISTIC MODELS

- The user's interaction with a computer is often viewed in terms of a language, so it is not surprising that several modeling formalisms have developed centered around this concept.

BNF

- Representative of the *linguistic approach* is Reisner's use of Backus–Naur Form (*BNF*) rules to describe the dialog grammar [301]. This views the dialog at a purely syntactic level, ignoring the semantics of the language. BNF has been used widely to specify the syntax of computer programming languages, and many system dialogs can be described easily using BNF rules.
- For example, imagine a graphics system that has a line-drawing function.
- To select the function the user must select the 'line' menu option. The line-drawing function allows the user to draw a polyline, that is a sequence of line arcs between points. The user selects the points by clicking the mouse button in the drawing area. The user double clicks to indicate the last point of the polyline.

```
draw-line ::= select-line + choose-points
+ last-point
select-line ::= position-mouse + CLICK-MOUSE
choose-points ::= choose-one
| choose-one + choose-points
choose-one ::= position-mouse + CLICK-MOUSE
last-point ::= position-mouse + DOUBLE-CLICK-MOUSE
position-mouse ::= empty | MOVE-MOUSE + position-mouse
```

- The names in the description are of two types: *non-terminals*, shown in lower case, and *terminals*, shown in upper case. Terminals represent the lowest level of user behavior, such as pressing a key, clicking a mouse button or moving the mouse.
- Non-terminals are higher-level abstractions. The non-terminals are defined in terms of other non-terminals and terminals by a definition of the form

Task–action grammar

- Measures based upon BNF have been criticized as not 'cognitive' enough. They ignore the advantages of consistency both in the language's structure and in its use of command names and letters.
- *Task–action grammar* (*TAG*) [284] attempts to deal with some of these problems by

including elements such as parametrized grammar rules to emphasize consistency and encoding the user's world knowledge (for example, up is the opposite of down).To illustrate consistency, we consider the three UNIX commands: cp (for copying files), mv (for moving files) and ln (for linking files). Each of these has twopossible forms.

- They either have two arguments, a source and destination filename, or have any number of source filenames followed by a destination directory:

```
copy ::= 'cp' + filename + filename
| 'cp' + filenames + directory
move ::= 'mv' + filename + filename
| 'mv' + filenames + directory
link ::= 'ln' + filename + filename
| 'ln' + filenames + directory
```

- Measures based upon BNF could not distinguish between these consistent commands and an inconsistent alternative – say if ln took its directory argument nfirst. Task–action grammar was designed to reveal just this sort of consistency. Its description of the UNIX commands would be

```
file-op[Op] := command[Op] + filename + filename
| command[Op] + filenames + directory
command[Op=copy] := 'cp'
command[Op=move] := 'mv'
command[Op=link] := 'ln'
```

THE CHALLENGE OF DISPLAY-BASED SYSTEMS

- Both goal hierarchical and grammar-based techniques were initially developed when most interactive systems were command line, or at most, keyboard and cursor based.
- There are significant worries, therefore, about how well these approaches can generalize to deal with more modern windowed and mouse-driven interfaces. Both families of techniques largely ignore system output – what the user sees.
- The implicit assumption is that the users know exactly what they want to do and execute the appropriate command sequences blindly.
- If this approach is taken, the detailed mouse movements and parsing of mouse events in the context of display information (menus, etc.) are abstracted away. Typically, even when this exploratory style is used at one level, we can see within it and around it more goal-oriented methods. So, for example, we might consider the high-level goal structure

```
WRITE_LETTER
. FIND_SIMILAR_LETTER
. COPY_IT
. EDIT_COPY
```

PHYSICAL AND DEVICE MODELS

3. Describe in detail about physical and device model?OVERVIEW:

√ **Keystroke-level model**

√ **Three-state model**

Keystroke-level model

- Compared with the deep cognitive understanding required to describe problemsolving activities, the human motor system is well understood. *KLM* (*Keystroke-Level Model* [55]) uses this understanding as a basis for detailed predictions about user performance.

- It is aimed at unit tasks within interaction – the execution of simple command sequences, typically taking no more than 20 seconds. Examples of this would be using a search and replace feature, or changing the font of a word. It does not extend to complex actions such as producing a diagram.
- The assumption is that these more complex tasks would be split into subtasks (as in GOMS) before the user attempts to map them into physical actions. The task is split into two phases:

  acquisition of the task, when the user builds a mental representation of the task;
  execution of the task using the system's facilities.
- KLM only gives predictions for the latter stage of activity. During the acquisition phase, the user will have decided how to accomplish the task using the primitives of the system, and thus, during the execution phase, there is no high-level mental activity
- The user is effectively expert. KLM is related to the GOMS model, and can be thought of as a very low-level GOMS model where the method is given.
- The model decomposes the execution phase into five different physical motor operators, a mental operator and a system response operator:

  K Keystroking, actually striking keys, including shifts and other modifier keys.
  B Pressing a mouse button.
  P Pointing, moving the mouse (or similar device) at a target.

  H Homing, switching the hand between mouse and keyboard.
  D Drawing lines using the mouse.
  M Mentally preparing for a physical action.
  R System response which may be ignored if the user does not have to wait for it, as in copy typing.
  The execution of a task will involve interleaved occurrences of the various operators.

For instance, imagine we are using a mouse-based editor. If we notice a single character error we will point at the error, delete the character and retype it, and then return to our previous typing point. This is decomposed as follows:

1. Move hand to mouse **H**[mouse]
2. Position mouse after bad character **PB**[LEFT]
3. Return to keyboard **H**[keyboard]
4. Delete character **MK**[DELETE]
5. Type correction **K**[char]
6. Reposition insertion point **H**[mouse]**MPB**[LEFT]

Notice that some operators have descriptions added to them, representing which device the hand homes to (for example, [mouse]) and what keys are hit (for example, LEFT – the left mouse button).

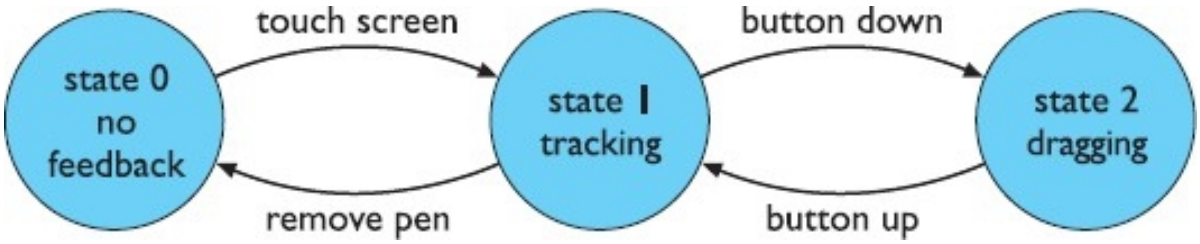
The model predicts the total time taken during the execution phase by adding the component times for each of the above activities. For example, if the time taken for one keystroke is tK, then the total time doing keystrokes is

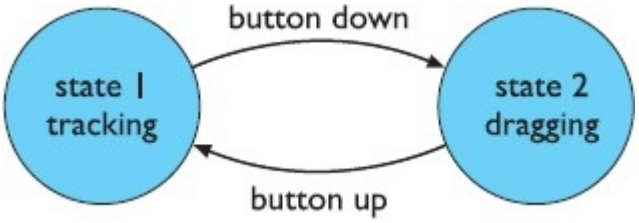
Operator	Remarks	Time (s)
K	Press key	
	good typist (90 wpm)	0.12
	poor typist (40 wpm)	0.28
	non-typist	1.20
B	Mouse button press	
	down or up	0.10
	click	0.20
P	Point with mouse	
	Fitts' law	0.1 $\log_2(D/S + 0.5)$
	average movement	1.10
H	Home hands to and from keyboard	0.40
D	Drawing – domain dependent	–
M	Mentally prepare	1.35
R	Response from system – measure	–

wpm = words per minute

Three-state model

- In Chapter 2, we saw that a range of pointing devices exists in addition to the mouse. Often these devices are considered logically equivalent, if the same inputs are available *to the application*.
- That is, so long as you can select a point on the screen, they are all the same. However, these different devices – mouse, trackball, light pen – feel very different. Although the devices are similar from the application's viewpoint, they have very different sensory–motor characteristics.
- A touchscreen is like the light pen with no button. While the user is not touching the screen, the system cannot track the finger – that is, state 0 again.
- When the user touches the screen, the system can begin to track – state 1. So a touchscreen is a state 0–1 device whereas a mouse is a state 1–2 device. As there is no difference between a state 0–2 and a state 0–1 device, there are only the three possibilities we have seen.





4. Explain in detail about cognitive architectures?

OVERVIEW:

- √ **Cognitive Architectures**
- √ **The Problem Space Model**
- √ **Interacting Cognitive Subsystems**

COGNITIVE ARCHITECTURES

- ❖ The formalisms we have seen so far have some implicit or explicit model of how the user performs the cognitive processing involved in carrying out a task. For instance, the concept of taking a problem and solving it by divide and conquer using subgoals is central to GOMS.
- ❖ CCT assumes the distinction between long- and short-term memory, with production rules being stored in long-term memory and 'matched' against the contents of short-term (or working) memory to determine which 'fire'.

The problem space model

- ❖ Rational behavior is characterized as behavior that is intended to achieve a specific goal. This element of rationality is often used to distinguish between intelligent and machine-like behavior.
- ❖ In the field of artificial intelligence (AI), a system exhibiting rational behavior is referred to as a *knowledge-level* system. A knowledge-level system contains an *agent* behaving in an environment.
- ❖ The agent has knowledge about itself and its environment, including its own goals. It can perform certain actions and sense information about its changing environment. As the agent behaves in its environment, it changes the environment and its own knowledge.
- ❖ We can view the overall behavior of the knowledge-level system as a sequence of environment and agent states as they progress in time. The goal of the agent is characterized as a preference over all possible sequences of agent environment states.

Interacting cognitive subsystems

- ❖ Barnard has proposed a very different cognitive architecture, called *interacting cognitive subsystems* (*ICS*) [24, 25, 27]. ICS provides a model of perception, cognition and action, but unlike other cognitive architectures, it is not intended to produce a description of the user in terms of sequences of actions that he performs.
- ❖ ICS provides a more holistic view of the user as an information-processing machine. The emphasis is on determining how easy particular procedures of action sequences become as they are made more automatic within the user.
- ❖ ICS attempts to incorporate two separate psychological traditions within one cognitive architecture. On the one hand is the architectural and general-purpose information-processing approach of short-term memory research.

- On the other hand is the computational and representational approach characteristic of psycholinguistic research and AI problem-solving literature.

5. Write in detail about socio-organizational issues and stakeholder requirements?OVERVIEW:

- √ **Socio-Organizational Issues And Stakeholder Requirements**
- √ **Organizational Issues**
- √ **Cooperation Or Conflict**
- √ **Changing Power Structures**
- √ **Free Rider Problem**
- √ **Critical Mass**

socio-organizational issues and stakeholder requirements

- It is used within a specific context, and is influenced by many factors within that context. The different people affected by the introduction of a system are known as stakeholders and their needs can be both complex and conflicting.
- In addition, we need to understand how the introduction of the system might actually change the organizational and work practices that are currently in place and what the impact of this might be.
- In this chapter we look in more detail at the socio-organizational context of use and discuss some of the generic issues that can affect the acceptance of technology nin organizations.
- We then look at a number of approaches to modeling the socioorganizational context and the requirements of the stakeholders within it.

ORGANIZATIONAL ISSUES

- In this section, we shall look at some of the organizational issues that affect the acceptance and relevance of information and communication systems. These factors often sit 'outside' the system as such, and may involve individuals who never use it.
- Yet it is often these factors more than any other that determine the success or failure of computer systems.
- Many systems supporting work in organizations are supporting groups of workers, but this may be through specialist groupware systems (see Chapter 19) or through shared data or processes.

Cooperation or conflict?

- The term 'computer-supported *cooperative* work' (CSCW) seems to assume that groups will be acting in a cooperative manner.
- This is obviously true to some extent; even opposing football teams cooperate to the extent that they keep (largely) within the rules of the game, but their cooperation only goes so far.
- People in organizations and groups have conflicting goals, and systems that ignore this are likely to fail spectacularly.

Changing power structures

- The identification of stakeholders will uncover information transfer and power relationships that cut across the organizational structure.

- Indeed, all organizations have these informal networks that support both social and functional contacts.
- However, the official lines of authority and information tend to flow up and down through line management. New communications media may challenge and disrupt these formal managerial structures.
- The physical layout of an organization often reflects the formal hierarchy: each department is on a different floor, with sections working in the same area of an office. If someone from sales wants to talk to someone from marketing then one of them must walk to the other's office. Many organizations are moving toward flatter management structures anyway, so from a strategic viewpoint these effects may be acceptable.
- But can the organization cope with a disaffected junior management during the transition?

For other organizations

- The effects may be less welcome and the system dropped or heavily regulated.
- In one case, an email system was introduced and was agreed to be functioning well, but the management, feeling a loss of control and suspicion over their subordinates' communications, introduced logging so that all email messages could be monitored.
  The system quickly fell into disuse
  The invisible worker
  The ability to work and collaborate at a distance can allow functional groups to be distributed over different sites. This can take the form of cross-functional neighbourhood centers, where workers from different departments do their jobs in electronic contact with their functional colleagues.
- Alternatively, distributed groupware can allow the true home-based teleworker to operate on similar terms to an office-based equivalent. The ecological and economic advantages of such working practices are now becoming well established, and it seems that communications and CSCW technology can overcome many of the traditional barriers.

Who benefits?

- One frequent reason for the failure of information systems is that the people who get the benefits from the system are not the same as those who do the work.
- One example, which we discuss in more detail in Chapter 19, is structured message systems such as Lens. In these systems the sender has to do work in putting information into fields appropriately, but it is the recipient who benefits.
- Another example is shared calendars. The beneficiary of the system is a manager who uses the system to arrange meeting times, but whose personal secretary does the work of keeping the calendar up to date.
- Subordinates are less likely to have secretarial support, yet must keep up the calendar with little perceived benefit.

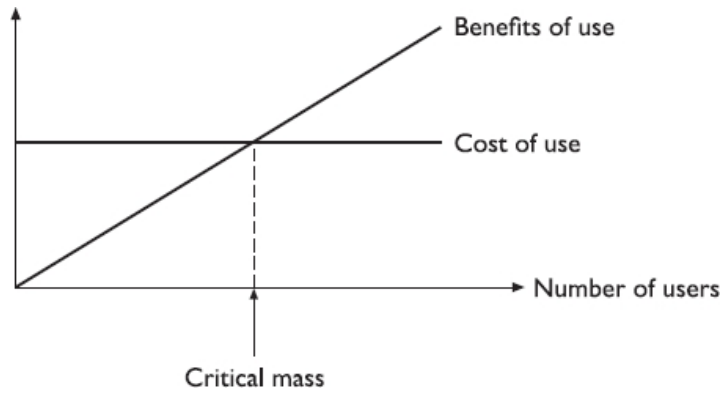
Free rider problem

- Even where there is no bias toward any particular people, a system may still not function symmetrically, which may be a problem, particularly with shared communication systems. One issue is the *free rider problem*. Take an electronic conferencing system.
- If there is plenty of discussion of relevant topics then there are obvious advantages to subscribing and reading the contributions.
- However, when considering writing a contribution, the effort of doing so may outweigh any benefits. The total benefit of the system for each user outweighs the costs, but for any particular decision the balance is overturned.

- To see this situation in a different context imagine an eccentric philanthropist who has gathered three strangers into a room.
- They are invited to throw money into a pot in the center. When they have done so, the philanthropist will double the money in the pot and then divide it up between them and send them on their way.

Critical mass

- Another issue related to the free rider problem is the need to develop a *critical mass*. When telephones were only in public places, their use as a form of pervasive interpersonal communication was limited.
- However, once a large number of people have telephones in their homes it becomes worthwhile paying to have a telephone installed.
- In cost/benefit terms, the early subscribers probably have a smaller benefit than the cost.
- Only when the number of subscribers increases beyond the critical mass does the benefit *for all* dominate the cost (see Figure 13.1). The situation for conferencing systems and email is, of course, very similar.



Automating processes – workflow and BPR

- The major task in many organizations is moving pieces of paper around. An order is received by phone and an order form filled in by the sales executive.
- The order form is passed to accounts who check the credit rating and if all is okay it is passed on to stores who check availability and collect the order together at the picking line.
- When the order is dispatched, a delivery note is packed with the order and a copy is returned to accounts, who send an invoice to the customer. Organizations have many such processes, and *workflow* systems aim to automate much of the process using electronic forms, which are forwarded to the relevant person based on pre-coded rules.
- Some workflow systems are built using specialpurpose groupware, often based on a notation for describing the desired workflow. Coordinator (see Section 14.3.6) is an example of a workflow system where the rules are heavily influenced by speech act theory.

Evaluating the benefits

- We have seen several problems that can arise from the mismatch between informationsystems and organizational and social factors.
- Let us assume that we have a system in place – and it has not fallen apart at the seams. Everyone seems happy with it and there are no secret resentments.
- Now it is time to count the cost – it was an expensive system to buy and install, but was it worth it? This is an almost impossible question to answer.
- The benefits from cooperative systems, especially organization-wide systems such as email or electronic conferencing, are in terms of job satisfaction or more fluid information flow. Some, such as the *video wall* (see Chapter 19), are expected primarily to help social contact within the organization.

7. Explain in detail about capturing requirements?OVERVIEW:

- √ **Capturing Requirements**
- √ **Socio-Technical Models**
- √ ***Custom Methodology***
- √ ***Open System Task Analysis (Osta)***
- √ **Soft Systems Methodology**

CAPTURING REQUIREMENTS

- As we have already seen, problems can arise when a system is introduced without a full understanding of all the people who will be affected by it.
- But how can we better understand and support complex organizational structures, workgroups and potentially conflicting stakeholder needs? We begin by capturing and analyzing requirements, but we need to do this within the work context, taking account of the complex mix of concerns felt by different stakeholders and the structures and processes operating in the workgroups

Who are the stakeholders?

- Understanding stakeholders is key to many of the approaches to requirements capture, since in an organizational setting it is not simply the end-user who is affected by the introduction of new technology. Imagine that a new billing system is to be introduced by a local gas supplier.
- Who will be affected by this decision? Obviously, the people who are responsible for producing and sending out bills – they will be the ones using the system directly. But where do they get the information from to produce the bills? To whom do they send the bills? Who determines the level of charging and on what grounds? Who stands to make a profit from increased revenue? Who will suffer if customers choose to switch supplier due to the improved service? Meter readers, customers, managers, regulators, shareholders and competitors are all stakeholders in the system.

  Primary stakeholders are people who actually use the system – the end-users.

  Secondary stakeholders are people who do not directly use the system, but receive output from it or provide input to it (for example, someone who receives a report produced by the system).

Tertiary stakeholders are people who do not fall into either of the first two categories but who are directly affected by the success or failure of the system (for example, a director whose profits increase or decrease depending on the success of the system).

Facilitating stakeholders are people who are involved with the design, development and maintenance of the system.

Socio-technical models

- Early in the twentieth century, studies of work focussed on how humans needed to adapt to technical innovations. *Technological determinism*, the view that social change is primarily dictated by technology, with human and social factors being secondary concerns, was prevalent

Methods vary but most attempt to capture certain common elements:

The problem being addressed: there is a need to understand why the technology is being proposed and what problem it is intended to solve.

The stakeholders affected, including primary, secondary, tertiary and facilitating, together with their objectives, goals and tasks.

The workgroups within the organization, both formal and informal.

The changes or transformations that will be supported.

The proposed technology and how it will work within the organization.

External constraints and influences and performance measures.

CUSTOM methodology

- CUSTOM is a socio-technical methodology designed to be practical to use in small organizations [200]. It is based on the User Skills and Task Match (USTM) approach, developed to allow design teams to understand and fully document user requirements [219]. CUSTOM focusses on establishing stakeholder requirements:

1. Describe the organizational context, including its primary goals, physical characteristics, political and economic background.
2. Identify and describe stakeholders. All stakeholders are named, categorized (as primary, secondary, tertiary or facilitating) and described with regard to personal issues, their role in the organization and their job. For example, CUSTOM addresses issues such as stakeholder motivation, disincentives, knowledge, skills, power and influence within the organization, daily tasks and so on.
3. Identify and describe work-groups. A work-group is any group of people who work together on a task, whether formally constituted or not. Again, work-groups are described in terms of their role within the organization and their characteristics.
4. Identify and describe task–object pairs. These are the tasks that must be performed, coupled with the objects that are used to perform them or to which they are applied.
5. Identify stakeholder needs. Stages 2–4 are described in terms of both the current system and the proposed system. Stakeholder needs are identified by considering the differences between the two. For example, if a stakeholder is identified as currently lacking a particular skill that is required in the proposed system then a need for training is identified.
6. Consolidate and check stakeholder requirements. Here the stakeholder needs list is checked against the criteria determined at earlier stages.

Open System Task Analysis (OSTA)

OSTA [116] is an alternative socio-technical approach, which attempts to describe what happens when a technical system is introduced into an organizational work environment. Like CUSTOM, OSTA specifies both social and technical aspects of the system. However, whereas in CUSTOM these aspects are framed in terms of stakeholder perspectives, in OSTA they are captured through a focus on tasks.

OSTA has eight main stages:

1. The primary task which the technology must support is identified in terms of users' goals.
2. Task inputs to the system are identified. These may have different sources and forms that may constrain the design.
3. The external environment into which the system will be introduced is described, including physical, economic and political aspects.
4. The transformation processes within the system are described in terms of actions performed on or with objects.
5. The social system is analyzed, considering existing work-groups and relationships within and external to the organization.
6. The technical system is described in terms of its configuration and integration with other systems.
7. Performance satisfaction criteria are established, indicating the social and technical requirements of the system.
8. The new technical system is specified.

OSTA uses notations familiar to designers, such as data flow diagrams and textual descriptions.

Soft systems methodology

The socio-technical models we have looked at focus on identifying requirements from both human and technical perspectives, but they assume a technological solution is being proposed. Soft systems methodology (SSM) arises from the same tradition but takes a view of the organization as a system of which technology and people are components. Root definitions are described in terms of specific elements, summarized using the acronym, CATWOE:

Clients – those who receive output or benefit from the system.

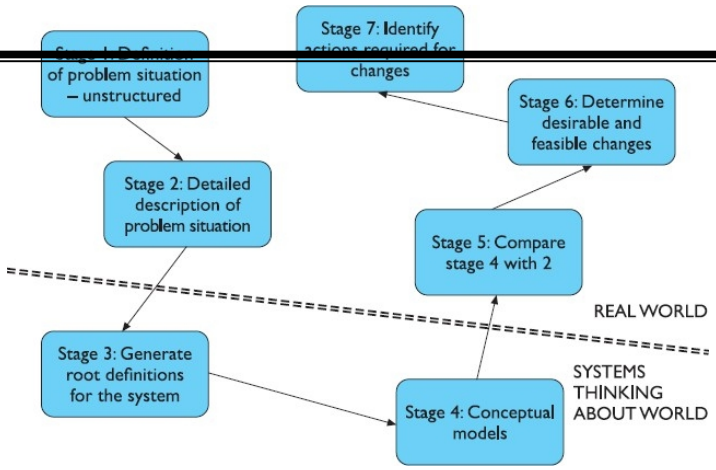
Actors – those who perform activities within the system.

Transformations – the changes that are effected by the system. This is a critical part of the root definition as it leads to the activities that need to be included in the next stage. These 'transform' the inputs of the system into the required outputs.

Weltanschauung – (from the German) meaning world view. This is how the system is perceived in a particular root definition.

Owner – those to whom the system belongs, to whom it is answerable and who can authorize changes to it.

Environment – the world in which the system operates and by which it is influenced.



Participatory design

- *Participatory design* is a philosophy that encompasses the whole design cycle. It is design in the workplace, where the user is involved not only as an experimental subject or as someone to be consulted when necessary but as a member of the design team.
- Users are therefore active collaborators in the design process, rather than passive participants whose involvement is entirely governed by the designer. The argument is that users are experts in the work context and a design canonly be effective within that context if these experts are allowed to contribute actively to the design process.
- The participatory design process utilizes a range of methods to help convey information between the user and designer. They include

Brainstorming This involves all participants in the design pooling ideas. This is informal and relatively unstructured although the process tends to involve 'onthe- fly' structuring of the ideas as they materialize.

Storyboarding This has been discussed in more detail in Chapter 6. Storyboards can be used as a means of describing the user's day-to-day activities as well as them potential designs and the impact they will have.

Workshops These can be used to fill in the missing knowledge of both user and designer and provide a more focussed view of the design. They may involve mutual enquiry in which both parties attempt to understand the context of the design from each other's point of view.

Pencil and paper exercises These allow designs to be talked through and evaluated with very little commitment in terms of resources. Users can 'walk through' typical tasks using paper mock-ups of the system design. This is intended to show up discrepancies between the user's requirements and the actual design as proposed. Such exercises provide a simple and cheap technique for early assessment o models. PICTIVE [242] is one such approach to paper prototyping, which includes representative stakeholders in a video recorded design session.However, participation is not always complete. Mumford recognizes three levels of participation:

Consultative – the weakest form of participation where participants are asked for their opinions but are not decision makers.

Representative – a representative of the participant group is involved in the decisionmaking process.

Consensus – all stakeholders are included in the decision-making process. The usual practice is that design groups are set up to include representatives from each stakeholder group and these groups make the design decisions, overseen by a steering committee of management and employee representatives.

The design groups then address the following issues and activities:

1. *Make the case for change*. Change for its own sake is inappropriate. If a case cannot be made for changing the current situation then the process ends and the system remains as it is.
2. *Identify system boundaries*. This focusses on the context of the current system and its interactions with other systems, in terms of business, existing technology, and internal and external organizational elements. How will the change impact upon each of these?
3. *Describe the existing system*, including a full analysis of inputs and outputs and the various other activities supported, such as operations, control and coordination.
4. *Define key objectives*, identifying the purpose and function of each area of the organization.
5. *Define key tasks*: what tasks need to be performed to meet these objectives?
6. *Define key information needs*, including those identified by analysis of the existing system and those highlighted by definition of key tasks.
7. *Diagnose efficiency needs*, those elements in the system that cause it to underperform or perform incorrectly. If these are internal they can be redesigned out of the new system; if they are external then the new system must be designed to cope with them.
8. *Diagnose job satisfaction needs*, with a view to increasing job satisfaction where it is low.
9. *Analyze likely future changes*, whether in technology, external constraints (such as legal requirements), economic climate or stakeholder attitudes. This is necessary to ensure that the system is flexible enough to cope with change.
10. *Specify and prioritize objectives based on efficiency*, job satisfaction and future needs. All stakeholders should be able to contribute here as it is a critical stage and conflicting priorities need to be negotiated. Objectives are grouped as either primary (must be met) or secondary (desirable to meet).

Ethnographic methods

- All of the approaches considered so far have included some level of consultation and observation of the stakeholders. However, the focus of this has been on gathering stakeholder perspectives rather than specifically studying actual work practice.
- It can be argued that work can only be understood and studied in context. This is consonant with the ideas of distributed cognition. Taking a worker away from the workplace changes the very nature of the worker's actions. Real action is *situated action*;

Contextual inquiry

- Contextual inquiry has much in common with the ethnographic tradition: it studies the user in context, trying to capture the reality of his work culture and practice.
- However, it is also an approach rooted in practice and it differs in a number of significant ways from pure ethnographic study: the intention is to understand and to interpret the

data gathered, and rather than attempting to take an open-ended view, the investigator acknowledges and challenges her particular focus.

- A number of models of the work are developed to capture what is important in the user's work situation:

  The sequence model elaborates the steps required to complete a specific task, as well as the triggers that initiate that sequence of steps.

  The physical model maps the physical work environment and how it impacts upon work practice, for example, an office plan showing where different work activities happen.

  The flow model shows the lines of coordination and communication between the user and other participants within and outside the workplace.

  The cultural model reflects the influences of work culture and policy and shows the scope of these influences. This may include official or unofficial codes of behavior, common expectations (which may or may not be explicit) and value systems.

  The artifact model describes the structure and use of a particular artifact within the work process

8. Explain in detail about communication and collaboration models?OVERVIEW:

- √ **Face-To-Face Communication**
- √ **Transfer Effects And Personal Space**
- √ **Eye Contact And Gaze**
- √ **Back Channels, Confirmation And Interruption**
- √ **Conversation**
- √ **Breakdown And Repair**

FACE-TO-FACE COMMUNICATION

- Face-to-face contact is the most primitive form of communication – primitive, that is, in terms of technology.
- If, on the other hand, we consider the style of communication, the interplay between different channels and productivity, we instead find that face-to-face is the most sophisticated communication mechanism available.

Transfer effects and personal space

- When we come to use computer-mediated forms of communication, we carry forward all our expectations and social norms from face-to-face communication.
- People are very adaptable and can learn new norms to go with new media (for example, the use of 'over' for turn-taking when using a walkie-talkie). However, success with new media is often dependent on whether the participants can use their existing norms.
- Furthermore, the rules of face-to-face conversation are not conscious, so, when they are broken, we do not always recognize the true problem. We may just have a feeling of unease, or we may feel that our colleague has been rude.
- An example of these problems concerns personal space. When we converse with one another we tend to stand with our heads a fairly constant distance apart.
- If people start to converse at opposite ends of a room, they will quickly move toward one another until they are a few feet apart. The exact distance depends somewhat on context; a high level of noise may make people come closer just to be heard.

Eye contact and gaze

- Long-term gazing into one another's eyes is usually reserved for lovers. However, normal conversation uses eye contact extensively, if not as intently. Our eyes tell us whether our colleague is listening or not; they can convey interest, confusion or boredom.
- Sporadic direct eye contact (both looking at one another's eyes) is important in establishing a sense of engagement and social presence. People who look away when you look at them may seem shifty and appear to be hiding something.
- Furthermore, relative frequency of eye contact and who 'gives way' from direct eye contact is closely linked to authority and power. Naturally, all these clues are lost if we have no visual contact.
- However, the misleading clues via a video connection can be worse. In Chapter 19 (Section 19.3.4) we discuss the problems of obtaining effective eye contact with standard video equipment. If the camera is strapped to the top of the monitor (a common arrangement) both participants will look as if their eyes are slightly dropped.

Gestures and body language

- In a similar but more direct way, we use our hands to indicate items of interest. This may be conscious and deliberate as we point to the item, or may be a slight wave of the hand or alignment of the body.
- Again, a video connection may not be sufficient to allow our colleagues to read our movements.
- This can be a serious problem since our conversation is full of expressions such as 'let's move this one there', where the 'this' and 'there' are indicated by gestures (or eyegaze).
- This is called *deictic reference* (see Chapter 19, Sections 19.4 and 19.6 for more details). Several groupware systems attempt to compensate for these losses.
- In Section 19.4.2, we discuss the idea of a group pointer, a mouse-controlled icon which can be used to point to things on a shared screen. Somewhat more esoteric, but more immediate, are the shared work surfaces (Section 19.4.3) which mix an image of the participants' hands with the electronic screen. The participants can then simply point at the relevant item on the screen, as they would face-to-face.

Back channels, confirmation and interruption

- It is easy to think of conversation as a sequence of utterances: A says something, then B says something, then back to A.
- This process is called *turn-taking* and is one of the fundamental structures of conversation. However, each utterance is itself the result of intricate negotiation and interaction. Consider the following transcript:

  Alison: Do you fancy that film . . . *er* . . . 'The Green' *um* . . . it starts at eight.
  Brian: Great!

- Alison has asked Brian whether he wants to go to the cinema (or possibly to watch the television at home).
- She is a bit vague about the film, but Brian obviously does not mind! However, if we had listened to the conversation more closely and watched Alison and Brian we would have seen more exchanges.
- As Alison says 'that film*er* . . .', she looks at Brian. From the quizzical look on his face he obviously does not know which film she is talking about. She begins to expand 'The Green *um* . . .', and light dawns; she can see it in his eyes and he probably makes a small affirmative sound 'uh huh'.
- Turn-taking As well as giving confirmation to the speaker that you understand, and indications when you do not, back channels can be used to interrupt politely.

- Starting to speak in the middle of someone's utterance can be rude, but one can say something like 'well uh' accompanied by a slight raising of the hands and a general tensing of the body and screwing of the eyes. This tells the speaker that you would like to interrupt, allowing a graceful transition.
- In this case, the listener *requested* the floor. *Turn-taking* is the process by which the roles of speaker and listener are exchanged. nBack channels are often a crucial part of this process.

CONVERSATION

- We have looked at the low-level issues of speech and gesture during face-to-face conversation. We now turn to the structure of the conversation itself.
- Most analysis of conversation focusses on two-person conversations, but this can range from informal social chat over the telephone to formal courtroom cross-examination.
- Secondly, they can be used as a guide for design decisions – an understanding of normal human–human conversation can help avoid blunders in the design of electronic media. Thirdly, and most controversially, they can be used to drive design – structuring the system around the theory.

Basic conversational structure

- Imagine we have a transcript of a conversation, recalling from Chapter 9 that the production of such a transcript is not a simple task. For example, a slightly different version of Alison and Brian's conversation may look like this:
  Alison: Do you fancy that film?
  Brian: The *uh* (*500 ms*) with the black cat – 'The Green whatsit'?
  Alison: Yeah, go at *uh* . . . (*looks at watch – 1.2 s*) . . . 20 to?
  Brian: Sure.
- This transcript is quite heavily annotated with the lengths of pauses and even Alison's action of looking at her watch. However, it certainly lacks the wealth of gesture and back channel activity that were present during the actual conversation.
- Transcripts may be less well documented, perhaps dropping the pause timings, or more detailed, adding more actions, where people were looking and some back channelling.
- Whilst thinking about the structure of conversation, the transcript above is sufficient Context Take a single utterance from a conversation, and it will usually be highly ambiguous if not meaningless: 'the *uh* with the black cat – "The Green whatsit"'.
- Each utterance and each fragment of conversation is heavily dependent on *context*, which must be used to *disambiguate* the utterance. We can identify two types of context within conversation:

  internal context – dependence on earlier utterances. For example, when Brian says 'masses' in the last transcript, this is meaningful in the light of Alison's question 'and lots of chocolate?'. This in turn is interpreted in the context of Brian's original offer of gateau.

  external context – dependence on the environment. For example, if Brian had said simply 'do you want one?', this could have meant a slice of gateau, or, if he had been holding a bottle, a glass of wine, or, if accompanied by a clenched fist, a punch on the nose. Topics, focus and forms of utterance Given that conversation is so dependent on context, it is important that the participants

have a shared focus. We have addressed this in terms of the external focus – the objects that are visible to the participants – but it is also true of the internal focus of the conversation.

Alison: Oh, look at your roses . . .
Brian: Mmm, but I've had trouble with greenfly.
Alison: They're the symbol of the English summer.
Brian: Greenfly?
Alison: No roses silly!

- Alison began the conversation with the *topic* of roses. Brian shifts to the related, but distinct, topic of greenfly.
- However, for some reason Alison has missed this shift inn focus, so when she makes her second utterance, her focus and Brian's differ, leading to the *breakdown* in communication.
- The last two utterances are a recovery which re-establishes a shared *dialog focus*. Another way of classifying utterances is by their relation to the task in hand.
- At one extreme the utterance may have no direct relevance at all, either a digression or purely social. Looking at the task-related conversation, the utterances can be classified into three kinds [335]:

  substantive directly relevant to the development of the topic;
  annotative points of clarification, elaborations, etc.;
  procedural talking about the process of collaboration itself.
- In addition, the procedural utterances may be related to the structure of collaboration itself, or may be about the technology supporting the collaboration. The latter is usually in response to a breakdown where the technology has intruded into the communication.

Breakdown and repair

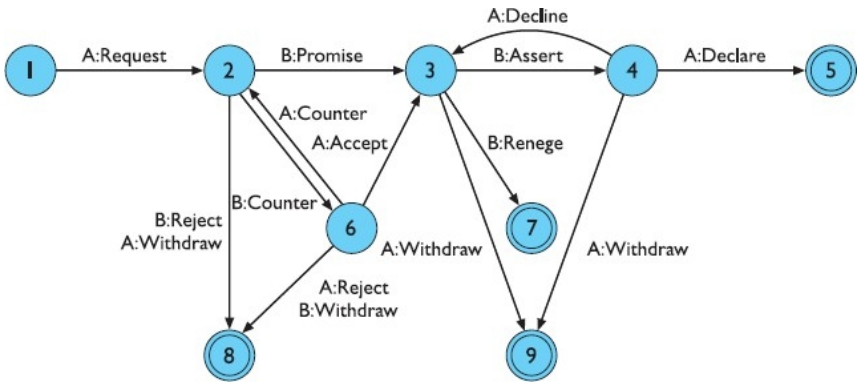
- We have already seen an example of *breakdown* in conversation. When Alison and Brian were talking about Brian's roses, they failed to maintain a shared focus.
- Brian tried to interpret Alison's utterance in terms of his focus and failed, or rather the meaning in that focus was unusual – greenfly are the symbol of the English summer?

Constructing a shared understanding

- We have seen that human conversation is in itself inherently ambiguous, relying on context and shared understanding between the parties to disambiguate the utterances.
- In some spheres, such as legal contracts, the precise meaning out of context becomes very important and thus highly stylized language is used to reduce ambiguity. 3 However, even the legal profession depends on a large body of shared knowledge and understanding about legal terms, case law, etc. Similarly, a book, such as this, attempts to use less ambiguous language and only commonly available knowledge.
- characterized by what they *do*. If you say 'I'm hungry', this has a certain *propositional meaning* – that you are feeling hungry. However, depending on who is talking and to whom, this may also carry the meaning 'get me some food' – the intent of the statement is to evoke an action on the part of the hearer.

Speech act theory

- Concerns itself with the way utterances interact with the actions of the participants.



- The numbered circles in Figure 14.1 are 'states' of the conversation, and the labeled arcs represent the speech acts, which move the conversation from state to state. Note that the speech acts are named slightly differently in different sources (by the same author even!), but the structure of a CfA is the same. The simplest route through the diagram is through states 1–5.

  Alison: Have you got the market survey on chocolate mousse?
  Brian: Sure.
  Rummages in filing cabinet and hands it to Alison
  Brian: There you are.
  Alison: Thanks.

9. Explain in detail about text-based communication?OVERVIEW:

- √ **Text-Based Communication**
- √ **Grounding Constraints**
- √ **Turn-Taking**
- √ **Pace And Granularity**
- √ **Linear Text Vs. Hypertext**
- √ **Group Communication**
- √ **Group Dynamics**
- √ **Distributed Cognition**

TEXT-BASED COMMUNICATION

- For *asynchronous* groupware (and even some synchronous systems), the major form of direct communication is text based.
- There are exceptions to this, for instance voice messaging systems and answer phones, and other media may be used in addition to text such as graphics, voice annotation or even video clips. But despite these, text is still the dominant medium.
- Text-based communication is familiar to most people, in that they will have written and received letters. However, the style of letter writing and that of face-toface communication are very different. The text-based communication in groupware systems is acting as a speech substitute, and, thus, there are some problems adapting between the two media.
- There are four types of textual communication in current groupware:
  discrete – directed message as in email. There is no explicit connection between different messages, except in so far as the text of the message refers to a previous one.
  linear – participants' messages are added in (usually temporal) order to the end of a single transcript.
  non-linear – when messages are linked to one another in a hypertext fashion.

spatial – where messages are arranged on a two-dimensional surface.

In addition, the communication may be connected to other shared computer artefacts, which will be described further in Chapter 19 (Section 19.6). In the case where the communication is an annotation, the annotation itself may be structured in any of the ways listed above.

- A special case of a linear transcript is structured message systems such as Coordinator, where not only the order but also the function of each message is determined.
- The other extreme is where the transcript is presented as a single stream, with no special fields except the name of the contributor. Figure 14.2 shows a screen shot of the York Conferencer system showing such a transcript on the left of the screen. On the right is an electronic pin-board, an example of spatially organized text.

Back channels and affective state

- One of the most profound differences between face-to-face and text-based communication is the lack of fine-grained channels.
- Much of the coordination of face-to-face conversation depends on back channels and interpretation of the listener's expressions. Text-based communication loses these back channels completely. Consider the effect of this on even a two-party conversation.
- In addition to this loss of back channels, the speaker's tone of voice and body language are of course absent.
- These normally convey the *affective state* of the speaker (happy, sad, angry, humorous) and the *illocutionary force* of the message (an important and urgent demand or a deferential request).
- Email users have developed explicit tokens of their affective state by the use of 'flaming' and 'smilies', using punctuation and acronyms; for example:

  :-) – smiling face, happy
  :-(– sad face, upset or angry
  ;-) – winking face, humorous
  LOL – laughing out loud.

Grounding constraints

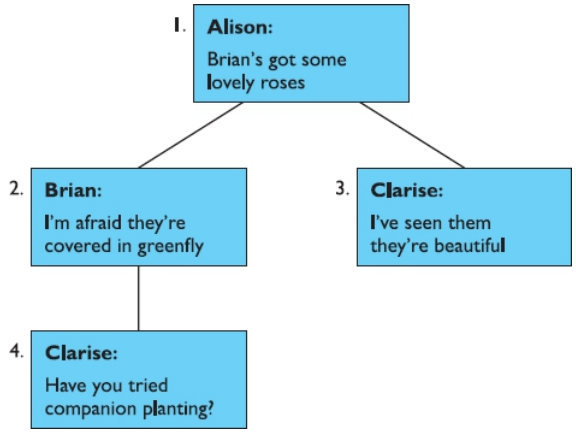
- In Section 14.3.5, we discussed the process by which conversants obtain common ground. This grounding process is linked strongly with the types of channels through which the conversants communicate. Clark and Brennan [71] describe the properties of these channels in terms of *grounding constraints*. These include:

  cotemporality – an utterance is heard as soon as it is said (or typed);
  simultaneity – the participants can send and receive at the same time;
  sequence – the utterances are ordered.

- These are all constraints which are weaker in text-based compared with face-to-faceinteraction. For example, simultaneity in face-to-face conversation allows back channel responses.
- Even where, say, two participants can see each other's typed messages as they are produced, the nature of typing makes it all but impossible to type your message whilst looking for your colleague's 'back channel' response.
- In a text-based system, different participants can compose simultaneously, but they lack cotemporality.
- As we saw, even if the messages appear as they are produced, they will not be read in real time. In addition, the messages may only be delivered when complete and even then may be delayed by slow communications networks.

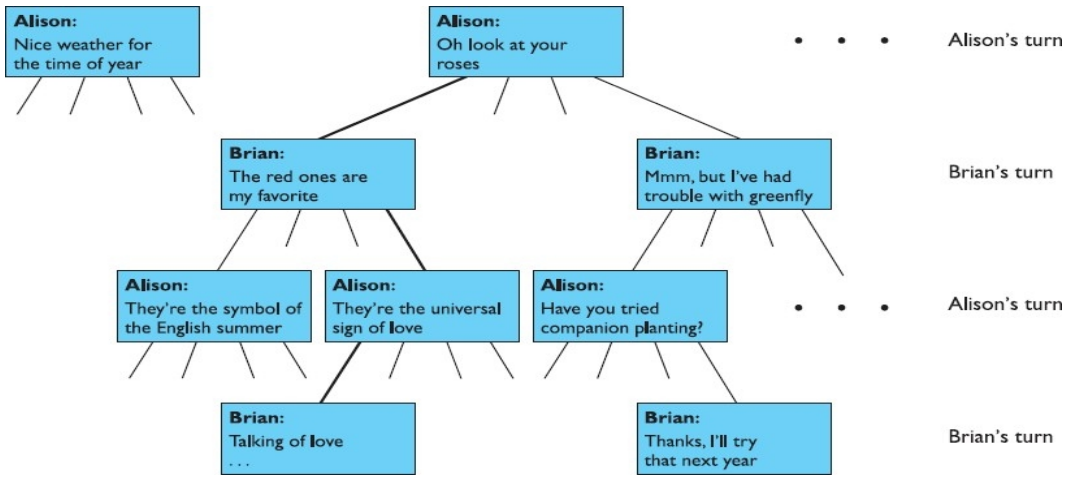
Turn-taking

- We saw that one of the fundamental structures of conversation was *turn-taking* (Section 14.2.5). The last transcript was an example of a breakdown in turn-taking.
- In fact, such breakdowns are quite rare in two-party electronic conversations and are quickly corrected.
- What is more surprising is that such breakdowns so rarely occur during letter writing, which is in some ways similar. However, when conversing by letter, one has an objective timescale with which to work out whether one's fellow conversant ought to have replied.
- One therefore does not send a second letter unless the conversant is very remiss in replying to the first missive Context and deixis We have seen how important context is in ordinary speech.
- Utterances are highly ambiguous and are only meaningful with respect to *external context*, the state of the world, and *internal context*, the state of the conversation. Both of these are problems in text-based communication. nThe very fact that the participants are not co-present makes it more difficult to use external context to disambiguate utterances.
- This is why many groupware systems strive so hard to make the participants' views the same; that is, to maintain WYSIWIS ('what you see is what I see'). In Chapter 19 we will look at an example that shows that this is an issue even when the participants have audio/video communications or are in the same room!



Pace and granularity

- In a spoken conversation, the turns are often only a few tens of seconds long. If we take into account minor confirmations and back channels, the pace is still faster, perhaps a turn or back channel response every few seconds. Compared with this, the pace of email is very slow: messages can take from a few seconds to hours to deliver.
- Even synchronous text-based conversations are limited by the participants' typing speed and have a pace of at most one turn every minute or so.
- The term *pace* is being used in a precise sense above. Imagine a message being composed and sent, the recipient reading (or hearing) the message and then composing and sending a reply.
- The pace of the conversation is the rate of such a sequence of connected messages and replies. Clearly, as the pace of a conversation reduces, there is a tendency for the *granularity* to increase.
- To get the same information across, you must send more per message. However, it is not as easy as that. We have seen the importance of feedback from listener to speaker in clarifying meaning and negotiating common ground.
- Even most monologs are interactive in the sense that the speaker is constantly looking for cues of comprehension in the listener. Reducithe pace of a conversation reduces its *interactivity*.



Linear text vs. hypertext

- Considerations of potential overlap suggest that hypertext-based communications may be better suited as a text-based communication medium.
- Similarly, the problems of pace may be partially solved in a hypertext. Multiplexed messages can be represented as updates to several parts of the hypertext, thus reducing the likelihood of breakdown and lost topics. In addition, if the messages themselves can be mini-hypertexts, then eager messages listing several possible courses of action can be explicitly represented by the message.
- On the other hand, hypertext has its disadvantages. Even static hypertexts, which have been carefully crafted by their authors, can be difficult to navigate.

GROUP WORKING

- So far we have been principally looking at the properties of direct communication, and largely two-party conversations.
- Group behavior is more complex still as we have to take into account the dynamic social relationships during group working. We will begin by looking at several factors which affect group working

Group dynamics

- Whereas organizational relationships such as supervisor/supervisee are relatively stable, the roles and relationships within a group may change dramatically within the lifetime of a task and even within a single work session.
- For example, studies of joint authoring have found that roles such as author, co-author and commentator change throughout the lifetime of a document [254, 295].
- This means that systems, such as co-authoring systems, which use a formal concept of *role*, must allow these roles to change together with the socially defined roles.
- Even the naming of roles can cause problems. A person may be an author of a book or paper, but never write the words in it, acting instead as a source of ideas and comments.
- A particular case of this is the biographical story where the individual concerned and a professional writer co-author the book, but only the professional author writes.

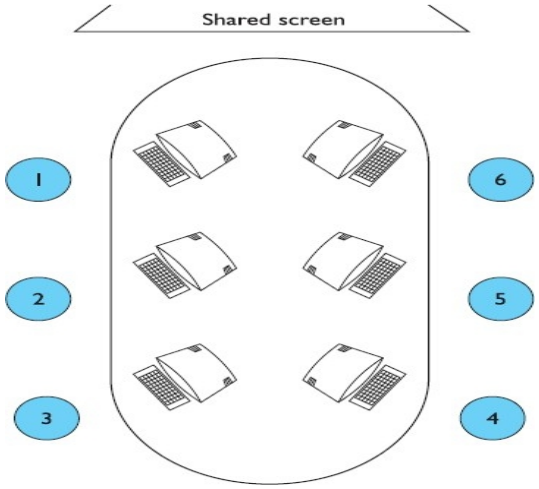
- A co-authoring system such as Quilt would call the non-writing author a 'commentator' or a 'reviewer', but *not* an 'author'. One can imagine some of the social friction such naming will cause.

Physical layout

- In Section 14.2, we discussed the importance of eyegaze and gesture in face-to-face communication and how these help to mediate turn-taking. In particular, we noted in Section 14.2.3 that we must ensure that monitors do not block the participants' views of one another.
- In general, the physical layout of a room has a profound effect upon the working relationship of those in it. This is particularly obvious for meeting rooms, but should be considered in any group-working environment.
- As well as being unobtrusive, the orientation of computing equipment can affect group working.
- If we wish to encourage conversation, as we do in a meeting room, the participants must be encouraged to look toward one another.
- Meeting rooms have a natural focus toward the screen at the front of the room, but inward-facing terminals can counteract this focus and thus encourage eye contact [226].

Distributed cognition

- In Chapter 1, we discussed human cognition, but the emphasis was, as in all traditional psychology, upon the activity *within* the person's head. A school of thinking has recently developed which regards thinking as happening not just within the head, but in the external relationships with things in the world and with other people.
- This viewpoint is called *distributed cognition* [208, 185]. In fact, this viewpoint is not as radical as it first appears. Traditional views talk about the movement of information between *working memory* and *long-term memory*: it is not so difficult then to regard bits of paper, books and computer systems as extensions to these internal memory systems.
- Similarly, many models of human cognition regard the mind as a set of interacting subsystems (see Chapter 12): the step to regarding several people as involved in joint thinking is not difficult.



10. Describe in detail about hypertext, Multimedia, WWW?OVERVIEW:

√ **Understanding Hypertext**

√ **Hypertext Definition – Text, Hypertext And Multimedia**

- √ **Rich Content**
- √ ***Animation***
- √ ***Video And Audio***
- √ **Delivery Technology**
- √ **On The Web**
- √ **On The Move**
- √ ***Rapid Prototyping***
- √ **Education And E Learning**
- √ **Indices, Directories And Search**

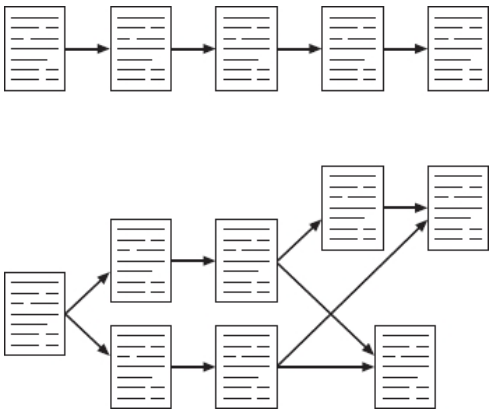
INTRODUCTION

- ❖ Increases in desktop computing power have enabled the rapid growth of the multimedia industry. CD-ROMs, stereo sound and often video input/output are now part of the specification of standard personal computers.
- ❖ Furthermore, the dreams of Vannevar Bush and Ted Nelson (see Chapter 4) have now become reality in the world wide web, which links computers, information and ultimately people across the world.
- ❖

UNDERSTANDING HYPERTEXT

Hypertext definition – text, hypertext and multimedia

- ❖ Although pictures and sculpture came first, it is text – the written word – that is commonly seen as the defining point of civilization; pre-literate and oral societies are often regarded as pre-history simply because they have no extant story.
- ❖ Hieroglyphics, Babylonian mud tablets, the Book of Kells, the Caxton press: these are the stepping-stones toward our current information-centric society.
- ❖ All these traditional texts share a common linear nature. Aristotle in his *Poetics* said that a story must have a beginning, a middle and an end, and even postmodern non-linear narrative is actually written in a linear fashion even though the events may not be causally connected.



Rich content

- ❖ As well as static material – text and static diagrams and photographs – hypertext systems may also include more dynamic material such as animation, video and audio clips, and even full computer applications.

Animation

- ❖ Animation is the term given to the addition of motion to images, making them move, alter and change in time. A simple example of animation in an interface is in the form of

a clock.

- Digital clocks can flick by the seconds, whilst others imitate Salvador Dali and bend and warp one numeral into the next.
- Analog clocks have moving hour and minute hands, with an optional second hand sweeping round the clock face. Such a desktop accessory is found in a lot of interface setups, and the additional processing time required to produce such effects is no longer a major factor.
- Another common use for animation in current windowing systems is to animate the cursor.
- Instead of simply having a basic pointer always on the screen, many interfaces now use the typical 16 16 bitmap that makes up the cursor to indicate more complex information. We saw in Chapter 3 that there are a number of different static cursors in use, but animation takes this one stage further by adding motion to the images.
- This is usually done to indicate that some process is in progress, to confirm to the user that something is actually happening.
- Animating the cursor means that messages do not need to be printed out to a window, making it a neat and concise way of presenting the desired information.

Video and audio

- In a media dominated world, there are strong arguments for using video or audio material as part of hypertext systems whether for education, entertainment or reference.
- Both audio and video material are expensive and time consuming to produce, but increasingly even home-PC systems include video and audio editing as standard. For example, the iMac includes a suite for editing video and burning DVDs.\ Combined with digital video cameras these bring the production of audio/video material into the reach of many.
- Furthermore, standard formats such as QuickTime allow this material to be embedded in web pages for easy distribution. Of course, quality video production requires extensive experience, but then so does text!

Computation, intelligence and interaction

- Of course the good thing about a computer is that not only can it *show* things that have previously been prepared, it can also *do* things. This book has an index, but it does not contain every word in the text (it may also refer to parts of the text that do not have a particular word). However, the web search can look through all the chapters and find any words you want.
- More interactive hypermedia may contain embedded games or applications. For example, Figure 21.3 shows a puzzle from the website of one of the authors (Alan), a sort of 2D Rubik's cube that you can play online.

Delivery technology

- *On the computer* Some hypertexts, in particular help systems, are downloaded or installed permanently on a computer.
- This has the advantage of instant access and such applications need not use a standard viewer but may include their own bespoke browsing software. However, with media-rich hypertexts containing substantial graphics, video and audio clips it may be impractical to store everything on hard disk.
- Also, for copyright protection, some systems will deliberately not allow themselves to be copied from their original distribution media. Many hypermedia systems are supplied on CD-ROM.
- This has the advantage of reasonably large capacity (650–700 Mbytes), but access is

slower than with installed systems. For highly dynamic material, such as educational media, a special player is installed; alternatively, material such as software documentation may use a standard format such as web pages.

On the web

- Of course, the *world wide web* is the best-known multimedia hypertext system of all. The world wide web offers a rich environment for the presentation of information.
- Documents can be constructed that are very different from paper versions; basic text can be augmented through the use of hypertext links to other documents, while graphics can easily be incorporated as pictures, photographs, icons, page dividing bars, or backgrounds. Pages can also have hypertext links embedded into different regions, which take the user to a different page or graphic if they are clicked on; these are known as active image maps.
- These features allow web pages to become interactive, acting as the interface to the information as well as its hold Dynamic material in the form of movies and sounds is also available to the designer; all these features push web page design well away from the conventional paper-based kind.

On the move

- Mobile phones, PDAs (personal digital assistants), and notebook computers have all increased the demand to have hypermedia available on the move.
- Furthermore, across many countries governments have sold franchises for high-bandwidth mobile services. After spending billions on these franchises the telecommunications giants *really* want people to use new mobile services!
- Notebook computers can use just the same mechanisms as desktop computers, using CD-ROM or DVD for standalone material, or connecting to the web through mwireless access points or through modems linked to mobile phone networks.
- However, the fact that the computer is mobile means that location can be used as a key into context-aware hypermedia showing different content depending on location. The 'stick e note' system developed by the University of Kent uses a sticky note metaphor with notes stuck to particular locations [49].

Application areas

- There are many applications of hypermedia, too many to describe in detail here. However, it is worth noting the type of domains in which hypermedia systems have proved successful, looking briefly at some example systems

Rapid prototyping

- Although now lacking the wealth of features expected of a hypermedia system,HyperCard on Macintosh computers has been very influential as a basis for experimental hypertext systems.
- HyperCard uses the metaphor of a card index, around which the user can navigate. Each card can hold text, diagrams, photographs,

Help and documentation

- Hypermedia systems are ideally suited to online manuals and other help system applications (see Chapter 10).
- They allow user-oriented access to the information, and support browsing. In addition the information can be organized hierarchically, with successive selections providing more detailed information.
- This supports the varying needs that users have, such as quick reference, usage

information, full details and so on. Many commercial help systems use hypermedia-style help. Good examples are the Sun Guide system, HyperCard help and Microsoft Windows help (used by many Windows applications).

Education and e-learning Hypertext and hypermedia are used extensively in educational settings, as they allow varied subjects to be related to each other in numerous ways so that the learner can investigate the links between different areas. In contrast to computer-aided learning (CAL) packages, hypermedia allows a student-controlled learning process.

- An early example of a hypermedia learning environment that inspired many subsequent systems was Intermedia [385]. This is a hypermedia system built and used at Brown University to support teaching in subjects as varied as English literature and biology. The system includes text, diagrams, photos and so on.
- Both learners and teachers can add information and links, giving students access to each other's opinions as well as those of their tutors. A map provides an overall view of the information for direct access and navigation, with links providing browsing facilities in the normal way. Intermedia has been successfully used for university-level teaching, and can be seen as a forerunner of the educational resources now facilitated by the web.

Collaboration and community

- Although strictly not hypertext, the web has become a central platform for collaborative applications and community.
- These use the hypertext structure of the web to structure and access shared resources and message areas. For example Yahoo!m Groups (groups.yahoo.com) allows mailing lists, shared images (such as family photo albums), web archives of the mailing list and chat, all accessed through a web interface.

E-commerce

- For some companies the web is simply another sales opportunity. Many readers will have used online stores such as Amazon or bought from an auction site such as eBay. Hypertext's use of hierarchies, links, images and so on, makes it ideal for displaying certain kinds of product.
- Actual buying and selling requires not only security at the level of the networks, websites, etc., but also trust.
- When you walk into a shop you can see the person you are dealing with, and the fact that it is physically there today gives you confidence that it will be there tomorrow if anything goes wrong. How to build and ensure this trust is an active area in HCI research.

FINDING THINGS

Lost in hyperspace

- Although the non-linear structure of hypertext is very powerful, it can also be confusing. It is easy to lose track of where you are, a problem that has been called 'lost in hyperspace'. There are two elements to this feeling of 'lostness'.
- The first is cognitive and related to content. In a linear text, when a topic is being described, the writer knows what the reader has already seen. In a hypertext, the reader can browse the text in any order.
- Each page or node has to be written virtually independently, but, of course, in reality it cannot be written entirely without any assumption of prior knowledge. As the reader encounters fragmentary information, it cannot be properly integrated, leading to confusion about the topic. The second is related to navigation and structure.

Designing structure

- We discussed the importance of good structural design in Chapter 5. Some of the task analysis techniques presented in Chapter 15 may be useful in giving a task-oriented or knowledge-orientated breakdown of a hypertext.
- In some areas there may be preexisting understood structures to mirror; for example, the faculty and departmental structure of a university, or the main disciplines (circulatory, neurological, etc.) within medicine.
- In a paper format one is stuck with a single structure, which can lead to tensions: for example, the fact that in this book structural design is discussed in several places.

Making navigation easier

- No matter how well designed the site structure is, there will still be problems: because the user does not understand the structure; or because the user has individual needs that the designer has not foreseen; or because even a good structure is not perfect.
- However, there are various things that can make it easier for users. One solution is to provide a map of the hypertext document, identifying the current position of the reader within it. Links to home or end points can then be identified and the user is less likely to get lost.
- This may be a separate part of the hypertext; for example, some websites have a site map link leading to a special page, and many help systems have a table of contents view.
- Alternatively, the site map can be woven into the layout of the document; for example, some sites have an outlinerstyle sidebar listing the main sections and drilling down to the current location.
- This acts both as an indication of where you are in the site (like the breadcrumbs discussed in Chapter 5) and as a constant reminder of the overall site structure.

History, bookmarks and external links

- Hypertext viewers and web browsers usually have some sort of history mechanism to allow you to see where you have been, and a more stack-based system using the 'back' button that allows you to backtrack through previously visited pages.
- The back button may be used where a user has followed a hyperlink and then decided it was to the wrong place, or alternatively, when browsing back and forth from a central page that contains lots of links.
- The latter is called *hub and spoke* browsing. In fact, in studies of web browsing the back button accounted for 30% of all navigation actions [63]. Other studies have shown frequent revisiting of pages during a single browsing session [341]. Although the back button is used extensively, it is used relatively little to go back more then one step.

Indices, directories and search

- As well as a hierarchical table of contents structure, many help systems, hypertexts, and for that matter paper books, have some kind of index. Note that an index is not a complete list of all words in a document. If this were the case then the index for this book would be as big as the rest of the book!
- On the web an index would be very big (!); however, directory services such as Yahoo! (www.yahoo.com) or the Open Directory Project (ODP) (www.dmoz.org) can be seen as a form of index.
- The main difference is that while an index is simply an alphabetical list of keywords, web directories give a hierarchical categorization to sites. The categorization is done either by self-submission, or, in the case of quality directories such as ODP or Yahoo!, by experts in the relevant field.

11. Explain in detail about web technologies and isuues?

OVERVIEW:

- √ **Web Technology And Issues**
- √ **Basics**
- √ **Web Servers And Web Clients**
- √ **Network Issues**

WEB TECHNOLOGY AND ISSUES

- ❖ The web has featured strongly already in this chapter and for the remainder of the chapter we will focus exclusively on it.

Basics

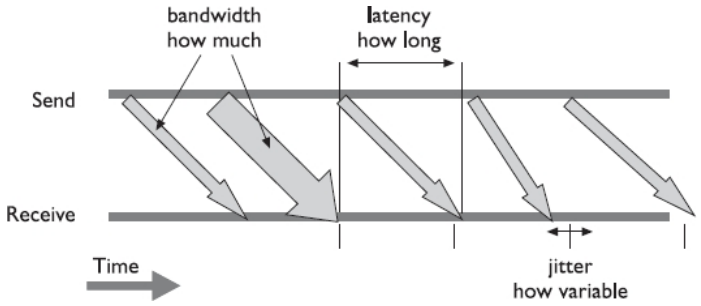
- ❖ The web consists of a set of protocols built on top of the *internet* that, in theory, allow multimedia documents to be created and read from any connected computer in the world.
- ❖ The web supports hypertext, graphics, sound and movies, and, to structure and describe the information, uses a language called *HTML* (hypertext markup language) or in some cases, XML (extensible markup language). HTML is a markup language that allows hypertext links, images, sounds and movies to be embedded into text, and it provides some facilities for describing how these components are laid out.
- ❖ HTML documents are interpreted by a viewer, known as a *browser*; there are many browsers, and each can interpret HTML in subtly different ways, or support different levels of functionality, which means that a web page viewed through one browser can look very different from the same page viewed through another.
- ❖ The web requires no particular multimedia capabilities from the machines that ru the browsers; for example, if sound is unavailable on a particular machine, then obviously no sound is heard but the browser still displays the text happily. applications. As well as publishing personal, corporate and governmental information, the web is used as a source of entertainment, an advertising medium, a communication environment, and more.

Web servers and web clients

- ❖ Whereas a conventional PC program runs and is displayed on one computer, the web is *distributed*.
- ❖ Different parts of it run on different computers, often in different countries of the world. They are linked, of course, by the internet, an enormous global computer network (see also Chapter 2, Section 2.9.3).
- ❖ The pages are stored on web servers that may be on a company's own premises or in special data centers.
- ❖ Because they are networked, the webmaster for a site can upload pages to the server from wherever she is.
- ❖ For example, the web pages for www.hcibook.com are stored in a data center several thousand miles from where any of the authors live! Your machine, the PC running the web browser, is called a *client* because it wants the pages from the servers. When you click on a link your web browser works out the full URL of the page it needs: say **'http://www.hcibook.com/e3/authors.html'**.
- ❖ It splits this into parts. The first part is the protocol 'http' which says how it talks to the server (other alternatives include 'ftp'). The second part 'www.hcibook.com' is the host name, that is the name of the web server containing the requested page. The last part '/e3/authors.html' gives the particular file on the site.

Network issues

- The fact that the web is networked raises a series of issues that can impact on usability.
- Network capacity is called *bandwidth*. This is a measure of the amount of information that can pass down the channel in a given time. For example, a typical modem speed is 56 kbs – that is 56 kilobits per second. This equates to about 6000 characters per second.
- This sounds fine until you realize that images may take many tens or hundreds of characters (bytes) to encode . . . this is why many have renamed the web the 'world wide wait'!
- However, bandwidth is not the only important measure. There is also the time it takes for a message to get across the network from your machine to the web server and back. This delay is called *latency*.
- As well as the underlying latency and jitter of the network, each layer of network software adds its own. The underlying internet protocols are *lossy*, that is messages can be lost. When this happens, higher level software (the TCP – transmission control protocol – layer) notices and resends messages to give reliable communication.



- These losses mean that the average delays increase, but also that the jitter increases – a lost and resent message takes more time than one that gets there first time.
- There may also be an appreciable amount of time setting up a new connection, which may outweigh the time taken to actually send data. This is particularly aproblem with sites containing many small images.

12. Describe in detail about static web content?OVERVIEW:

- √ **The Message And The Medium**
- √ **Text**
- √ ***Obtaining Graphics***
- √ **Icons**
- √ ***Graphics And Color***
- √ **Movies And Sound**

STATIC WEB CONTENT

The message and the medium

- One thing is often forgotten when web pages are created. It is of vital importance, and hence will be discussed first. It is content.
- Many people assume that because they can make information available on the web, they should. Unfortunately, because it is very easy to publish information, much less care is taken with the actual content. Material may be nonsense, it may be incorrect, it may not read well, or be incomplete, or inane.
- Excellent page design can make useless material look attractive, but it still remains useless material.
- On the other hand, poor design can mean that excellent material is never seen by

potential readers, as they have become bored, or intolerant of the medium, or confused, or for a host of other reasons have aborted their attempts to download and view the information.

- Pages do have to look immediately interesting nand attractive if people are to spend time, effort and, because of the communication costs, money, in viewing them; the user-centered nature of the medium makes this imperative.
- This is in marked contrast to television or cinema or other dynamic media, which are not under any direct user control, where information is presented to a passive audience. With web documents, people have actually to want to see the information.

Text

- Because web pages are displayed on many different machines, there are only a small set of fonts that can be guaranteed to be available: a standard font and a typewriter font (e.g. courier) with bold and italic versions in different sizes. However, it is possible to specify preferred fonts and many of these such as *Arial*, Verdana or Comic Sans are available on most web platforms.
- The difficult thing is to balance fine tuning the appearance of the text on one platform with making it readable on all.
  The various structured styles such as headings allow the web designer to create material that will lay out passably on all platforms. But these offer a fairly coarse level of control. The size and boldness of the heading should be chosen carefully; for example, huge dark fonts on a page can look loud and brash.

Obtaining graphics

- There are a number of sites on the web that contain archives of graphical images, icons, backgrounds and so on.
- There are also paint and image manipulation packages available on almost all computer systems, and scanners and digital cameras, where available, enable the input of photographs and diagrams.

Using graphics

- While graphics and icons tend to play a significant role in web page design, their use should be carefully thought out. Graphical images take longer to load than text, and this may become a problem.
- Text uses 8 bits to represent a character: some rough calculations show that approximately 2000 characters represent about a screenful of information, and so 16,000 bits (2 K) are required. For graphics, one pixel may use 8 bits to represent its color: a page-sized image will be at least 600 by 400 pixels, which will take 1,920,000 bits (240 K), or 120 times as long to load.

Icons

- Icons often appear on web pages, and while there are many available to choose from, they should be used with care. On web pages, icons are typically used in one of two ways.
- They are either visual cues, associating some small picture with different parts of the text (for example, some pages have icon-sized characters that appear next to instructions).
- Alternatively, they are used in much the same way as in a standard WIMP interface to represent aspects of the functionality of the underlying pages. In this latter case, they must represent their associated functionality in either a concrete or an abstract form.
- This means that the design of the individual icon has to be carefully thought out, as a lot of information may have to be represented in a small area of screen estate.
- However, icons are rarely seen on their own, and when placed next to their neighbors, the

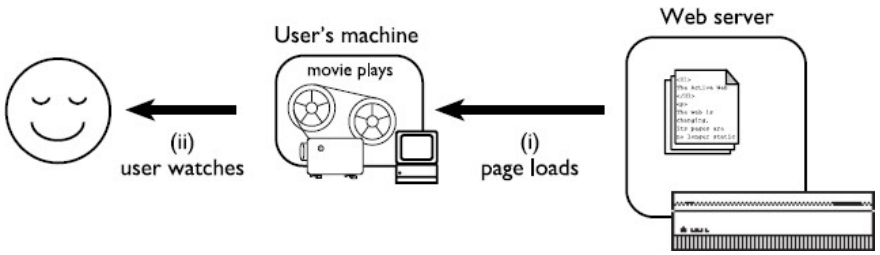
whole effect has to be pleasing rather than disruptive and garish. Therefore, the group of icons has to be designed together, with a coherent and recognizable style.

Graphics and color

- Using many different colors within graphics may well result in the browsers for older machines running out of entries in the colormap, with unpredictable consequences.
- This is often problematical as the browser may be running in tandem with other color applications, and only has a restricted range of colors to begin with.
- For many consumer markets, for example in the UK and the US, this is unlikely to be a problem as home machines are often relatively recent. However, many businesses continue to use older PCs so long as they 'do the job' and PDAs may not have a full color palette. Furthermore, in economically deprived areas, where there is computer access it may well be through older or second-hand machines.
- If universal access is required it is therefore still wise, where possible, to restrict images to a limited number of colors, taken from the standard 216 color web palette, and to reduce complex color images to simpler approximations.
- Reducing the number of colors used also allows the depth of the images to be reduced; a change from a default of 8 bits to, say, 4 bits will produce a twofold speedup in image loading. The earlier comments on the use of color obviously apply as much to graphics as they do to text.

Movies and sound

- Movies and sound are both available to users of the web, and hence to page designers.One problem associated with them is actually obtaining appropriate sound and video clips, as they usually require some sort of multimedia capability on behalf of the host machine in order to be able to digitize sound and capture and digitize video.
- Video suffers from the same problems as graphics, magnified by an order of magnitude or two; it can take extremely large amounts of time for a video segment to download.



- Video is also not well integrated into the web, requiring the creation of a process to run it that is separate from the page from whence it came. Not all receiving machines have the capability to play video, or sound, and so it is unwise for a designer to rely on these dynamic media to convey information without replicating it elsewhere.
- The use of sound and video moves page design further away from the typesetter and toward the sound engineer and cinematographer;
- the integration of these cinematic media with the enhanced textual capabilities offered by the web is a new domain, in which the techniques that work and those that fail have not yet been fully explored, let alone understood.

13. Explain in detail about dynamic web content?OVERVIEW:

√ **The Active Web**
√ ***The User View***

- √ ***Technology And Security***
- √ **Automatic Generation**
- √ **Batch Generation**
- √ **Dynamic Content**

DYNAMIC WEB CONTENT

The active web

- In the early days, the web was simply a collection of (largely text) pages linked together. The material was static or slowly changing and much of it authored and updated by hand.
- Some pages were generated on the fly, in particular the gateways into ftp servers and to gophers, which were so important in adding 'free' content to the web [97]. However, even here the user's model was still of a static repository of information.
- Web surfers may not have always known where they were, but they had a pretty good idea of what they were seeing and that if they came back it would be the same.
- It was a pleasant, if somewhat boring world, but from a usability viewpoint it was wonderful – a consistent interface to terabytes of information. Who could ask for more? Indeed, this is one of the key arguments Nielsen brings against frames-rich\ sites in his famous alertbox, *Why frames suck (most of the time)* [263] – frames break this simple user model and hence cause trouble.
- Nielsen calls for a new richer model for the web, which preserves the simplicity of the old model, but which can accommodate and guide the development of new features. Well, if frames cause trouble, what about
- It has a major impact on the pace of interaction, both *feedback*, how fast users see the effects of their own actions, and *feedthrough*, how fast they see the effects of others' actions. Also, where the computation happens influences where data has to be moved to with corresponding effects on download times and on the security of the data.

The user view

One set of issues is based on what the end-user sees, the end-user here being the web viewer.

What changes? This may be a media stream (video, audio or animation) which is changing simply because it is the fundamental nature of the medium. It may be the presentation or view the user has of the underlying content; for example, sorting by different categories or choosing text-only views for blind users. A special form of presentation change is when only a selection of the full data set is shown, and that selection changes. The deepest form of change is when the actual content changes.

By whom? Who effects the changes? In the case of a media stream or animation, the changes are largely automatic – made by the computer. The other principal sources of change are the site author and the user. However, in complex sites users may see each other's changes – feedthrough.

How often? Finally, what is the pace of change? Months, days, or while you watch? We'll use the 'what changes?' categories as we examine alternatives and trade-offs in more detail below. But first we also need to look at the technological constraints.

Technology and security

The fundamental question here is where 'computation' is happening. If pages are changing, there must be some form of 'computation' of those changes. Where does it happen?

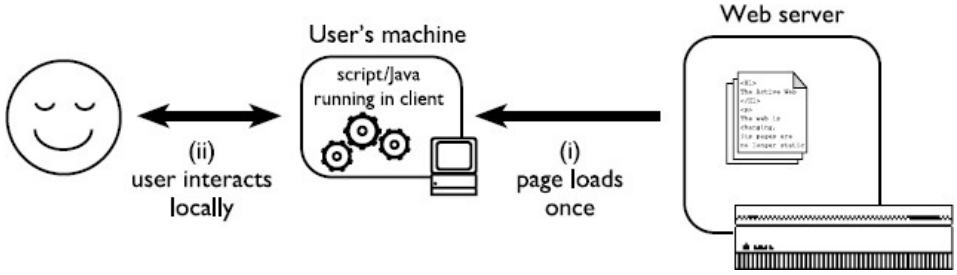
Client One answer is in the user's web-browsing client enabled by Java applets, various plug-ins such as Flash, scripting using JavaScript or VBScript with dynamic HTML

layers, CSS and DOM (Domain Object Model).

Server A second possibility is at the web server using CGI scripts (written in Perl, C, UNIX shell, Java or whatever you like!), Java Servlets, Active Server Pages or one of the other server-specific scripting languages such as PHP. In addition, client-side Java applets are only allowed to connect to networked resources on the same machine as they came from. This means that databases accessed from clientside JDBC (Java database connectivity) must run on the web server (see below).

Another machine Although the pages are *delivered* from the web server, they may be *constructed* elsewhere. For hand-produced pages, this will usually be on the page author's desktop PC. For generated pages, this may be a PC or a central database server.

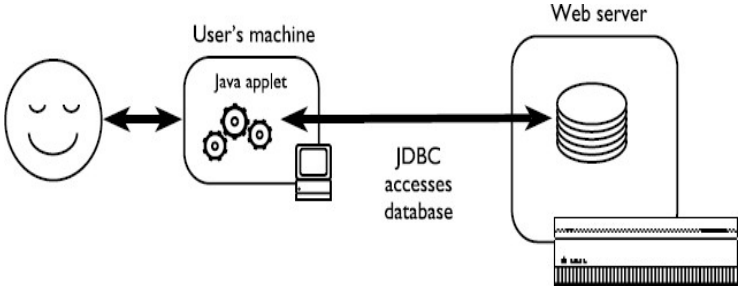
People Of course, as noted earlier, the process of production and update may even involve people! Fixed content – local interaction and changing views Probably the most hyped aspect of the web in recent years has been Java. In fact, Java can be used to write server-end software and platform independent standalone programs (not to mention the embedded systems for which it was originally designed!), but the aspect that most people think of is Java applets.



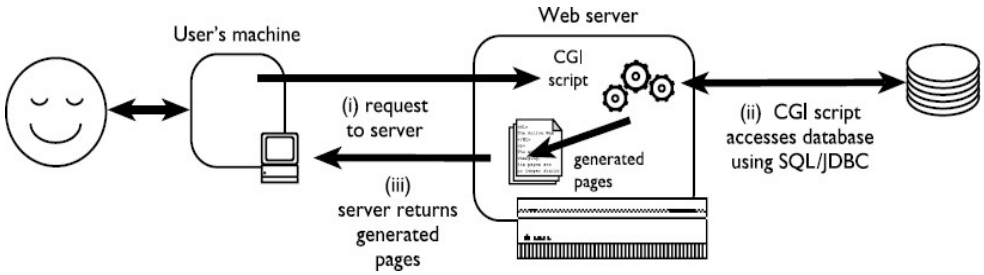
- Applets are just one of the techniques that can be added to give client-end interaction (and about the least well integrated into the rest of the page).
- The most common alternatives are JavaScript, Flash and if you are prepared to limit yourself to Windows platforms, ActiveX plug-ins.
- These techniques share the characteristic that they are downloaded to the user's own machine (see Figure 21.9) and thereafter all interaction
- The user's keywords are submitted to the server using an HTML form, they are compared against pre-prepared indexes at the server and all matching paragraphs in the book] are returned (Figure 21.12). This also reminds us of another reason for not downloading all the text to the user's machine – security; we don't want to distribute the full electronic text for free!

Automatic generation

- It was evident in the earliest days of the web that a key problem for the future would be maintenance. In the first rush of enthusiasm, individuals and organizations produced extensive and fascinating sites built largely of hand-crafted HTML.



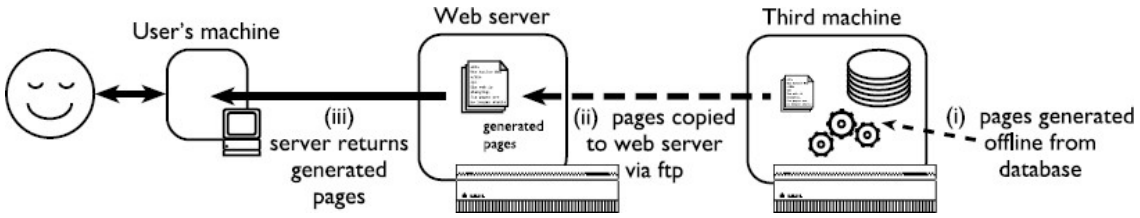
- Not surprisingly, vast areas of the web are not just static but in perpetual stasis. Web surfing sometimes seems not so much a watersport, but an exercise in archaeology.
- From the beginning it was clear that websites would eventually need to be createdfrom databases of content combined with some form of templates or layout description. However, at that stage there were no tools available and those who saw the database future used a variety of ad hoc methods.



- Happily, there are now a (sometimes bewildering) array of products for automating web production from existing and bespoke databases. These include vendor-specific products such as Oracle Web Server and Domino (for publishing Lotus Notes), and also more general techniques such as using SQL (structured query language) or JDBC to access databases from CGI scripts or even from running Java applets.'

Batch generation

- A low-tech but very secure solution is to generate pages offline from a database and then upload them to the web server (Figure 21.15).
- Many of Alan's earliest web pages were generated in this way from HyperCard stacks. This is certainly a simple solution as it separates out the task of page generation from that of page delivery. Pages can be generated directly using many standarddatabase packages such as Access or HyperCard.



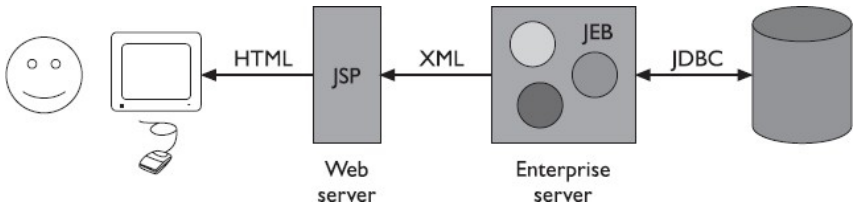
- Alternatively, standalone programs in languages such as Visual Basic, Java or C can access a database and output HTML pages. These programs can run on a central computer, or on your own PC. The generating program simply produces a set of HTML pages on your own disk that can be checked locally and then copied onto the web server using ftp or shared network disks.
- Many people think that this will be difficult, but in reality it is remarkably easy, as you can use the tools you are used to – if you can create a text file you can create HTML. In fact, the snippet of Visual Basic in Figure 21.16 is a trivial but fully

```
functioning HTML generator!
Set db = openDatabase("C:\test.mdb");
sql = "select Name, Address from
Personnel;"
Set query = db.OpenRecordset(sql)
Open "out.html" For Output As #1
Print #1, "<h1>Address List</h1>"
query.MoveFirst
```

```
While Not query.EOF
Print #1, "<p>" & query("Name") & " ";
query("Address")
query.MoveNext
Wend
Close #1
query.Close
```

Dynamic content

- The mechanisms we have been discussing manage the feedthrough when the database is updated by some non-web means.
- Perhaps the most 'active' web pages are those where the content of the pages reacts to and is updateable by the web user. If pages are generated from database content using either the Java-applet JDBC method or the CGI method, the same mechanisms can as easily be used to update as to access the database.
- The feedback of changes to the user is thus effectively instantaneous – you check for seat availability on the theatre web page, select a seat, menter your credit card details and not only is the seat booked, but you can see it change from free to booked on the web page. This sort of web application opens up many additional problems. You may need to add some form of login or authentication.
- If credit card numbers are supplied you need to ensure that the web server is secure. Also, without care it is easy to design solutions that accidentally book multiple seats if the user presses the back button and ends up on what appears to be a simple confirmation screen.
- Going in the direction of greater complexity, many business applications operate an *n*-tier web architecture.
- This involves multiple layers of software where the outer layers are concerned more with the user interface and the inner layers more with business functionality. Figure 21.17 shows this using several web standards.
- The user interacts through a web browser with a web server. The pages are generated using Java Servlet Pages (JSP). To generate the page the servlets connect to Java Enterprise Beans (JEB) on an enterprise server. These are components that encapsulate 'business logic'.



- For example, in a banking system this could include rules on whether a particular transaction is allowed. These Java Enterprise Beans draw their data from the corporate database using JDBC connections

UNIT IV MOBILE HCI

Mobile Ecosystem: Platforms, Application frameworks- Types of Mobile Applications: Widgets, Applications, Games- Mobile Information Architecture, Mobile 2.0, Mobile Design: Elements of Mobile Design, Tools.

1. What is Mobile EcoSystem?

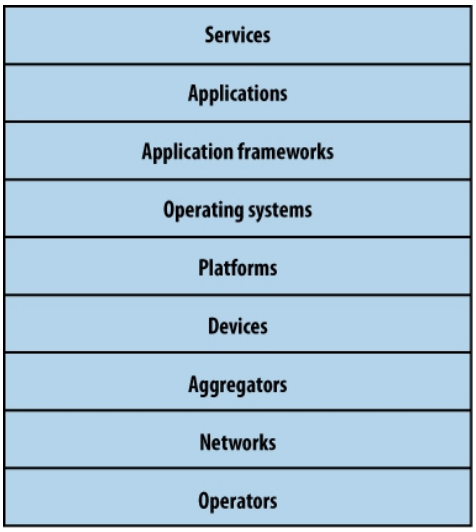
OR

Q1)Give detail description about Mobile EcoSystem:

Overview:

Operators
Networks
Aggregators
Devices
Platforms
Operating Systems
Application frameworks
Applications
Services

Def:



Operators:

The base layer in the mobile ecosystem is the ***operator***. Operators can be referred to as **Mobile Network Operators** (MNOs); mobile service providers, wireless carriers, or simply carriers; mobile phone operators; or cellular companies.

In the mobile community, officially refer them as operators, though in the United States, there is a tendency to call them ***carriers.*** Operators are what essentially make the entire mobile ecosystem work. They are the gatekeepers to the kingdom.

They install cellular towers, operate the cellular network, make services (such as the Internet) available for mobile subscribers, and they often maintain relationships with the subscribers, handling billing and support, and offering subsidized device sales and a network of retail stores.

The operator's role in the ecosystem is to create and maintain a specific set of wireless services over a reliable cellular network. Often the mobile startups and companies that succeed are the ones with the best "carrier relations man," or person with the best relationship to the operators.

Table 2-1 lists the rank, markets, technologies used, and subscriber numbers for the

Networks:

Operators operate wireless networks. Remember that cellular technology is just a radio that receives a signal from an antenna. The type of radio and antenna determines the capability of the network and the services you can enable on it.

That the vast majority of networks around the world use the GSM standard (see Table 2-2) using GPRS or GPRS EDGE for 2G data and UMTS or HSDPA for 3G.

The CDMA (Code Division Multiple Access) and its 2.5G hybrid CDMA2000, which offers greater coverage than its more widely adopted rival.

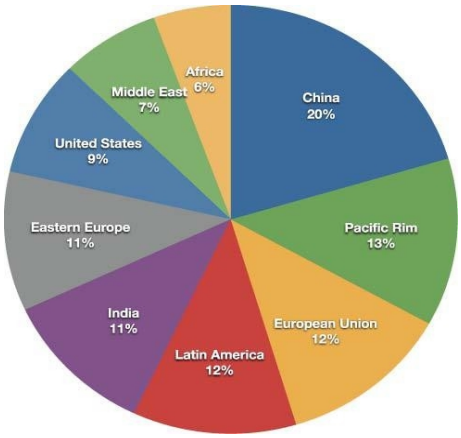
Table 2-2. GSM mobile network evolutions

2G Second generation of mobile phone standards and technology		**Theoretical max data speed**
GSM	Global System for Mobile communications	12.2 KB/sec
GPRS	General Packet Radio Service Max	60 KB/sec
EDGE	Enhanced Data rates for GSM Evolution	59.2 KB/sec
HSCSD	High-Speed Circuit-Switched Data	57.6 KB/

3G Third generation of mobile phone standards and technology		**Theoretical max data speed**
W-C DMA	Wideband Code Division Multiple Access	14.4 MB/sec
UMTS	Universal Mobile Telecommunications System	3.6 MB/sec
TD-CDMA	Time Divided Code Division Multiple Access	16 MB/sec
HSPA	High-Speed Packet Access	14.4 MB/sec
HSDPA	High-Speed Downlink Packet Access	14.4 MB/sec
HSUPA	High-Speed Uplink Packet Access	5.76 MB/sec

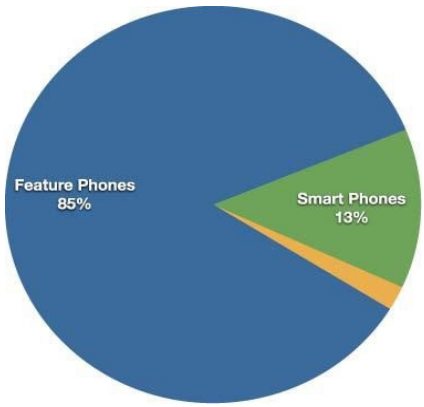
Devices:

The biggest slice of the device pie—mobile phones. As of 2008, there are about 3.6 billion mobile phones currently in use around the world; just more than half the planet's population has a mobile phone (see Figure 2-2).



Most of these devices are feature phones, making up the majority of the marketplace. Smartphones make up a small sliver of worldwide market share and maintain a healthy percentage in the United States and the European Union; Smartphone market share is growing with the introduction of the iPhone and devices based on the Android platform.

As next-generation devices become a reality, the distinction between feature phones and smartphones will go away. In the next few years, feature phones will largely be located in emerging and developing markets. Figure 2-3 shows a breakdown of devices.



Platforms:

A mobile platform's primary duty is to provide access to the devices. To run software and services on each of these devices, need a *platform*, or a core programming language in which all of your software is written.

All software platforms, these are split into three categories:

i) Licensed
ii) Proprietary

iii) Open source.

i) Licensed:

Licensed platforms are sold to device makers for nonexclusive distribution on devices. The goal is to create a common platform of development Application Programming Interfaces (APIs) that work similarly across multiple devices with the least possible effort required to adapt for device differences, although this is hardly reality.

Following are the licensed platforms:

Java Micro Edition (Java ME):

Formerly known as J2ME, Java ME is by far the most predominant software platform of any kind in the mobile ecosystem. It is a licensed subset of the Java platform and provides a collection of Java APIs for the development of software for resource constrained devices such as phones.

Binary Runtime Environment for Wireless (BREW):

BREW is a licensed platform created by Qualcomm for mobile devices, mostly for the U.S. market. It is an interface-independent platform that runs a variety of application frameworks, such as C/C++, Java, and Flash Lite.

Windows Mobile:

Windows Mobile is a licensable and compact version of the Windows operating system, combined with a suite of basic applications for mobile devices that is based on the Microsoft Win32 API.

LiMo:

LiMo is a Linux-based mobile platform created by the LiMo Foundation. Although Linux is open source, LiMo is a licensed mobile platform used for mobile devices.LiMo includes SDKs for creating Java, native, or mobile web applications using the WebKit browser framework.

ii) Proprietary:

Proprietary platforms are designed and developed by device makers for use on their devices. They are not available for use by competing device makers. These include:

Palm:

Palm uses three different proprietary platforms. Their first and most recognizable is the Palm OS platform based on the C/C++ programming language; this was initially developed for their Palm Pilot line, but is now used in low-end smartphones

such as the Centro line. As Palm moved into higher-end smartphones, they started using the Windows Mobile-based platform for devices like the Treo line. The most recent platform is called webOS, is based on the WebKit browser framework, and is used in the Prē line.

BlackBerry:

Research in Motion maintains their own proprietary Java-based platform, used exclusively by their BlackBerry devices.

iPhone:

Apple uses a proprietary version of Mac OS X as a platform for their iPhone and iPod touch line of devices, which is based on Unix.

iii) Open Source:

Open source platforms are mobile platforms that are freely available for users to download, alter, and edit. Open source mobile platforms are newer and slightly controversial, but they are increasingly gaining traction with device makers and developers.

Android is one of these platforms. It is developed by the Open Handset Alliance, which is spearheaded by Google. The Alliance seeks to develop an open source mobile platform based on the Java programming language.

Operating Systems:

Operating systems often have core services or toolkits that enable applications to talk to each other and share data or services. Mobile devices without operating systems typically run "walled" applications that do not talk to anything else.

Although not all phones have operating systems, the following are some of the most common:

Symbian:

Symbian OS is a open source operating system designed for mobile devices, with associated libraries, user interface frameworks, and reference implementations of common tools.

Windows Mobile:

Windows Mobile is the mobile operating system that runs on top of the Windows Mobile platform.

Palm OS:

Palm OS is the operating system used in Palm's lower-end Centro line of mobile phones.

Linux:

The open source Linux is being increasingly used as an operating system to power smartphones, including Motorola's RAZR2.

Mac OS X:

A specialized version of Mac OS X is the operating system used in Apple's iPhone and iPod touch.

Android:

Android runs its own open source operating system, which can be customized by operators and device manufacturers.

Application Frameworks:

Application frameworks often run on top of operating systems, sharing core services such as communications, messaging, graphics, location, security, authentication, and many others.

Java:

Applications written in the Java ME framework can often be deployed across the majority of Java-based devices, but given the diversity of device screen size and processor power, cross-device deployment can be a challenge.

Most Java applications are purchased and distributed through the operator, but they can also be downloaded and installed via cable or over the air.

S60 :

The S60 platform, formerly known as Series 60, is the application platform for devices that run the Symbian OS. S60 is often associated with Nokia devices—Nokia owns the platform—but it also runs on several non-Nokia devices. S60 is an open source framework.

S60 applications can be created in Java, the Symbian C++ framework, or even Flash Lite.

BREW:

Applications written in the BREW application framework can be deployed across the majority of BREW-based devices, with slightly less cross-device adaption than other frameworks.

However BREW applications must go through a costly and timely certification process and can be distributed only through an operator.

Flash Lite;

Adobe Flash Lite is an application framework that uses the Flash Lite and ActionScript frameworks to create vector-based applications. Flash Lite applications can be run within the Flash Lite Player, which is available in a handful of devices around the world.

Flash Lite is a promising and powerful platform, but there has been some difficulty getting it on devices. A distribution service for applications written in Flash Lite is long overdue.

Windows Mobile :

Applications written using the Win32 API can be deployed across the majority of Windows Mobile-based devices. Like Java, Windows Mobile applications can be downloaded and installed over the air or loaded via a cable-connected computer.

Cocoa Touch:

Cocoa Touch is the API used to create native applications for the iPhone and iPod touch. Cocoa Touch applications must be submitted and certified by Apple before being included in the App Store.

Once in the App Store, applications can be purchased, downloaded, and installed over the air or via a cable-connected computer.

Android SDK:

The Android SDK allows developers to create native applications for any device that runs the Android platform.

By using the Android SDK, developers can write applications in C/C++ or use a Java virtual machine included in the OS that allows the creation of applications with Java, which is more common in the mobile ecosystem.

Web Runtimes (WRTs):

Nokia, Opera, and Yahoo! provide various Web Runtimes, or WRTs. These are meant to be miniframeworks, based on web standards, to create mobile widgets. Both Opera's and Nokia's WRTs meet the W3C-recommended specifications for mobile widgets.

Although WRTs are very interesting and provide access to some device functions using mobile web principles, I've found them to be more complex than just creating a simple mobile web app, as they force the developer to code within an SDK rather than just code a simple web app.

WebKit:

With Palm's introduction of webOS, a mobile platform based on WebKit, and given its predominance as a mobile browser included in mobile platforms like the iPhone, Android, and S60, and that the vast majority of mobile web apps are written specifically for WebKit, I believe we can now refer to WebKit as a mobile framework in its own right.

WebKit is a browser technology, so applications can be created simply by using web technologies such as HTML, CSS, and JavaScript. WebKit also supports a number of recommended standards not yet implemented in many desktop browsers.

Applications can be run and tested in any WebKit browser, desktop, or mobile device.

The Web:

The Web is the only application framework that works across virtually all devices and all platforms. Innovation and usage of the Web as an application framework in mobile has been lacking for many years, increased demand to offer products and services outside of operator control, together with a desire to support more devices in shorter development cycles, has made the Web one of the most rapidly growing mobile application platforms to date.

Applications:

Application frameworks are used to create applications, such as a game, a web browser, acamera, or media player.

Although the frameworks are well standardized, the devices are not. The largest challenge of deploying applications is knowing the specific device attributes and capabilities.

Services:

Finally, the last layer in the mobile ecosystem: services. Services include tasks such as accessing the Internet, sending a text message, or being able to get a location—basically, anything the user is trying to do.

Q2: Briefly explain about Mobile Application Medium Types:

The *mobile medium type* is the type of application framework or mobile technology that presents content or information to the user. It is a technical approach regarding which type of medium to use.

Overview:

SMS
Mobile Website
Mobile Web Widgets
Mobile Web Applications
Native Applications
Games
Mobile application Media Matrix

SMS:

The most basic mobile application is an SMS application. Although it might seem odd to consider text messages applications, they are nonetheless a designed experience.

SMS applications can be both “free,” meaning that there is no additional charge beyond the text message fees an operator charges, and “premium,” meaning that you are charged an additional fee in exchange for access to premium content.

The most common uses of SMS applications are mobile content, such ringtones and images, and to interact with actual goods and services. Some vending machines can dispense beverages when

you send them an SMS; SMS messages can also be used to purchase time at a parking meter or pay lot.

A great example of how SMS adds incredible value would be Twitter, where users can receive SMS alerts from their friends and post to their timeline from any mobile device.

Pros:
The pros of SMS applications include:
- They work on any mobile device nearly instantaneously.
- They're useful for sending timely alerts to the user.
- They can be incorporated into any web or mobile application.
- They can be simple to set up and manage.

Cons:
The cons of SMS applications include:
- They're limited to 160 characters.
- They provide a limited text-based experience.
- They can be very expensive

Mobile Websites:

A mobile website is a website designed specifically for mobile devices, not to be confused with viewing a site made for desktop browsers on a mobile browser. Mobile websites are characterized by their simple "drill-down" architecture, or the simple presentation of navigation links.

Mobile websites have made up the majority of what we consider the mobile web for the past decade, starting with the early WML-based sites and moving to today's websites, with a richer experience that more closely resembles the visual aesthetic users have come to expect with web content.

Though mobile websites are fairly easy to create, they fail to display consistently across multiple mobile browsers—a trait common to all mobile web mediums.

The mobile web has been gradually increasing in usage over the years in most major markets, but the limited experience offered little incentive to the user. Many compare the mobile web to a 10-year-old version of the Web: slow, expensive to use, and not much to look at.

As better mobile browsers started being introduced to device platforms like the iPhone and Android, the quality of mobile websites began to improve dramatically, and with it, usage improved.

Pros:
The pros of mobile websites are:
- They are easy to create, maintain, and publish.
- They can use all the same tools and techniques you might already use for desktop sites.
- Nearly all mobile devices can view mobile websites.

Cons:

The cons of mobile websites are:

- They can be difficult to support across multiple devices.
- They offer users a limited experience.
- Most mobile websites are simply desktop content reformatted for mobile devices.
- They can load pages slowly, due to network latency.

Mobile Web Widgets:

A mobile web widget is another attempt by the mobile industry to hype a technology that no one wants.A component of a user interface that operates in a particular way. The ever-trusty Wikipedia defines a web widget this way:

A portable chunk of code that can be installed and executed within any separate HTMLbased web page by an end user without requiring additional compilation.

Between these two definitions is a better answer: A mobile web widget is a standalone chunk of HTML-based code that is executed by the end user in a particular way.

Mobile web widgets are small web applications that can't run by themselves; they need to be executed on top of something else. Opera Widgets, Nokia Web RunTime (WRT), Yahoo! Blueprint, and Adobe Flash Lite are all examples of widget platforms that work on a number of mobile handsets.

Using a basic knowledge of HTML (or vector graphics in the case of Flash), one can create compelling user experiences that tap into device features and, in many cases, can run while the device is offline.

Pros:

The pros of mobile web widgets are:

- They are easy to create, using basic HTML, CSS, and JavaScript knowledge.
- They can be simple to deploy across multiple handsets.
- They offer an improved user experience and a richer design, tapping into device features and offline use.

Cons:

The cons of mobile web widgets are:

- They typically require a compatible widget platform to be installed on the device.
- They cannot run in any mobile web browser.
- They require learning additional proprietary, non-web-standard techniques.

Mobile Web Applications:

Mobile web applications are mobile applications that do not need to be installed or compiled on the target device. Using XHTML, CSS, and JavaScript, they are able to provide an application-like experience to the end user while running in any mobile web browser.

Web applications allow users to interact with content in real time, where a click or touch performs an action within the current view.

The Web 2.0 movement brought user-centered design principles to the desktop web, and those same principles were sorely needed in the mobile web space as well.

The challenge, as always, was device fragmentation. The mobile browsers were years behind the desktop browsers, making it nearly impossible for a mobile device to render a comparable experience.

While XHTML support had become fairly commonplace across devices, the rendering of CSS2 was wildly inconsistent, and support for Java- Script, necessary or simple DHTML, and Ajax was completely nonexistent.

With the introduction of the first iPhone, a cataclysmic change across the board.Using WebKit, the iPhone could render web applications not optimized for mobile devices as perfectly usable, including DHTML- and Ajax-powered content. Developers quickly got on board, creating mobile web applications optimized mostly for the iPhone.

Pros:

- They are easy to create, using basic HTML, CSS, and JavaScript knowledge.
- They are simple to deploy across multiple handsets.
- They offer a better user experience and a rich design, tapping into device features and offline use.
- Content is accessible on any mobile web browser.

Cons:

- The optimal experience might not be available on all handsets.
- They can be challenging (but not impossible) to support across multiple devices.
- They don’t always support native application features, like offline mode, location lookup, file system access, camera, and so on.

Native Applications:

The next mobile application medium is the oldest and the most common; it is referred to as native applications, which is actually a misnomer because a mobile web app or mobile web widget can target the native features of the device as well.

These applications actually should be called "platform applications," as they have to be developed and compiled for each mobile platform.

These native or platform applications are built specifically for devices that run the platform in question. The most common of all platforms is Java ME (formerly J2ME).

In theory, a device written as a Java ME MIDlet should work on the vast majority of feature phones sold around the world. The reality is that even an application written as a Java ME MIDlet still requires some adaptation and testing for each device it is deployed on.

In the smartphone space, the platform SDKs get much more specific. Although many smartphones are also powered by Java, an operating system layer and APIs added to allow developers to more easily offload complex tasks to the API instead of writing methods from scratch. In addition to Java, other smartphone programming languages.

Games:

The final mobile medium is games, the most popular of all media available to mobile devices. Technically games are really just native applications that use the similar platform SDKs to create immersive experiences.

The reason games are relatively easy to port ("relatively" being the key word), is that the bulk of the gaming experience is in the graphics and actually uses very little of the device APIs. The game mechanics are the only thing that needs to adapted to the various platforms. Like in console gaming, there are a great number of mobile game porting shops that can quickly take a game written in one language and port it to another.

Pros:

The pros of game applications are:
- They provide a simple and easy way to create an immersive experience.
- They can be ported to multiple devices relatively easily.

Cons:

The cons of game applications are:
- They can be costly to develop as an original game title.
- They cannot easily be ported to the mobile web.

Mobile Application Media Matrix :

In Mobile Application Media, context is the surroundings in which information is processed, and the application user experience is no different.

Application Context :

Once your application medium is decided upon, it is time to look at the application context, or the appropriate type of application to present to the user in order for the user to process and understand the information presented and complete the goals.

Where the application medium refers mostly to the technical approach of creating an application, the application context deals with the user experience. Applications can be presented in a variety of ways.

Utility Context :

The most basic application context is the utility, or a simple user experience metaphor that is meant to address short, task-based scenarios. Information is meant to be presented in a minimal fashion, often using the least amount of user input as possible.

The goal of the utility is to give users at-a-glance information, therefore offering users a minimal design aesthetic, focusing the design around the content in view, and often using larger type and a sparse layout.

Locale Context:

When creating locale apps, it is important to ensure that the user's present location is always clearly identified, as well as a means of adding data to it.

This could be another location, in the case of finding point-to-point directions, or it could be a keyword query to find people, places, or things nearby.

Use locale applications for location-based applications, applications that might contain a dynamic map, and listing multiple location-based points of interest.

Informative Applications:

The informative application is an application context in which the one and only goal is to provide information, like a news site, an online directory, a marketing site, or even a mobile commerce site.

Immersive Full-Screen Applications:

The final application context is an immersive full-screen application, like a game, a media player, or possibly even a single-screen utility. These applications are meant to consume the user's focus, often doing so by filling the entire screen and leaving no trace of the device user interface to distract the user.

Application Context Matrix :

Tthe application contexts into Table 6-2, comparing and contrasting their benefits to determine what is best for your application.

Table 6-2. Application context matrix

User experience type	Task type	Task duration	Combine with
Utility	At-a-glance	Information recall	Very short
Immersive			
Locale	Location-based	Contextual information	Quick
Immersive			
Informative	Content-based	Seek information	Quick
Locale			
Productivity	Task-based	Content management	Long
Utility			
Immersive	Full screen	Entertainment	Long
Utility, locale			

Q3.What is Mobile Information Architecture ? Explain it with neat diagram.

Overview:

Information Architecture
The elements of user Interface
Site Mapes
Confirm the path by teasing content
Click Stream
Wire Frame
Prototype

Definition of Mobile Information Architecture:

The structural design of shared information environments. The combination of organizations, labeling, search, and navigation systems within websites and intranets.

The art and science of shaping information products and experiences to support usability and find ability.

An emerging discipline and community of practice focused on bringing principles of design and architecture to the digital landscape.

Similar to how mobile technology has many facets, so does information architecture, as it is often used as an umbrella term to describe several unique disciplines, including the following:

Information architecture:

The organization of data within an informational space. In other words, how the user will get to information or perform tasks within a website or application.

Interaction design:

The design of how the user can participate with the information present, either in a direct or indirect way, meaning how the user will interact with the website of application to create a more meaningful experience and accomplish her goals.

Information design:

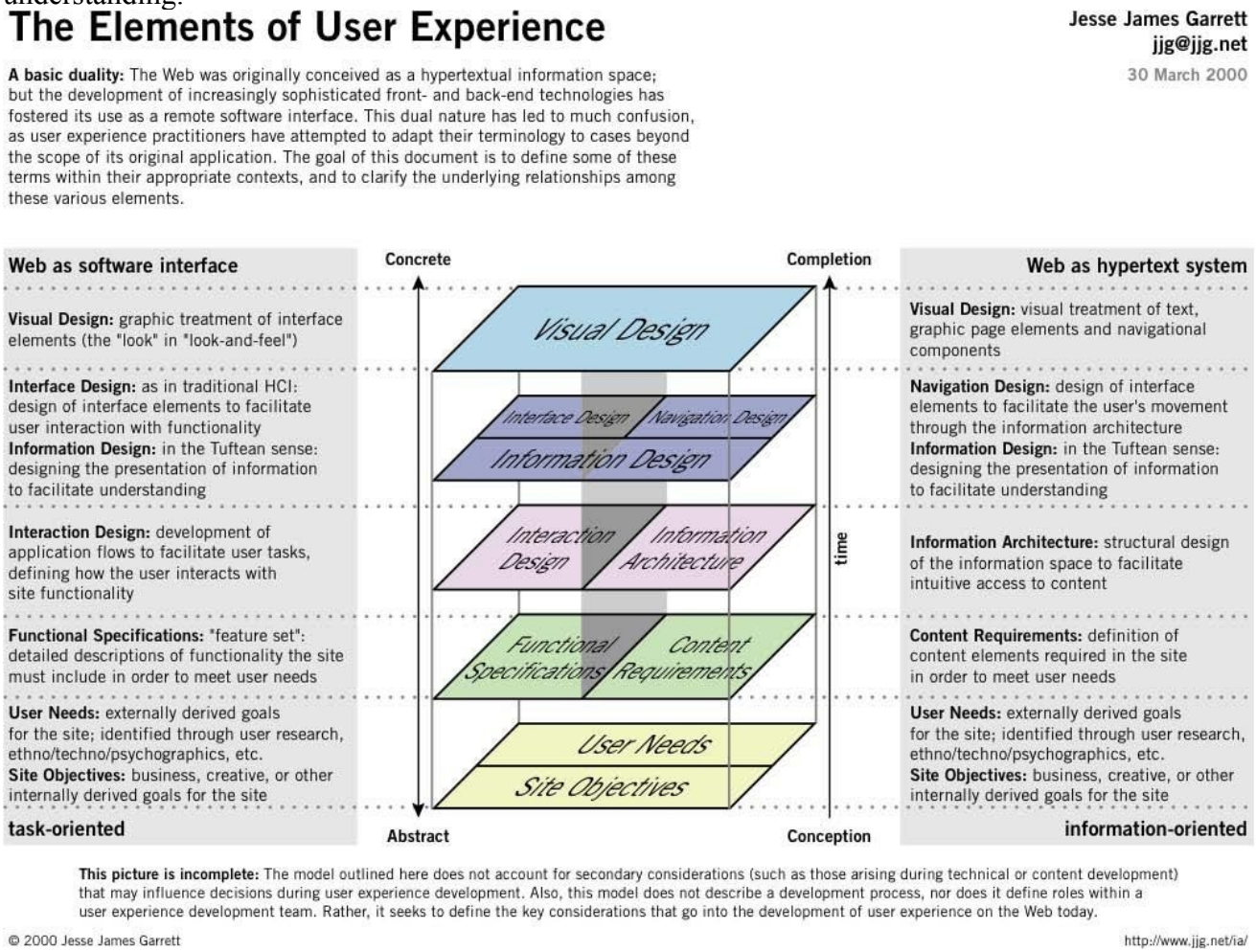
The visual layout of information or how the user will assess meaning and direction given the information presented to him.

Navigation design:

The words used to describe information spaces; the labels or triggers used to tell the users what something is and to establish the expectation of what they will find.

Interface design:

The design of the visual paradigms used to create action or understanding.

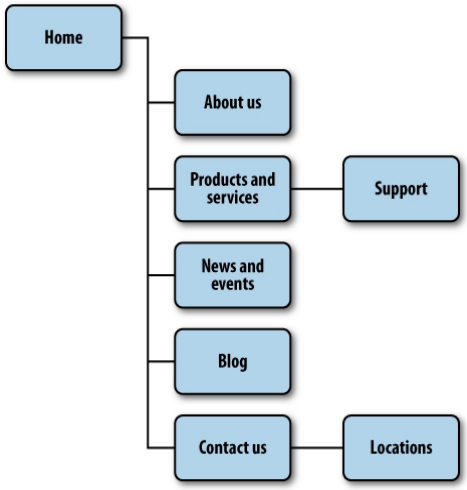


Site Maps:

The first deliverable we use to define mobile information architecture is the site map. Site maps are a classic information architecture deliverable.

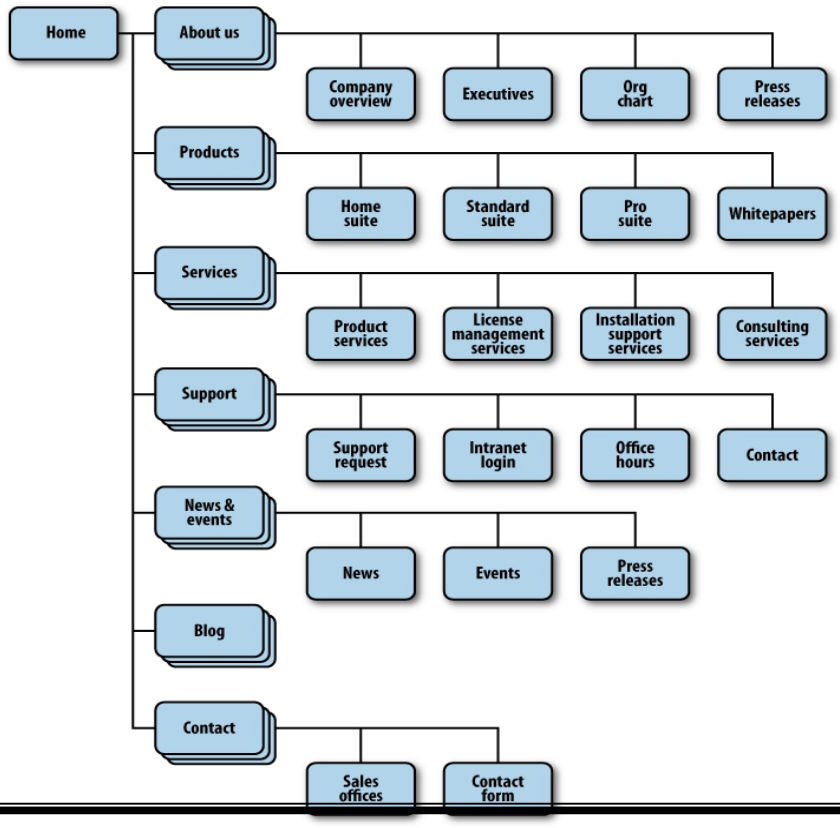
They visually represent the relationship of content to other content and provide a map for how the user will travel through the informational space, as shown in Figure 7-4.

Mobile site maps aren't that dissimilar from site maps we might use on the Web. But there are a few tips specific to mobile that we want to consider



Limit opportunities for mistakes:

Poorly designed mobile information architecture that too closely mimics its desktop cousin; it was not designed with the mobile user in mind.



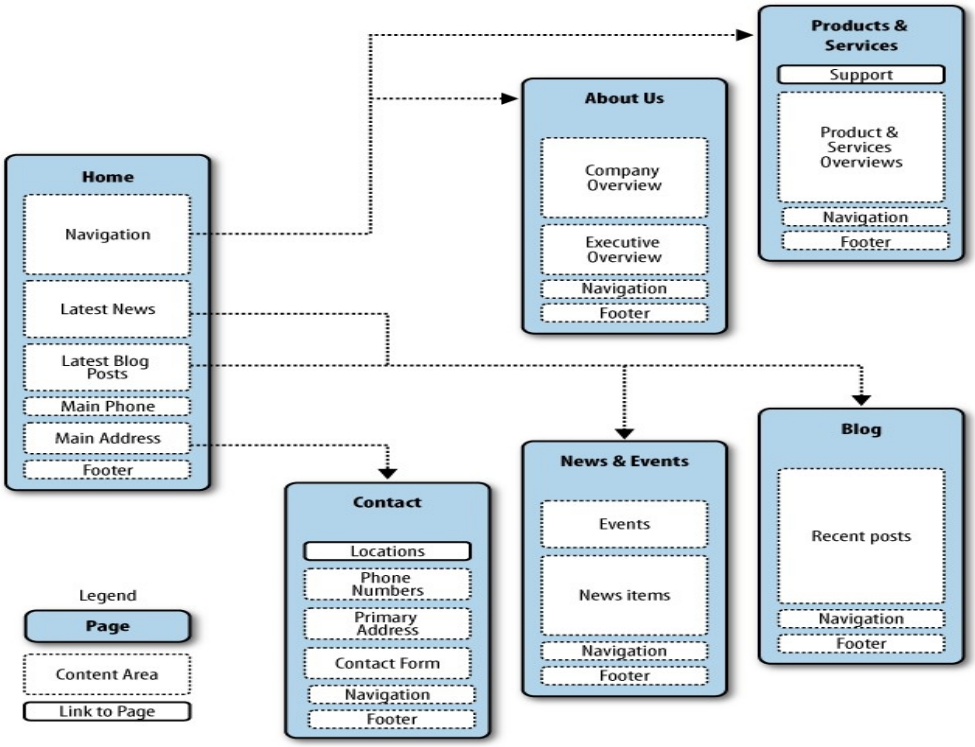
But in mobile, we cannot make this assumption. In the mobile context, tasks are short and users have limited time to perform them.

And with mobile websites, we can't assume that the users have access to a reliable broadband connection that allows them to quickly go back to the previous page.

Confirm the path by teasing content:

After the users have selected a path, it isn't always clear whether they are getting to where they need to be. Information-heavy sites and applications often employ nested or drill-down architectures, forcing the user to select category after category to get to their target.

In Figure 7-6, you can see in a constrained screen that teasing the first few items of the page provides the user with a much more intuitive interface, immediately indicating what type of content the user can



expect.

Clickstreams:

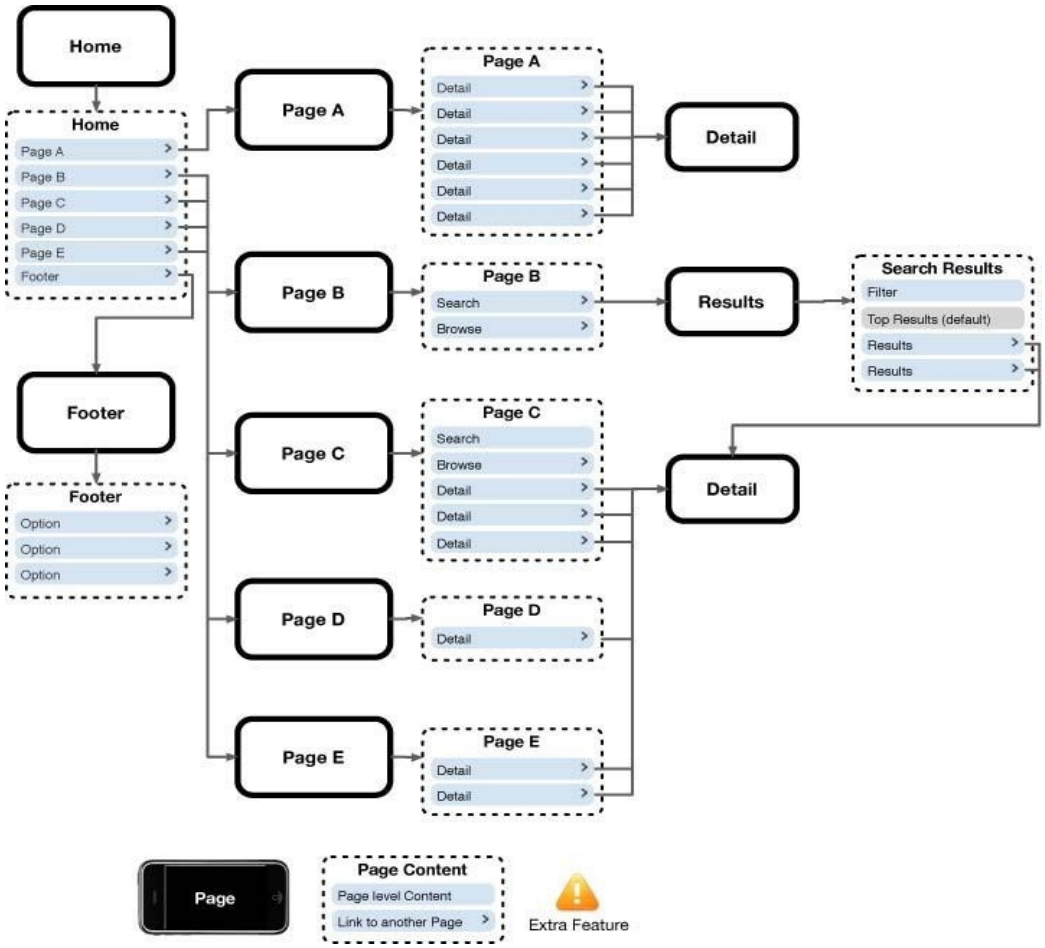
Clickstream is a term used for showing the behavior on websites, displaying the order in which users travel through a site's information architecture, usually based on data gathered from server logs.

Clickstreams are usually historical, used to see the flaws in your information architecture, typically using heat-mapping or simple percentages to show where your users are going.

However, information architecture in mobile is more like software than it is the Web, meaning that create clickstreams in the beginning, not the end.

This maps the ideal path the user will take to perform common tasks. Being able to visually lay out the path users will take gives you a holistic or bird's-eye view of your mobile information architecture, just as a road map does.

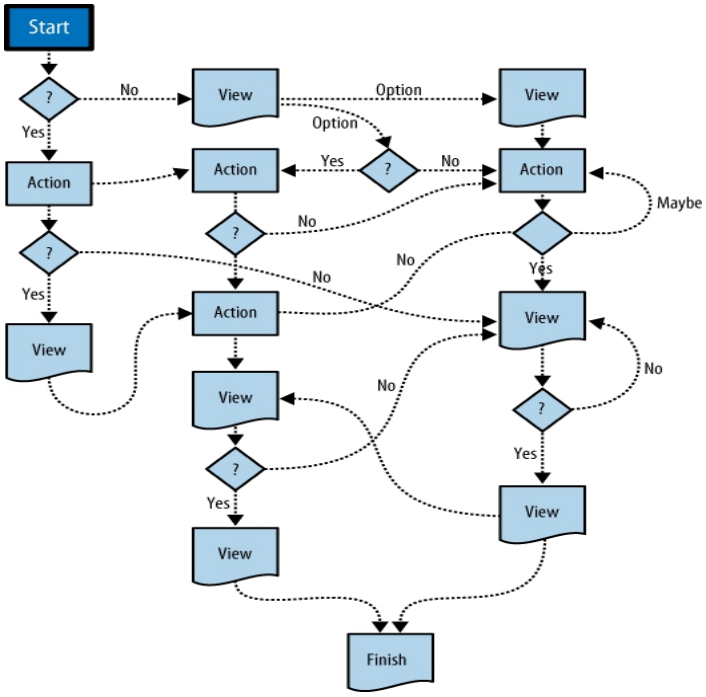
When you can see all the paths next to each other and take a step back, you start to see shortcuts and how you can get users to their goalfaster or easier, as shown in below diagram.



The esoteric diagram shown in Diagram, which is made up of boxes and diamonds that look more like circuit board diagrams than an information architecture.

The information architecture deliverables from the perspective of the user, using the same metaphors it will use to make a way through information architecture— in this case, either a screen or page metaphor.

A good architect's job is to create a map of user goals, not map out every technical contingency or edge case.

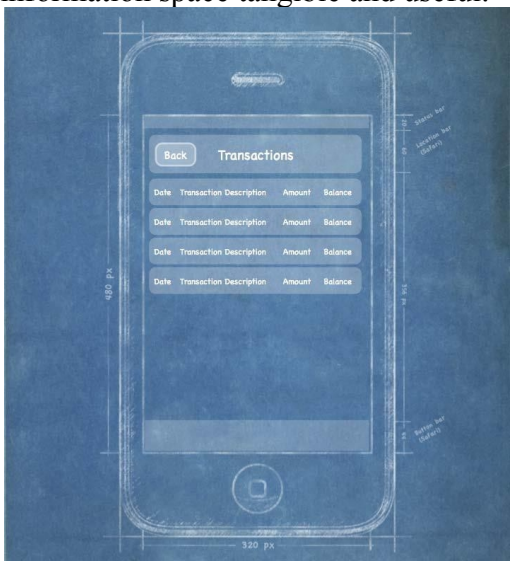


Wireframes:

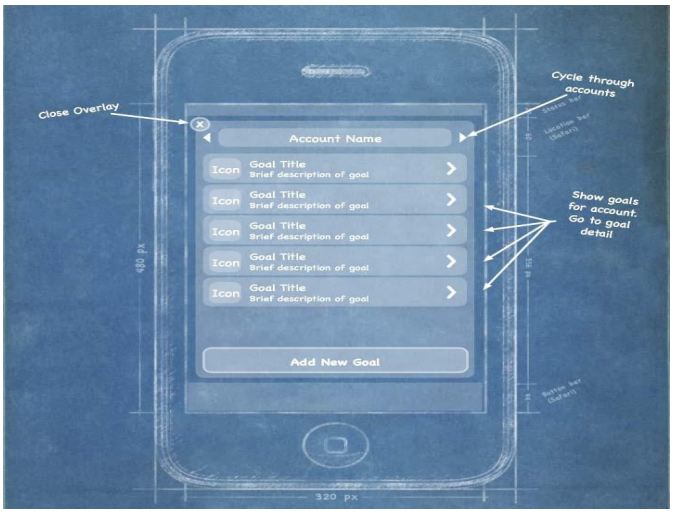
The next information architecture tool at our disposal is wireframes. *Wireframes* are a way to lay out information on the page, also referred to as *information design*.

Site maps show how our content is organized in our informational space; wireframes show how the user will directly interact with it.

Wireframes are like the peanut butter to the site map jelly in our information architecture sandwich. It's the stuff that sticks. Wireframes like the one in diagram serve to make our information space tangible and useful.



These interactions can include copious amounts of annotation, describing each content area in as much length as you can fit in the margins of the page, as shown in diagram.



Prototyping:

As mentioned before, wireframes lack the capability to communicate more complex, often in-place, interactions of mobile experiences. This is where prototypes come in.

Prototypes might sound like a scary (or costly) step in the process. Some view them as redundant or too time-consuming, preferring to jump in and start coding things.

In wireframes each product built out some sort of prototype has saved both time and money.

Paper prototypes:

The most basic level we have is paper prototyping: taking our printed-out wireframes or even drawings of our interface, like the one shown in Figure 7-11, and putting them in front of people.

Context prototype:

The next step is creating a context prototype (diagram). Take a higher-end device that enables you to load full-screen images on it. Take your wireframes or sketches and load them onto the device, sized to fill the device screen.

HTML prototypes:

The third step is creating a lightweight, semifunctional static prototype using XHTML, CSS, and JavaScript, if available.

This is a prototype that you can actually load onto a device and produce the nearest experience to the final product, but with static dummy content and data (diagram). It takes a little extra time, but it is worth the effort.With a static XHTML prototype, you use all the device metaphors of navigation

Q4:What is Mobile 2.0 ? Explain about Mobile 2.0 in detail.

Overview:

Mobile 2.0: The Convergence of the Web and Mobile:

The Mobile Web Browser As the Next Killer App:

Mobile Web Applications Are the Future:

JavaScript Is the Next Frontier:

Rich interactions kill battery life:

Mobile Widgets Are the Next Big Thing

Carrier Is the New "C" Word

We Are Creators, Not Consumers:

Mobile 2.0: The Convergence of the Web and Mobile:

Mobile 2.0 is the Web. At this point, the mobile web has always been viewed as a second-class citizen within the mobile ecosystem, for many reasons, as discussed later.

Mobile is already a medium, but the consensus is that by leveraging the power of the Web, integrating web services into the mobile medium is the future of mobile development.

When the iPhone exploded onto the scene, it increased the usage of the mobile web by its users to levels never seen before. The spur of new mobile web apps created just for the iPhone doubled the number of mobile websites available in under a year.

If Web 2.0 taught us that the Web is the platform, then Mobile 2.0 tells us that mobile will be the primary context in which we leverage the Web in the future.

The Mobile Web Browser As the Next Killer App:

If the future of mobile is the Web, then it only makes sense that the mobile web browser is the next killer app of mobile. Again, this is something confirmed with WebKit in the iPhone and later in Android. But the single biggest challenge in mobile remains device fragmentation.

The mobile browser enables us to penetrate the problem by not having our content locked so specifically
to the device abilities, screen size, and form factor, but device fragmentation still causes old, outdated browsers to remain in the market long after they should be put out to pasture.

What appears to be solving browser fragmentation is actually the iPhone. The Mobile Safari browser included with the iPhone provided such an excellent web experience on a mobile device that it drove use of the mobile web to huge levels, which means big profits for the operators.

This also means that the mobile web is no longer a secondclass citizen. In the post-iPhone market, all new devices are judged by the quality of their mobile web browser. Operators know it and therefore are demanding better browsers from device makers and browser makers.

Mobile Web Applications Are the Future:

Creating mobile web applications instead of mobile software applications has remained an area of significant motivation and interest. The mobile community is looking at the Web 2.0 revolution for inspiration, being able to create products and get them to market quickly and at little cost.

Developers have been keen for years to shift away from the costly mobile applications that are difficult to publish through the mobile service provider, require massive testing cycles and costly porting to multiple devices, and can easily miss the mark with users after loads of money have been dumped into them.

The iPhone App Store and the other mobile device marketplaces have made it far easier to publish and sell, but developers still have to face difficult approval processes, dealing with operator and device maker terms and porting challenges.

Mobile software has two fundamental problems that mobile web applications solve. The first is forcing users through a single marketplace. We know from years of this model that an app sold through a marketplace can earn huge profits if promoted correctly.

Being *promoted correctly* is the key phrase. What gets promoted and why is a nebulous process with no guarantees. One thing is certainly true: the companies that know how to work the system are the ones that get the big prizes, making it increasingly hard for the small developer to see any kind of success.

But the mobile web provides any size of developer with the ability to promote and distribute their app on their terms, building a relationship directly with their customers and not by proxy.

The second problem is the ability to update your application. It is certainly possible on modern marketplaces like the App Store, but we are still years from that being the norm.

Mobile web apps enable you to make sure that you never ship a broken app, or if your app breaks in the future due to a new device, to be able to fix it the same day the device hits the street. This flexibility isn't possible in the downloaded app market.

JavaScript Is the Next Frontier:

It is JavaScript, a web technology that has largely been ignored with most mobile web browsers. Ajax is great, but just being able to do a little show/hide or change a style after you click or touch it goes a long way toward improving the user experience.

This is probably where mobile web browsers fall behind desktop browsers the most. Because

they both support XHTML and CSS relatively well, JavaScript has been a nogo in mobile for years. In order for mobile web apps to rival native applications, you have to support some JavaScript.

Modern mobile browsers have made much progress over the last few years, but there is still plenty of work to be done. For example, accessing the device capabilities like the phone book or filesystem with JavaScript doesn't work in a consistent way. These problems still need to be solved in order to truly reap the benefits of the Web.

Rich interactions kill battery life:

JavaScript and Ajax have been ignored because using an Ajax-based web application on your phone can drain your battery at a rate of four to five times than normal power consumption. The number of reasons for why this happens from mobile hardware?

- JavaScript consumes more processor power and therefore more battery life.

- Ajax apps fetch more data from the network, meaning more use of the radio and more battery life.

Unless you are in the habit of carrying around a bunch of extra batteries, expect to charge your phone every hour or two as a penalty for using the modern mobile web.

Apple and the open source WebKit browser have made huge strides by releasing a JavaScript engine that is incredibly efficient on mobile devices, though the other big mobile browser technologies aren't far behind. This problem is going away quickly as the mobile browsers get better, batteries improve efficiency, and devices get more powerful.

Mobile Widgets Are the Next Big Thing

At many Mobile 2.0 events a lot of buzz about mobile widgets, though no one can tell how mobile widgets would define a mobile widget, or how they are different from mobile web apps. The consensus seems to be that the solution for the challenges with the mobile web is to create a series of "small webs" targeted at a specific user or task.

Carrier Is the New "C" Word

A strong tendency over the years for people in the mobile industry to avoid uttering the word "carrier." Even the more European equivalent term "operator" seems to be on the decline. An example of how much carriers are hated:

It is clear that one of the key drivers of Mobile 2.0 and the focus on the mobile web is to find a way to build a business that doesn't rely on carrier control.

We Are Creators, Not Consumers:

The final principle of Mobile 2.0 is recognizing that we are in a new age of consumerism. Yesterday's consumer does not look anything like today's consumer. The people of today's market don't view themselves as consumers, but rather as creators.

Our primary task online is to read, to gain information. During the early days of the Web, it took tools and know-how in order to publish to the Web. But early in the Web 2.0 evolution, we saw a rise in tools that allowed us to publish to the Web easily, giving individuals a voice online, with a massive audience.

This democratization of the Web took many forms that some call "social media," like blogging, social networks, media sharing, microblogging, and lifestreams. Although social media may have many facets, they all share the same goal: to empower normal, everyday people to become creators and publishers of content. It started with the written word, then music, then photos, and more recently video was added.

Entire markets have been created to provide today's consumer with gadgets, software, and web services to record and publish content so that we can share it with our friends and loved ones. At the center of this revolution in publishing is the mobile device.

As networked portable devices become more powerful, allowing us to capture, record, and share content in the moment, we are able to add a new kind of context to content—the likes of which we haven't seen since satellite television.

Tony Fish, coauthor of *Mobile Web 2.0* (futuretext), says: When everyone has the tools to create content, in addition to zero-cost publishing, we do not consume content, we create it.*

It probably is no coincidence that Web 2.0 occurred around the same time that laptop computers became affordable for the average person, making the Web a more personal medium.With Mobile 2.0, the personal relevance of the content matches how personal the device is and how personally it applies to our everyday situations or our context.

Q5: Explain about the Elements of Mobile Design:

Starting with the context and layering in visual elements or laying out content to achieve the design goal.

Overview:

Context
Message
Layout
Different Layouts for different devices
Color
Color palettes
Typography
Graphics
Iconography
Photos and Images

The context is core to the mobile experience. As the designer, it is a job to make sure that the user can figure out how to address context using your app. Make sure you do your homework to answer the following questions:

• Who are the users? What do you know about them? What type of behavior can you assume or predict about the users?

• What is happening? What are the circumstances in which the users will best absorb the content you intend to present?

• When will they interact? Are they at home and have large amounts of time? Are they at work where they have short periods of time? Will they have idle periods of time while waiting for a train, for example?

• Where are the users? Are they in a public space or a private space? Are they inside or outside? Is it day or is it night?

• Why will they use your app? What value will they gain from your content or services in their present situation?

• How are they using their mobile device? Is it held in their hand or in their pocket?

How are they holding it? Open or closed? Portrait or landscape?

Messages:

Another design element is your message, or what you are trying to say about your site or application visually.

Which of the following designs provide a message?

Yahoo!:

Yahoo! sort of delivers a message. This app provides a clean interface, putting a focus on search and location, using color to separate it from the news content. Words you might use to describe the message are crisp, clean, and sharp.

ESPN:

The ESPN site clearly is missing a message. It is heavily text-based, trying to put a lot of content above the fold, but doesn't exactly deliver a message of any kind. If you took out the ESPN logo, you likely would have indifferent expectations of this site; it could be about anything, as the design doesn't help set expectations for the user in any way. Words you might use to describe the message: bold, cluttered, and content-heavy.

Disney:

Disney creates a message with its design. It gives you a lot to look at—probably too much—but it clearly tries to say that the company is about characters for a younger audience. Words might use to describe the message: bold, busy, and disorienting.

Wikipedia:

The Wikipedia design clearly establishes a message. With a prominent search and text-heavy layout featuring an article, what you are getting with this design. Words you might use to describe the message: clean, minimal, and text-heavy.

Amazon:

Amazon sort of creates a message. Although there are some wasted opportunities above the fold with the odd ad placement, that it is mostly about products (which is improved even more if you scroll down). Words you might use to describe the message: minimal but messy, product-heavy, and disorienting.

The words you might use to describe these designs might be completely different than mine—thus the beauty and the curse of visual design.

Layout:

Layout is an important design element, because it is how the user will visually process the page, but the structural and visual components of layout often get merged together, creating confusion and making the design more difficult to produce.

Why define the layout before the mobile design?
Design is just too subjective of an issue. If one creating a design for anyone but yourself, chances are good that there will be multiple loosely-based-on-experience opinions that will be offered and debated.

Different layouts for different devices:

The second part of layout design is how to visually represent content. In mobile design, the primary content element is navigation.

To design a site or app, one need to provide users with methods of performing tasks, navigating to other pages, or reading and interacting with content. This can vary, depending on the devices.

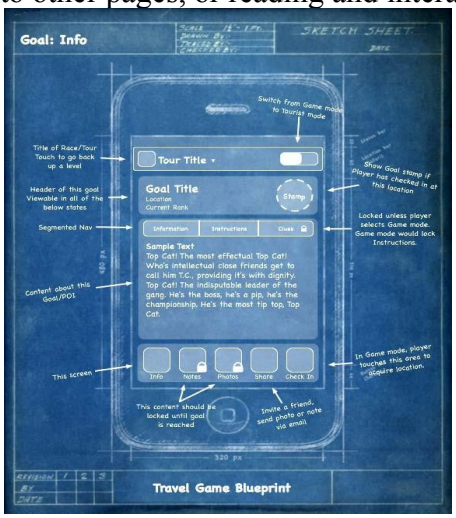


Figure 8-7. Using a low-fidelity wireframe to define the layout design element before visual design

There are two distinct types of navigation layouts for mobile devices: touch and scroll. Therefore, navigation can be anywhere on the screen.

But we tend to see most of the primary actions or navigation areas living at the bottom of the

screen and secondary actions living at the top of the screen, with the area in between serving as the content area, like what is shown in fig 8.8

Figure 8-8. iPhone HIG, showing the layout dimensions of Safari on the iPhone

This is the opposite of the scroll navigation type, where the device's D-pad is used to go left, right, up, or down. When designing for this type of device, the primary and often the secondary actions should live at the top of the screen.

This is so the user doesn't have to press down dozens of times to get to the important stuff. In Figure 8-9, you can actually see by the bold outline that the first item selected on the screen is the link around the logo.

When dealing with scroll navigation, to make the choice of whether to display navigation horizontally or vertically.

Visually, horizontally makes a bit more sense, but that it forces the user to awkwardly move left and right, it can quickly become a bit cumbersome for the user to deal with.

There is no right or wrong way to do it, but my advice is just to try and keep it as simple as

possible.

Figure 8-9. Example layout of a scroll-based application, where the user had to press the D-pad past each link to scroll the page

Fixed versus fluid:

Another layout consideration is how your design will scale as the device orientation changes, for example if the device is rotated from portrait mode to landscape and vice versa.

This is typically described as either being fixed (a set number of pixels wide), or fluid (having the ability to scale to the full width of the screen regardless of the device orientation).

Orientation switching has become commonplace in mobile devices, and your design should always provide the user with a means to scale the interface to take full advantage of screen real estate.

Color:

The fifth design element, color, is hard to talk about in a black-and-white book. Maybe it is fitting, because it wasn't that long ago that mobile screens were available only in black and white (well, technically, it was black on a green screen).

These days, we have nearly the entire spectrum of colors to choose from for mobile designs.

The most common obstacle you encounter when dealing with color is mobile screens, which come in a number of different color or bit depths, meaning the number of bits (binary digits) used to represent the color of a single pixel in a bitmapped image.

When complex designs are displayed on different mobile devices, the limited color depth on one device can cause banding, or unwanted posterization in the image.

For an example of posterization, the technical term for when the gradation of tone is replaced with regions of fewer tones, see in Figure 8-10 how dramatically the color depth can affect the quality of a photo or gradient, producing banding in several parts

Figure 8-10. An example of different levels of posterization that can occur across multiple device colorDepths

The psychology of color:

People respond to different colors differently. It is fairly well known that different colors produce different emotions in people, but surprisingly few talk about it outside of art school.

Thinking about the emotions that colors evoke in people is an important aspect of mobile design, which is such a personal medium that tends to be used in personal ways. Using the right colors can be useful for delivering the right message and setting expectations.

For the purposes of reference, Table 8-2 provides some of the characteristics of various colors that naturally evoke certain emotions in people.

Color Represents

White Light, reverence, purity, truth, snow, peace, innocence, cleanliness, simplicity, security, humility, sterility, winter, coldness, surrender, fearfulness, lack of imagination, air, death (in Eastern cultures), life, marriage (in Western cultures), hope, bland

Black Absence, modernity, power, sophistication, formality, elegance, wealth, mystery, style, e Evil, death , fear, seriousness, conventionality, rebellion, anarchism, unity, sorrow, professionalism

Gray Elegance, humility, respect, reverence, stability, subtlety, wisdom, old age, pessimism, boredom, decay, decrepitude, dullness, pollution, urban sprawl, strong emotions, balance, neutrality, mourning, formality

Yellow Sunlight, joy, happiness, earth, optimism, intelligence, idealism, wealth (gold), summer, hope, air, liberalism, cowardice, illness (quarantine), fear, hazards, dishonesty, avarice, weakness, greed, decay or aging, femininity, gladness, sociability, friendship

Green Intelligence, nature, spring, fertility, youth, environment, wealth, money (U.S.), good luck, vigor, generosity, go, grass, aggression, coldness, jealousy, disgrace (China), illness, greed, drug culture, corruption (North Africa), life eternal, air, earth (classical element), sincerity, renewal, natural abundance, growth

Blue Seas, men, productiveness, interiors, skies, peace, unity, harmony, tranquility, calmness, trust, coolness, confidence, conservatism, water, ice, loyalty, dependability, cleanliness, technology, winter, depression, coldness, idealism, air, wisdom, royalty, nobility, Earth (planet), strength, steadfastness, light, friendliness, peace, truthfulness, love, liberalism (U.S. politics), and conservatism (UK, Canadian, and European politics)

Violet Nobility, envy, sensuality, spirituality, creativity, wealth, royalty, ceremony, mystery, wisdom, enlightenment, arrogance, flamboyance, gaudiness, mourning, exaggeration, profanity, bisexuality, confusion, pride

Red Passion, strength, energy, fire, sex, love, romance, excitement, speed, heat, arrogance, ambition, leadership, masculinity, power, danger, gaudiness, blood, war, anger, revolution, radicalism, aggression, respect, martyrs, conservatism (U.S. politics), Liberalism (Canadian politics), wealth (China), and marriage (India)

Orange Energy, enthusiasm, balance, happiness, heat, fire, flamboyance, playfulness, aggression, arrogance, gaudiness, overemotion, warning, danger, autumn, desire

Pink Spring, gratitude, appreciation, admiration, sympathy, socialism, femininity, health, love, romance, marriage, joy, flirtatiousness, innocence and child-like qualities,

Brown Calm, boldness, depth, nature, richness, rustic things, stability, tradition, anachronism, boorishness, dirt, dullness, heaviness, poverty, roughness, earth

Color palettes:

Defining color palettes can be useful for maintaining a consistent use of color in your mobile design. Color palettes typically consist of a predefined number of colors to use throughout the design.

Selecting what colors to use varies from designer to designer, each having different techniques and strategies for deciding on the colors.

Sequential :

There are primary, secondary, and tertiary colors. Often the primary color is reserved as the "brand" color or the color that most closely resembles the brand's meaning. The secondary and tertiary colors are often complementary colors selected using a color wheel.

Adaptive:

An adaptive palette is one in which you leverage the most common colors present in a supporting graphic or image. When creating a design that is meant to look native on the device, an adaptive palette to make sure that colors are consistent with the target mobile platform.

Inspired:

This is a design that is created from the great pieces of design one might see online, or offline, in which a picture of the design might inspire you.

This could be anything from an old poster in an alley, a business card, or some packaging. When I sit down with a new design, I thumb through some of materials to create an inspired palette. Like with the adaptive palette, you actually extract the colors from the source image, though you should never ever use the source material in a design.

Typography:

The sixth element of mobile design is typography, which in the past would bring to mind the famous statement by Henry Ford:

Any customer can have a car painted any color that he wants so long as it is black.

Traditionally in mobile design, you had only one typeface that you could use (Figure 8-12), and that was the device font. The only control over the presentation was the

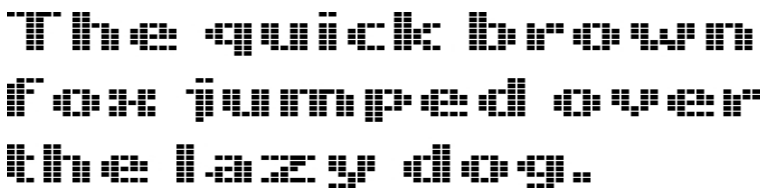


Figure 8-12. What most mobile designers think of when it comes to mobile typography

As devices improved, so did their fonts. Higher-resolution screens allowed for a more robust catalog of fonts than just the device font. First, let's understand how mobile screens work.

Subpixels and pixel density:

There seem to be two basic approaches to how type is rendered on mobile screens:

Using subpixel-based screens or having a greater pixel density or pixels per inch (PPI). A subpixel is the division of each pixel into a red, green, and blue (or RGB) unit at a microscopic level, enabling a greater level of antialiasing for each font character or glyph.

The addition of these RGB subpixels enables the eye to see greater variations of gray, creating sharper antialiasing and crisp text.

In Figure 8-13, you can see three examples of text rendering. The first line shows a simple black and white example, the second shows text with grayscale antialiasing, and the third line shows how text on a subpixel display would render

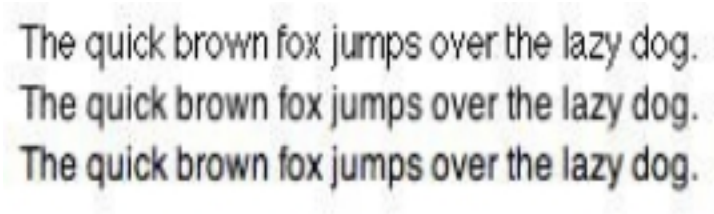


Figure 8-13. Different ways text can render on mobile screens

The Microsoft Windows Mobile platform uses the subpixel technique with its Clear-Type technology, as shown in Figure 8-14.

The second approach is to use a great pixel density, or pixels per inch. We often refer to screens by either their actual physical dimensions ("I have a 15.4-inch laptop screen") or their pixel dimensions, or resolution ("The resolution of my laptop is 1440×900pixels").

The pixel density is determined by dividing the width of the display area in pixels by the width of the display area in inches. So the pixel density for my 15.4-inch laptop would be 110 PPI. In comparison, a 1080p HD television has a PPI of 52.

Graphics:

The final design element is graphics, or the images that are used to establish or aid a visual experience. Graphics can be used to supplement the look and feel, or as content displayed inline with the text.

For example, Ribot's Little Spender application for the iPhone and the S60 platform. The use of graphical icons in the iPhone experience helps to establish a visual language for the user to interact with to quickly categorize entries.

On the S60 application, the wallet photo in the upper-right corner helps communicate the message of the application to the user.

Iconography:

The most common form of graphics used in mobile design is icons. Iconography is useful to communicate ideas and actions to users in a constrained visual space. The challenge is making sure that the meaning of the icon is clear to the user.

For example, some helpful icons that clearly communicate an idea .

Photos and images:

Photos and images are used to add meaning to content, often by showing a visual display of a concept, or to add meaning to a design. Using photos and images isn't as common in mobile design as you might think.

Because images have a defined height and width, they need to be scaled to the appropriate device

size, either by the server, using a content adaptation model, or using the resizing properties of the device.

In the latter approach, this can have a cost in performance. Loading larger images takes longer and therefore costs the user more.

Using graphics to add meaning to a design can be a useful visual, but you can encounter issues regarding how that image will display in a flexible UI—for example, when the device orientation is changed. In Figure 8-19, you can see how the pig graphic is designed to be positioned to the right regardless of the device orientation.

Q6. Give detail description about Mobile Design:

When building mobile experiences, it is impossible to create a great experience without three ingredients: context, information architecture, and visual design. This leads us to the most significant challenge in mobile design: creativity.

Overview:

1. The Mobile Design Tent-Pole
2. Designing for the Best Possible Experience

On computers, there is a strategy called "lowest common denominator": in order to reach the widest possible number of platforms, one create a product that works on the most common architectural components on all platforms (see diagram).

The Mobile Design Tent-Pole

In Hollywood, executives like to use the term “tent-pole” to describe their movies and television shows. The term has dual meanings: one is business, and the other creative.

The business goal of a tent-pole production is to support or prop up the losses from other productions. However, to create a tent-pole production, the creators involved must make an artistic work that they know will appeal to the largest possible audience, providing something for everyone. Tent-pole movies as “blockbusters”; in television, they are known as “anchor” shows.

Trying to reach the widest possible audience poses a problem. Hollywood is learning with great pains that with so many entertainment options and with today’s audience being so hard to reach through traditional advertising channels, tent-pole productions are failing.

As the number of social niches increases, it becomes difficult to satisfy the specific tastes of each social group. What one group finds hysterically funny, several other groups might find offensive. Today, tent-pole productions often come off as bland and half-hearted, failing to appease anyone.

One of the most interesting examples of how the tide turned in entertainment is with the animated films of Disney versus those of Pixar. For years, Disney produced tent pole family fare quite successfully.

Apple’s App Store quickly changed that. You can clearly see that the best-selling games and applications for the iPhone are the ones with the best designs in diagram.

Users look at multiple screenshots (below diagram), read the user reviews, and judge the product based on the quality of its icon and of the screenshots before they buy.

The Apple App Store is proving everyday that mobile design doesn't have to start with tent-pole lowest-common-denominator products—it can instead start with providing the best possible experience and tailoring that experience to the market that wants it.

Designing for the Best Possible Experience

Here is a design solution: design for the best possible experience. Actually, don't just design for it: focus on creating the best possible experience with unwavering passion and commitment. Iterate, tweak, and fine-tune until you get it right. Anything less is simply unacceptable.

Trying to create a mobile design in the context of the device constraints isn't where you start; it is where you should end.

Q7.Give detail description about Mobile Design Tools:

Mobile design requires understanding the design elements and specific tools. The closest thing to a common design tool is Adobe Photoshop, though each framework has a different method of implementing the design into the application.

Overview:

Designing for the Right Device:
Designing for Different Screen Sizes:

Some frameworks provide a complete interface toolkit, allowing designers or developers to simply piece together the interface, while others leave it to the designer to define from scratch.

Table 8-4. Design tools and interface toolkits

Mobile framework	Design tool	Interface toolkits
Java ME	Photoshop, NetBeans JavaFX, Capuchin	
BREW	Photoshop, Flash	BREW UI Toolkit, uiOne, Flash
Flash Lite	Flash	Flash Lite
iPhone	Photoshop, Interface Builder	iPhone SDK
Android	Photoshop, XML-based themes Android SDK	
Palm webOS	Photoshop, HTML, CSS, and JavaScript	Mojo SDK

Designing for the Right Device:

iPhone users consume more mobile content and products than the average mobile user. This platform has an easy-to-learn framework and excellent documentation, for both web and native products, and an excellent display and performance means.

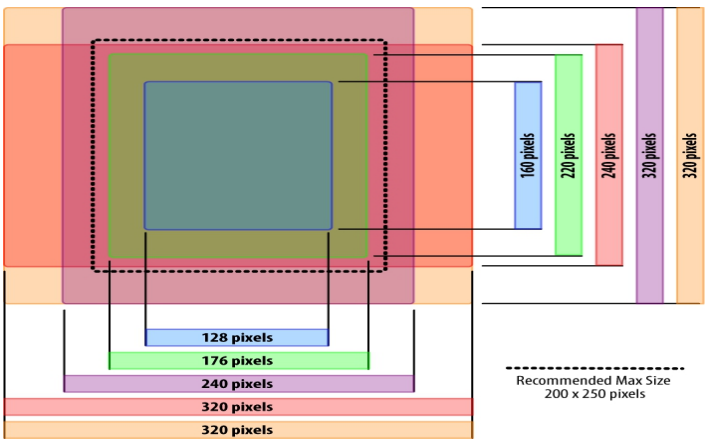
Although iPhone users might not be the majority the ability to create the best possible design and get it in front of those users presents the least expensive product to produce with the lowest risk.

With a successful single device launch, one can start to adapt designs from the best possible experience to the second best possible experience, then the third, and fourth, and so on.

Designing for Different Screen Sizes:

Mobile devices come in all shapes and sizes. Choice is great for consumers, but bad for design. It can be incredibly difficult to create that best possible experience for a plethora of different screen sizes.

The good news is that the vast majority of mobile device screens share the same vertical or portrait orientation, even though they vary greatly in dimension, as shown in Figure 8-20.



There are some devices by default in a horizontal orientation, and many smartphones that can switch between the two orientations, but most people use their mobile devices in portrait mode.

This is a big shift in thinking if you are coming from interactive design, as up to this point, screens have been getting wider, not taller.

In software, tasks flow from left to right. With vertical designs, the goal is to think of your design as a cascade of content from top to bottom (Figure 8-21),similar to a newspaper. The most contextual information lives at the top, and the content consumes the majority of the screen. Any exit points live at the bottom. Mobile is no different.

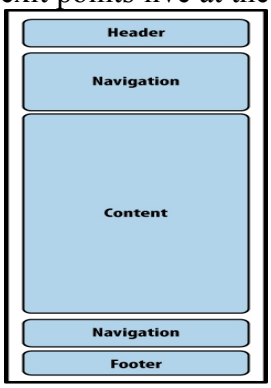


Figure 8-21. The typical flow of information on mobile devices

The greatest challenge to creating a design that works well on multiple screen sizes is filling the width. For content-heavy sites and applications, the width of mobile devices is almost the perfect readability, presenting not too many words per line of text.

The problem is when you have to present a number of tasks or actions. The easiest and most compatible way is to present a stacked list of links or buttons, basically one action per line. It isn't the most effective use of space, but presenting too many actions on the adapt to other devices.

Although most mobile web browsers and device framework APIs enable it in principle, its execution across multiple devices is a little anticlimatic.

Mobile websites usually employ a fixed-width layout for the lowest common denominator, and native applications are often resized for multiple screen sizes during development.

As devices get larger, denser screens, you will see an increase in the use of touch, forcing the size of content to increase to fingertip size—typically 40 pixels wide and 40 pixels tall (Figure 8-22). This actually solves part of the horizontal axis problem, simply by making content larger for larger screens.

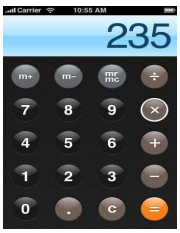


Figure 8-22. The iPhone calculator application uses common fingertip-size controls

1. Explain in detail about drag and drop operations?

OVERVIEW

- √ **Drag and Drop Module**
- √ **Drag and Drop List**
- √ **Drag and Drop Object**
- √ **Drag and Drop Action**
- √ **Drag and Drop Collection**

One of the great innovations that the Macintosh brought to the world in 1984 was Drag and Drop. Influenced by the graphical user interface work on Xerox PARC's Star Information System and subsequent lessons learned from the Apple Lisa, the Macintosh team invented drag and drop as an easy way to move, copy, and delete files on the user's desktop.

Interesting Moments

At first blush, drag and drop seems simple. Just grab an object and drop it somewhere. But, as always, the devil is in the details. There are a number of individual states at which interaction is possible. We call these microstates *interesting moments*:†

- How will users know what is draggable?
- What does it mean to drag and drop an object?
- Where can you drop an object, and where is it not valid to drop an object?
- What visual affordance will be used to indicate draggability?

The Events

There are at least 15 events available for cueing the user during a drag and drop interaction:

Page Load

Before any interaction occurs, you can pre-signify the availability of drag and drop. For example, you could display a tip on the page to indicate draggability.

Mouse Hover

The mouse pointer hovers over an object that is draggable.

Mouse Down

The user holds down the mouse button on the draggable object.

Drag Initiated

After the mouse drag starts (usually some threshold—3 pixels).

Drag Leaves Original Location

After the drag object is pulled from its location or object that contains it.

Drag Re-Enters Original Location

When the object re-enters the original location.

Drag Enters Valid Target

Dragging over a valid drop target.

Drag Exits Valid Target

Dragging back out of a valid drop target.

Drag Enters Specific Invalid Target

Dragging over an invalid drop target.

Drag Is Over No Specific Target

Dragging over neither a valid or invalid target. Do you treat all areas outside of valid targets as invalid?

Drag Hovers Over Valid Target

User pauses over the valid target without dropping the object. This is usually when a spring

loaded drop target can open up. For example, drag over a folder and pause, the folder opens revealing a new area to drag into

Drag Hovers Over Invalid Target

User pauses over an invalid target without dropping the object. Do you care? Will you want additional feedback as to why it is not a valid target?

Drop Accepted
Drop occurs over a valid target and drop has been accepted.
Drop Rejected
Drop occurs over an invalid target and drop has been rejected.
Do you zoom back the dropped object?
Drop on Parent Container
Is the place where the object was dragged from special? Usually this is not the case, but it may carry special meaning in some contexts.

The Actors

During each event you can visually manipulate a number of *actors*. The page elements available include:

- Page (e.g., static messaging on the page)
- Cursor
- Tool Tip
- Drag Object (or some portion of the drag object, e.g., title area of a module)
- Drag Object's Parent Container
- Drop Target

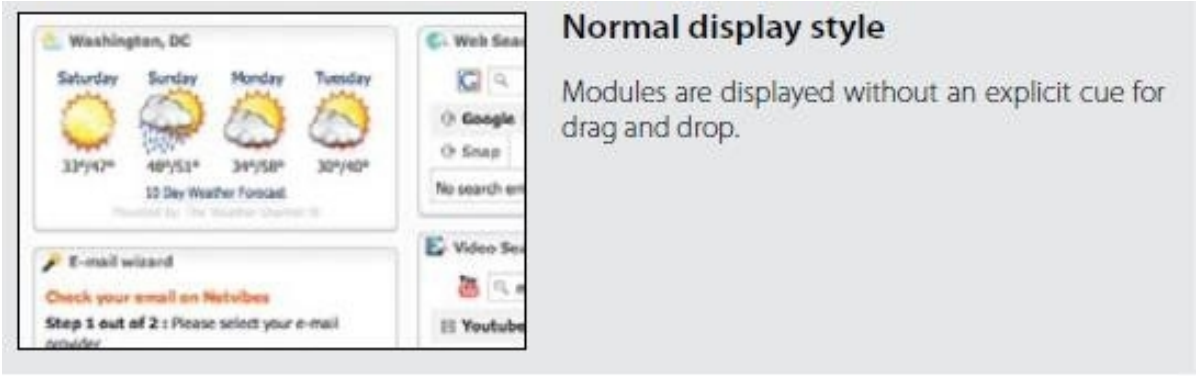
Purpose of Drag and Drop

Drag and drop can be a powerful idiom if used correctly. Specifically it is useful for:

Drag and Drop Module
Rearranging modules on a page.
Drag and Drop List
Rearranging lists.
Drag and Drop Object
Changing relationships between objects.
Drag and Drop Action
Invoking actions on a dropped object.
Drag and Drop Collection
Maintaining collections through drag and drop.

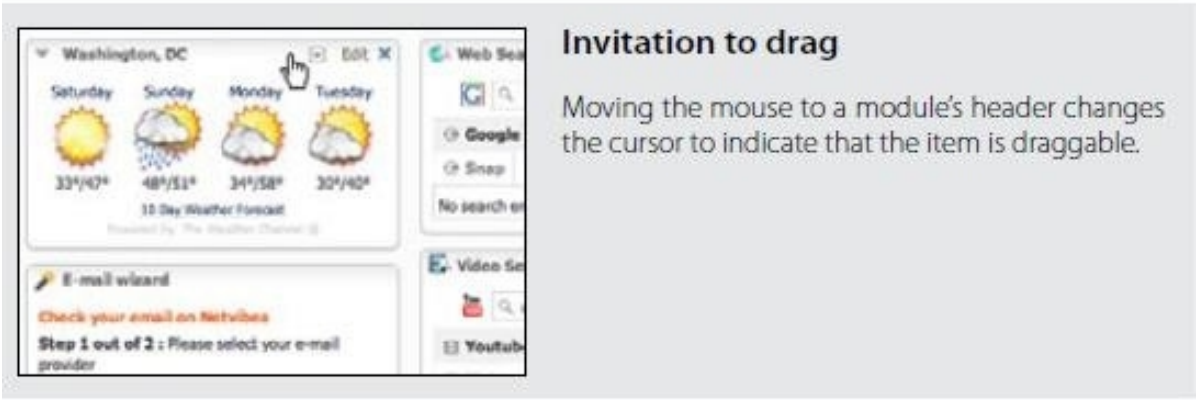
Drag and Drop Module

- One of the most useful purposes of drag and drop is to allow the user to directly place objects where she wants them on the page. A typical pattern is Drag and Drop Modules on a page. Netvibes provides a good example of this interaction pattern (Figure 2-3).



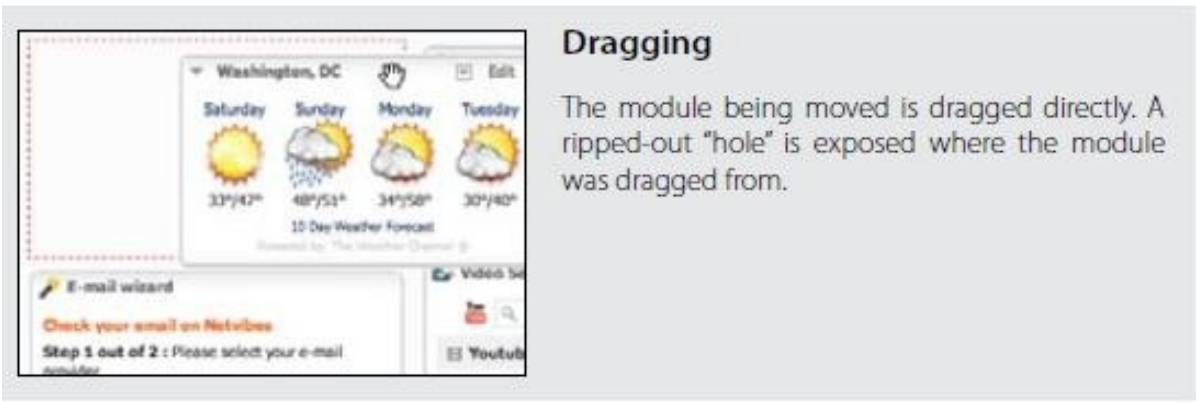
Normal display style

Modules are displayed without an explicit cue for drag and drop.



Invitation to drag

Moving the mouse to a module's header changes the cursor to indicate that the item is draggable.

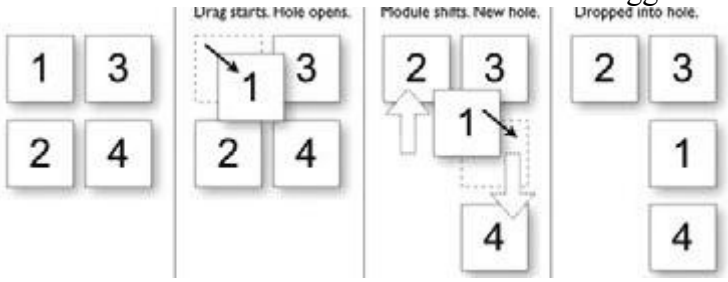


Dragging

The module being moved is dragged directly. A ripped-out "hole" is exposed where the module was dragged from.

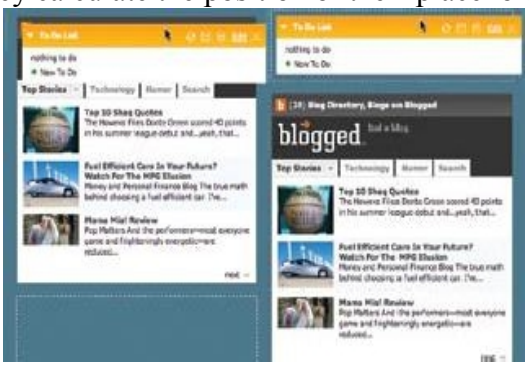
Placeholder target

- Net vibes uses a placeholder (hole with dashed outline) as the drop target.
- The idea (illustrated in Figure 2-5) is to always position a hole in the spot where the drop would occur. When module 1 starts dragging, it gets "ripped" out of the spot.
- In its place is the placeholder target (dashed outline). As 1 gets dragged to the spot between 3 and 4, the placeholder target jumps to fill in this spot as 4 moves out of the way.
- The hole serves as a placeholder and always marks the spot that the dragged module will land when dropped. It also previews what the page will look like (in relation to the other modules) if the drop occurs thereFor module drag and drop, the other modules only slide up or down within a vertical column to make room for the dragged module.



Boundary-based placement.

- Since most sites that use placeholder targeting drag the module in its original size, targeting is determined by the boundaries of the dragged object and the boundaries of the dragged-over object.
- The mouse position is usually ignored because modules are only draggable in the title (a small region). Both Netvibes and iGoogle take the boundary-based approach. But, interestingly, they calculate the position of their placeholders differently.

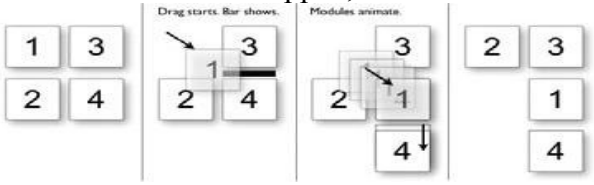


- In contrast, moving the small module below the large module actually requires less drag distance since you only have to get the title bar of the small module below the title bar of the large module

Insertion target

- Placeholder positioning is a common approach, but it is not the only way to indicate drop targeting.
- An alternate approach is to keep the page as stable as possible and only move around an insertion target (usually an insertion bar). A previous version of My Yahoo! Used the insertion bar approach as the dragged module was moved around

- While the module is dragged, the page remains stable. No modules move around. Instead an insertion bar marks where the module will be placed when dropped. This technique is illustrated in Figure 2-14.
- When module 1 is dragged to the position between 3 and 4, an insertion bar is placed there. This indicates that if 1 is dropped, then 4 will slide down to open up the drop spot.



Drag distance

- Dragging the thumbnail around does have other issues. Since the object being dragged is small, it does not intersect a large area. It requires moving the small thumbnail directly to the place it will be dropped.
- With iGoogle, the complete module is dragged. Since the module will always be larger than the thumbnail, it intersects a drop target with much less movement. The result is a shorter drag distance to accomplish a move.

Drag rendering

- How should the dragged object be represented? Should it be rendered with a slight transparency (ghost)? Or should it be shown fully opaque? Should a thumbnail representation be used instead?
- As shown earlier, My Yahoo! uses a small gray rectangle to represent a module (Figure 2-15). Netvibes represents the dragged module in full size as opaque (shown back in Figure 2-3), while iGoogle uses partial transparency (Figure 2-17).
- The transparency (ghosting) effect communicates that the object being dragged is actually a representation of the dragged object. It also keeps more of the page visible, thus giving a clearer picture of the final result of a drop.

Drag and Drop List

Rearranging lists is very similar to rearranging modules on the page but with the added constraint of being in a single dimension (up/down or left/right). **The Drag and Drop** List pattern defines interactions for rearranging items in a list. 37 Signal's Backpackit allows to-do items to be rearranged with Drag and Drop List (Figure 2-18).

Considerations

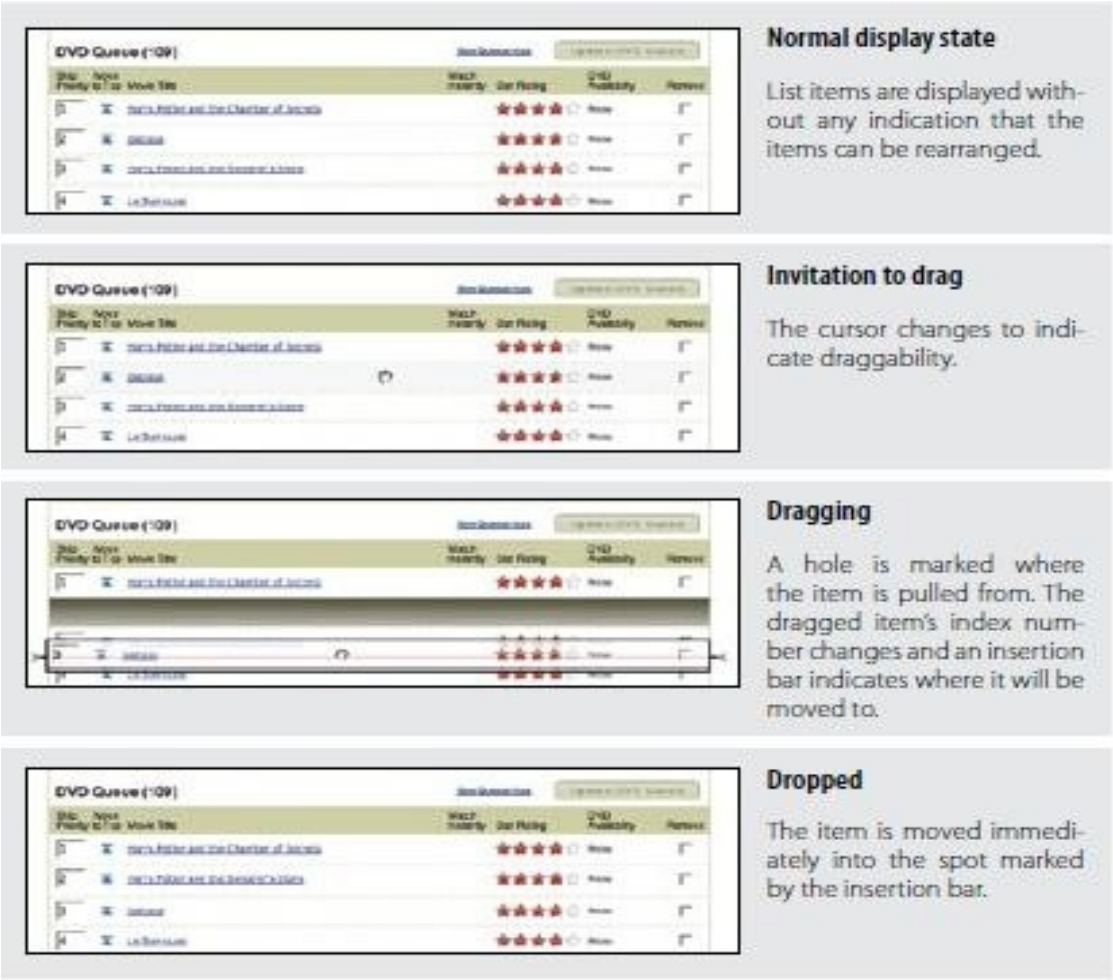
- Backpacked takes a real-time approach to dragging items. Since the list is constrained, this is a natural approach to moving objects around in a list. You immediately see the result of the drag.

Placeholder target

- This is essentially the same placeholder target approach we discussed earlier for dragging and dropping modules.
- The difference is that when moving an item in a list, we are constrained to a single dimension.
- Less feedback is needed. Instead of a "ripped-out" area (represented earlier with a dotted rectangle), a simple hole can be exposed where the object will be placed when dropped.

Insertion target

- Just as with Drag and Drop Modules, placeholder targeting is not the only game in town. You can also use an insertion bar within a list to indicate where a dropped item will land.
- Netflix uses an *insertion target* when movies are dragged to a new location in a user' movie queue (Figure 2-20).



Non–drag and drop alternative

- Besides drag and drop, the Netflix queue actually supports two other ways to move objects around:
- Edit the row number and then p • ress the "Update DVD Queue" button.
- Click the "Move to Top" icon to pop a movie to the top.
- Modifying the row number is straightforward.
- It's a way to rearrange items without drag and drop. The "Move to Top" button is a little more direct and fairly straightforward (if the user really understands that this icon means "move to top").
- Drag and drop is the least discoverable of the three, but it is the most direct, visual way to rearrange the list. Since rearranging the queue is central to the Netflix customer's satisfaction, it is appropriate to allow multiple ways to do so.

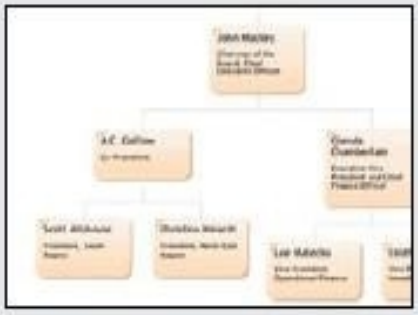
Drag lens

- Drag and drop works well when a list is short or the items are all visible on the page. But when the list is long, drag and drop becomes painful. Providing alternative ways to rearrange is one way to get around this issue. Another is to provide a *drag lens* while dragging.

- A drag lens provides a view into a different part of the list that can serve as a shortcut target. It could be a fixed area that is always visible, or it could be a miniature view of the list that provides more rows for targeting

Drag and Drop Object

- Another common use for drag and drop is to change relationships between objects. This is appropriate when the relationships can be represented visually.
- Drag and drop as a means of visually manipulating relationships is a powerful tool. Cogmap is a wiki for organizational charts. Drag and Drop Object is used to rearrange members of the organization (Figure 2-23).

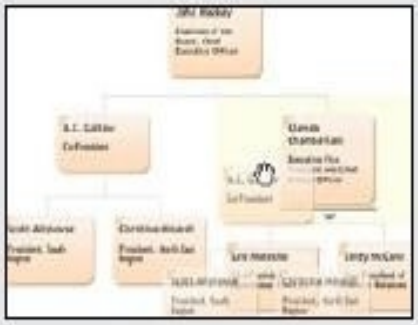


Normal display state

An organizational chart visually represents relationships.

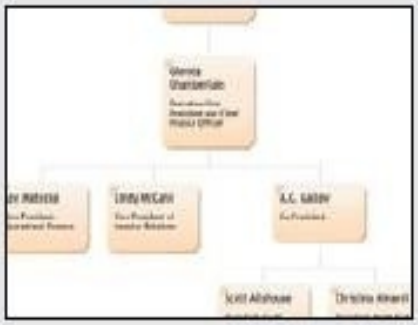
Invitation to drag

When the mouse hovers over a member of the organization, the cursor changes to show dragga-bility. In addition, the texture in the top-left corner changes to represent a dimpled surface. This hints at draggability.



Dragging

An insertion bar is used to indicate where the member will be inserted when dropped.



Dropped

When the dragged member is dropped, the chart is rearranged to accommodate the new location.

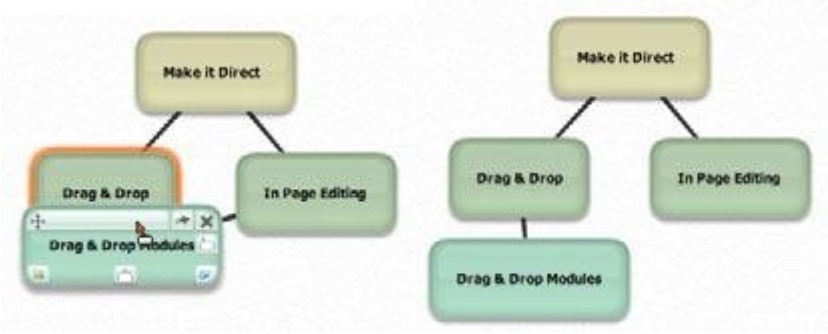
Considerations

- When object relationships can be clearly represented visually, drag and drop is a naturalchoice to make these type of changes. Cogmap uses the target insertion approach. This

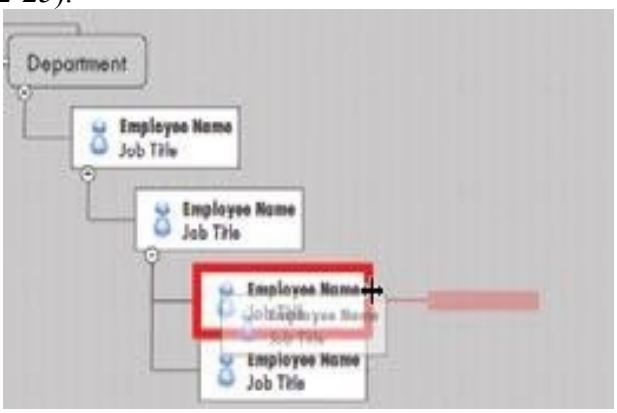
allows the dragging to be nondistracting, since the chart does not have to be disturbed during targeting.

Drag feedback: Highlighting

- Bubbl.us, an online mind-mapping tool, simply highlights the node that will be the new parent (Figure 2-24).

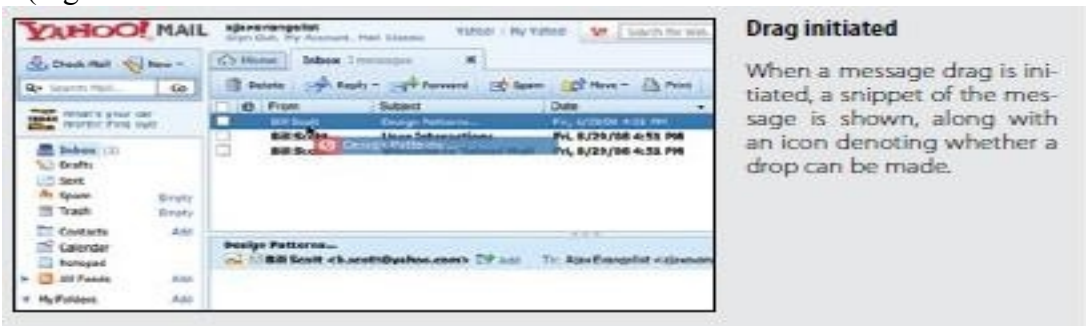


- In both cases, immediate preview is avoided since it is difficult to render the relationships in real time without becoming unnecessarily distracting.
- Looking outside the world of the Web, the desktop application Mind Manager also uses highlighting to indicate the parent in which insertion will occur. In addition, it provides insertion targeting to give a preview of where the employee will be positioned once dropped (Figure 2-25).



Drag feedback: Dragged object versus drop target

- As we mentioned at the beginning of this chapter, one of the first serious uses for drag and drop was in the Oddpost web mail application. Oddpost was eventually acquired by Yahoo! and is now the Yahoo! Mail application.
- Yahoo! Mail uses drag and drop objects for organizing email messages into folders (Figure



- Instead of signaling that a drop is valid or invalid by changing the visual appearance of the area dragged over, Yahoo! Mail shows validity through the dragged object. When a drop will be invalid (Figure 2-27, left):
  - The dragged object's icon becomes a red invalid sign.
  - If over an invalid folder, the folder is highlighted as well.

  When a drop will be valid (Figure 2-27, right):
  - The dragged object's icon changes to a green checkmark.
  - The drop target highlights.

Drag and Drop Action

- Drag and drop is also useful for invoking an action or actions on a dropped object. The Drag and Drop Action is a common pattern. Its most familiar example is dropping an item in the trash to perform the delete action.
- Normally uploading files to a web application includes pressing the upload button and browsing for a photo. This process is repeated for each photo.
- When Yahoo! Photos was relaunched in 2006, it included a drag and drop upload feature. It allowed the user to drag photos directly into the upload page. The drop signified the upload action (Figure 2-30).

Normal display state

"Add Photos" allows either browsing for photos or simply dragging and dropping them into the target zone below.

Invitation to drag

The invitation is clear. By using the drop target area as an advertisement for the drag feature, the process is discoverable (as well as natural).

Dropped

Photos dropped are collected into an upload area. Pressing "Start Upload" starts the uploading process.

Completed

All items are marked Complete when finished.

***

Considerations

- This is not a trivial implementation. But it does clearly illustrate the benefit of drag and drop for operating on a set of files.

- The traditional model requires each photo to be selected individually for upload. Drag and drop frees you to use whatever browsing method is available on your system and then drop those photos for upload.

Anti-pattern: Artificial Visual Construct

- Unfortunately, drag and drop can sometimes drive the design of an interface instead of being an extension of a natural interface.
- These interactions are almost always doomed, as they are the tail wagging the proverbial dog. Rating movies, books, and music is a common feature found on many sites. But what happens if you try to use drag and drop to rate movies?
- In Figure 2-31 you can rate movies by dragging them into three buckets: "Loved It", m"Haven't Seen It", or "Loathed It".

Too much effort

- Requires too much user effort for a simple task. The user needs to employ mouse gymnastics to simply rate a movie. Drag and drop involves these discrete steps: target, then drag, then target, and then drop. The user has to carefully pick the movie, drag it to the right bucket, and release.

Too much space

- Requires a lot of visual space on the page to support the idiom. Is it worth this amount of screen real estate?
- Direct rating systems (thumbs up/down, star ratings, etc.) are a much simpler way to rate a movie than using an Artificial Visual Construct. A set of stars is an intuitive, compact, and simple way to rate a movie (Figure 2-32).

Natural Visual Construct

- Another example of Drag and Drop Action is demonstrated in Google Maps. A route is visually represented on the map with a dark purple line. Dragging an arbitrary route point to a new location changes the route in real time (Figure 2-33).

Normal display state

Route is shown in dark purple.

Invitation to drag

Hovering over any part of the route provides a draggable circle (route point) with a tool tip saying "Drag to change route".

Dragging

We want to stay on the east side of the bay and cross the San Mateo bridge. Dragging the route bubble back over the bridge will reroute our trip.

Dropped

The route changes as we drag. Dropping completes the rerouting action.

Drag and Drop Collection

- A variation on dragging objects is collecting objects for purchase, bookmarking, or savinginto a temporary area. This type of interaction is called Drag and Drop Collection.
- Drag and drop is a nice way to grab items of interest and save them to a list. The Laszlo shopping cart example illustrates this nicely (Figure 2-34).

Normal display state

The shopping cart is docked on the right part of the screen.

Considerations

There are a few issues to consider in this example.

Discoverability

- Drag and drop is a natural way collect items for purchase. It mimics the shopping experience in the real world. Grab an item.
- Drop it in your basket. This is fast and convenient once you know about the feature. However, as a general rule, you should never rely solely on drag and drop for remembering items.

Teachable moment

- When providing alternates to drag and drop, it is a good idea to hint that dragging is an option.
- In the Laszlo example, clicking the "+ cart" button causes the shopping cart tray to bump slightly open and then closed again.
- This points to the physicality of the cart. Using another interaction as a teachable moment to guide the user to richer interactions is a good way to solve discoverability issues.

The Challenges of Drag and Drop

- As you can see from the discussion in this chapter, Drag and Drop is complex. There are four broad areas where Drag and Drop may be employed: Drag and Drop Module, Drag and Drop List, Drag and Drop Object, and Drag and Drop Action. And in each area, there are a large number of interesting moments that may be handled in numerous ways.
- Being consistent in visual and interaction styles across all of these moments for all of these types of interactions is a challenge in itself.

- And keeping the user informed throughout the process with just the right amount of hints requires design finesse. In Chapter 10, we explore some ways to bring this finesse into Drag and Drop.

2. Describe in detail in about Direct Selection?<u>OVERVIEW</u>

Toggle Selection
Collected Selection
Object Selection
Hybrid Selection

- When the Macintosh was introduced, it ushered into the popular mainstream the ability to directly select objects and apply actions to them. Folders and files became first-class citizens. Instead of a command line to delete a file, you simply dragged a file to the trashcan

- Treating elements in the interface as directly selectable is a clear application of the *MakeIt Direct* principle.
- On the desktop, the most common approach is to initiate a selection by directly clicking on the object itself. We call this selection pattern Object Selection

Toggle Selection
Checkbox or control-based selection.
Collected Selection
Selection that spans multiple pages.
Object Selection
Direct object selection.
Hybrid Selection
Combination of Toggle Selection and Object Selection.

Toggle Selection

- The most common form of selection on the Web is Toggle Selection. Checkboxes and toggle buttons are the familiar interface for selecting elements on most web pages. An example of this can be seen in Figure 3-3 with Yahoo! Mail Classic

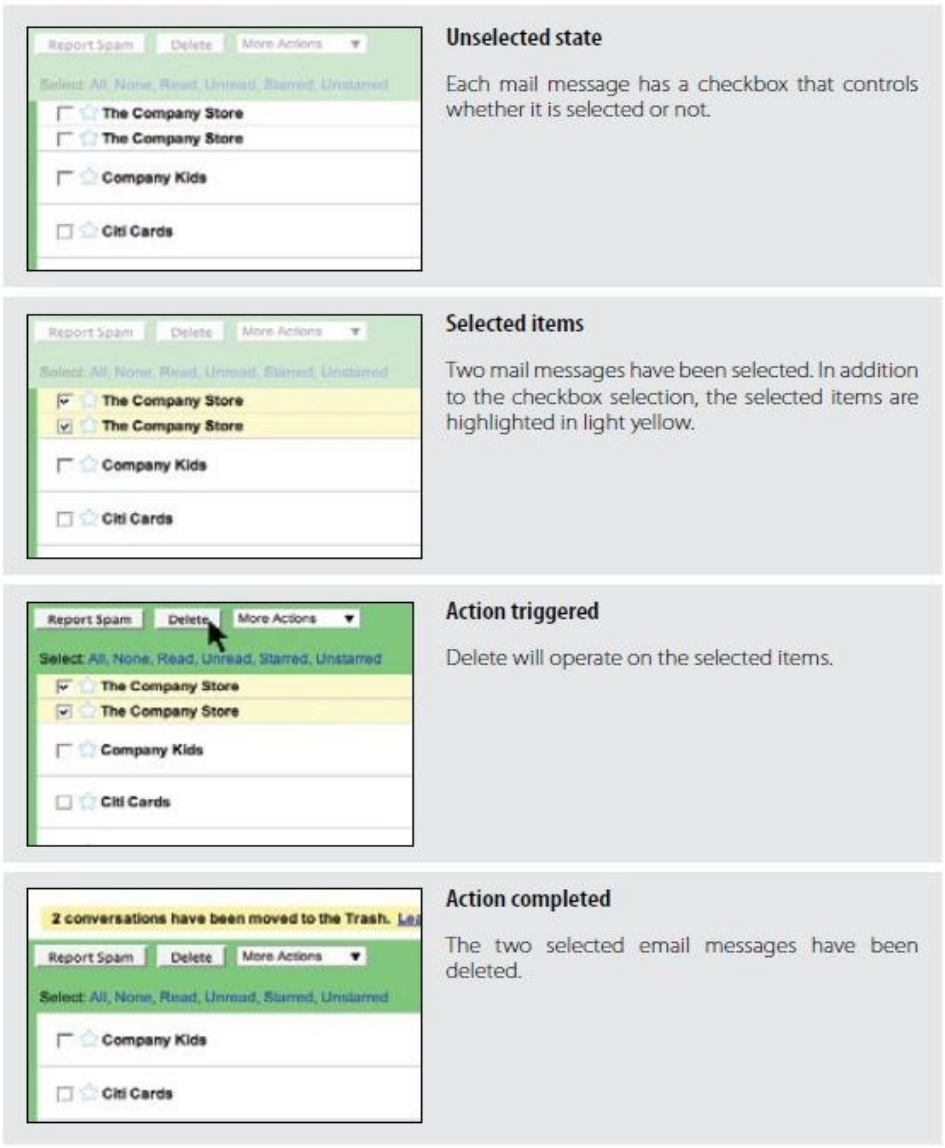
- The way to select an individual mail message is through the row's checkbox. Clicking on the row itself does not select the message. We call this pattern of selection Toggle Selection since toggle-style controls are typically used for selecting items.

Considerations

Toggle Selection with checkboxes has some nice attributes:

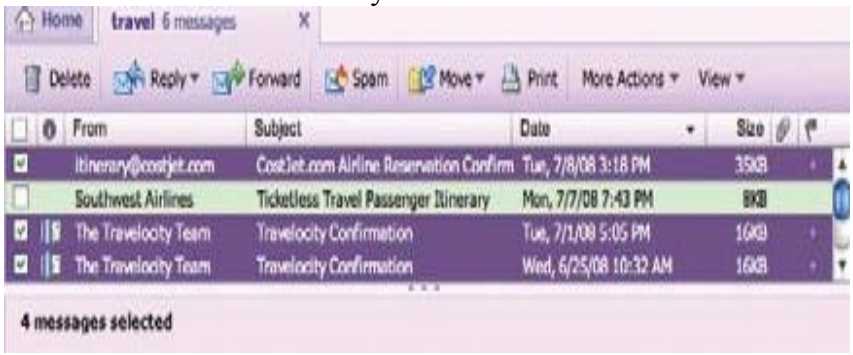
Clear targeting, with no ambiguity about h • ow to select the item or deselect it.

- Straightforward discontinuous selection, and no need to know about Shift or Controlkey ways to extend a selection. Just click the checkboxes in any order, either in a continuous or discontinuous manner.
- Clear indication of what has been selected.



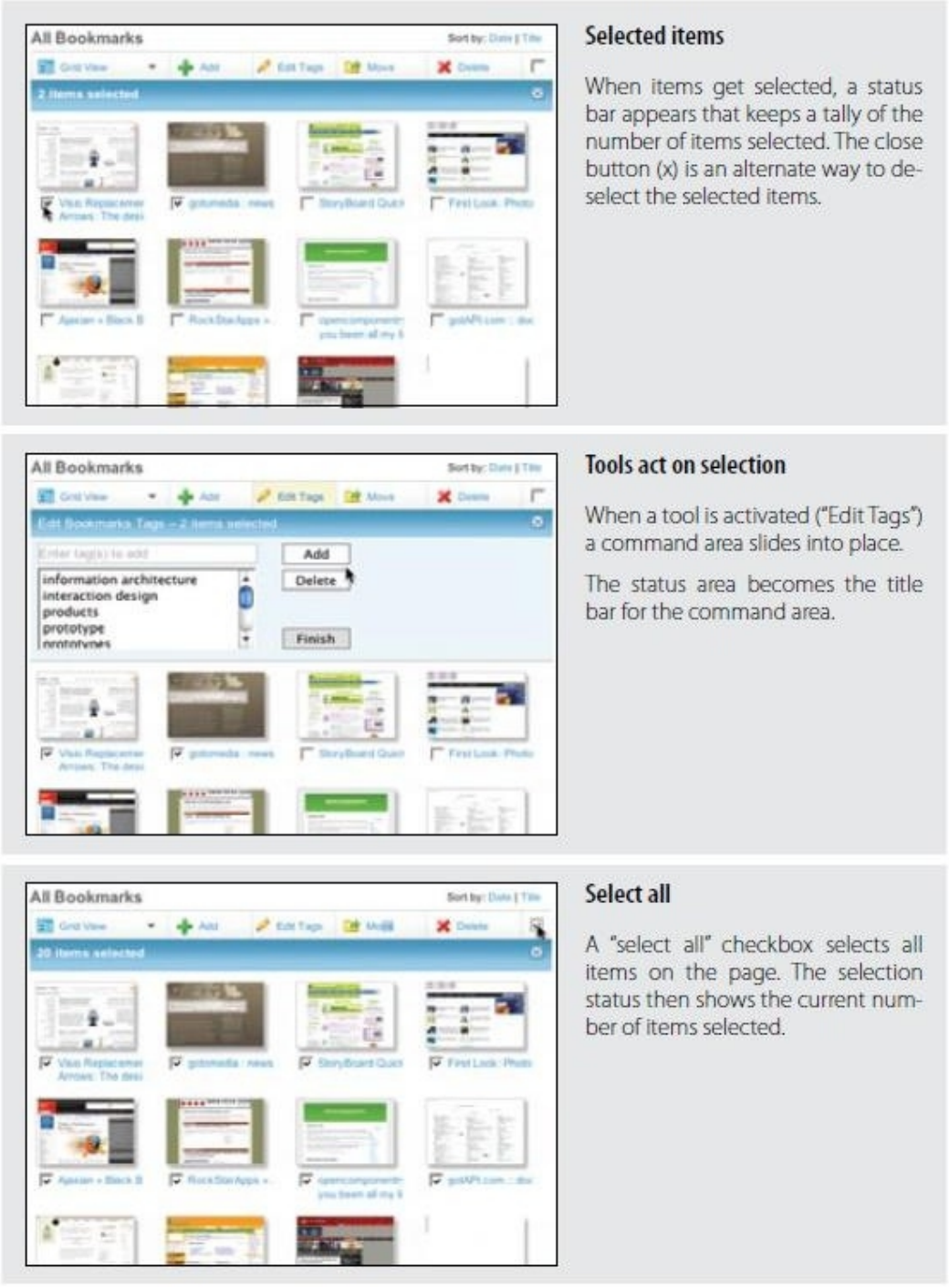
Scrolling versus paging

- The previous examples were with paged lists. But what about a scrolled list? Yahoo! Mail uses a scrolled list to show all of its mail messages (Figure 3-5). While not all messages are visible at a time, the user knows that scrolling through the list retains the currently selected items.



Making selection explicit

With Yahoo! Bookmarks you can manage your bookmarks by selecting bookmarked pages and then acting on them. The selection model is visually explicit (Figure 3-6).



The advantage of this method is that it is always clear how many items have been selected.

- Visualizing the underlying selection model is generally a good approach. This direct approach to selection and acting on bookmarks creates a straightforward interface. One interesting question: what happens when nothing is selected?
- One approach is to disable any actions that require at least one selected item. Yahoo! Bookmarks takes a different approach. Since buttons on the Web do not follow a standard convention, you often can't rely on a color change to let you know something is not clickable.
- Yahoo! Bookmarks chose to make selection very explicit and make it clear when a command is invalid because nothing is selected ("No selection" in Figure 3-6).

- This is not normally the optimal way to handle errors. Generally, the earlier you can prevent errors, the better the user experience.

Collected Selection

- Toggle Selection is great for showing a list of items on a single page. But what happens if you want to collect selected items across multiple pages? Collected Selection is a pattern for keeping track of selection as it spans multiple pages.
- In Gmail, you can select items as you move from page to page. The selections are remembered for each page.
- If you select two items on page one, then move to page two and select three items, there are only three items selected.
- This is because actions only operate on a single page. This makes sense, as users do not normally expect selected items to be remembered across different pages.

Considerations

- Gmail does provide a way to select all items across different pages. When selecting all items on a individual page (with the "All" link), a prompt appears inviting the user to "Select all 2785 conversations in Spam". Clicking that will select all items across all pages (Figure 3-8). The "Delete Forever" action will operate on all 2785 conversations, not just the 25 selected on the page.

Keeping the selection visible

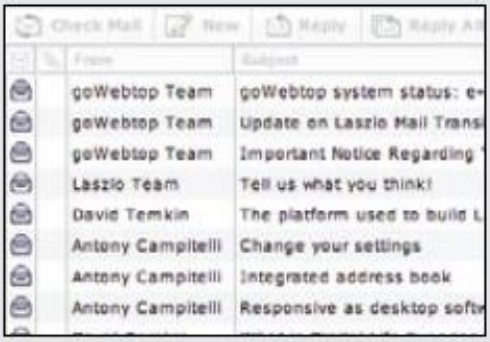
- The real challenge for multi-page selection is finding a way to show selections gathered across multiple pages.
- You need a way to collect and show the selection as it is being created. Here is one way that Collected Selection comes into play**.**

Collected Selection and actions

- When Yahoo! Photos was working its way through an early design of its Photo Gallery (see Figure 3-13, later in this chapter), the plan was to show all photos in a single, continuous scrolling page (we discuss virtual scrolling in Chapter 7). In a long virtual list, the selection model is simple.
- Photos are shown in a single page and selection is easily understood in the context of this single page. However, due to performance issues, the design was changed. Instead of a virtual page, photos had to be chunked into pages.
- In order to support Collected Selection, Yahoo!
- Photos introduced the concept of a "tray" into the interface (Figure 3-10). On any page, photos can be dragged into the tray. The tray keeps its contents as the user moves from page to page.
- So, adding a photo from page one and three more from page four would yield four items in the tray. As a nice touch, the tray would make itself visible (by sliding into view) even when the user was scrolled down below the fold.

Object Selection

- As we mentioned earlier, Toggle Selection is the most common type of selection on the Web.
- The other type of selection, Object Selection, is when selection is made directly on objects within the interface.
- Sometimes using a checkbox does not fit in with the style of interaction desired. Laszlo's WebTop mail allows the user to select messages by clicking anywhere in the row. The result is that the whole row gets highlighted to indicate selection (Figure 3-11).

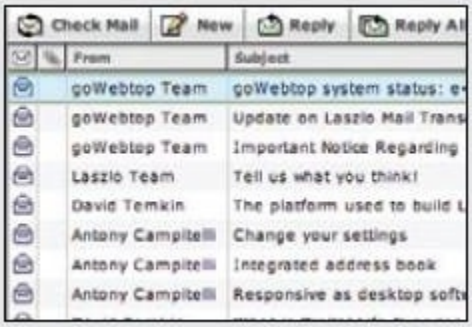


Nothing selected

Normal view when nothing is selected and the mouse is not over a message.

Hovered state

When the mouse hovers over a row, the row is subtly highlighted to indicate focus and what will be selected if the user clicks.



Selected state

When the user clicks on a message, the whole row gets selected.

Considerations

- Desktop applications tend to use Object Selection. It is also natural that web-based mail applications that mimic desktop interactions employ this same style of selection.
- Instead of showing a control (like a checkbox), the object itself can be selected and acted on directly.
- Object Selection can be extended by holding down the Shift key while clicking on a different item.
- The Command key (Macintosh) or Control key (Windows) can be used to individually add items in a discontinuous manner. The downside to this approach is that it is not obvious to use the modifier keys for extending the selection.
- Toggle Selection's use of toggle buttons makes the selection extension model completely obvious.

Desktop-style selection

- For now Object Selection is not as common on the Web. Given that most sites have been content-oriented, there have been few objects to select. Also, with the Web's simple event model, Object Selection was not easy to implement. In

- typical web pages, keyboard events have rarely made sense since they are also shared with the browser. However, all of this is changing as the capabilities of web technologies continue to improve.

Hybrid Selection

Mixing Toggle Selection and Object Selection in the same interface can lead to a confusinginterface. Referring back to Yahoo! Bookmarks, you'll see an odd situation arise during drag and drop (Figure 3-14).

Confusing two models

- In the left panel of Figure 3-14, one bookmark element is selected (notice the checkbox Toggle Selection).
- The second bookmark element ("Dr. Dobb's") is unselected (the checkbox is clear). In the right panel of Figure 3-14, clicking and dragging on the unselected bookmark element initiates a drag.
- The drag includes both the selected element and the unselected element. Since only one is shown as selected, this creates a confusing situation.

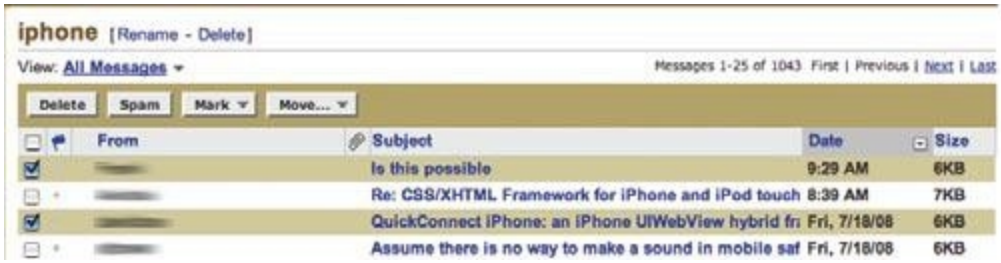
This occurs because three things are happening in the same space:

- Toggle Selection is used for selecting bookmarks for editing, deleting, etc.
- Object Selection is used for initiating a drag drop.
- Mouse click is used to open the bookmark on a separate page.

- The problem is that more than one interaction idiom is applied to the same place on the same page.
- In this case, if you happen to try to drag, but instead click, you will be taken to a new page. And if you drag an unselected item, you now have two items selected for drag but only one shown as selected for other operations (Figure 3-14, right).
- This is definitely confusing. Simply selecting the item (automatically checking the box) when the drag starts would keep the selection model consistent in the interface.
- However, it might lead the user to expect a single click to also do the same (which it cannot since it opens the bookmark).
- So, mixing the two selection models together can be problematic. However, there is a way to integrate the Toggle Selection and Object Selection and have them coexist peacefully as well as create an improved user experience.

Blending two models

- Yahoo! Mail originally started with the Toggle Selection model (Figure 3-15). When the new Yahoo! Mail Beta was released, it used Object Selection exclusively (Figure 3-16). But since there are advantages to both approaches, the most recent version of Yahoo! Mail incorporates both approaches in a Hybrid Selection (Figure 3-17).



- Hybrid Selection brings with it the best of both worlds. You can use the checkbox selection model as well as normal row selection.
- You get the benefit of explicit selection and simplified multiple selection that Toggle Selection brings.
- And you get the benefit of interacting with the message itself and direct object highlighting.

3. Explain in detail about Contextual tools?

OVERVIEW:

- √ **Always-Visible Tools**
- √ **Hover-Reveal Tools**
- √ **Toggle-Reveal Tools**
- √ **Multi-Level Tools**
- √ **Secondary Menus**

Interaction in Context

- Most desktop applications separate functionality from data. Menu bars, toolbars, and palettes form islands of application functionality.
- Either the user chooses a tool to use on the data or makes a selection and then applies the tool.
- Early websites were just the opposite. They were completely content-oriented. Rich tool sets were not needed for simply viewing and linking to content pages.
- Even in e-commerce sites like Amazon or eBay, the most functionality needed was the hyperlink and "Submit" button.
- However, this simplistic approach no longer exists in the current web application landscape. As the Web has matured, a wide variety of application styles has emerged.

- Touch-based interfaces were the stuff of research labs and, more recently, interesting You- Tube videos.
- But now they're as close as our phones. Most notably, the Apple iPhone brought touch to the masses (Figure 4-2). Gesture-based interfaces seemed even further out. Yet these became reality with the Nintendo Wii.

Fitts's Law

- Fitts's Law is an ergonomic principle that ties the size of a target and its contextual proximity to ease of use. Bruce Tognazzini restates it simply as:

The time to acquire a target is a function of the distance to and size of the target.

- In other words, if a tool is close at hand and large enough to target, then we can improve the user's interaction. Putting tools in context makes for lightweight interaction.

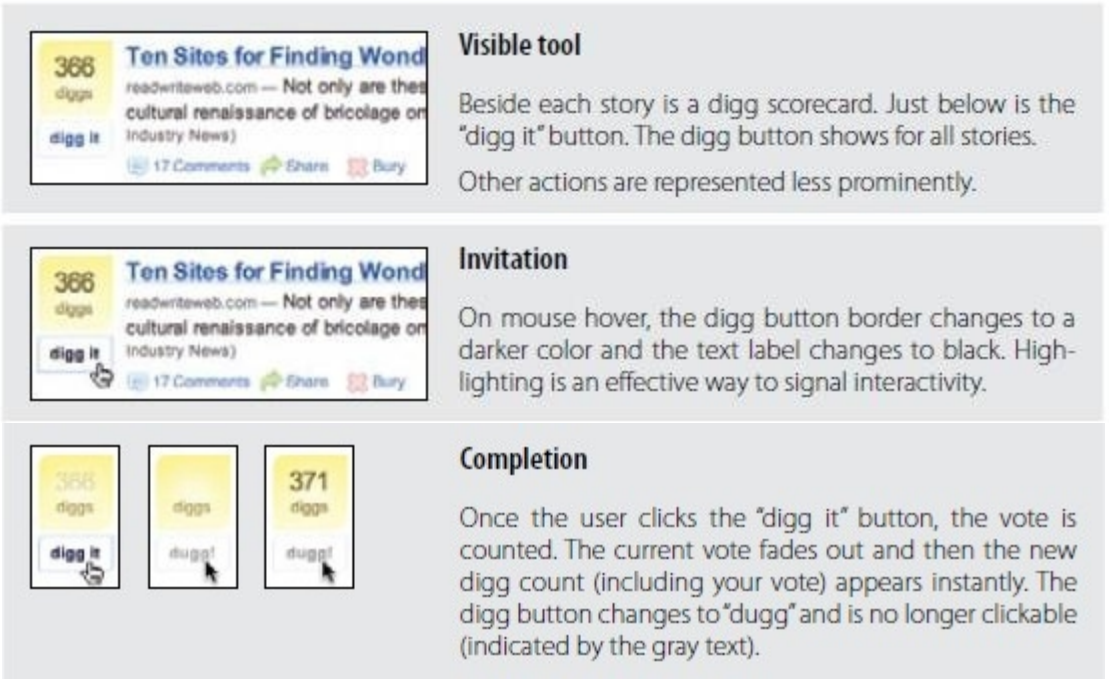
Contextual Tools

- We could simply isolate our functionality into islands of tools (toolbars and menus). But this would work against Fitts's Law by requiring more effort from the user.
- It would also add more visual weight to the page. Instead of interacting with the functionality separately, we can bring the functionality into the content with Contextual Tools.
- Contextual Tools are the Web's version of the desktop's right-click menus. Instead of having to right-click to reveal a menu, we can reveal tools in context with the content. We can do this in a number of ways:

  Always-Visible Tools
  Place Contextual Tools directly in the content.
  Hover-Reveal Tools
  Show Contextual Tools on mouse hover.
  Toggle-Reveal Tools
  A master switch to toggle on/off Contextual Tools for the page.
  Multi-Level Tools
  Progressively reveal actions based on user interaction.
  Secondary Menus
  Show a secondary menu (usually by right-clicking on an object).

Always-Visible Tools

- The simplest version of Contextual Tools is to use Always-Visible Tools. Digg is an example of making Contextual Tools always visible (Figure 4-3).



Considerations

- The "digg it" button and Digg scorecard provide Always-Visible Tools next to each story.
- Clear call to action Why not hide the tools and only reveal them when the mouse is over the story? Since digging stories is central to the business of Digg, always showing the tool provides a *clear call to action*.
- There are other actions associated with news stories (comments, share, bury, etc.) but they are represented less prominently. In the case of Digg, the designers chose to show these at all times.
- An alternate approach would be to hide them and show them on mouse hover (we will discuss this approach in the next section).
- It turns out that voting and rating systems are the most common places to make tools always visible. Netflix was the earliest to use a one-click rating system (Figure 4-4).

Relative importance

- One way to clarify this process is to decide on the relative importance of each exposed action. Is the "digg it" action as important as the "bury it" action? In the case of Digg, the answer is no.
- The "digg it" action is represented as a button and placed prominently in the context of the story. The "bury it" action is represented as a hyperlink along with other "minor" actions just below the story.

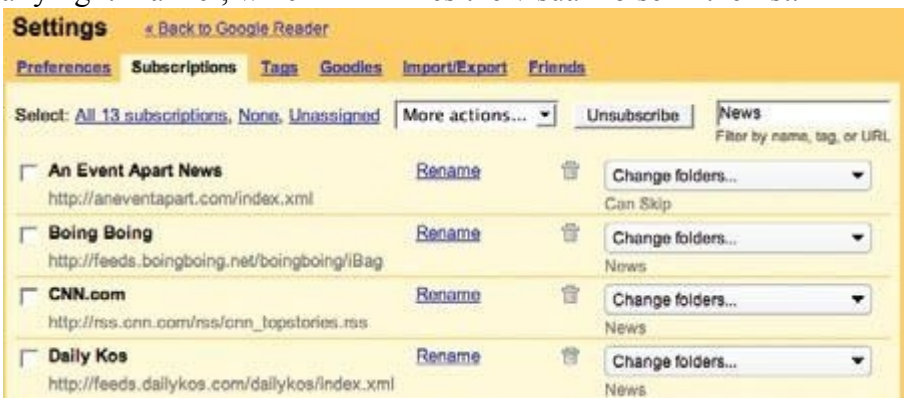
Discoverability

- Discoverability is a primary reason to choose Always-Visible Tools. On the flip side, it can lead to more visual clutter.
- In the case of Digg and Netflix, there is a good deal of visual

space given to each item (story, movie).

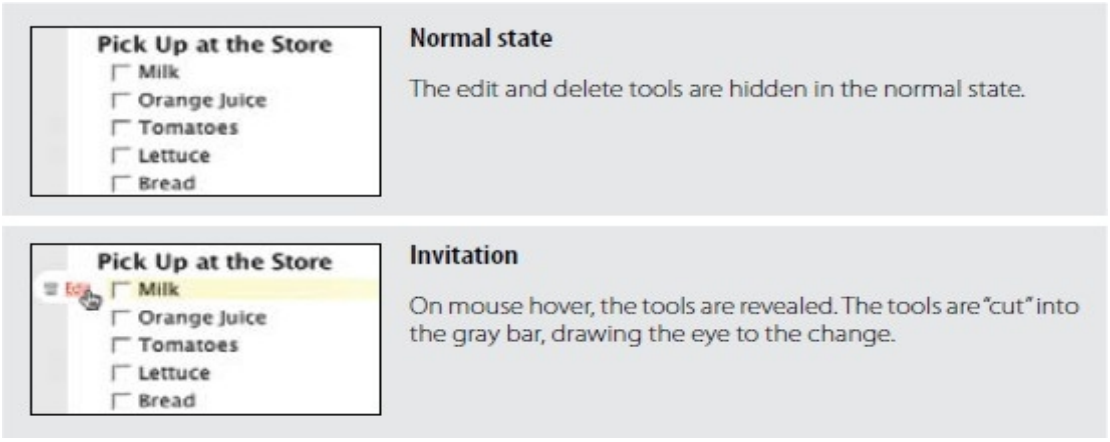
- But what happens when the items you want to act on are in a list or table? Generally Contextual Tools in a list work well when the number of actions is kept to a minimum.
- Gmail provides a single Always-Visible Tool in its list of messages—the star rating—for flagging emails (Figure 4-5).

- Simply clicking the star flags the message as important. The unstarred state is rendered in a visually light manner, which minimizes the visual noise in the list.



Hover-Reveal Tools

- Instead of making Contextual Tools always visible, we can show them on demand. One way to do this is to reveal the tools when the user pauses the mouse over an object.
- The Hover-Reveal Tools pattern is most clearly illustrated by 37 Signal's Backpackit (Figure m4-8). To-do items may be deleted or edited directly in the interface. The tools to accomplish this are revealed on mouse hover.



Considerations

- The gray bar on the left is a nice visual reinforcement for the interaction. By allowing the tools to "cut" into the sidebar, the designers draw your eye to the available tools.
- The light yellow background draws attention to the to-do item being acted on. These two simple treatments make it clear which line has the focus and that additional tools have been revealed.

Visual noise

- Showing the items on hover decreases the visual noise in the interface. Imagine if instead the delete and edit actions were always shown for all to-do items. Figure 4-9 shows just how visually noisy that approach would have been.

- Yahoo! Buzz reveals its tools on mouse hover for both its Top Searches (Figure 4-10) and Top Stories (Figure 4-11). For Top Searches, it is important to keep the top-ten list as simple as possible. Showing tools would compete with the list itself.
- Since the actions "Search Results" and "Top Articles" (Figure 4-10, right) are less important, they are revealed on hover

Discoverability

- A serious design consideration for Hover-Reveal Tools is just how discoverable the additional functionality will be.
- In the earlier Backpackit example (Figure 4-8), while the Contextual Tools are revealed on hover, the checkbox is always visible for each to-do item. To check off an item, users have to move the mouse over it.
- When they do, they will discover the additional functionality.
- Flicker provides a set of tools for contacts. To avoid clutter, contact profile photos are shown without any tool adornment.
- When the mouse hovers over the contact's photo, a drop-down arrow is revealed (Figure 4-12). Clicking reveals a menu with a set of actions for the contact. This works because users often know to click on an image to get more information.
- Being drawn to the content is a good way to get the user to move the mouse over the area and discover the additional functionality.

Contextual Tools in an overlay

- Sometimes there are several actions available for a focused object. Instead of placing tools beside the object being acted on, the revealed tools can be placed in an overlay.
- However, there can be issues with showing contextual tools in an overlay:

  1. Providing an overlay feels heavier. An overlay creates a slight contextual switch for the user's attention.

  2. The overlay will usually cover other information—information that often provides context for the tools being offered.

3. Most implementations shift the content slightly between the normal view and the overlay view, causing the users to take a moment to adjust to the change.

4. The overlay may get in the way of navigation. Because an overlay hides at least part of the next item, it becomes harder to move the mouse through the content without stepping into a "landmine."

Anti-pattern: Hover and Cover

- For Teachers,* hovering over a clipped item brought in three tools: copy, delete, and preview.
- However, when these tools were placed in an overlay, it covered the item to the right, making it hard to see that content and even navigate to it. In addition, since the overlay had some additional padding (as well as rounded corners), the image shown in the overlay was about two pixels off from the non-overlay version.
- This slight jiggle was distracting. To add insult to injury, the overlay was sluggish to bring into view.

- The final straw was if users wanted to delete several items, they would hover over the image, wait for the overlay, click Delete, then be forced to move out and back in again to activate the next image's Contextual Tools (Figure 4-15). Hover and Cover is a common anti-pattern that occurs when exposing an overlay on hover and hiding important context or further navigation.

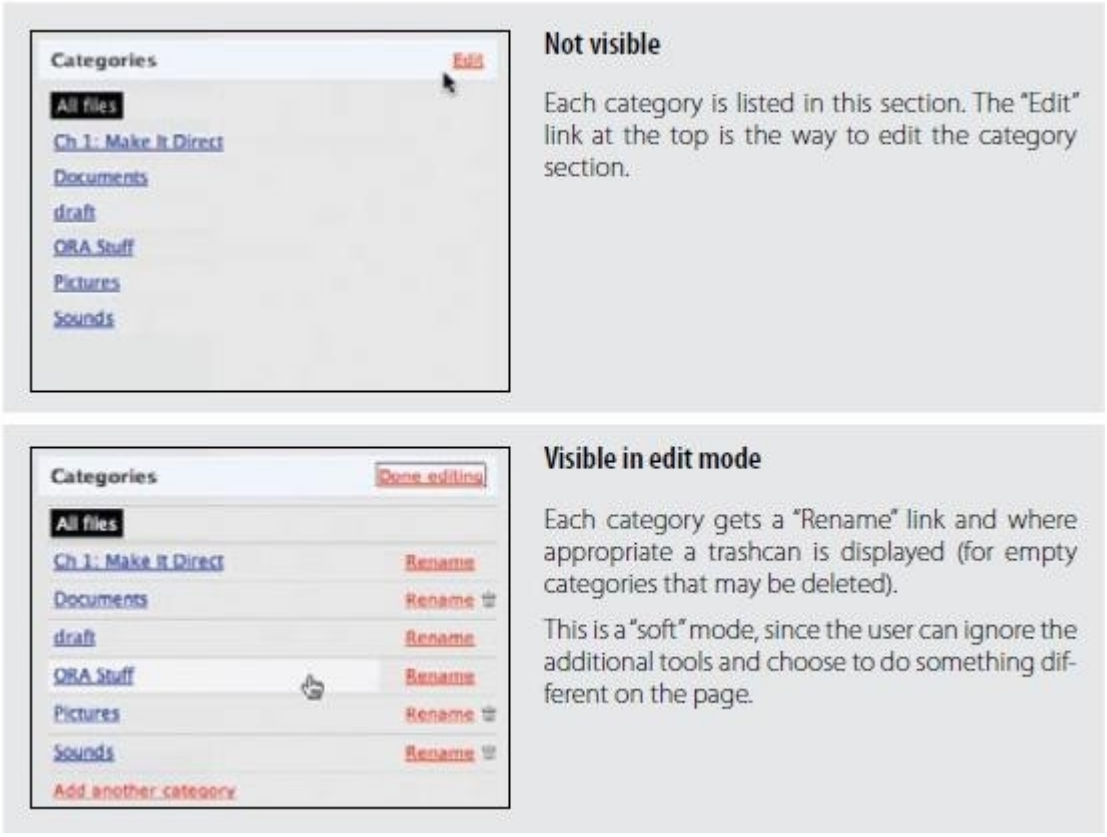
- **Hover and Cover** was resolved by no longer using an overlay. Instead, additional margin space was added to each image, and the Contextual Tools were hidden. On mouse hover, the tools were simply revealed, along with a border defining the image being acted on

Anti-pattern: Mystery Meat

- Have you ever found a can in the back of the pantry whose label has long since fallen off? The only way to identify this *mystery meat* is to open it. Unidentifiable icons are pretty much the same as a row of unlabeled cans. You have to hover over each icon and hope for a tool tip to label it. Even worse is when no tool tip is available.
- The easiest way to avoid this predicament is to use either a text label or combine an icon with a text label.
- Mystery Meat is a common anti-pattern that occurs when you have to hover over an item to understand how to use it.

Toggle-Reveal Tools

- A variation on the two previous approaches is to not show any Contextual Tools until a special mode is set on the page.
- A good example of Toggle-Reveal Tools is in Basecamp's category editing, which we discussed in Chapter 1 (Figure 4-19).



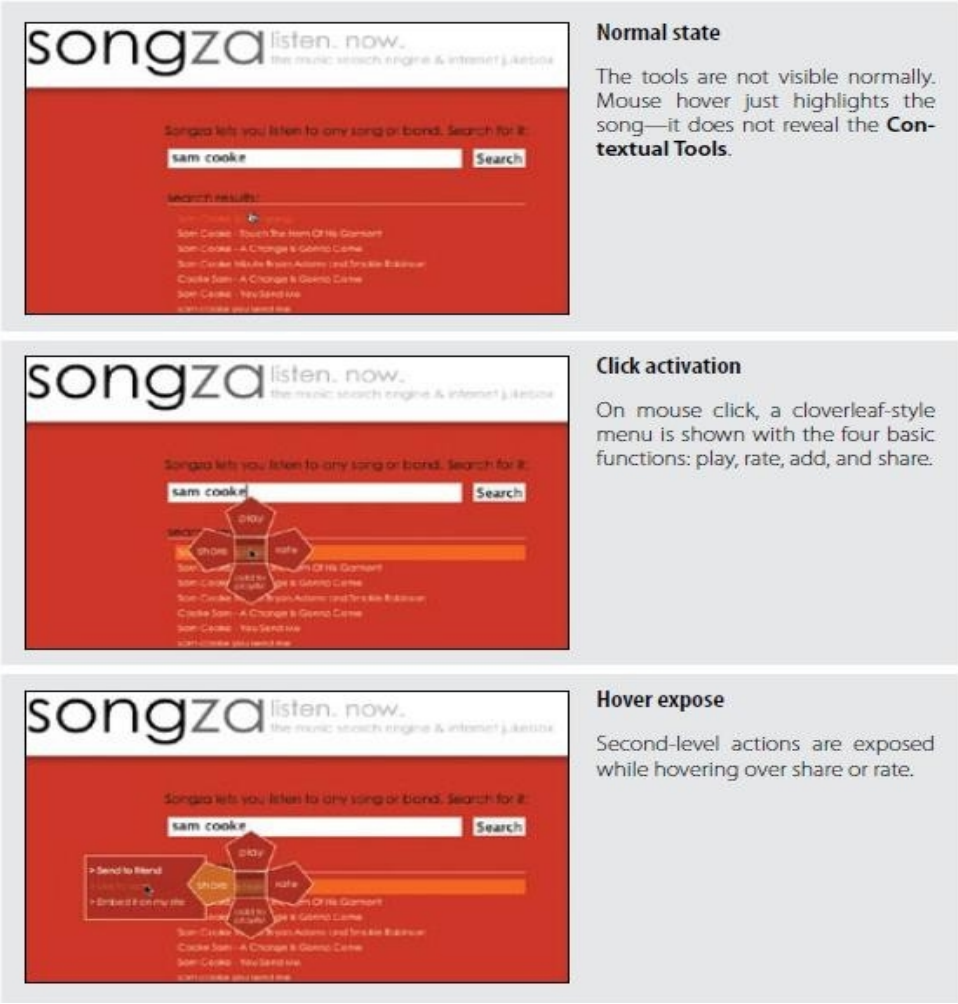
Considerations

- Here are a few considerations to keep in mind when using Toggle-Reveal Tools. Soft mode Generally, it is a good thing to avoid specific modes in an interface.
- However, if a mode is *soft* it is usually acceptable. By "soft" we mean the user is not trapped in the mode.
- With Basecamp, the user can choose to ignore the tools turned on. It just adds visual noise and does not restrict the user from doing other actions. This is a nice way to keep the interaction lightweight.
- When would you use this technique? When the actions are not the main thing and you want to reduce visual noise.
- This fits the category example perfectly. Items are renamed or deleted occasionally. It is common, however, to want to click through and see the contents of a category (the

category is always hyperlinked). Hence, make it readable and easily navigable in the normal case—but still give the user a way to manage the items in context.

Multi-Level Tools

- Contextual Tools can be revealed progressively with Multi-Level Tools. Songza* provides a set of tools that get revealed after a user clicks on a song. Additional tools are revealed when hovering over the newly visible tools (Figure 4-21).



Considerations

- Songza reveals the four options "play", "rate", "share", and "add to playlist" after the user clicks on a song title. Hovering over "share" or "rate" reveals a secondary set of menu items

Radial menus

- *Radial menus** such as in Songza have been shown to have some advantages over more traditional menus.

- First, experienced users can rely on muscle memory rather than having to look directly at the menu items. Second, the proximity and targeting size make the menu easy to navigate since the revealed menu items are all equally close at hand (recall Fitts's Law).
- The one potential downside to this approach is that rating a song requires several steps: an initial click on the song, moving the mouse over the "rate" menu item, then clicking either the thumbs up or thumbs down option.
- If rating songs was an important activity, the extra effort might prevent some users from doing so. An alternate approach would be to replace "rate" directly with the thumbs up and the thumbs down options.

Activation

- Another interesting decision Songza made was to not activate the radial menu on hover. Instead, the user must click on a song to reveal the menu. Activating on click makes the user intent more explicit.
- Making activation more explicit avoids the issues described earlier in the Hover and Cover anti-pattern. The user has chosen to interact with the song. Conversely, with a mouse hover, it's never quite clear if the user meant to activate the menu or just happened to pause over a song title.

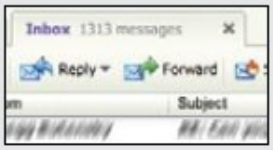
Default action

- However, this does mean there is no way to start a song playing with just a simple click. bPlaying a song requires moving to the top leaf. One possible solution would be to place the "play" option in the middle of the menu (at the stem) instead of in one of the leaves.
- Clicking once would activate the menu. Clicking a second time (without moving the mouse) would start playing the song. This interaction is very similar to one commonly used in desktop applications: allowing a double-click to activate the first item (default action) in a right-click menu.

Muttons

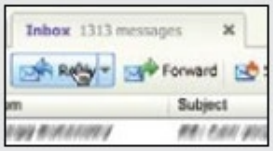
- Another variation on Multi-Level Tools is the "mutton" (menu + button = mutton). Muttons are useful when there are multiple actions and we want one of the actions to be the default. Yahoo! Mail uses a mutton for its "Reply" button (Figure 4-23).

  Clicking "Reply" performs the individual reply. To reply to all, the menu has to be activated
  by clicking on the drop-down arrow to show the menu.
  Muttons are used to:
  Provide a defa • ult button action ("Reply to Sender")
  • Provide a clue that there are additional actions
  • Provide additional actions in the drop-down



Normal state

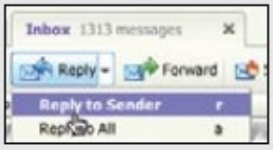
Yahoo! Mail displays the "Reply" mutton in its toolbar as a button with a drop-down arrow control.



As a button

On mouse hover, the button gets a 3D treatment and color highlight. The drop-down arrow gets the same treatment to call out its functionality.

Clicking the "Reply" button at this point will trigger a reply without activating the menu.



As a menu

Clicking on the drop-down arrow reveals two commands: "Reply to Sender" is the same as the default "Reply" button action; "Reply to All" is an additional action that was hidden until the menu was revealed.

Anti-pattern: Tiny Targets

- At the beginning of this chapter, we discussed Fitts's Law. Recall that the time it takes to acquire a target is a function of both distance and size. Even if tools are placed close by in context, don't forget to make them large enough to target.
- Both Flickr and Yahoo! Mail provide a reasonable-size target for the drop-down arrow. Compare this with the expand/collapse arrow used in an early version of Yahoo! For Teachers (Figure 4-24).

Secondary Menu

- Desktop applications have provided Contextual Tools for a long time in the form of Secondary Menus.
- These menus have been rare on the Web. Google Maps uses a secondary menu that is activated by a right-click on a route. It shows additional route commands

Normal view of route

Routes give no indication of additional functionality when not hovered over.

Invitation

When the mouse is over the route, potential stops are marked with a white circle.

Menu

Right-clicking on the item exposes four commands that act on the point selected: "Add a destination", "Zoom in", "Zoom out", and "Center map here".

Considerations

- Secondary Menus have not been common in web applications.
- Conflict with browser menu One problem is the browser inserts its own right-click menu. Replacing the menu in normal content areas can confuse users, as they may not know if the standard browser menu or the application-specific menu will be shown.
- It will depend on whether it is clear that an object exists in the interface (as in the route line above), and if the menu is styled differently enough to disambiguate the menus.

Discoverability

- As a general rule, never put anything in the Secondary Menu that can't be accomplished elsewhere. Secondary Menus are generally less discoverable.
- More advanced items or shortcuts, however, can be placed in the Secondary Menu as an alternate way to accomplish the same task.

Accessibility

- Right-click is not the only way to activate a Secondary Menu. You can activate the menu by holding down the mouse for about one second.
- This provides a more accessible approach to popping up a Secondary Menu. This technique is used in the Macintosh Dock. Clicking and holding down on an application in the dock will reveal the Secondary Menu without requiring a right-click activation.

Acting on multiple objects

- Keep in mind that all of the other Contextual Tools presented in this chapter have a limitation on the number of items they can operate on. Always-Visible Tools, Hover-Reveal Tools, Toggle-Reveal Tools, and Multi-Level Tools all operate on a single item at a time (even Toggle-Reveal Tools just shows a tool per item).
- Secondary Menus are different. They can be combined with a selection model (as described in Chapter 3) to perform actions on selected set of items.

4. Describe in detail about overlay and its types?OVERVIEW:

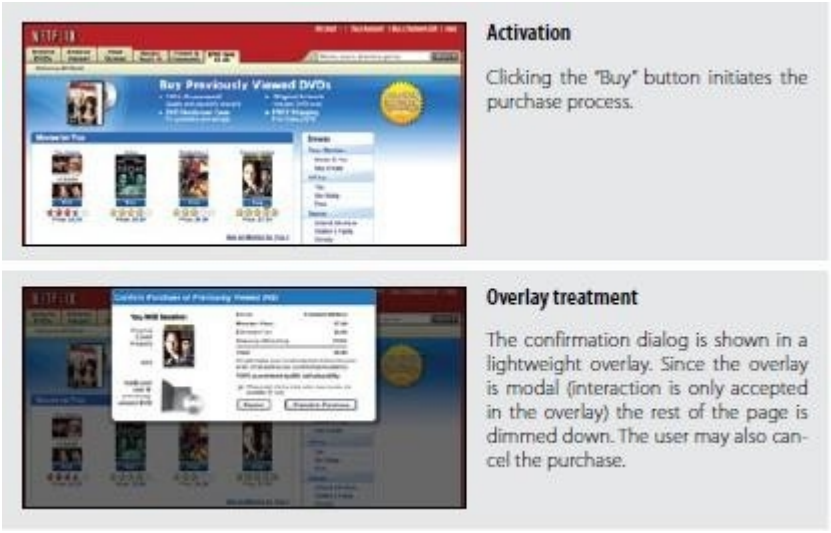
Dialog Overlay
Detail Overlay
Input Overlay

- Overlays are really just lightweight pop ups. We use the term *lightweight* to make a clear distinction between it and the normal idea of a *browser pop up*. Browser pop ups are created as a new browser window (Figure 5-1). *Lightweight overlays* are shown within the browser page as an overlay (Figure 5-2).
- Older style browser pop ups are undesirable because: Browser pop ups display a new browser window. As a result • these windows often take time and a sizeable chunk of system resources to create.
- Browser pop ups often display browser interface controls (e.g., a URL bar). Due to security concerns, in Internet Explorer 7 the URL bar is a permanent fixture on any browser pop-up window.

Dialog Overlay

- Dialog Overlays replace the old style browser pop ups. Netflix provides a clear example of a very simple Dialog Overlay
- . In the "previously viewed movies for sale" section, a user can click on a "Buy" button to purchase a DVD. Since the customer purchasing the DVD is a member of Netflix, all the pertinent shipping and purchasing information is already on record. The complete checkout experience can be provided in a single overlay

**



Considerations

- Because the overlay is a lightweight pop up, the confirmation can be displayed more rapidly and the application has complete control over its look and placement.

Lightbox Effect

- One technique employed here is the use of a Lightbox Effect. In photography a lightbox provides a backlit area to view slides.
- On the Web, this technique has come to mean bringing something into view by making it brighter than the background. In practice, this is done by dimming down the background.

that the user should not ignore. Both the Netflix Purchase dialog and the Flickr Rotate dialog are good candidates for the Lightbox Effect.
If the overlay contains optional information, then the Lightbox Effect is overkill and should not be used.

Modality

- Overlays can be modal* or non-modal. A modal overlay requires the user to interact with it before she can return to the application. In both the Netflix example (Figure 5-3) and the Flickr example (Figure 5-4), the overlays are *modal*: users cannot interact with the main Netflix or Flickr page until they perform the action or cancel the overlay.
- In both cases, modality is reinforced with the Lightbox Effect. Dimming down the background cues the user that this portion of the interface cannot be interacted with.

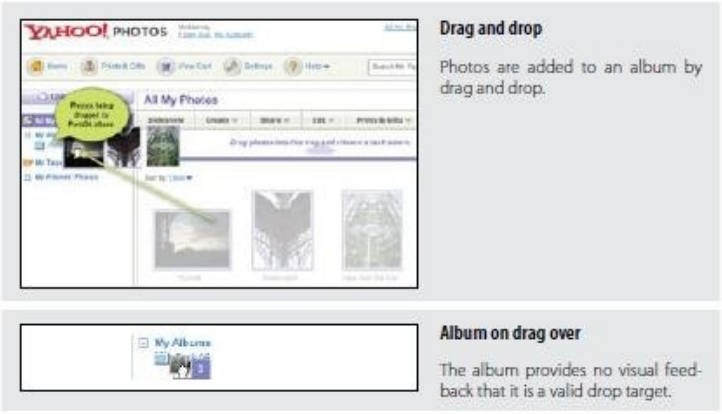
Staying in the flow

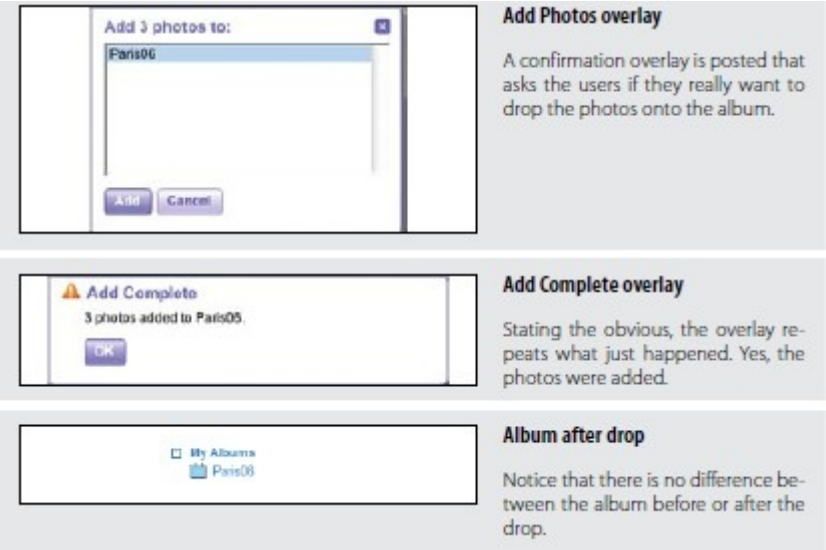
- Overlays are a good way to avoid sending a user to a new page. This allows the user to stay within the context of the original page.
- However, since overlays are quick to display and inexpensive to produce, sometimes they can be tempting to use too freely, and in the process, may actually break the user's flow.

Anti-pattern: Idiot Boxes Alan Cooper states a simple principle:
*Don't stop the proceedings with idiocy.**

- This is a clear anti-pattern that should be avoided. We call these types of overlays Idiot Boxes. One of the clearest examples of Idiot Boxes is the way certain confirmation overlays were used in Yahoo! Photos.
- When users selected a set of photos and dragged and then dropped them onto an album, they were treated with not just one confirmation, but with two (Figure 5-6).

**





Using JavaScript alerts

- It is tempting to use the alert mechanism built into the browser for some confirmations. The problem with using this type of alerts is two-fold.
- First, they do not present themselves consistently across different operating systems. Under Microsoft Windows they will appear centered in the browser window, but with the Macintosh they will slide out from under the title bar. Depending on where the action takes place, users may have to move their mouse a lot further each time they need to dismiss the alert (Figure 5-7).

Detail Overlay

- The second type of overlay is somewhat new to web applications. The Detail Overlay allows an overlay to present additional information when the user clicks or hovers over a link or section of content.
- Toolkits now make it easier to create overlays across different browsers and to request additional information from the server without refreshing the page.
- Taking another example from Netflix, information about a specific movie is displayed as the user hovers over the movie's box shot (Figure 5-8).

Activation

- The overlay is displayed when the mouse hovers over a box shot. There is about a halfsecond delay after the user pauses over a movie.
- The delay on activation prevents users from accidentally activating the overlay as they move the cursor around the screen.
- Once the user moves the cursor outside the box shot, the overlay is removed immediately. Removing it quickly gives the user a fast way to dismiss it without having to look for a "Close" box.

Anti-pattern: Mouse Traps

- It is important to avoid activating the Detail Overlay too easily. We have seen usability studies that removed the delay in activation, and users reported that the interface was "too noisy" and "felt like a trap".
- We label this anti-pattern the Mouse Trap. The reasoning for this is not clear, but Amazon uses the Mouse Trap anti-pattern in one of its "associate widgets".
- In Figure 5-10 the link "Ray! Original Motion Picture Soundtrack" activates an overlay providing information on the soundtrack and a purchase option. Presumably, this approach is intended to drive purchases—but it also presents an annoying experience.

Anti-pattern: Non-Symmetrical Activation/Deactivation

- When the user moves her mouse over the link, the overlay springs up immediately.
- The only way she can remove the overlay is by clicking the small close button in the upper right.
- Using Non-Symmetrical Activation/Deactivation is also a general anti-pattern that should be avoided. It should take the same amount of effort to dismiss an overlay as it took to open it.

Anti-pattern: Needless Fanfare

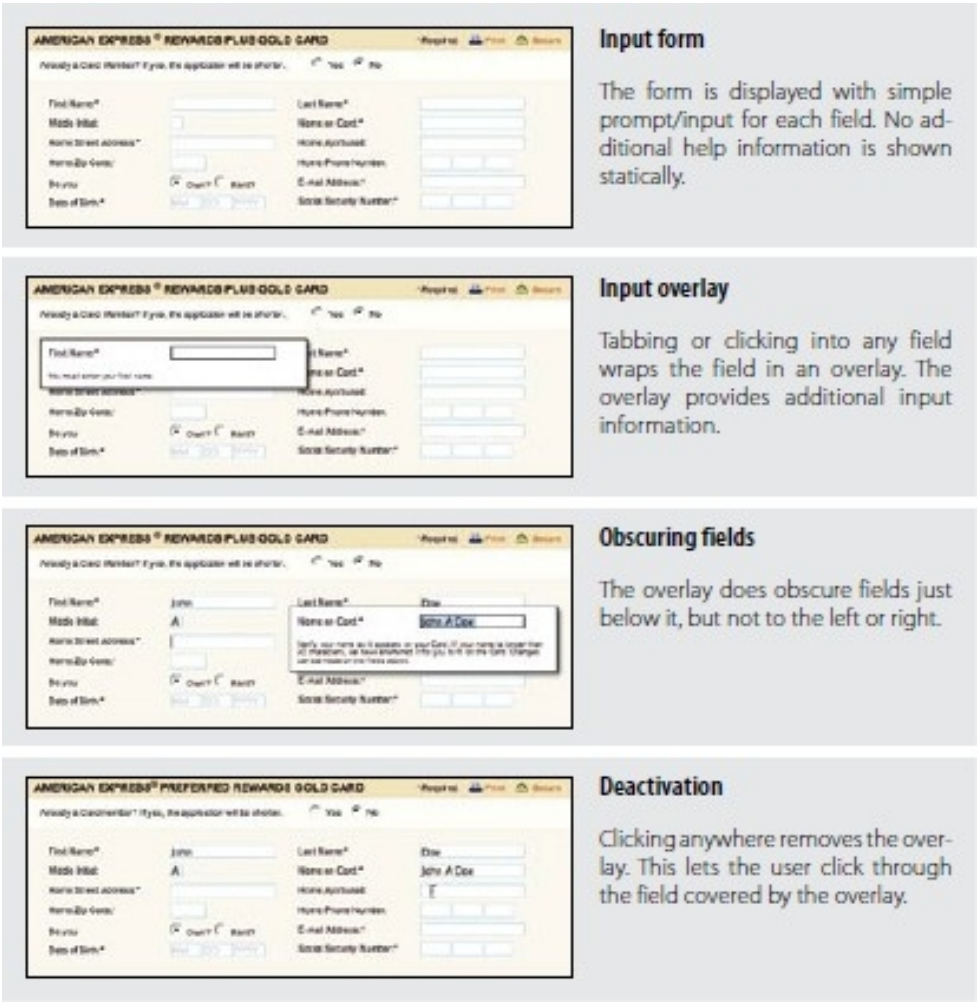
- One of the advantages of a lightweight overlay is the ability to pop it up quickly. After a slight delay in activation (recall the half-second delay used by Netflix), you would not want or need the overlay to come up slowly. But in the case of Borders online, this is preciselythe approach taken (Figure 5-12). First the activation is immediate (no delay

Anti-pattern: Hover and Cover

- We discussed the anti-pattern Hover and Cover in Chapter 2, and it's important to keep this anti-pattern in mind when providing a Detail Overlay. In the Netflix example (Figure 5-7), the overlay does not get in the way of moving to the next box shot

Input Overlay

- Input Overlay is a lightweight overlay that brings additional input information for each field tabbed into. American Express uses this technique in its registration for premium cards such as its gold card (Figure 5-15).



**

Considerations

- There are a few things to keep in mind when using Input Overlays.
- Clear focus As the user tabs or clicks from field to field, the field gets wrapped in an overlay. The overlay contains additional input help information.
- This allows the normal display of the form to be displayed in a visually simple manner (just prompts and inputs). The overlay creates focus on the given input field. Instead of seeing an ocean of inputs, the user is focused on just entering one field.

Display versus editing

- Additionally, when the Input Overlay is shown, the prompt is displayed in exactly the same manner as when the overlay doesn't show. This is critical, as it makes the overlay fee even more lightweight.
- If the overlay prompt were bold, for example, the change would be slightly distracting and take the focus away from input. The only difference between the non-overlay field and the overlay version is a slightly thicker input field border. This draws the eye to the task at hand—input.

Anti-pattern: Hover and Cover

- But what about the anti-pattern, Hover and Cover? Doesn't this cause the same issues that we saw earlier? For example, in Figure 5-15 ("Obscuring fields"), the "Name on Card" overlay hides the "Home Apt/Suite#" and "Home Phone Number Fields" fields below it.
- There are several reasons that American Express was able to employ this overlay in forms and "get away" with covering some fields during input:
  - √ *Field traversal*
  - √ *Tab navigation*
  - √ *One-click deactivation*

5. Explain in detail in inlay and its types?

OVERVIEW

- √ ***Dialog Inlay***
- √ ***List Inlay***
- √ **Detail Inlay**

- Not every bit of additional control, information, or dialog with the user needs to be an overlay. Another approach is to inlay the information directly within the page itself. To distinguish from the pop-up overlay, we call these in-page panels Inlays.

Dialog Inlay

- A simple technique is to expand a part of the page, revealing a dialog area within the page. The BBC recently began experimenting with using a Dialog Inlay as a way to reveal customization controls for its home page

Considerations

- Of course an overlay could have been used instead. However, the problem with overlays is that no matter where they get placed, they will end up hiding information.
- Inlays get around this problem by inserting themselves directly into the context of the page. Here's one more example, which shows how a Dialog Inlay can be used to operate on specific objects within an interface.
- In Yahoo! Bookmarks, selected bookmarks can be edited, deleted, etc. Because a pop up will often hide the items being operated on (the user ends up having to move the dialog pop up out of the way to make sure the right object is being deleted), an inlay can work nicely when tied in

- By being in proximity to the toolbar and keeping the objects visible, the Dialog Inlay provides a clear way to expose additional interface elements in context.
- In all of the previous examples, the panel is revealed with an animated slide in transition. This provides a nice way to smooth out the experience (we will discuss transitions in Chapters 11 and 12).

List Inlay

- Lists are a great place to use Inlays. Instead of requiring the user to navigate to a new page for an item's detail or popping up the information in an Overlay, the information can be shown with a List Inlay in context. The List Inlay works as an effective way to hide detail until needed—while at the same time preserving space on the page for high-level overview information.
- Google Reader provides an expanded view and a list view for unread blog articles. In the list view, an individual article can be expanded in place as a List Inlay (Figure 6-5)

.

Considerations

- By allowing the reader to move through a list of article titles (by mouse or keyboard), the articles can be scanned quickly so she can decide which should be looked at in detail.
- Clicking on an article title expands the article in place. Showing one item at a time focuses the reader on the current content.
- Accordion: One-at-a-time expand The Accordion is an interface element that employs the List Inlay pattern to show only one open panel in a list at a time. The following weather widget demo uses weather.com to display real-time weather information in an accordion (Figure 6-6).

Accordions work best when:

- Each pane of content is independent
- Only one pane is visible at a time
- Each pane can be logically titled

Accordion: More than one pane visible at a time

You may want to allow multiple panes to be visible at a time if:

- Detail panes provide context for one another
- The vertical height of the accordion area is not fixed
- Different panels of content may be different heights

Parallel content

- The Yahoo! Autos Car Finder tool (Figure 6-7) uses an accordion-style interaction for search filters that allows more than one pane to be open at a time.
- This choice makes sense because the decisions needed for one detail pane may be affected by the details of another pane.
- However, one problem with this specific implementation is the lack of information when a pane is closed. For example, no summary information is given for the "Price" tab.
- Looking at that button, it is not clear whether search criteria has been set or what it might be set to without first opening the pane.

Detail Inlay

- A common idiom is to provide additional detail about items shown on a page. We saw this with the example of the Netflix movie detail pop up in Chapter 5 (Figure 5-8).
- Hovering over a movie revealed a Detail Overlay calling out the back-of-the-box information. Details can be shown inline as well. Roost allows house photos to be viewed in-context for a real estate listing with a Detail Inlay (Figure 6-10).

**

Considerations

- One of the more difficult things to do on most real estate sites is get a view of the house

in context without having to navigate from page to page.

- The curb appeal, inside view, and backyard are all key factors in driving interest for a house. Knowing this, the team at Roost wanted to make it really easy to get to the photos quickly.

Combining inlays and overlays

- Roost's solution was to combine several patterns. First, it uses the Hover Reveal, a Contextual Tools pattern, to reveal a set of tools when the user hovers over a listing.
- Second, it uses the Detail Inlay pattern to show a carousel of photos when the user clicks on the "View photos" link. And finally, it uses a Detail Overlay to blow up a thumbnail when clicked on.
- Compare this to the traditional approach, one that requires the user to navigate from the listing page to a photo page and back again. The Roost team actually expended a Herculean effort in setting up this convenience, as it is dealing with hundreds of MLS listings with different contractual requirements for displaying real estate photos.
- The Roost team worked out the difficulties behind the scenes to create a nice user experience.Tabs Lest we forget, there are some very traditional interface elements that can be used to inlay details.
- *Tabs*, for instance, can be used as a Detail Inlay. Instead of moving the user from page to page (site navigation), tabs can be used to bring in content within the page, keeping the user in the page.

Considerations

- In Figure 6-11 you can see not just one type of tab for Yahoo!'s home page, but three! Three different tab styles and interactions can be confusing. In fact, user-testing revealed this to be the case.
- However, the design sidestepped some of the problem by giving the three tab areas each a different visual style. Let's look at each type of tab interaction

Traditional tabs

- The Featured, Entertainment, Sports, and Video sections are styled as a traditional tab notebook (Figure 6-11, callout 1). Going from section to section requires the user to click on the corresponding tab.
- Most studies on tabs find that the first and second tabs get the most activity and the subsequent tabs get clicked less frequently. This is the rationale for placing the "Featured" content first.

Content tabs

- Each section has four featured stories. In this example, the "Cute hamster" is tied to the thumbnail and story lead (Figure 6-11, callout 2). Clicking on a story lead switches the content inside the tab. Effectively this is a secondary tab control, but visually it appears as a content story. By switching in the story without leaving the page, the user can get a peek at the top stories.

Personal assistant tabs

- On the right side of the page, Yahoo! provides what it calls a *Personal Assistant*. Each tab in this area (Mail, Messenger, etc.) is activated by hovering over the tab.
- In our example the mouse is hovered over the Mail tab (Figure 6-11, callout 3) and it automatically expands open.
- Clicking on the link actually takes the user to Yahoo! Mail. The three types of tabs vary greatly, visually and interactively. However, Yahoo! is able to pull this off because:

Normal users of Yahoo! will discover these interactions over time.

• Creating the contrast makes a more visually compelling interface, as well as making the interaction feel deeper (inviting exploration).

• It is a great improvement over the old Yahoo! home page, which was completely static. Every link took the user to a different page. Keeping users on the page until they are ready to leave actually creates a happier user experience.

6. Explain in detail about virtual pages and its types?OVERVIEW

- Virtual Scrolling
- Inline Paging
- Scrolled Paging
- Panning
- Zoomable User Interface

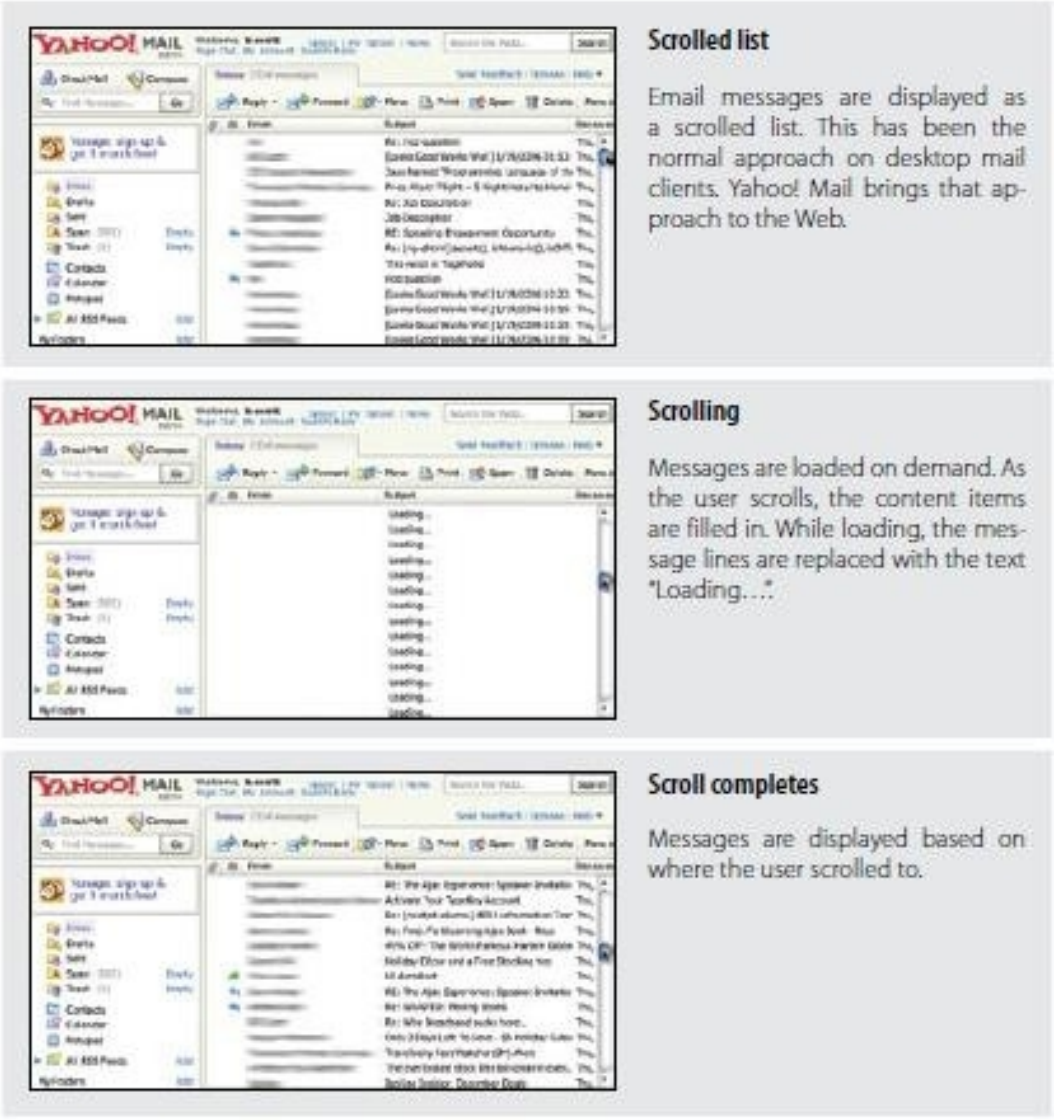
- Overlays allow you to bring additional interactions or content in a layer above the current page. Inlays allow you to do this within the page itself. However, another powerful approach to keeping users engaged on the current page is to create a *virtual page*.
- That is to say, we create the illusion of a larger virtual page.

Patterns that support virtual pages include:

- Virtual Scrolling
- Inline Paging
- Scrolled Paging
- Panning
- Zoomable User Interface

Virtual Scrolling

- The traditional Web is defined by the "page." In practically every implementation of websites (for about the first 10 years of the Web's existence) pagination was the key way to get to additional content. Of course, websites could preload data and allow the user to scroll through it. However, this process led to long delays in loading the page.
- So most sites kept it simple: go fetch 10 items and display them as a page and let the user request the next page of content. Each fetch resulted in a page refresh.
- The classic example of this is Google Search. Each page shows 10 results. Moving through the content uses the now-famous Google pagination control (Figure 7-1).
- Another approach is to remove the artificial page boundaries created by paginating the data with Virtual Scrolling. In Yahoo! Mail, mail messages are displayed in a scrolled list that loads additional messages on demand as the user scrolls (Figure 7-2).



**

Considerations

- In some sense Virtual Scrolling turns the scrollbar into a "pagination control." But instead of a page refresh, the messages are seamlessly brought into the message-list pane, giving the illusion of a larger virtual space.

Desktop-style applications

- In testing Virtual Scrolling, Yahoo! found that users naturally understood the scrolling paradigm, most likely because they were already accustomed to this feature on desktop mail clients.
- Since the Yahoo! Mail Web application looks very similar to desktop web applications, the expectation for scrolling already exists in the user's mind.

Scrolled list

12,500,000 image results are represented as a scrolled list. Obviously there is no way to accurately represent that many items in a list with a scrollbar. Notice the scrollbar shows size relative to the amount of data that has been loaded.

Scrolling

By scrolling into the area where results have not been loaded, images are initially represented as gray squares to indicate that they are currently not loaded.

As each image is loaded it replaces the gray squares.

At the top, the start and end range of the visible images is displayed ("Images 46–70 of 12,500,00").

Scroll completes

Image results are fully loaded, and the scrollbar is updated to reflect where this page is in relation to the previously loaded content.

**

Loading status

- There are a few downsides to the Yahoo! Mail version of Virtual Scrolling. First, if the loading is slow, it spoils the illusion that the data is continuous.
- Second, since the scrollbar does not give any indication of where users are located in the data, they have to guess how far down to scroll. A remedy would be to apply a constantly updating status while the user is scrolling.

Progressive loading

- Microsoft has applied Virtual Scrolling to its image search. However, it implements it in a different manner than Yahoo! Mail.
- Instead of all content being virtually loaded (and the scrollbar reflecting this), the scrollbar reflects what has been loaded. Scrolling to the bottom causes more content to load into the page (Figure 7-3).

Inline Paging

- What if instead of scrolling through content we just wanted to make pagination feel less like a page switch? By only switching the content in and leaving the rest of the page stable, we can create an Inline Paging experience.
- This is what Amazon's Endless.com site does with its search results (Figure 7-6).

**

Paginated results

Searching for "Men's athletic shoes" displays a traditional-looking set of search results. The pagination controls are familiar (shown as an exploded callout).

Inline Paging

Clicking to "page 3" causes just the search results area to update with the third page of results.

Page remains stable

The rest of the page stays stable when the new "page" of results is brought into view.

Considerations

There are some issues to consider with Inline Paging.

In-page update

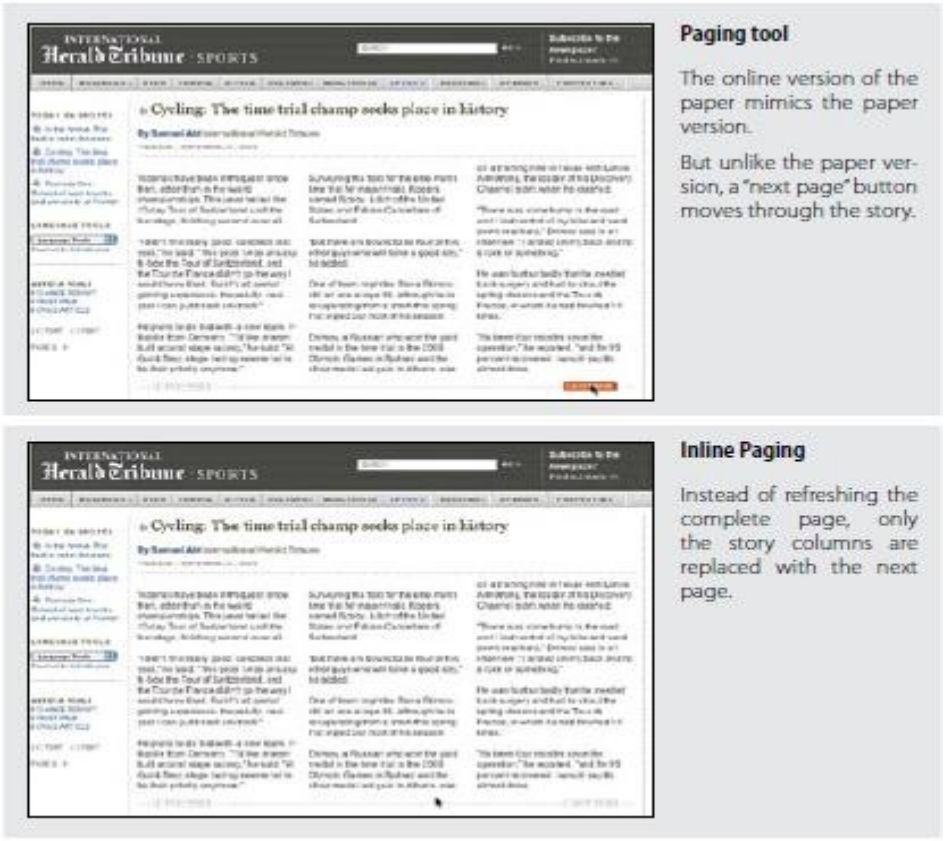
- Endless.com provides the normal pagination controls. But instead of the whole page refreshing, only the results area is updated. Keeping the context stable creates a better flow

experience. With Inline Paging it feels like the user never leaves the page even though new virtual pages of results are being brought into view.

Natural chunking

- Inline Paging can also be useful when reading news content online. The *International Herald Tribune* applied this as a way to page through an article while keeping the

surrounding context visible at all times (Figure



Back button

- The biggest issue with Inline Paging is whether the back button works correctly. One criticism of Endless.com is that if the user pages through search results and then hits the back button, it jumps to the page just before the search.
- This unexpected result could be fixed by making the back button respect the virtual pages as well. This is the way Gmail handles the back button.* Clicking back moves you through the virtual pages.

Interactive content loading

- The iPhone employs inline paging when displaying search results in the iTunes store (Figure 7-9).Each tap of the "Load 25 More Results…" button loads 25 more songs.
- The number of songs loaded is cumulative. With the first tap there are now 50 songs; the third tap, 75 songs; and so on. Normally no scrollbar is shown. However, if the user places a finger on the list to move up or down, the scrollbar is displayed (Figure 7-10). Since the finger is

the "scroller," the scrollbar becomes just an indicator of how many songs are loaded and where the user is in the list.

Scrolled Paging: Carousel

- Besides Virtual Scrolling and Virtual Paging, there is another option. You can combine both scrolling and paging into Scrolled Paging. Paging is performed as normal. But instead the content is "scrolled" into view.
- The Carousel pattern takes this approach. A Carousel provides a way to page-in more data by scrolling it into view.
- On one hand it is a variation on the Virtual Scrolling pattern. In other ways it is like Virtual Paging since most carousels have paging controls. The additional effect is to animate the scrolled content into view.

**

Timeline

The top section provides a navigation control through various Underground articles. "Previously" and "Up Next" indicate where the user can go.

Animation

Animation reinforces the fact that the articles are from the past (the content moves in from the left to the right).

Considerations

There are some issues to consider when using Scrolled Paging.

Time-based

- Carousels work well for time-based content. Flickr employs a Carousel to let users navigate back and forth through their photo collection (Figure 7-12

Virtual Panning

- One way to create a virtual canvas is to allow users the freedom to roam in two-dimensional space. A great place for Virtual Panning is on a map. Google Maps allows you to pan in any

direction by clicking the mouse down and dragging the map around (Figure 7-14

Considerations

There are some issues to consider while using Virtual Panning.

Natural Visual Construct

- When we discussed dragging routes in Google Maps, we pointed out that drag and drop worked well since it fit with the natural visual representation of routes on the map (see Figure 2-26).
- In the same way, panning around in a map is a natural visual metaphor. Extending the visual space to a larger virtual space is a natural fit.

Gestures

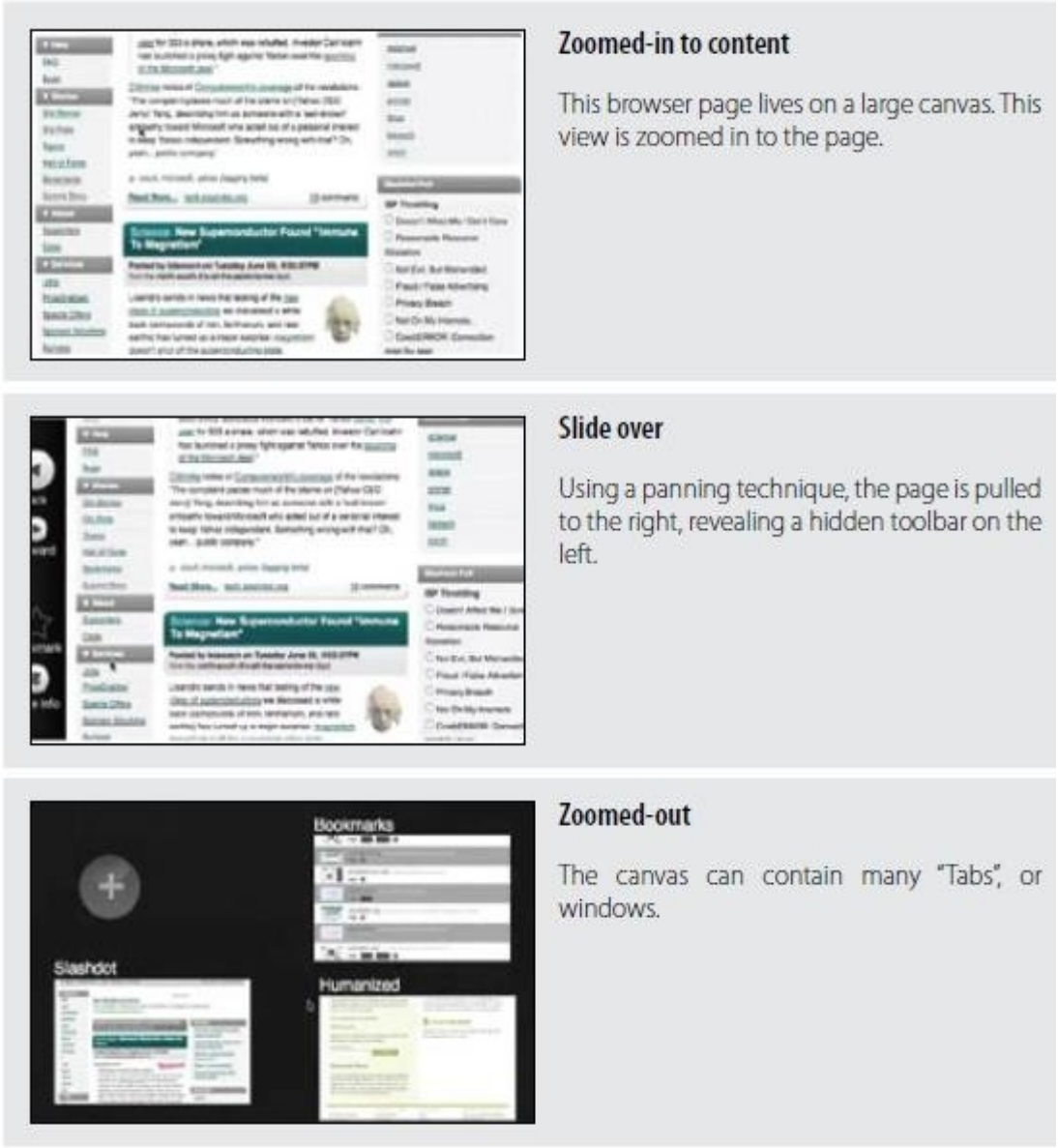
- Besides map applications, the idea of Virtual Panning has been extended to other devices thanks to gesture-based interfaces.*
- With the introduction of the iPhone, the user can simply "flick" through weather locations, photos, or an iTunes playlist.

Zoomable User Interface

- A Zoomable User Interface (ZUI) is another way to create a virtual canvas. Unlike panning or flicking through a flat, two-dimensional space, a ZUI allows the user to also zoom in to elements on the page.
- This freedom of motion in both 2D and 3D supports the concept of an infinite interface. Practically speaking, ZUIs have rarely been available in everyday software applications, much less on the Web. But with more advanced features added to Flash and the advent of Silverlight, this type of interface is starting to emerge and may be commonplace in the not-too-distant future.

Considerations

- Hard Rock Café has a large rock and roll memorabilia collection. Recently, it digitized photos of the collection and placed them online for virtual viewing.
- The memorabilia application uses a ZUI interface to move around from artifact to artifact and zoom in to see details on each item (Figure 7-15).



**

Paging Versus Scrolling

- ❖ Leading web designers and companies have taken different approaches to solving thesame problems. Yahoo! Mail chose Virtual Scrolling. Gmail chose Inline Paging.
- ❖ How do you choose between paging and scrolling? While there are no hard and fast rules, here are some things to consider when making the decision:

• When the data feels "more owned" by the user—in other words, the data is not transient but something users want to interact with in various ways. If they want to sort it, filter it, and so on, consider Virtual Scrolling (as in Yahoo! Mail).

• When the data is more transient (as in search results) and will get less and less relevant the further users go in the data, Inline Paging works well (as with the iPhone).

• For transient data, if you don't care about jumping around in the data to specific sections, consider using Virtual Scrolling (as in Live Image Search).

• If you are concerned about scalability and performance, paging is usually the best choice. Originally Microsoft's Live Web Search also provided a scrollbar. However, the scrollbar increased server-load considerably since users are more likely to scroll than page.
• If the content is really continuous, scrolling is more natural than paging.
• If you get your revenue by page impressions, scrolling may not be an option for your business model.
• If paging causes actions for the content to become cumbersome, move to a scrolling model. This is an issue in Gmail. The user can only operate on the current page. Changing items across page boundaries is unexpected. Changing items in a continuous scrolled list is intuitive.

8. Describe in detail about process flow and its types?OVERVIEW:

Interactive Single-Page Process
- Inline Assistant Process
- Configurator Process
- Overlay Process
- Static Single-Page Process

- In the last three chapters we've been discussing the principle *Stay on the Page*. Sometimes tasks are unfamiliar or complicated and require leading the user step-by-step through a Process Flow.
- It has long been common practice on the Web to turn each step into a separate page. While this may be the simplest way break down the problem, it may not lead to the best solution. For some Process Flows it makes sense to keep the user on the same page throughout the process.

Google Blogger

- The popular site Google Blogger generally makes it easy to create and publish blogs. One thing it does not make easy, though, is deleting comments that others may leave on your blog.
- This is especially difficult when you are the victim of hundreds of spam comments left by nefarious companies hoping to increase their search ranking. Blogger forces you to delete these comments through a three-step process. Each step is an individual page, all punctuated with a page refresh (Figure 8-1).

or more spam comments. That means that I had to delete more than 400 spam comments. Given the way Google Blogger implemented comment deleting, I had to follow these steps for each comment on each blog article:

1. Scroll to find the offending comment.

2. Click the trash icon to delete the comment.
3. After page refreshes, click the "Remove Forever" checkbox.
4. Click the "Delete Comment" button.
5. After the page refreshes, click the link to return to my blog article.
6. Repeat steps 1–5 for each article with spam comments.

- ❖ It took 1,600 clicks, 1,200 page refreshes, 400 scroll operations, and several rid myself of all of the spam comments.
- ❖ If the delete action could have been completed in the same page as the comments, that would have eliminated hundreds of clicks and well over a thousand page refreshes, and scrolling would have been all but eliminated.
- ❖ I would not have wasted all the mental energy to reorient myself after each page transition. And I would have been a much happier man. This is a common interaction flow on the Web.
- ❖ It turns out to be simpler to design and implement a process as a series of pages rather than a single interactive space.

The Magic Principle

- ❖ Alan Cooper discusses a wonderful technique for getting away from a technology-driven approach and discovering the underlying mental model of the user.
- ❖ He calls it the "magic principle."* Ask the question, "What if when trying to complete a task the user could invoke some magic?" For example, let's look at the problem of taking and sharing photos.

The process for this task breaks down like this:

- Take pictures with a digital camera.
- Sometime later, upload the photos to a photo site like Flickr. This involves:
  - Finding the cable.
  - Starting iTunes.
  - Importing all photos.

- ❖ Using a second program, such as Flickr Uploadr, to upload the photos to Flickr. Copying the link for a Flickr set (which involves first locating the page for the uploaded set).
- ❖ Send the link in email to appropriate friends.

If some magic were invoked, here is how it might happen:

- The camera would be event-aware. It would know that is your daughter's eighth birthday.
- When finished taking pictures of the event, the camera would upload the pictures to Flickr.
- Flickr would notify family and friends that the pictures of the birthday party are available.

- ❖ Thinking along these lines gets some of the artifacts out of the way. Of course the magic could be taken to the extreme: just eliminate the camera altogether! But by leaving some elements in the equation, the potentially unnecessary technology pieces can be exposed.
- ❖ How about the cable? What if the camera could talk magically to the computer?
- ❖ This kind of thinking led to some recent products that allow users to upload photos automaticallyfrom their digital camera to their favorite photo site. The camera's memory card actually contains a Wi-Fi connection, giving it direct access to a photo-upload service.*

In this chapter we will apply a little magic, hopefully get rid of the nuisance of page transitions where possible, and in the end treat the user to a better experience. Specifically, we will look at these Process Flow patterns:

- Interactive Single-Page Process
- Inline Assistant Process
- Configurator Process
- Overlay Process

• Static Single-Page Process

Interactive Single-Page Process

- Consumer products come in a variety of shapes, sizes, textures, colors, etc. Online shoppers will not only have to decide that they want shoes, but do they want blue suede shoes?
- And what size and width do they want them in? In the end the selection is constrained by the available inventory.
- As the user makes decisions, the set of choices gets more and more limited. This type of product selection is typically handled with a multipage workflow.
- On one page, the user selects a shirt and its color and size. After submitting the choice, a new page is displayed. Only when the user arrives at this second page does he find outthat the "true navy" shirt is not available in the medium size. The Gap accomplishes this kind of product selection in a single page (Figure 8-3) using**Interactive Single-Page Process.**

Considerations

There are some issues to consider when using an Interactive Single-Page Process.

Responsiveness

- The user's taste preference comes first. Either the color or the size can be chosen. If the item is out of stock for any color/size combination, it is displayed as unavailable (by showing the color or size as disabled).
- By placing this process in a few simple interactions, the user can quickly find something available to buy. With any online shopping experience, the potential for the user to bail out is a real concern. In-place interactions like this reduce these *bailout moments*.

Selecting sizes

Sizes are shown in a drop-down menu. The sizes that are not available for the selected color are disabled.

If a disabled size is selected, the unavailable colors will be displayed in a dimmed state.

Selecting color

The colors available for this size neck are clearly shown in the color selection palette.

Unavailable

If a combination that is unavailable is selected, a clear warning is displayed and no links to purchase are made available.

Benefits

- Adobe calls out the Broadmoor one-page reservation interface in its Adobe Showcase.* It states the benefits of this method:
  - Reduces entire reservation process to a single screen.
  - Reduces the number of screens in the online reservation process from five to one. Other online reservation applications average 5 to 10 screens.
  - Seventy-five percent of users choose OneScreen in favor of the HTML version.
  - Allows users to vary purchase parameters at will and immediately view results.
  - Reduces the time it takes to make a reservation from at least three minutes to less than one.

- Additionally, Adobe notes that conversion rates (users who make it through the reservation process) are much higher with the Interactive Single-Page Process.

Inline Assistant Process

- Another common place where multiple pages are used to complete a process is when adding items to a shopping cart. As mentioned earlier, Amazon provides the typical experience (Figure 8-2).
- So what magic can we apply to move this from a multi-page experience to a single-page experience? Instead of thinking about the cart as a process, we can think about it as a real-world object.
- Given this mindset, the cart can be realized in the interface as an object and be made available on the page.
- The Gap employed an Inline Assistant Process pattern for its shopping cart when it re-launched its site a few years back .

"Add to Bag" button in context

"Add to Bag" is available in the context of selecting the sweater pattern and size.

Inline shopping cart

The shopping cart is not a separate page. Instead it is an interface element that is tied to each page.

It is always available as a drop-down shade. It is readily available to add items to or to open up and manage at any time.

Considerations

There are some things to consider when using the Inline Assistant Process.

Quick and easy

- The Gap integrates the shopping cart into its entire site as a drop-down shade. In fact, the Gap, Old Navy, Banana Republic, and PiperLime all share the same

Inline Assistant

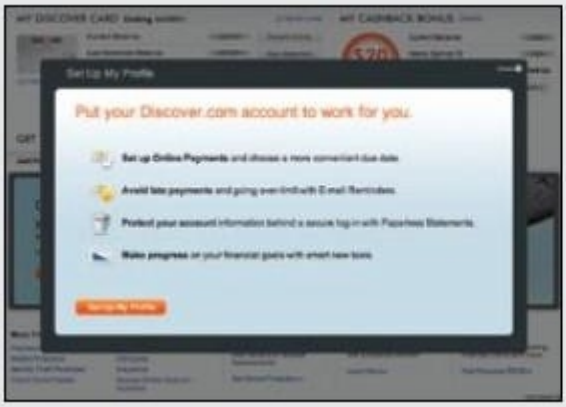
- Process-style shopping cart. The Gap is betting that making it quick and easy to add items to the cart across four stores will equal more sales. Additional step Amazon, on the other hand, is betting on its recommendation engine.
- By going to a second page, Amazon can display other shirts like the one added—as well as advertise the Amazon.com Visa card (Figure 8-8).

Blending quick and easy with the additional step

- Is there a way to combine both the single-page experience and the recommendation experience? This is what Netflix does when a user adds movies to his shipping queue

Dialog Overlay Process

- As mentioned before, any page switch is an interruption to the user's mental flow. In addition, any context switch is a chance for a user to leave the site.
- We seek an experience that has as little mental friction as possible. But sometimes the step-by-step flow is necessary. The Netflix approach just described (Figure 8-9) uses a Dialog Overlay Process to encapsulate a multi-step flow inside a Dialog Overlay.
- We looked at Overlays in detail in Chapter 5. Overlays allow us to keep the context of the page yet present a virtual space to conduct a conversation with the user.
- Discover.com recently expanded its account section with a more detailed profile.
- The profile captures things like your payment date, mobile fraud alerts, paperless statements, and general contact information (Figure 8-11). The overlay pops up when you first enter your account.



Invitation to set up profile

When entering an account, a **Dialog Overlay** is presented with an invitation to set up a profile.

A **Lightbox Effect** is used to focus the user on the task of profile setup. The pleasing visuals add a level to the engagement factor.

Multiple steps

Each step is presented is presented as a separate "page" within the overlay.

Each "page" has a simple, clear call to action.

Final step

Filling out the contact information is left until the end. If this step occurs too early in the process, it might make the user think all the steps will be this involved.

Considerations

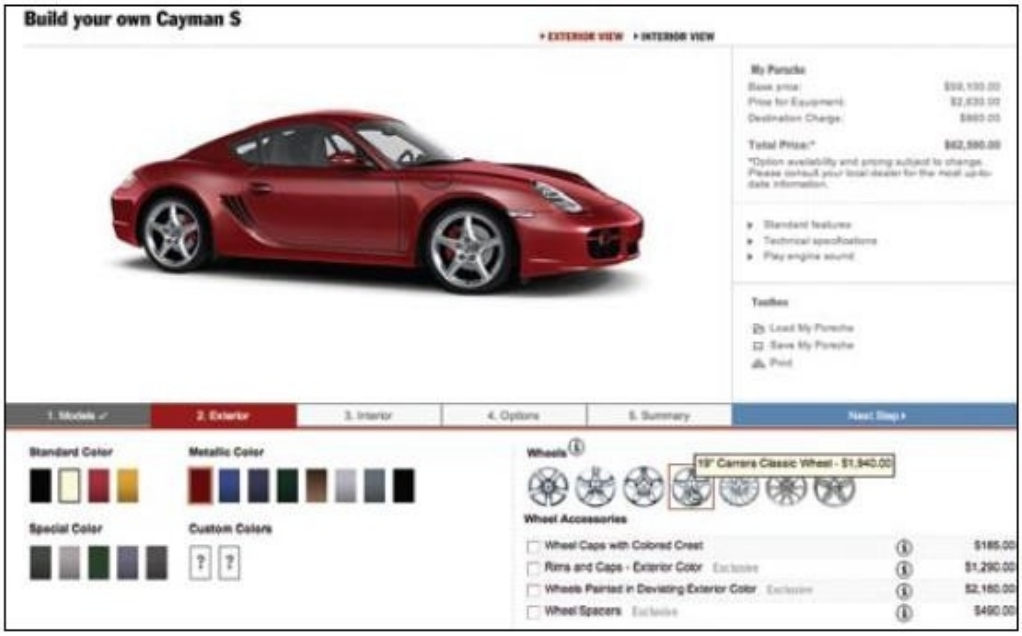
- There are some issues to consider when using the Dialog Overlay Process. Making it feel lightweight The Lightbox Effect is nice because it focuses the user on the task.
- But what is smart about the flow is the first page is colorful and rendered in a simple, pleasing manner. The process looks like it might be simple and feels engaging.
- The in-between steps are usually a single action. There is a whole page dedicated just to selecting the due date for payments (Figure 8-12).
- Making the in-between steps clear and visually appealing with a single call to action makes the process feel lightweight. The last step is the biggest.
- By this time the user has committed to the process to some degree. Most of the user information is already filled in from the account, so the step does not feel too involved.
- But imagine if this step were first. It could cause users to more frequently bail out from the process, thinking that each step would be as involved as the first one.

Clear status

- The other interesting touch in this example is providing an indication of the number of steps (Figure 8-13). There are any number of ways to indicate this information; the important thing is to give some indication of what the users are dealing with when they start. Usually three steps are ideal. In this case, there are five steps. But as we mentioned, the early steps are single actions.

Configurator Process

- Sometimes a Process Flow is meant to invoke delight. In these cases, it is the engagement factor that becomes most important. This is true with various Configurator Process interfaces on the Web.
- We can see this especially at play with car configurators. Porsche provides a configurator that allows users to build their own Porsche (Figure 8-15).



Considerations

There are some issues to consider when using a Configurator Process.

Immediate feedback

- In the case of the Porsche site, when the user clicks on various body colors, the car is reconfigured in the selected color and rendered in photorealistic shading. Most configurators allow a 360-degree view of the car as it is configured.

Static Single-Page Process

- The Apple example illustrates another way to get rid of multiple pages in a Process Flow. Just put the complete flow on one page in a Static Single-Page Process. The user sees all the tasks needed to complete the full process.
- This can be both good and bad. Seeing just one step to complete the process can encourage users to finish the task. But if the single step seems too long or too confusing, the user will most likely bail out of the process early.
- In other words, if placing all the tasks on a single page is enough to cause the user to bail out, it is not a good idea.
- In the case of the Apple store, each item is optionally set, and it's just a single click to include or exclude an item from the purchase.

Figure 8-17. *eBay's simplified one-page flow*

Considerations

There are some issues to keep in mind when using a Static Single-Page Process.

Making it feel lightweight

- In this Static Single-Page Process, many options are defaulted and a simplified form is presented to the user.
- Multiple pages are compressed into a single page. Of course a long page like this can also be daunting. But eBay did a good job of getting the essentials into a single page.
- Each step is clearly called out with a clear border and a large step number. In addition, color is used to further simplify the steps.
- The two steps in green are about the item the user is selling. The last two steps in blue are about money. The color coding gives the impression there are only four steps total: item, photo, description, and pricing. Multiple pages are not necessarily evil The eBay example (Figures 8-16 through 8-18) illustrates that there is more than one way to deal with a step-by-step Process Flow.
- For a very complex flow, a Static Single-Page Process may work well. Sometimes it is good to break what could be a Static Single-Page Process into a multipage process.
- Multiple pages can provide the natural chunking needed. They say "You are done with that step, now move onto the next." Netflix has a problem-reporting interface that does just that. When reporting a scratched disc, clicking on the "DVD is damaged…" link takes the user to a secondary page (Figure 8-20).

First page

The first page focuses the users on just deciding what the issue is.

Final page

The second page allows them to describe the details of that problem.