ARUNAI ENGINEERING COLLEGE

DEPARTMENT OF CSE

IV YEAR - VII SEMESTER

CS8079 – HUMAN COMPUTER INTERACTION (R2017)

UNIT I FOUNDATIONS OF HCI

The Human: I/O channels – Memory – Reasoning and problem solving; The computer: Devices – Memory – processing and networks; Interaction: Models – frameworks – Ergonomics – styles – elements – interactivity- Paradigms. Case Studies

FOUNDATIONS OF HCI

The h*uman–computer interaction* is the systematic study of human performance. Ergonomists have been concerned primarily with the physical characteristics of machines and systems, and how these affect user performance. Human Factors incorporates these issues and more cognitive issues as well. Both of these disciplines are concerned with user performance in the context of any system, whether computer, mechanical or manual. As computer use became more widespread, an increasing number of researchers specialized in studying the interaction between people and computers, concerning themselves with the physical, psychological and theoretical aspects of this process. This research originally went under the name *man–machine interaction*, but this became *human–computer interaction* in recognition of the particular interest in computers and the composition of the user population.

HCI involves the design, implementation and evaluation of interactive systems in the context of the user's task and work. By ***user*** we may mean an individual user, a group of users working together, or a sequence of users in an organization, each dealing with some part of the task or process. The user is whoever is trying to get the job done using the technology. By ***computer*** we mean any technology ranging from the general desktop computer to a large-scale computer system, a process control system or an embedded system. The system may include non-computerized parts, including other people. By ***interaction*** we mean any communication between a user and computer, be it direct or indirect. **Direct interaction** involves a dialog with feedback and control throughout performance of the task. **Indirect interaction** may involve batch processing or intelligent sensors controlling the environment. The important thing is that the user is interacting with the computer in order to accomplish something.

WHO IS INVOLVED IN HCI?

The ideal designer of an interactive system would have expertise in a range of topics: psychology and cognitive science to give her knowledge of the user's perceptual, cognitive and problem-solving skills; ergonomics for the user's physical capabilities; sociology to help her under-stand the wider context of the interaction; computer science and engineering to be able to

build the necessary technology; business to be able to market it; graphic design to produce an effective interface presentation; technical writing to produce the manuals, and so it goes on. There is obviously too much expertise here to be held by one person. So, we must be pragmatists rather than theorists: we want to know how to apply the theory to the problem rather than just acquire a deep understanding of the theory. Our goal, then, is to be multi-disciplinary but practical.

THEORY AND HCI

There is an underlying principle that forms the basis of the views on HCI, and claims that people use computers to accomplish work. This outlines the three major issues of concern: **the people, the computers and the tasks** that are performed. The system must support the user's task, which gives us a **fourth focus, usability**: if the system forces the user to adopt an unacceptable mode of work then it is not usable.

The word 'task' or the focus on accomplishing 'work' is also problematic when we think of areas such as domestic appliances, consumer electronics and e-commerce. There are three 'use' words that must all be true for a product to be successful; it must be:

useful – accomplish what is required: play music, cook dinner, format a document;

usable – do it easily and naturally, without danger of error, etc.;

used – make people want to use it, be attractive, engaging, fun, etc

The other issues like motivation, enjoyment and experience are increasingly important.

The most impressive structures, the most beautiful buildings, the innovative and imaginative creations that provide aesthetic pleasure, all require inventive inspiration in design and a sense of artistry, and in this sense the discipline is a craft. So it is for HCI, beautiful and/or novel interfaces are artistically pleasing *and* capable of fulfilling the tasks. Innovative ideas lead to more usable systems, but in order to maximize the potential benefit from the ideas, we need to understand not only that they work, but how and why they work. we have to pro-vide them with an understanding of the concepts involved, a scientific view of the reasons why certain things are successful whilst others are not, and then allow their creative nature to feed off this information: creative flow, underpinned with science; or maybe scientific method, accelerated by artistic insight. The truth is that HCI is required to be both a craft and a science in order to be successful.

The Human

Introduction

The human, the ***user*****,** is, after all, the one whom computer systems are de-signed to assist. The requirements of the user should therefore be our first priority.

The aspects of **cognitive psychology** have a bearing on the use of computer systems, how humans perceive the world around them, how they store and process information and solve problems, and how they physically manipulate objects.

The ***Model Human Processor***, which is a simplified view of the human processing involved in interacting with computer systems. The model comprises three subsystems: **the perceptual system**, handling sensory stimulus from the outside world, **the motor system**, which controls actions, and **the cognitive system**, which provides the processing needed to connect the two. Each of these subsystems has its own processor and memory, although obviously the complexity of these varies depending on the complexity of the tasks the subsystem has to perform. The model also includes a number of *principles of operation* which dictate the behavior of the systems under certain conditions.

In conventional computer system, the three components are input–output, memory and processing. In the human, dealing is with an intelligent information-processing system, and processing therefore includes problem solving, learning, and, consequently, making mistakes. The human, unlike the computer, is also influenced by external factors such as the social and organizational environment, and we need to be aware of these influences as well. These factors are ignored for now and concentration is on the human's information processing capabilities only.

Input– Output Channels

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.

Input in the human occurs mainly through the senses and output through the motor control of the effectors. There are five major senses: sight, hearing, touch, taste and smell. Of these, the first three are the most important to HCI. Taste and smell do not currently play a significant role in HCI.

Similarly there are a number of effectors, including the limbs, fingers, eyes, head and vocal system. In the interaction with the computer, the fingers play the primary role, through typing or mouse control, with some use of voice, and eye, head and body position.

Vision

Human vision is a highly complex activity with a range of physical and perceptual limitations, yet it is the primary source of information for the average person. We can roughly divide visual perception into two stages: the physical reception of the stimulus from the outside world, and the processing and interpretation of that stimulus. On the one hand the physical properties of the eye and the visual system mean that there are certain things that cannot be seen by the human; on the other the interpretative capabilities of visual processing allow images to be constructed from incomplete information

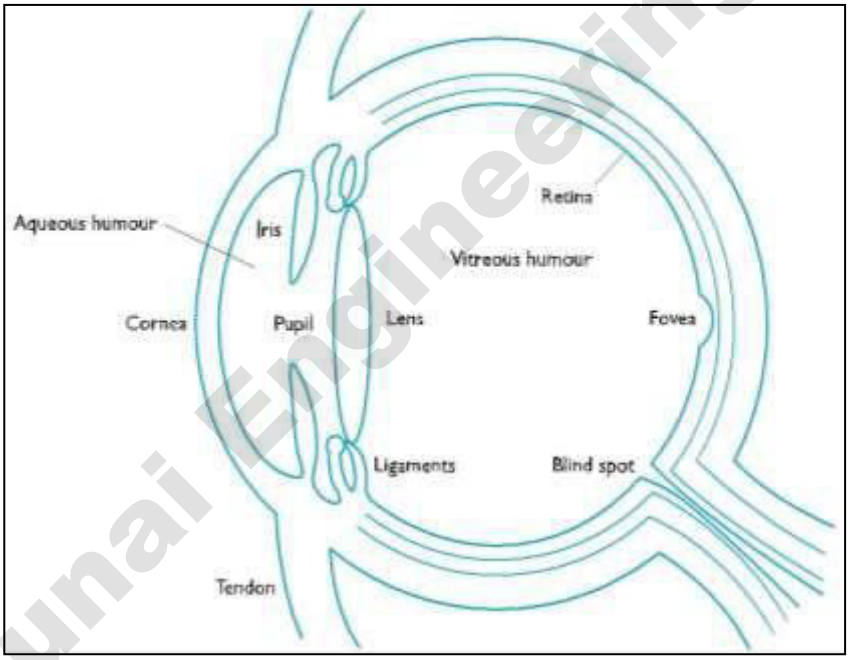
The human eye

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 focused 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 (Figure 1.1). The *cornea* and *lens* at the front of the eye focus the light into a 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 from a 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. There are approximately 120 million rods per eye which are mainly situated towards the edges of the retina. Rods therefore dominate peripheral vision.

Cones are the second type of receptor in the eye. They are less sensitive to light than the rods and can therefore tolerate more light. There are three types of cone, each sensitive to a different wavelength of light. This allows color vision. The eye has approximately 6 million cones, mainly concentrated on the *fovea*, a small area of the retina on which images are fixated.



Although the retina is mainly covered with photoreceptors there is one *blind spot* where the optic nerve enters the eye. The blind spot has no rods or cones, yet our visual system compensates for this so that in normal circumstances we are unaware of it.

The retina also has specialized nerve cells called *ganglion cells*. There are two types: X-cells, which are concentrated in the fovea and are responsible for the early detection of pattern; and Y-cells which are more widely distributed in the retina and are responsible for the early detection of movement. The distribution of these cells means that, while we may not be

able to detect changes in pattern in peripheral vision, we can perceive movement

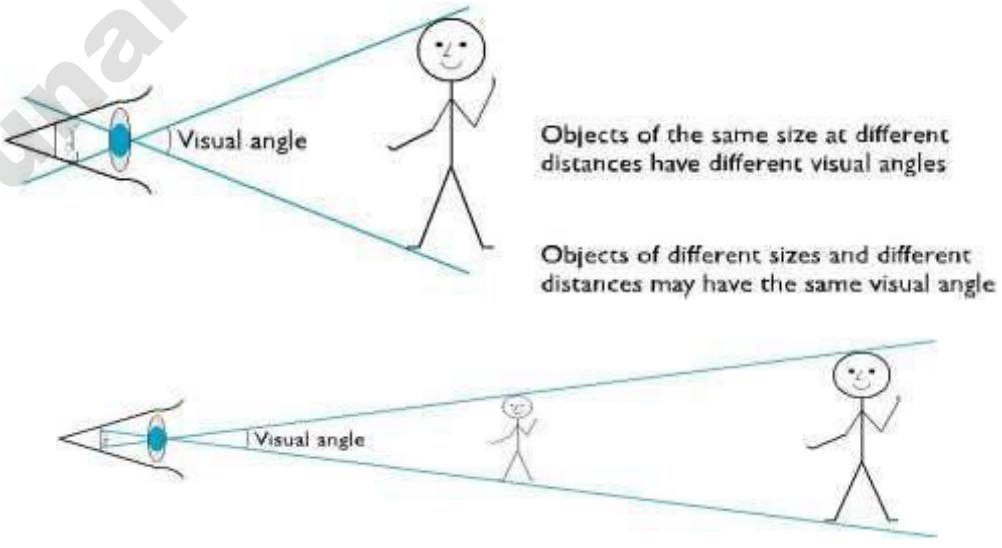
Visual perception

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. The factors that are crucial to the design of effective visual interfaces are how we perceive size and depth, brightness and color.

Perceiving size and depth Imagine you are standing on a hilltop. Describing such a scene the notions of size and distance predominate. To understand how does the eye perceive size, depth and relative distances, we must consider how the image appears on the retina. Reflected light from the object forms an upside-down image on the retina. The size of that image is specified as a *visual angle*. Figure 1.2 illustrates how the visual angle is calculated.

If we were to draw a line from the top of the object to a central point on the front of the eye and a second line from the bottom of the object to the same point, the visual angle of the object is the angle between these two lines. Visual angle is affected by both the size of the object and its distance from the eye. Therefore if two objects are at the same distance, the larger one will have the larger visual angle. Similarly, if two objects of the same size are placed at different distances from the eye, the furthest one will have the smaller visual angle. The visual angle indicates how much of the field of view is taken by the object. The visual angle measurement is given in either degrees or *minutes of arc*, where 1 degree is equivalent to 60 minutes of arc, and 1 minute of arc to 60 seconds of arc.

So, an object's visual angle affect our perception of its size. First, if the visual angle of an object is too small we will be unable to perceive it at all. ***Visual acuity*** is the ability of a person to perceive fine detail. A number of measurements have been established to test visual acuity, most of which are included in standard eye tests. For example, a person with normal vision can detect a single line if it has a visual angle of 0.5 seconds of arc. Spaces between lines can be detected at 30 seconds to 1 minute of visual arc. These represent the limits of human visual acuity.



Our perception of an object's size remains constant even if its visual angle changes. So a person's height is perceived as constant even if they move further from you. This is the ***law of size constancy***, and it indicates that our perception of size relies on factors other than the visual angle.

One of these factors is our perception of depth. If we return to the hilltop scene there are a number of *cues* which we can use to determine the relative positions and distances of the objects which we see. If objects overlap, the object which is partially covered is perceived to be in the background, and therefore further away. Similarly, the size and height of the object in our field of view provides a cue to its distance.

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

The visual system itself also compensates for changes in brightness. In dim lighting, the rods predominate vision. Since there are fewer rods on the fovea, objects in low lighting can be seen less easily when fixated upon, and are more visible in peripheral vision. In normal lighting, the cones take over.

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. Intensity is the brightness of the color, and saturation is the amount of whiteness in the color. By varying these two, we can perceive in the region of 7 million different colors.

The eye perceives color because the cones are sensitive to light of different wave-lengths. There are **three different types of cone**, each sensitive to a different color (blue, green and red). Color vision is best in the fovea, and worst at the periphery where rods predominate.

The capabilities and limitations of visual processing:

Visual processing compensates for the movement of the image on the retina which occurs as we move around and as the object which we see moves. Although the retinal image is moving, the image that we perceive is stable. Similarly, color and brightness of objects are perceived as constant, in spite of changes in luminance.

This ability to interpret and exploit our expectations can be used to resolve ambiguity.

Figure 1.3:Ambiguous shape? **Figure 1.4:BABC** **Figure 1.4:12 13 14**

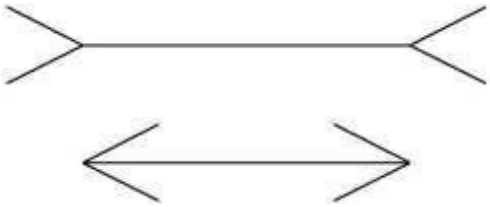
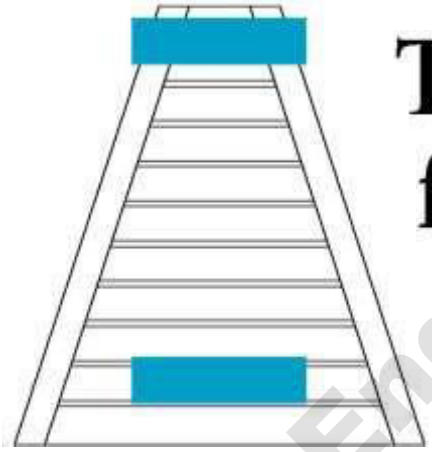


Figure 1.6 : The Muller–Lyer illusion – which line is longer?

Consider Figure 1.6. Which line is longer? Most people when presented with this will say that the top line is longer than the bottom. In fact, the two lines are the same length.

A similar illusion is the Ponzo illusion (Figure 1.7). Here the top line appears longer, owing to the distance effect, although both lines are the same length. These illusions demonstrate that our perception of size is not completely reliable.

Another illusion created by our expectations compensating an image is the proof-reading illusion. Read the text in Figure 1.8 quickly. What does it say? Most people reading this rapidly will read it correctly, although closer inspection shows that the word 'the' is repeated in the second and third line.



The quick brown fox jumps over the the lazy dog.

Figure 1.8 Is this text correct?

Figure 1.7:The Ponzo illusion – are these the same size?

Reading :

During reading, the eye makes jerky movements called *saccades* followed by fixations. Perception occurs during the fixation periods, which account for approximately 94% of the time elapsed. The eye moves backwards over the text as well as forwards, in what are known as *regressions*. If the text is complex there will be more regressions. Adults read approximately 250 words a minute. It is unlikely that words are scanned serially, character by character, since experiments have shown that words can be recognized as quickly as single characters. Instead, familiar words are recognized using word shape.

Hearing

The sense of hearing is often considered secondary to sight. 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 near my house, and which direction it is traveling in.

The human ear

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 is the 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 sensitive middle 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.

The **middle ear** is a small cavity connected to the outer ear by the *tympanic membrane*, or ear drum, and to the inner ear by the *cochlea*. Within the cavity are the *ossicles*, the smallest bones in the body. Sound waves pass along the auditory canal and vibrate the ear drum which in turn vibrates the ossicles, which transmit the vibrations to the cochlea, and so into the inner ear. This 'relay' is required because, unlike the air-filled outer and middle ears, the inner ear is filled with a denser cochlean liquid. If passed directly from the air to the liquid, the transmission of the sound waves would be poor. By transmitting them via the ossicles the sound waves are concentrated and amplified.

The waves are passed into the liquid-filled cochlea in the inner ear. Within the cochlea are delicate hair cells or cilia that bend because of the vibrations in the cochlean liquid and release a chemical transmitter which causes impulses in the auditory nerve.

Processing Sound

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. The human ear can hear frequencies from about 20 Hz to 15 kHz. The auditory system performs some filtering of the sounds received, allowing us to ignore background noise and concentrate on important information.

Touch

The third and last of the senses is touch or ***haptic perception***. 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. Touch is therefore an important means of feedback, and this is no less so in using computer systems. Feeling buttons depress is an important part of the task of pressing the button. Also, we should be aware that, although for the average person, haptic perception is a secondary source of informa-tion, for those whose other senses are impaired, it may be vitally important. For such

users, interfaces such as braille may be the primary source of information in the interaction. We should not therefore underestimate the importance of touch.

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.

There are two kinds of mechanoreceptor, which respond to different types of pressure. ***Rapidly adapting mechanoreceptors*** respond to immediate pressure as the skin is indented. These receptors also react more quickly with increased pressure. However, they stop responding if continuous pressure is applied. ***Slowly adapting mechanoreceptors*** respond to continuously applied pressure.

Although the whole of the body contains such receptors, some areas have greater sensitivity or acuity than others. It is possible to measure the acuity of different areas of the body using the ***two-point threshold test***. Take two pencils, held so their tips are about 12 mm apart. Touch the points to your thumb and see if you can feel two points. If you cannot, move the points a little further apart. When you can feel two points, measure the distance between them. The greater the distance, the lower the sensitivity. You can repeat this test on different parts of your body. You should find that the measure on the forearm is around 10 times that of the finger or thumb. The fingers and thumbs have the highest acuity.

A second aspect of haptic perception is ***kinesthesis***: awareness of the position of the body and limbs. This is due to receptors in the joints. Again there are three types: rapidly adapting, which respond when a limb is moved in a particular direc-tion; slowly adapting, which respond to both movement and static position; and positional receptors, which only respond when a limb is in a static position. This perception affects both comfort and performance. For example, for a touch typist, awareness of the relative positions of the fingers and feedback from the keyboard are very important.

Movement

The stimulus 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 is dependent largely on the physical characteristics of the subjects: their age and fitness, for example. Reaction time varies according to the sensory channel through which the stimulus is received.

A second measure of motor skill is accuracy. One question that we should ask is whether speed of reaction results in reduced accuracy. This is dependent on the task and the user. In some cases, requiring increased reaction time reduces accuracy. This is the premise behind many arcade and video games where less skilled users fail at levels of play that require faster responses. However, for skilled operators this is not necessarily the case. Studies of keyboard operators have shown that, although the faster operators were up to twice as fast as the others, the slower ones made 10 times the errors.

Speed and accuracy of movement are important considerations in the design of interactive systems, primarily in terms of the time taken to move to a particular target on a screen. The target may be a button, a menu item or an icon, for example. The time taken to hit a target is a function of the size of the target and the distance that has to be moved. This is formalized in *Fitts' law* [135]. There are many vari-ations of this formula, which have varying constants, but they are all very similar. One common form is

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

where a and b are empirically determined constants.

This affects the type of target we design. Since users will find it more difficult to manipulate small objects, targets should generally be as large as possible and the distance to be moved as small as possible. This has led to suggestions that pie-chart-shaped menus are preferable to lists since all options are equidistant. However, the trade-off is increased use of screen estate, so the choice may not be so simple.

MEMORY

- Much of our everyday activity relies on memory. As well as storing all our factual knowledge, our memory contains our knowledge of actions or procedures. 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 the second part of our model of the human as an information-processing 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 as shown in figure 1.9.

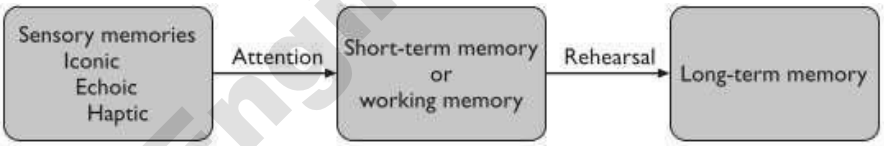


Figure 1.9: A model of the structure of memory

a. Sensory memory:

- The sensory memories act as buffers for stimuli received through the senses. A sensory 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.
- Similarly, the existence of echoic memory is evidenced by our ability to ascertain the direction from which a sound originates. This is due to information being received by both ears. However, since this information is received at different times, we must store the stimulus in the meantime. Echoic memory allows brief 'play-back' of information.

- Information is passed from sensory memory into short-term memory by attention, thereby filtering the stimuli to only those which are of interest at a given time. Attention is the concentration of the mind on one out of a number of competing stimuli or thoughts. It is clear that we are able to focus our attention selectively, choosing to attend to one thing rather than another. This is due to the limited capacity of our sensory and mental processes. If we did not selectively attend to the stimuli coming into our senses, we would be overloaded. We can choose which stimuli to attend to, and this choice is governed to an extent by our arousal, our level of interest or need.

b. 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 = 70 followed by 3 × 70.
- To perform calculations such as this we need to store the intermediate stages for use later. For this task 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.Short-term memory also has a limited capacity.
- There are two basic methods for measuring memory capacity. The first involves determining the length of a sequence which can be remembered in order. The second allows items to be freely recalled in any order. Using the first measure, the average person can remember 7 ± 2 digits. A generalization of the 7 ± 2 rule is that we can remember 7 ± 2 chunks of information.
- Therefore chunking information can increase the short-term memory capacity. The limited capacity of short-term memory produces a subconscious desire to create chunks, and so optimize the use of the memory. The successful formation of a chunk is known as closure.
- In experiments where subjects were able to recall words freely, evidence shows that recall of the last words presented is better than recall of those in the middle, this is known as the recency effect. However, if the subject is asked to perform another task between presentation and recall (for example, counting backwards) the recency effect is eliminated. The recall of the other words is unaffected. This suggests that short-term memory recall is damaged by interference of other information. (refer Fig.1.10)

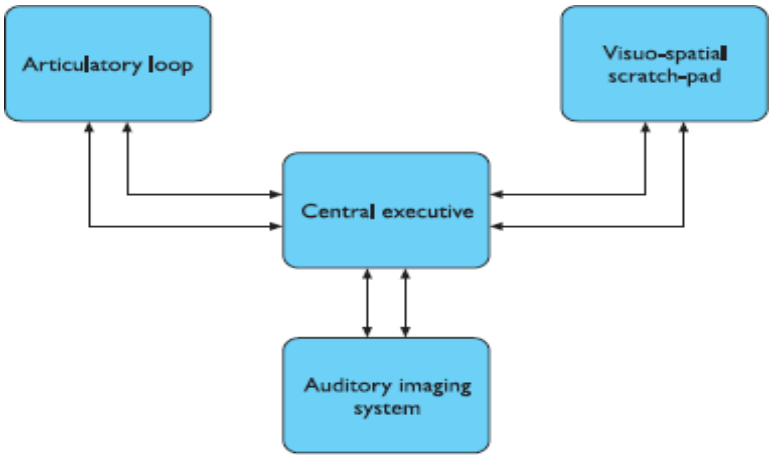


Fig 1.10 : A more detailed model of short-term memory

c. Long-term memory:

- Short-term memory is our working memory or 'scratch-pad', long-term memory is 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 tenth of 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 record of 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.

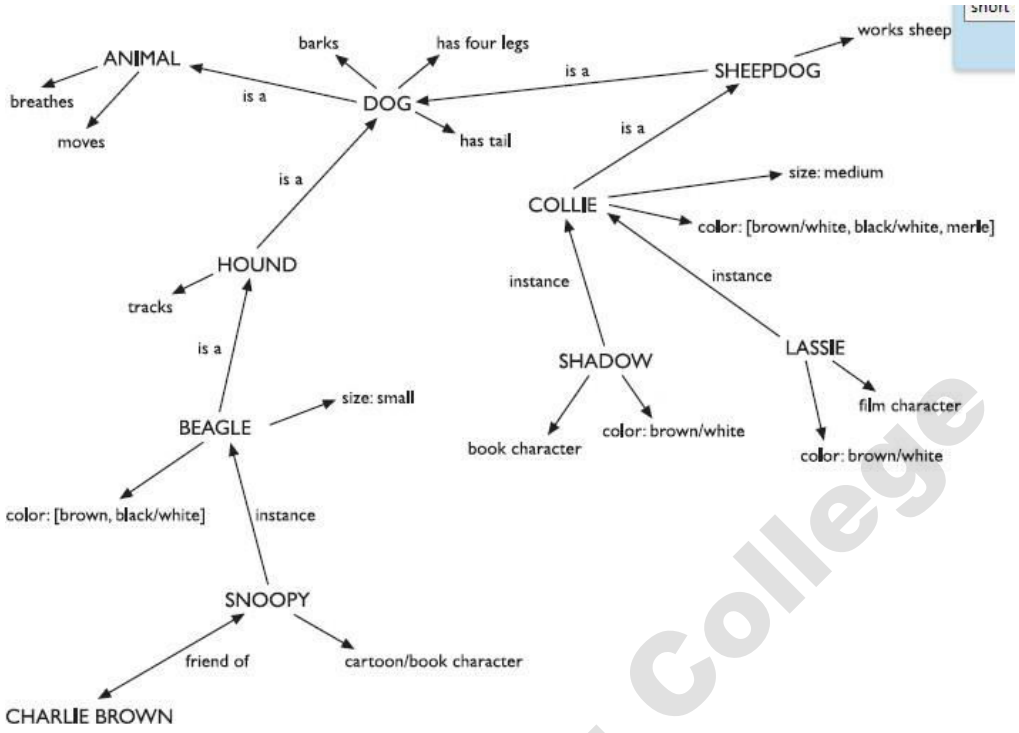


Fig 1.11 : Long-term memory may store information in a semantic network

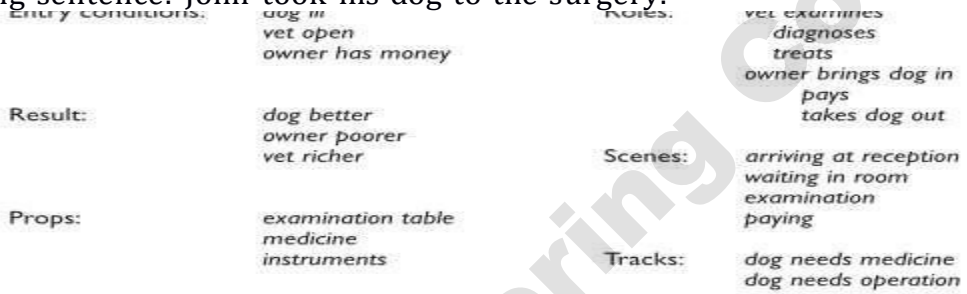
- Semantic memory is structured in some way to allow access to information, representation of relationships between pieces of information, and inference. One model for the way in which semantic memory is structured is as a network. Items are associated to each other in classes, and may inherit attributes from parent classes. This model is known as a semantic network. As an example, our knowledge about dogs may be stored in a network such as that shown in figure 1.12.
- Specific breed attributes may be stored with each given breed, yet general dog information is stored at a higher level. This allows us to generalize about specific cases. For instance, we may not have been told that the sheepdog Shadow has four legs and a tail, but we can infer this information from our general knowledge about sheepdogs and dogs in general. Note also that there are connections within the network which link into other domains of knowledge, for example cartoon characters.
- A number of other memory structures have been proposed to explain how we represent and store different types of knowledge. Each of these represents a different aspect of knowledge and, as such, the models can be viewed as complementary rather than mutually exclusive.

Figure1.12: A frame based representation of knowledge

- Semantic networks represent the associations and relationships between single items in memory. However, they do not allow us to model the representation of more complex objects or events, which are perhaps composed of a number of

items or activities. Structured representations such as frames and scripts organize information into data structures. Slots in these structures allow attribute values to be added. Frame slots may contain default, fixed or variable information.

- A frame is instantiated when the slots are filled with appropriate values. Frames and scripts can be linked together in networks to represent hierarchical structured knowledge. Returning to the 'dog' domain, a frame-based representation of the knowledge may look something as said above.
- The fixed slots are those for which the attribute value is set, default slots represent the usual attribute value, although this may be overridden in particular instantiations and variable slots can be filled with particular values in a given instance. Slots can also contain procedural knowledge.
- Frames extend semantic nets to include structured, hierarchical information. They represent knowledge items in a way which makes explicit the relative importance of each piece of information. Scripts attempt to model the representation of stereotypical knowledge about situations. Consider the following sentence: John took his dog to the surgery.



- After seeing the vet, he left. From our knowledge of the activities of dog owners and vets, we may fill in a substantial amount of detail. The animal was ill. The vet examined and treated the animal. John paid for the treatment before leaving. We are less likely to assume the alternative reading of the sentence, that John took an instant dislike to the vet on sight and did not stay long enough to talk to him!
- 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:

- There are three main activities related to long-term memory: storage or remembering of information, forgetting and information retrieval. 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.

- Sentences are easier still to memorize. To retell the story replacing unfamiliar words and concepts with words which were meaningful was easier.
- Stories were effectively translated into the subject's own culture. This is related to the semantic structuring of long-term memory: if information is meaningful and familiar, it can be related to existing structures and more easily incorporated into memory.
- So if structure, familiarity and concreteness help us in learning information, what causes us to lose this information, to forget? There are two main theories of forget- ting: decay and interference. The first theory suggests that the information held in long-term memory may eventually be forgotten.
- From an 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.
- A common example of this is the fact that if you change telephone numbers, learning your new number makes it more difficult to remember your old number. This is because the new association masks the old. However, sometimes the old memory trace breaks through and interferes with new information.
- This is called proactive inhibition. An example of this is when you find yourself driving to your old house rather than your new one. Forgetting is also affected by emotional factors.
- In experiments, subjects given emotive words and non-emotive words found the former harder to remember in the short term but easier in the long term. Indeed, this observation tallies with our experience of selective memory. We tend to remember positive information rather than negative (hence nostalgia for the 'good old days'), and highly emotive events rather than mundane.

REASONING AND PROBLEM SOLVING

- 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. Thinking can require different amounts of knowledge.

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

Deductive reasoning: Deductive reasoning derives the logically necessary conclusion from the given premises.

Inductive reasoning: Induction is generalizing from cases we have seen to infer information about cases we have not seen.

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.

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, is the 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.
- A second major theory, proposed in the 1970s by Newell and Simon, was the problem space theory, which takes the view that the mind is a limited information processor. Later variations on this drew on the earlier theory and attempted to reinterpret Gestalt theory in terms of information processing theories. We will look briefly at each of these views.

Gestalt theory:

- Gestalt psychologists were answering the claim, made by behaviorists, that problem solving 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.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 these states using legal state transition operators. The problem has an initial state 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. In means–ends analysis the initial state is compared with the goal state and an operator chosen to reduce the difference between the two.
- For example, imagine you are reorganizing your office and you want to move your desk from the north wall of the room to the window. Your initial state is that the desk is at the north wall. The goal state is that the desk is by the window. The main difference between these two is the location of your desk.
- You have a number of operators which you can apply to moving things: you can carry them or push them or drag them, etc. However, you know that to carry something it must be light and that your desk is heavy. You therefore have a new subgoal: to make the desk light. Your operators for this may involve removing drawers, and so on.

Analogy in problem solving:

- A third element of problem solving is the use of analogy. Here we are interested in how 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 analogical mapping. Similarities between the known domain and the new one are noted and operators from the known domain are transferred to the new one. This process has been investigated using analogous stories.

Skill acquisition:

- All of the 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.

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, as we see when 'human error' results in a plane crash or nuclear plant leak.
- There are several different types of error. 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. Other errors result from an incorrect understanding, or model, of a situation or system.
- People build their own theories to understand the causal behavior of systems. These have been termed mental models. They have a number of characteristics. Mental models are often partial: the person does not have a full understanding of the working of the whole system. They are unstable and are subject to change.

THE COMPUTER

- There is the computer 'box' itself, a keyboard, a mouse and a color screen. 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 data that may have to be entered into and obtained from a system, and there are also many different types of user, each with their own unique requirements. A computer system comprises various elements, each of which affects the user of the system.
- Input devices for interactive use, allowing text entry, drawing and selection from the screen:
    a. text entry: traditional keyboard, phone text entry, speech and handwriting
    b. pointing: principally the mouse, but also touchpad, stylus, and others
    c. 3D interaction devices
- Output display devices for interactive use:
    a. different types of screen mostly using some form of bitmap display
    b. large displays and situated displays for shared and public use
    c. digital paper may be usable in the near future

- Virtual reality systems and 3D visualization have special interaction and display devices. Various devices in the physical world:
    - physical controls and dedicated displays
    - sound, smell and haptic feedback
    - sensors for nearly everything including movement, temperature, bio-signs
- Paper output and input: the paperless office and the less-paper office:
    - different types of printers and their characteristics, character styles and fonts
    - scanners and optical character recognition
- Memory:
    - short-term memory: RAM
    - long-term memory: magnetic and optical disks
    - capacity limitations related to document and video storage
    - access methods as they limit or help the user
- Processing:
    - the effects when systems run too slow or too fast, the myth of the infinitely fast machine
    - limitations on processing speed
    - networks and their impact on system performance.

DEVICES

Input devices for interactive use, allowing text entry, drawing and selection from the screen.

TEXT ENTRY DEVICES: 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 QWERTY keyboard: The layout of the digits and letters on a QWERTY keyboard is fixed, but non-alphanumeric keys vary between keyboards. The standard layout is also subject to variation in the placement of brackets, backslashes and such like.

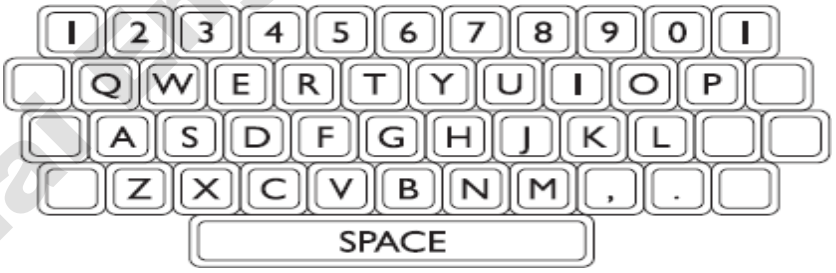


Fig.1.13 QWERTY Keyboard

- Ease of learning – alphabetic keyboard: One of the most obvious layouts to be produced is the alphabetic keyboard, in which the letters are arranged alphabetically across the keyboard. It might be expected that such a layout would make it quicker for untrained typists to use.
- 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. The layout of the keys also attempts to ensure that the

majority of keystrokes alternate between hands, thereby increasing the potential speed.

Chord keyboards:

- Chord keyboards are significantly different from normal alphanumeric keyboards. Only a few keys, four or five, are used and letters are produced bypressing one or more of the keys at once.
- For example, in the Microwriter, the pattern of multiple keypresses is chosen to reflect the actual letter shape.Such keyboards have a number of advantages. They are extremely compact: simply reducing the size of a conventional keyboard makes the keys too small and close together, with a correspondingly large increase in the difficulty of using it.

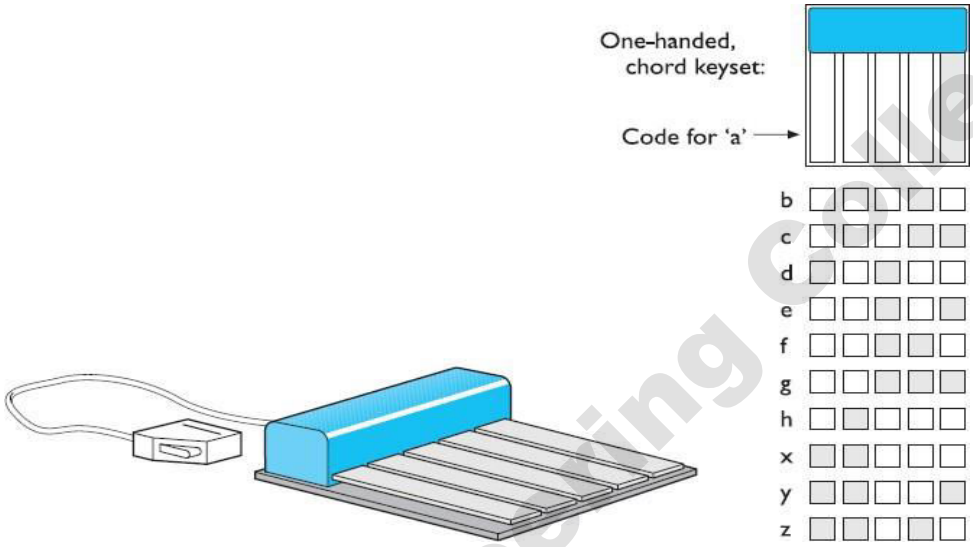


Fig.1.14 : A very early chord keyboard (left) and its lettercodes (right)

Phone pad and T9 entry:

- With mobile phones being used for SMS text messaging and WAP, 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 shows a typical mapping of digits to letters.
- 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.
- Most phones have at least two modes for the numeric buttons: one where the keys mean the digits and one where they mean letters. Some have additional modes to make entering accented characters easier. Also a special mode or setting is needed for capital letters although many phones use rules to reduce this.

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, but with the computer taking this form of input and converting it to text, we can see that it is an intuitive and simple way of interacting with the computer.

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 unacceptible (sic)! Note also that this performance is usually quoted only for a restricted vocabulary of command words.
- Trying to extend such systems to the level of understanding natural language, with its inherent vagueness, imprecision and pauses, opens up many more problems that have not been satisfactorily solved even for keyboard-entered natural language.

Positioning, Pointing And Drawing

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.
- 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. The changing values of these potentiometers can be directly related to changes in position of the ball. The potentiometers are aligned in different directions so that they can detect both horizontal and vertical motion.
- The relative motion information is passed to the computer via a wire attached to the box, or in some cases using wireless or infrared, and moves a pointer on the screen, called the cursor. The whole arrangement tends to look rodent-like, with the box acting as the body and the wire as the tail; hence the term 'mouse'.

Touchpad:

- Touchpads 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.

Trackball and thumbwheel:

- The trackball is really just an upside-down mouse! A weighted ball faces upwards and is rotated inside a static housing, the motion being detected in the same way as for a 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. It is fairly accurate, but is hard to draw with, as long movements are difficult.

Joystick and keyboard nipple:

- The joystick is an indirect input device, taking up very little space. Consisting of a small palm-sized box with a stick or shaped grip sticking up from it, the joystick is a simple 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.

Touch-sensitive screens (touchscreens):

- Touchscreens are another method of allowing the user to point and select objects on the screen, but they are much more direct than the mouse, as they detect the presence of the user's finger, or a stylus, on the screen itself.
- They work in one of a number of different ways: by the finger (or stylus) interrupting a matrix of light beams, or by capacitance changes on a grid overlaying the screen, or by ultrasonic reflections. 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 touch- sensitive surfaces often emply a stylus. Instead of pointing at the screen directly a small pen-like plastic stick is used to point and draw on the screen. This is particularly popular in PDAs, but they are also being used in some laptop computers.

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.
- The tablet provides positional information by measuring the position of some device on a special pad, or tablet, and can work in a number of ways. The resistive tablet detects point contact between two separated conducting sheets.
- It has advantages in that it can be operated without a specialized stylus – a pen or the user's finger is sufficient. The magnetic tablet detects current pulses in a magnetic field using a small loop coil housed in a special pen. There are also capacitative and electrostatic tablets that work in a similar way.
- The sonic tablet is similar to the above but requires no special surface. An ultrasonic pulse is emitted by a special pen which is detected by two or more microphones which then triangulate the pen position. This device can be adapted to provide 3D input, if required.

Eyegaze:

- Eyegaze systems allow you to control the computer by simply looking at it! Some systems require you to wear special glasses or a small head-mounted box, others are built into the screen or sit as a small box below the screen.
- A low-power laser is shone into the eye and is reflected off the retina. The reflection changes as the angle of the eye alters, and by tracking the reflected beam the eyegaze system can determine the direction in which the eye is looking.
- The system needs to be calibrated, typically by staring at a series of dots on the screen, but thereafter can be used to move the screen cursor or for other more specialized uses.

- Eyegaze is a very fast and accurate device, but the more accurate versions can be expensive. It is fine for selection but not for drawing since the eye does not move in smooth lines. Also in real applications it can be difficult to distinguish deliberately gazing at something and accidentally glancing at it.

Cursor keys and discrete positioning:

- All of the devices we have discussed are capable of giving near continuous 2D positioning, with varying degrees of accuracy. For many applications we are only interested in positioning within a sequential list such as a menu or amongst 2D cells as in a spreadsheet.
- Even for moving within text discrete up/down left/right keys can sometimes be preferable to using a mouse. Cursor keys are available on most keyboards. Four keys on the keyboard are used to control the cursor, one each for up, down, left and right.

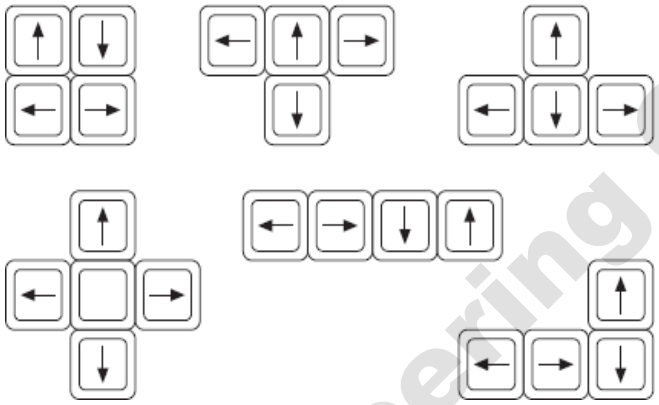


Fig.1.15 : Various cursor key layouts

Display Devices:

- The vast majority of interactive computer systems would be unthinkable without some sort of display screen, but many such systems do exist, though usually in specialized applications only.
- 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 pixels may be limited to black and white (for example, the small display on many TV remote controls), in grayscale, or full color.
- The color or, for monochrome screens, the intensity at each pixel is held by the computer's video card. One bit per pixel can store on/off information, and hence only black and white (the term 'bitmap' dates from such displays).
- More bits per pixel give rise to more color or intensity possibilities. As well as the number of colors that can be displayed at each pixel, the other measure that is important is the resolution of the screen. Actually the word 'resolution' is used in 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 4:3 ratio, perhaps 1024 pixels across by 768 down, or 1600 × 1200; for PDAs this will be 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 (see Section 2.7 below) this density varies little between 72 and 96 pixels per inch. To add to the confusion, a monitor, liquid crystal display (LCD) screen or other display device will quote its maximum resolution, but the computer may actually give it less than this.

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 The electron beam is scanned from left to right, and then flicked back to rescan the next line, from top to bottom. This is repeated, at about 30 Hz (that is, 30 times a second), per frame, although higher scan rates are sometimes used to reduce the flicker on the screen.
- Another way of reducing flicker is to use interlacing, in which the odd lines on the screen are all scanned first, followed by the even lines. Using a high-persistence phosphor, which glows for a longer time when excited, also reduces flicker, but causes image smearing especially if there is significant animation.

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 can be accessed without the need for scanning.

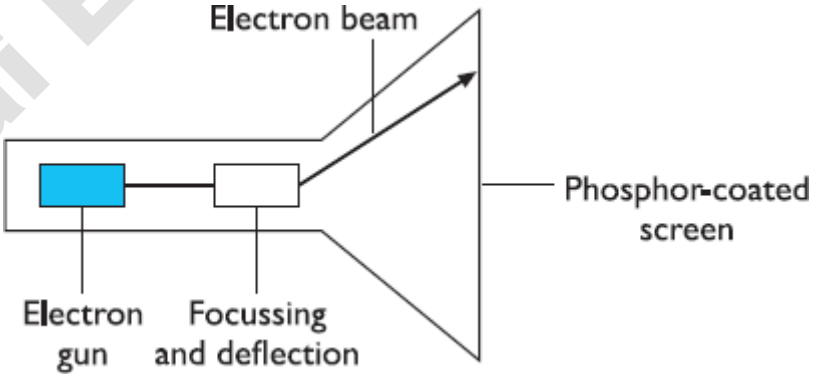


Fig.1.16 : CRT screen

Special displays:

- There are a number of other display technologies used in niche markets. The one you are most likely to see is the gas plasma display, which is used in large screens. The random scan display, also known as the directed beam refresh, or vector display, works differently from the bitmap display, also known as raster scan.

- Instead of scanning the whole screen sequentially and horizontally, the random scan draws the lines to be displayed directly. By updating the screen at at least 30 Hz to reduce flicker, the direct drawing of lines at any angle means that jaggies are not created, and higher resolutions are possible, up to 4096 × 4096 pixels.
- Color on such displays is achieved using beam penetration technology, and is generally of a poorer quality.The direct view storage tube is used extensively as the display for an analog storage oscilloscope, which is probably the only place that these displays are used in any great numbers.
- They are similar in operation to the random scan CRT but the image is maintained by flood guns which have the advantage of producing a stable display with no flicker. The screen image can be incrementally updated but not selectively erased; removing items has to be done by redrawing the new image on a completely erased screen. The screens have a high resolution, typically about 4096 × 3120 pixels, but suffer from low contrast, low brightness and a difficulty in displaying color.

Large displays and situated displays:

- Displays are no longer just things you have on your desktop or laptop. There are several types of large screen display. Some use gas plasma technology to create large flat bitmap displays. These behave just like a normal screen except they are big and usually have the HDTV (high definition television) wide screen format which has an aspect ratio of 16:9 instead of the 4:3 on traditional TV and monitors.
- Where very large screen areas are required, several smaller screens, either LCD or CRT, can be placed together in a video wall. These can display separate images, or a single TV or computer image can be split up by software or hardware so that each screen displays a portion of the whole and the result is an enormous image. This is the technique often used in large concerts to display the artists or video images during the performance.
- Possibly the large display you are most likely to have encountered is some sort of projector. There are two variants of these. In very large lecture theatres, especially older ones, you see projectors with large red, green and blue lenses.
- These each scan light across the screen to build a full color image. In smaller lecture theatres and in small meetings you are likely to see LCD projectors. Usually the size of a large book, these are like ordinary slide projectors except that where the slide would be there is a small LCD screen instead. The light from the projector passes through the tiny screen and is then focussed by the lens onto the screen.

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.

Devices for Virtual Reality And 3d Interaction:

These require you to navigate and interact in a three-dimensional space.

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 involves a change from two degrees of freedom 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. In many PC games and desktop virtual reality, 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

The 3D mouse:

- There are a variety of devices that act as 3D versions of a mouse. Rather than just moving the mouse on a tabletop, you can pick it up, move it in three dimensions, rotate the mouse and tip it forward and backward.
- The 3D mouse has a full six degrees of freedom as its position can be tracked (three degrees), and also its up/down angle (called pitch), its left/right orientation (called yaw) and the amount it is twisted about its own axis (called roll). Various sensors are used to track the mouse position and orientation: magnetic coils, ultrasound or even mechanical joints where the mouse is mounted rather like an angle-poise lamp.

Fig.1.17 : Pitch, yaw and roll

Dataglove:

- One of the mainstays of high-end VR systems, the dataglove is a 3D input 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 of wrist 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 of 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 display the 3D world to each eye and (ii) they allow the user's head

position to be tracked. The head tracking is used primarily to feed into the output side. As the user's head moves around the user ought to see different parts of the scene.

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.

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.

Seeing in 3D:

- Our eyes use many cues to perceive depth in the real world. 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).
- Because each eye is looking at an object from a slightly different angle each sees a different image and our brain is able to use this to assess the relative distance of different objects. In desktop VR this stereoscopic effect is absent. However, various devices exist to deliver true stereoscopic images.

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-mounted displays, some virtual reality systems work by putting the user within an environment where the virtual world is displayed upon it.
- The most obvious examples of this are large flight simulators – you go inside a mock-up of an aircraft cockpit and the scenes 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 fill the 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.

Physical Controls, Sensors And Special Devices

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 process control systems.

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

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 when a car was about to go off the track. In some VR applications, such as the use 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.
- Haptic devices are not limited to virtual environments, but are used in specialist interfaces in the real world too.

Physical controls:

- A desktop computer system has to serve many functions and so has generic keys and 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.
- 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 triggered by a small switch in the car door.

MEMORY

- 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).
- RAM and short-term memory (STM): At the lowest level of computer memory are the registers on the computer chip, but these 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 random access memory (RAM).
- Different forms of RAM differ as to their precise access times, power consumption and characteristics. Typical access times are of the order of 10 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.
- 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.

a. 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. There are two main kinds of technology used in disks: magnetic disks and optical disks.
- 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 and 1.4 Mbytes, but as they are removable, you can have as many as you have room for on 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.
- Optical disks use laser light to read and (sometimes) write the information on the disk. There are various high capacity specialist optical devices, but the most common is the CD-ROM, using the same technology as audio compact discs.
- CD-ROMs have a capacity of around 650 megabytes, but cannot be written to at all. They are useful for published material such as online reference books, multimedia and software distribution. Recordable CDs are a form of WORM device (write-once read-many) and are more flexible in that information can be written, but (as the name suggests) only once at any location – more like a piece of paper than a blackboard. They are obviously very useful for backups and for producing very secure audit information.

PROCESSING AND NETWORKS

- 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.
- 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.
- The first type of fault is functional fault in which the program did the wrong thing. 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. A similar problem, icon wars, occurs on window systems. The user clicks the mouse on a menu or icon, and nothing happens; for some reason the machine is busy or slow. So the user clicks again, tries something else – then, suddenly, all the buffered

mouse clicks are interpreted and the screen becomes a blur of flashing windows and menus.

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.

Storage channel bound:

- 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.
- 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 taken by common graphics operations, and there is tremendous variability in performance between programs running on the same hardware. Most computers include a special-purpose graphics card to handle many of the most common graphics operations.
- This is optimized for graphics operations and allows the main processor to do other work such as manipulating documents and other user data.

Network capacity:

- Most computers are linked by networks. At the simplest this can mean using shared files on a remote machine. When accessing such files it can be the speed of the network rather than that of the memory which limits performance.

INTERACTION

- There are a number of ways in which the user can communicate with the system. At one extreme is batch input, in which the user provides all the information to the computer at once and leaves the machine to perform the task.
- This approach does involve an interaction between the user and computer but does not support many tasks well. At the other extreme are highly interactive input devices and paradigms, such as direct manipulation and the applications of virtual reality.
- Here the user is constantly providing instruction and receiving feedback. These are the types of interactive system we are considering.

MODELS

- 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 at a 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.

The terms of interaction:

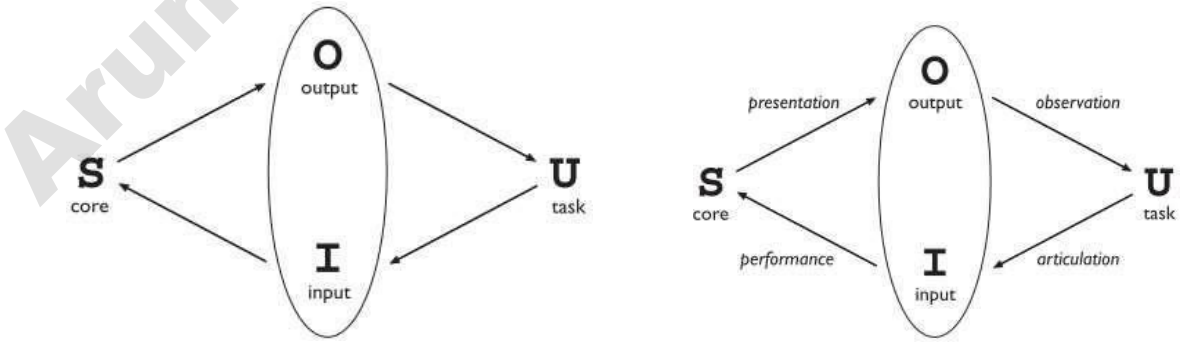
- The purpose of an interactive system is to aid a user in accomplishing goals from some application domain. A domain defines an area of expertise and knowledge in some real-world activity. Tasks are operations to manipulate the concepts of a domain. A goal is the desired output from a performed task.
- An intention is a specific action required to meet the goal. Task analysis involves the identification of the problem space for the user of an interactive system in terms of the domain, goals, intentions and tasks. 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.

The execution–evaluation cycle:

- Norman's model of interaction is perhaps the most influential in Human–Computer Interaction, possibly because of its closeness to our intuitive understanding of the interaction between human user and computer. The user formulates a plan of action, which is then executed at the computer interface.
- When the plan, or part of the plan, has been executed, the user observes the computer interface to evaluate the result 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 in Norman's model of interaction are as follows:
    a. Establishing the goal.
    b. Forming the intention.
    c. Specifying the action sequence.
    d. Executing the action.
    e. Perceiving the system state.
    f. Interpreting the system state.
- 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 into 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 attempts a more realistic description of interaction by including the system explicitly, and breaks it into four main components, as shown in figure below. 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. 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 the interactive cycle, each corresponding to a translation from one component to another, as shown by the labelled arcs in figure below.
- 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 articulated within the input language. The input language is translated into the core language as operations to be performed by the System.
- The System then transforms itself as described by the operations; the execution phase of the cycle is complete and the evaluation phase now begins. The System is in a new state, which must now be communicated to the User.
- The current values of system attributes are rendered as concepts or features of the Output. It is then up to the User to observe the Output and assess the results of the interaction relative to the original goal, ending the evaluation phase and, hence, the interactive cycle. There are four main translations involved in the interaction: articulation, performance, presentation and observation



.

Fig1.18:a.General Interaction Framework b.Translations between components

- The general interaction framework Translations between componentsis shown in figure 1.18. 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.

- As pointed out above, this articulation is judged in terms of the coverage from tasks to input and the relative ease with which the translation can be accomplished. The task is phrased in terms of certain psychological attributes that highlight the important features of the domain for the User. If these psychological attributes map clearly onto the input language, then articulation of the task will be made much simpler.
- At the next stage, the responses of the Input are translated to stimuli for the System. Of interest in assessing this translation is whether the translated input language can reach as many states of the System as is possible using the System stimuli directly. The ease with which this translation from Input to System takes place is of less importance because the effort is not expended by the user.
- However, there can be a real effort expended by the designer and programmer. In this case, the ease of the translation is viewed in terms of the cost of implementation. Once a state transition has occurred within the System, the execution phase of the interaction is complete and the evaluation phase begins.
- The new state of the System must be communicated to the User, and this begins by translating the System responses to the transition into stimuli for the Output component. This presentation translation must preserve the relevant system attributes from the domain in the limited expressiveness of the output devices.
- The ability to capture the domain concepts of the System within the Output is a question of expressiveness for this translation. The response from the Output is translated to stimuli for the User which trigger assessment. The observation translation will address the ease and coverage of this final translation.
- For example, it is difficult to tell the time accurately on an unmarked analog clock, especially if it is not oriented properly. It is difficult in a command line interface to determine the result of copying and moving files in a hierarchical file system. Developing a website using a markup language like HTML would be virtually impossible without being able to preview the output through a browser.

FRAMEWORKS

- 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. The Figure 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 as in figure 1.19.
- 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.

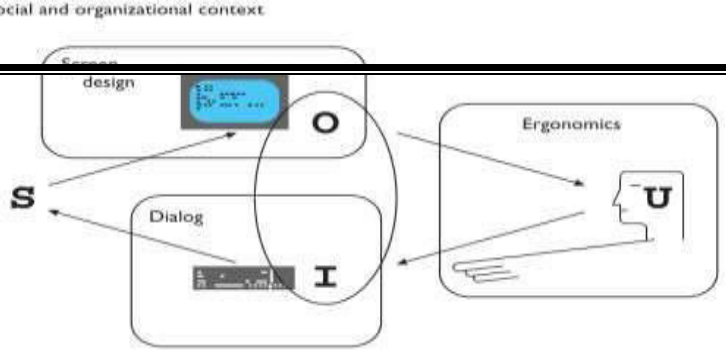


Figure1.19: A framework for human-computer interaction

- 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.
- A framework for human–computer interaction. Adapted from ACM SIGCHI Curriculum Development Group

ERGONOMICS

- 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 from this.

Arrangement of controls and displays:

- 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.
- 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 with a huge range of displays and controls. Indeed, returning to the less critical PC application, inappropriate placement of controls and displays can lead to inefficiency and frustration.
- 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.
- 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.

- 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:

- 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. It has been suggested that excessive use of CRT displays can be harmful to users, particularly pregnant women.

The use of color:

- 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.
- 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.
- Colors used in the display should be as distinct as possible and the distinction should not be affected by changes in contrast. The colors used should also correspond to common conventions and user expectations.

STYLES

- The choice of interface style can have a profound effect on the nature. 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.

As the WIMP interface is the most common and complex.

Command line interface:

- Command line interfaces are powerful in that they offer direct access to system functionality (as opposed to the hierarchical nature of menus), and can be combined to apply a number of tools to the same data. They are also flexible: the command often has a number of options or parameters that will vary its behavior in some way, and it can be applied to many objects at once, making it useful for repetitive tasks.
- However, this flexibility and power brings with it difficulty in use and learning. Commands must be remembered, as no cue is provided in the command line to indicate which command is needed. They are therefore better for expert users than for novices.
- This problem can be alleviated a little by using consistent and meaningful commands and abbreviations. The commands used should be terms within the vocabulary of the user rather than the technician. Unfortunately, commands are often obscure and vary across systems, causing confusion to the user and increasing the overhead of learning.Typical example: the Unix system

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. The grouping and naming of menu options then provides the only cue for the user to find the required option.
- Such systems either can be purely text based, with the menu options being presented as numbered choices, or may have a graphical component in which the menu appears within a rectangular box and choices are made, perhaps by typing the initial letter of the desired selection, or by entering the associated number, or by moving around the menu with the arrow keys.

Fig.1.20 : Menu-driven interface

Natural language:

- 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. It is important in interfaces which use natural language in this restricted form that the user is aware of the limitations of the system and does not expect too much understanding.
- The use of natural language in restricted domains is relatively successful, but it is debatable whether this can really be called natural language. The user still has to learn which phrases the computer understands and may become frustrated if too much is expected. However, it is also not clear how useful a general natural language interface would be. Language is by nature vague and imprecise: this gives it its flexibility and allows creativity in expression. Computers, on the other hand, require precise instructions.

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.
- 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 search the 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.

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. 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. Most form-filling

interfaces allow easy movement around the form and allow some fields to be left blank.

- They also require correction facilities, as users may change their minds or make a mistake about the value that belongs in each field. The dialog style is useful primarily for data entry applications and, as it is easy to learn and use, for novice users. However, assuming a design that allows flexible entry, form filling is also appropriate for expert users.
- Spreadsheets are a sophisticated variation of form filling. The spreadsheet comprises a grid of cells, each of which can contain a value or a formula. The formula can involve the values of other cells (for example, the total of all cells in this column).
- The user can enter and alter values and formulae in any order and the system will maintain consistency amongst the values displayed, ensuring that all formulae are obeyed. The user can therefore manipulate values to see the effects of changing different parameters.
- Spreadsheets are an attractive medium for interaction: the user is free to manipulate values at will and the distinction between input and output is blurred, making the interface more flexible and natural.

WIMP:

- 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 of WIMP interfaces include Microsoft Windows for IBM PC compatibles, MacOS for Apple Macintosh compatibles and various X Windows-based systems for UNIX.

Point and click:

- 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. 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 touchscreen information systems. In this case, it is often combined with a menu-driven interface.

Three-dimensional interfaces:

- 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. Unfortunately, some interfaces make indiscriminate use of sculptural effects, on every text area, border and menu, so all sense of differentiation is lost.
- A more complex technique uses interfaces with 3D workspaces. The objects displayed in such systems are usually flat, but are displayed in perspective when at an angle to the viewer and shrink when they are 'further away'.

ELEMENTS OF WIMP INTERFACE

- WIMP stands for windows, icons, menus and pointers (sometimes windows, icons, mice and pull-down menus).There are also many additional interaction objects and techniques commonly used in WIMP interfaces, some designed for specific purposes and others more general. Together, these elements of the WIMP interfaces are called widgets, and they comprise the toolkit for interaction between user and system.

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. Usually, windows have various things associated with them that increase their usefulness.
- Scrollbars are one such attachment, allowing the user to move the contents of the window up and down, or from side to side. This makes the window behave as if it were a real window onto a much larger world, where new information is brought into view by manipulating the scrollbars.

Icons:

- 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.
- Icons can also be used to represent other aspects of the system, such as a waste-basket for throwing unwanted files into, or various disks, programs or functions that are accessible to the user. Icons can take many forms: they can be realistic representations of the objects that they stand for, or they can be highly stylized. They can even be arbitrary symbols, but these can be difficult for users to interpret.

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 icons. The mouse provides an input device capable of such tasks, although joysticks and trackballs are other alternatives.

- The different shapes of cursor are often used to distinguish modes, for example the normal pointer cursor may be an arrow, but change to cross-hairs when drawing a line. Cursors are also used to tell the user about system activity, for example a watch or hour-glass cursor may be displayed when the system is busy reading a file. Pointer cursors are like icons, being small bitmap images, but in addition all cursors have a hot-spot, the location to which they point.

Menus:

- A menu is an interaction technique that 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. The pointing device is used to indicate the desired option.
- As the pointer moves to the position of a menu item, the item is usually highlighted (by inverse video, or some similar strategy) to indicate that it is the potential candidate for selection.
- Selection usually requires some additional user action, such as pressing a button on the mouse that controls the pointer cursor on the screen or pressing some special key on the keyboard.
- Menus are inefficient when they have too many items, and so cascading menus are utilized, in which item selection opens up another menu adjacent to the item, allowing refinement of the selection. Several layers of cascading menus can be used. The main menu can be visible to the user all the time, as a menu bar and submenus can be pulled down or across from it upon request.
- Menu bars are often placed at the top of the screen or at the top of each window. Alternatives include menu bars along one side of the screen, or even placed amongst the windows in the main 'desktop' area. Websites use a variety of menu bar locations, including top, bottom and either side of the screen. Alternatively, the main menu can be hidden and upon request it will pop up onto the screen.
- Pull-down menus are dragged down from the title at the top of the screen, by moving the mouse pointer into the title bar area and pressing the button. Fall-down menus are similar, except that the menu automatically appears when the mouse pointer enters the title bar, without the user having to press the button.
- Some menus are pin-up menus, in that they can be 'pinned' to the screen, staying in place until explicitly asked to go away. Pop-up menus appear when a particular region of the screen, maybe designated by an icon, is selected, but they only stay as long as the mouse button is depressed.
- Another approach to menu selection is to arrange the options in a circular fashion. The pointer appears in the center of the circle, and so there is the same distance to travel to any of the selections. This has the advantages that it is easier to select items, since they can each have a larger target area, and that the selection time for each item is the same, since the pointer is equidistant from them all.

Fig.1.21 : Elements of the WIMP interface

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 to toggle 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.
- Such toggle buttons can be grouped together to allow a user to select one feature from a set of mutually exclusive options, such as the size in points of the current font. These are called radio buttons, since the collection functions much like the old-fashioned mechanical control buttons on car radios.
- If a set of options is not mutually exclusive, such as font characteristics like bold, italics and underlining, then a set of toggle buttons can be used to indicate the on/off status of the options. This type of collection of buttons is sometimes referred to as check boxes.

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. 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 of icons 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 subdialog between user and system for a very specific task that will normally be embedded within some larger task.
- 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 subdialog is complete, the dialog box will disappear.
- Just as windows are used to separate the different threads of user–system dialog, so too are dialog boxes used to factor out auxiliary task threads from the main task dialog.

INTERACTIVITY

- 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.
- 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!).
- Menus are a major difference between the MacOS and Microsoft Windows environments: in MacOS you have to keep the mouse depressed through- out 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.
- 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.
- 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. Slips and mistakes is a way 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. This ability to detect and correct is important both at the small scale of button presses and keystrokes and also at the large scale.

PARADIGMS

- Paradigms is
    - Predominant theoretical frameworks or scientific world views
        - e.g., Aristotelian, Newtonian, Einsteinian (relativistic) paradigms in physics
    - Understanding HCI history is largely about understanding a series of paradigm shifts
        - Not all listed here are necessarily "paradigm" shifts, but are at least candidates
        - History will judge which are true shifts

Paradigms of interaction

- New computing technologies arrive, creating a new perception of the human computer relationship. We can trace some of these shifts in the history of interactive technologies.The initial paradigm
    - Batch processing
    - Time-sharing
    - Networking
    - Graphical displays
    - Microprocessor
    - WWW
    - Ubiquitous Computing

Time-sharing

- 1940s and 1950s – explosive technological growth
- 1960s – need to channel the power
- J.C.R. Licklider at ARPA
- single computer supporting multiple users

Video Display Units

- more suitable medium than paper
- 1962 – Sutherland's Sketchpad

- computers for visualizing and manipulating data
- one person's contribution could drastically change the history of computing

Programming toolkits

- Engelbart at Stanford Research Institute
- 1963 – augmenting man's intellect
- 1968 NLS/Augment system demonstration
- the right programming toolkit provides building blocks to producing complex interactive systems

Personal computing

- 1970s – Papert's LOGO language for simple graphics programming by children
- A system is more powerful as it becomes easier to user
- Future of computing in small, powerful machines dedicated to the individual
- Kay at Xerox PARC – the Dynabook as the ultimate personal computer

Window systems and the WIMP interface

- humans can pursue more than one task at a time
- windows used for dialogue partitioning, to "change the topic"
- 1981 – Xerox Star first commercial windowing system
- windows, icons, menus and pointers now familiar interaction mechanisms

Metaphor

- relating computing to other real-world activity is effective teaching technique
  - LOGO's turtle dragging its tail
  - file management on an office desktop
  - word processing as typing
  - financial analysis on spreadsheets
  - virtual reality – user inside the metaphor
- Problems
  - some tasks do not fit into a given metaphor
  - cultural bias

Direct manipulation

- 1982 – Shneiderman describes appeal of graphically-based interaction
  - visibility of objects
  - incremental action and rapid feedback
  - reversibility encourages exploration
  - syntactic correctness of all actions
  - replace language with action
  - 1984 – Apple Macintosh
- the model-world metaphor
- What You See Is What You Get (WYSIWYG)

Language versus Action

- actions do not always speak louder than words!
- DM – interface replaces underlying system
- language paradigm
- interface as mediator
- interface acts as intelligent agent
- programming by example is both action and language

Hypertext

- 1945 – Vannevar Bush and the memex
- key to success in managing explosion of information

- mid 1960s – Nelson describes hypertext as non-linear browsing structure
- hypermedia and multimedia
- Nelson's Xanadu project still a dream today

Multimodality

- a mode is a human communication channel
- emphasis on simultaneous use of multiple channels for input and output

Computer Supported Cooperative Work (CSCW)

- CSCW removes bias of single user / single computer system
- Can no longer neglect the social aspects
- Electronic mail is most prominent success

The World Wide Web

- Hypertext, as originally realized, was a closed system
- Simple, universal protocols (e.g. HTTP) and mark-up languages (e.g. HTML) made publishing and accessing easy
- Critical mass of users lead to a complete transformation of our information economy.

Agent-based Interfaces

- Original interfaces
  - Commands given to computer
  - Language-based
- Direct Manipulation/WIMP
  - Commands performed on "world" representation
  - Action based
- Agents - return to language by instilling proactivity and "intelligence" in command processor
  - Avatars, natural language processing

Ubiquitous Computing

"The most profound technologies are those that disappear."
Mark Weiser, 1991
Late 1980's: computer was very apparent
How to make it disappear?

- Shrink and embed/distribute it in the physical world
- Design interactions that don't demand our intention

Sensor-based and Context-aware Interaction

- Humans are good at recognizing the "context" of a situation and reacting appropriately
- Automatically sensing physical phenomena (e.g., light, temp, location, identity) becoming easier

Case Studies

1. **Automatic Syringe:**

The eventual users should be involved in the design process. They have vital knowledge and will soon find flaws. A mechanical syringe was once being developed and a prototype was demonstrated to hospital staff. Happily they quickly noticed the potentially fatal flaw in its interface.

The doses were entered via a numeric keypad: **an accidental keypress** and the **dose could be out by a factor of 10!** The production version had individual increment/decrement buttons for each digit.

People are complicated, so you won't get it right first time. Programming an interface can be a very difficult and time-consuming Process. So, the result becomes precious and the builder will want to defend it and minimize changes. Making early prototypes less precious and easier to throw away is crucial.

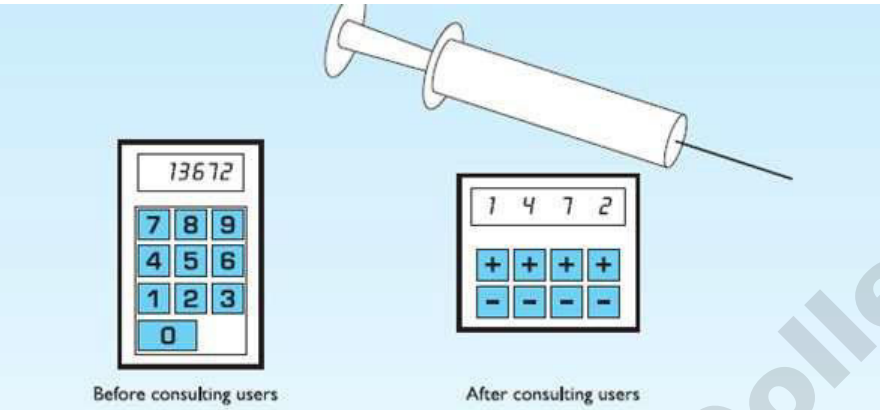


Figure 0.1 Automatic syringe: setting the dose to 1372. The effect of one key slip before and after user involvement

2. Handling the goods

E-commerce has become very successful in some areas of sales, such as travel services, books and CDs, and food. However, in some retail areas, such as clothes **shopping**, e-commerce has been less successful. Why?

When buying train and airline tickets and, to some extent, books and food, the experience of shopping is less important than the convenience. So, as long as we know what we want, we are happy to shop online. With clothes, the experience of shopping is far more important. We need to be able to **handle the goods, feel the texture of the material, check the weight to test quality**. Even if we know that something will fit us we still want to be able to handle it before buying.

Research into **haptic interaction** is looking at ways of solving this problem. By using special force feedback and tactile hardware, users are able to feel surfaces and shape. For example, a demonstration environment called "**TouchCity**" allows people to walk around a **virtual shopping mall**, pick up products and feel their texture and weight. A key problem with the commercial use of such an application, however, is that the haptic experience requires expensive hardware not yet available to the average e-shopper.

3. Cashing in

Early automatic teller machines (**ATMs**) gave the **customer money before returning their bank card.** On receiving the money the customer would reach closure and hence often forget to take the card. when we complete some part of a task , our minds have a tendency to flush short-term memory in order to get on with the next job.

Modern ATMs **design changed** and it **returns the card first**! .

4. 7 ± 2 revisited

When we looked at short-term memory, we noted the general rule that people can hold 7 ± 2 items or chunks of information in short-term memory. It is a principle that people tend to remember but it can be misapplied. For example, it is often suggested that this means that **lists, menus and other groups of items** should be designed to be **no more than 7 items long**. But use of menus and lists of course has little to do with short-term memory – they are available in the environment as cues and so do not need to be remembered.

On the other hand the 7 ± 2 rule would apply in command line interfaces. Imagine a scenario where a **UNIX** user looks up a command in the manual. Perhaps the **command has a number of parameters of options**, to be applied in a particular order, and it is going to be applied to several files that have long path names. The user then has to hold the command, its parameters and the file path names in short term memory while he types them in. Here we could say that the task may cause problems if the number of items or chunks in the command line string is more than 7.

5. Memorable or secure?

As online activities become more widespread, people are having to remember more and more access information, such as **passwords and security checks.** The average active internet user may have separate passwords and user names for several email accounts, mailing lists, e-shopping sites, e-banking, online auctions and more! Remembering these passwords is not easy. From a security perspective it is important that passwords are random. Words and names are very easy to crack, hence the recommendation that passwords are frequently changed and constructed from random strings of letters and numbers. But in reality these are the hardest things for people to commit to memory. Hence many people will use the **same password for all their online activities** and will choose a word or a name that is easy for them to remember, in spite of the obviously increased security risks. Security here is in conflict with memorability!

A solution to this is to **construct a nonsense password** out of letters or numbers that will have meaning to you but will not make up a word in a dictionary (e.g. initials of names, numbers from significant dates or postcodes, and so on). Then what is remembered is the meaningful rule for

constructing the password, and not a meaningless string of alphanumeric characters.

6. Improve your memory

Many people can perform excellence in memory, where we have exercises like recalling the sequence of cards in a pack , or recounting π to 1000 decimal places.

There are exercises to improve memory abilities , one example discussed below,

Look at the list below of numbers and associated words:

1 bun	6 sticks
2 shoe	7 heaven
3 tree	8 gate
4 door	9 wine
5 hive	10 hen

Notice that the words sound similar to the numbers. Now think about the words one at a time and visualize them, in as much detail as possible. For example, for '1', think of a large, sticky iced bun, the base spiralling round and round, with raisins in it, covered in sweet, white, gooey icing. Now do the rest, using as much visualization as you can muster: imagine how things would look, smell, taste, sound, and so on.

This is your reference list, and you need to know it off by heart.

Having learnt it, look at a pile of at least a dozen odd items collected together by a colleague. The task is to look at the collection of objects for only 30 seconds, and then list as many as possible without making a mistake or viewing the collection again. Most people can manage between five and eight items, if they do not know any memory-enhancing techniques.

7. Chess: of human and artificial intelligence

A few years ago, Deep Blue, a **chess-playing computer**, beat Gary Kasparov, the world's top Grand Master, in a full tournament. This was the long-awaited breakthrough for the **artificial intelligence (AI)** community, who have traditionally seen chess as the ultimate test of their art.

However, despite the fact that computer chess programs can play at Grand Master level against human players, this does not mean they play in the same way. For each move played, Deep Blue investigated many millions of alternative moves and counter-moves. In contrast, a human chess player will only consider a few dozen. But, if the human player is good, these will usually be the right few dozen.

The ability to spot patterns allows a human to address a problem with far less effort than a **brute force approach**. In chess, the number of moves is such that finally brute force, applied fast enough, has **overcome human pattern-matching skill.** Many models of the mental processes have been heavily influenced by computation. It is worth remembering that although

there are similarities, computer 'intelligence' is very different from that of humans.

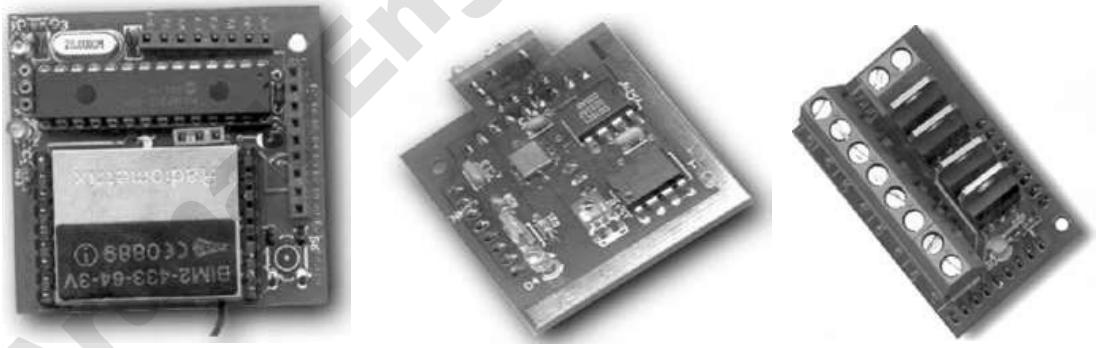
8. Feeling the road

In the **BMW 7 Series** you will find a single haptic feedback control for many of the functions that would normally have dedicated controls. It uses technology developed by Immersion Corporation who are also behind the force feedback found in many medical and entertainment haptic devices. The iDrive control slides backwards and forwards and rotates to give access to various menus and lists of options. The **haptic feedback** allows the user to feel 'clicks' appropriate to the number of items in the various menu lists

9. Smart-Its – making using sensors easy

Building systems with physical sensors is no easy task. You need a soldering iron, plenty of experience in electronics, and even more patience. Although some issues are unique to each sensor or project, many of the basic building blocks are similar – connecting simple microprocessors to memory and networks, connecting various standard sensors such as temperature, tilt, etc.

The **Smart-Its project** has made this job easier by creating a collection of components and an architecture for adding new sensors. There are a number of basic Smart-It boards – the photo on the left shows a microprocessor with wireless connectivity. Onto these boards are plugged a variety of modules – in the center is a sensor board including temperature and light, and on the right is a power controller.



10. Video recorder

A simple example of programming a **VCR from a remote control** shows that all **four translations** in the interaction cycle can affect the overall interaction.

Articulatory problem → Ineffective interaction is indicated by the user not being sure the VCR is set to record properly. This could be because the user has pressed the keys on the remote control unit in the wrong order.

performance translation → the VCR is able to record on any channel but the remote control lacks the ability to select channels, indicating a coverage problem

presentation problem → It may be the case that the VCR display panel does not indicate that the program has been set,

observational error → maybe the user does not interpret the feedback properly.

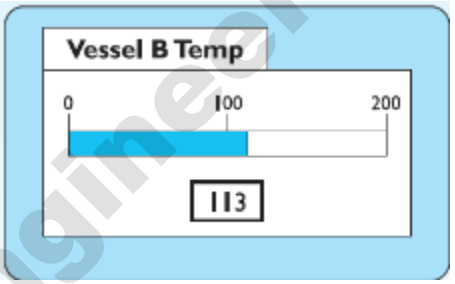
Any one or more of these deficiencies would give rise to ineffective interaction.

11. Industrial interfaces

Industrial interfaces raise some additional design issues rarely encountered in the office.

Glass interfaces vs. dials and knobs

The traditional machine interface consists of dials and knobs directly wired or piped to the equipment. Increasingly, some or all of the controls are replaced with a glass interface, a computer screen through which the equipment is monitored and controlled. Many of the issues are similar for the two kinds of interface, but glass interfaces do have some special advantages and problems. For a complex system, a glass interface can be both cheaper and more flexible, and it is easy to show the same information in multiple forms as shown below



For example, a data value might be given both in a precise numeric field and also in a quick to assimilate graphical form.

Indirect manipulation

The phrase 'direct manipulation' dominates office system design as in fig 3.5.

In a direct manipulation system, the user interacts with an artificial world inside the computer (for example, the electronic desktop). In contrast, an industrial interface is merely an intermediary between the operator and the real world. One implication of this indirectness is that the interface must provide feedback at two levels

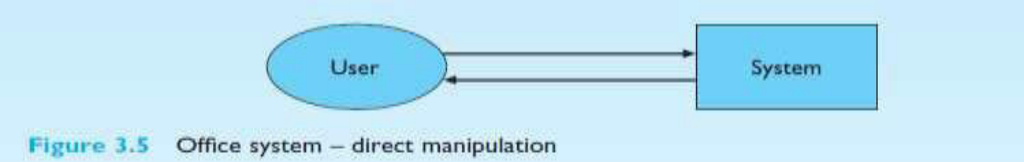


Figure 3.5 Office system – direct manipulation

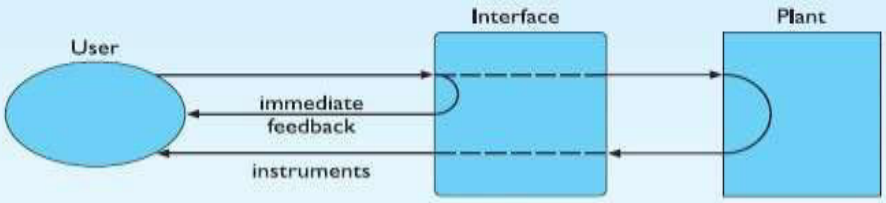


Figure 3.6 Indirect manipulation – two kinds of feedback

As in Figure 3.6 , At one level, the user must receive immediate feedback, generated by the interface, that keystrokes and other actions have been received. In addition, the user's actions will have some effect on the equipment controlled by the interface and adequate monitoring must be provided for this.

The indirectness also causes problems with simple monitoring tasks. Delays due to periodic sampling, slow communication and digital processing often mean that the data displayed are somewhat out of date. If the operator is not aware of these delays, diagnoses of system state may be wrong. These problems are compounded if the interface produces summary information displays. If the data comprising such a display are of different timeliness the result may be misleading.

12. Mixing styles

The UNIX windowing environments are interesting as the contents of many of the windows are often themselves simply command line or character-based programs . In fact, this mixing of interface styles in the same system is quite common, especially where older legacy systems are used at the same time as more modern applications.

It can be a problem if users attempt to **use commands and methods suitable for one environment** in another.

On the Apple Macintosh, HyperCard uses a point-and-click style. However, HyperCard stack buttons look very like Macintosh folders. If you double click on them, as you would to open a folder, your two mouse clicks are treated as separate actions. The first click opens the stack , but the second is then interpreted in the context of the newly opened stack, behaving in an apparently arbitrary fashion! This is an example of the **importance of consistency in the interface.**

13. Bank managers don't type . . .

The safe in most banks is operated by at least two keys, held by different employees of the bank. This makes it difficult for a bank robber to obtain both keys, and also protects the bank against light-fingered managers! ATMs contain a lot of cash and so need to be protected by similar

measures. In one bank, which shall remain nameless, the **ATM had an electronic locking device**. The machine could not be opened to replenish or remove cash until a long key sequence had been entered. In order to preserve security, the bank gave half the sequence to one manager and half to another,

so both managers had to be present in order to open the ATM. However, these were traditional bank managers who were not used to typing – that was a job for a secretary! So they each gave their part of the key sequence to a secretary to type in when they wanted to gain entry to the ATM. In fact, they both gave their respective parts of the key sequence to the same secretary. Happily the secretary was honest.

but the **moral** is **you cannot ignore social expectations and relationships when designing any sort of computer system, however simple it may be.**

14. Half the picture?

When systems are not designed to match the way people actually work, then users end up having to do 'work arounds'. **Integrated student records** systems are becoming popular in **universities in the UK**. They bring the benefits of integrating examination systems with enrolment and finance systems so all data can be maintained together and cross-checked. All very useful and time saving – in theory.

However, one commonly used system only holds a single overall mark per module for each student, whereas many modules on UK courses have multiple elements of assessment. Knowing a student's mark on each part of the assessment is often useful to academics making decisions in examination boards as it provides a more detailed picture of performance.

In many cases **staff** are therefore supplementing the official records system with their own **unofficial spreadsheets to provide** this information – making **additional work** for staff and **increased opportunity for error**.

15. Worked exercise (Unit –II)

Discuss the ways in which a full-page word processor is or is not a direct manipulation interface for editing a document using **Shneiderman's** criteria. What features of a modern word processor break the metaphor of composition with pen (or typewriter) and paper?

Answer We will answer the first point by evaluating the word processor relative to the criteria for direct manipulation given by Shneiderman.

Visibility of the objects of interest

The most important objects of interest in a word processor are the words themselves. Indeed, the visibility of the text on a continual basis was one of the major usability advances in moving from line-oriented to display-oriented editors.

Depending on the user's application, there may be other objects of interest in word processing that may or may not be visible. For example, are the margins for the text on screen similar to the ones which would eventually be printed? Is the spacing within a line and the line breaks similar? Are the different fonts and formatting characteristics of the text visible (without altering the spacing)? Expressed in this way, we can see the visibility criterion for direct manipulation as very similar to the criteria for a WYSIWYG interface.

Incremental action at the interface with rapid feedback on all actions

We expect from a word processor that characters appear in the text as we type them in at the keyboard, with little delay. If we are inserting text on a page, we might also expect that the format of the page adjust immediately to accommodate the new changes.

Various word processors do this reformatting immediately, whereas with others changes in page breaks may take some time to be reflected. One of the other important actions which requires incremental and rapid feedback is movement of the window using the scroll button. If there is a significant delay between the input command to move the window down and the actual movement of the window on screen, it is quite possible that the user will 'overshoot' the target when using the scrollbar button.

Reversibility of all actions, so that users are encouraged to explore without severe penalties

Single-step undo commands in most word processors allow the user to recover from the last action performed. One problem with this is that the user must recognize the error before doing any other action. More sophisticated undo facilities allow the user to retrace back more than one command at a time. The kind of exploration this reversibility provides in a word processor is best evidenced with the ease of experimentation that is now available for formatting changes in a document (font types and sizes and margin changes).

One problem with the ease of exploration is that emphasis may move to the look of a document rather than what the text actually says (style over content).

Syntactic correctness of all actions, so that every user action is a legal operation

WYSIWYG word processors usually provide menus and buttons which the user uses to articulate many commands. These interaction mechanisms serve to constrain the input language to allow only legal input from the user. Document markup systems, such as HTML and LaTeX, force the user to insert textual commands (which may be erroneously entered by the user) to achieve desired formatting effects.

Replacement of complex command languages with actions to manipulate directly the visible objects

The case for word processors is similar to that described above for syntactic correctness.

In addition, operations on portions of text are achieved many times by allowing the user to highlight the text directly with a mouse (or arrow keys). Subsequent action on that text, such as moving it or copying it to somewhere else, can then be achieved more directly by allowing the user to 'drag' the selected text via the mouse to its new location.

To answer the second question concerning the drawback of the pen (or typewriter) metaphor for word processing, compares the functionality of the space key in typewriting versus word processing. For a typewriter, the space key is passive; it merely moves the insertion point one space to the right. In a word processor, the space key is active, as it inserts a character (the space character) into the document. The functionality of the typewriter space key is

produced by the movement keys for the word processor (typically an arrow key pointing right to move forward within one line). In fact, much of the functionality that we have come to expect of a word processor is radically different from that expected of a typewriter, so much so that the typewriter as a metaphor for word processing is not all that instructive. In practice, modern typewriters have begun to borrow from word processors when defining their functionality

ARUNAI ENGINEERING COLLEGEDEPARTMENT OF

CSE

IV YEAR - VII SEMESTER

CS8079 – HUMAN COMPUTER INTERACTION (R2017)

UNIT 2 - 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.

INTERACTIVE DESIGN BASICS

A simple definition of design is: achieving goals within constraints

- goals - purpose
  - who is it for, why do they want it
- constraints
  - materials, platforms used
- trade-offs

Choosing which goals or constraints can be relaxed so that others can be met. For example, we might find that an eye-mounted video display, a bit like those used in virtual reality, would give the most stable image while walking along. However, this would not allow you to show friends, and might be dangerous if you were watching a gripping part of the movie as you crossed the road.

The golden rule of design

- The designs we produce may be different, but often the raw materials are the same. This leads us to the golden rule of design:

understand your materials.

- For Human–Computer Interaction the obvious materials are the human and the computer. That is we must:
  - understand computers
    – limitations, capacities, tools, platforms
  - understand people
    – psychological, social aspects, human error.

To err is human

- People make mistakes. This is not 'human error', an excuse to hide behind in accident reports, it is human nature. We are not infallible consistent creatures, but often make slips, errors and omissions.
- If you design using a physical material, you need to understand how and where failures would occur and strengthen the construction, build in safety features or redundancy. Similarly, if you treat the human with as much consideration as a piece of steel or concrete, it is obvious that you need to

understand the way human failures occur and build the rest of the interface accordingly

The central message – the user

This is the core of interaction design: put the user first, keep the user in the center and remember the user at the end.

THE PROCESS OF DESIGN

- A system has been designed and built, and only when it proves unusable do they think to ask how to do it right! In other companies usability is seen as equivalent to testing – checking whether people can use it and fixing problems, rather than making sure they can from the beginning. Simplified view of four main phases plus an iteration loop, focused on the design of interaction.
  a. **Requirements**: **what is wanted** 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
  b. **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.
  c. **Design**: there is a central stage when you move from what you want, to how to do it. There are numerous rules, guidelines and design principles that can be used to help.
  d. **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
  e. **Implementation and deployment**: Finally, when we are happy 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

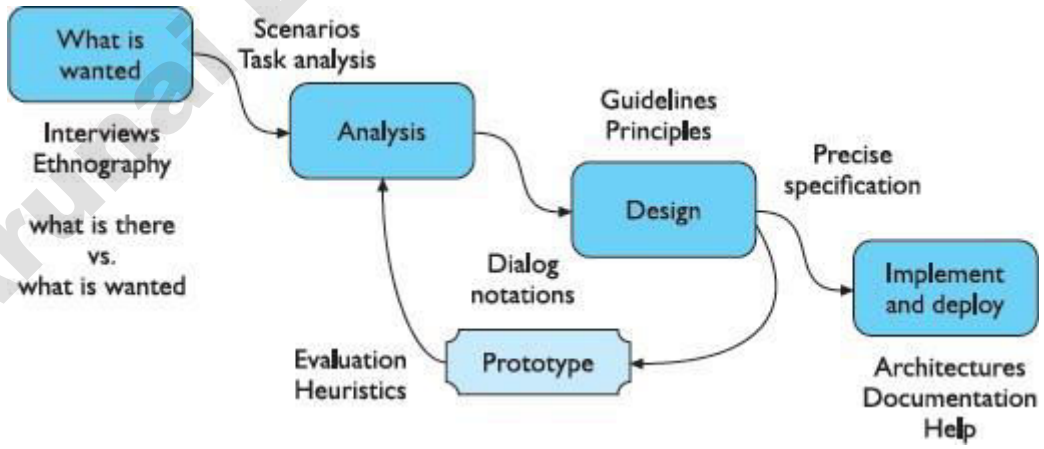


Fig 2.1: Interface Design Process

USER FOCUS

As we've already said, the start of any interaction design exercise must be the intended user or users. This is often stated as:

know your user

Know your user

- The start of any interaction design exercise must be the intended user or users. This is often stated as: know your users. So, how do you get to know your users?
  - **Who are they?:** The first thing to find out is who your users are. Are they young or old, experienced computer users or novices?
  - **Probably not like you!:** When designing a system it is easy to design it as if you were the main user: you assume your own interests and abilities. So often you hear a designer say 'but it's obvious what to do'. It may be obvious for her! This is not helped by the fact that many software houses are primarily filled with male developers
  - **Talk to them:** It is hard to get yourself inside someone else's head, so the best thing is usually to ask them. This can take many forms: structured interviews about their job or life, open-ended discussions, or bringing the potential users fully into the design process. The last of these is called participatory design
  - **Watch them:** A professional in any field is very practiced and can do things in the domain. An academic in the same field may not be able to do things, but she knows about the things in the domain. These are different kinds of knowledge and skill.

Persona

- One method that has been quite successful in helping design teams produce user focussed designs is the persona.
- the aspect of someone's character that is presented to or perceived by others
- description of an 'example' user
  - not necessarily a real person
- use as substitute user

example persona

Betty is 37 years old, She has been Warehouse Manager for five years and worked for Simpkins Brothers Engineering for twelve years. She didn't go to university, but has studied in her evenings for a business diploma. She has two children aged 15 and 7 and does not like to work late. She did part of an introductory in-house computer course some years ago, but it was interrupted when she was promoted and could no longer afford to take the time. Her vision is perfect, but her right-hand movement is slightly restricted following an industrial accident 3 years ago. She is enthusiastic about her work and is happy to delegate responsibility and take suggestions from her staff. However, she does feel threatened by the introduction of yet another new computer system (the third in her time at SBE).

Cultural probes

Cultural probes are small packs of items designed to provoke and record comments in various ways. They are given to people to take away and to open and use in their own environment. For example, one probe pack for the domestic environment includes a glass with a paper sleeve

- direct observation sometimes hard
    - in the home
    - psychiatric patients, …
- probe packs are introduced with items to prompt responses
    - e.g. glass to listen at wall, camera, postcard
- given to people to open in their own environment
  they record what is meaningful to them
- used to …
  - inform interviews, prompt ideas, enculture designers

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. This gives answers for the following questions
    - what will users want to do?
    - step-by-step walkthrough
        - what can they see (sketches, screen shots)
        - what do they do (keyboard, mouse etc.)
        - what are they thinking?
    - use and reuse throughout design

scenario – movie player

Brian would like to see the new film "Moments of Significance" and wants to invite Alison, but he knows she doesn't like "arty" films. He decides to take a look at it to see if she would like it and so connects to one of the movie sharing networks. He uses his work machine as it has a higher bandwidth connection, but feels a bit guilty. He knows he will be getting an illegal copy of the film, but decides it is OK as he is intending to go to the cinema to watch it. After it downloads to his machine he takes out his new personal movie player. He presses the 'menu' button and on the small LCD screen he scrolls using the arrow keys to 'bluetooth connect' and presses the select button. On his computer the movie download program now has an icon showing that it has recognised a compatible device and he drags the icon of the film over the icon for the player.

On the player the LCD screen says "downloading now", a percent done indicator and small whirling icon. … … …

Scenarios can be used to:

i) 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.

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

iii) Express dynamics Individual screen shots and pictures give you a sense of what a system would look like, but not how it behaves. This linearity has both positive and negative points:

iv) 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.

v) 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.

NAVIGATION

- 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 focusing on the computer system itself. You interact at several levels:

a. 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.

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

c. 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.

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!

d. Environment The word processor has to read documents from disk, perhaps some are on remote networks. You swap between applications, perhaps cut and paste.Here we will consider two main kinds of issue:

- local structure
  – looking from one screen or page out
- 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.
- 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.
- To do this goal seeking, each state of the system or each screen needs to give the 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.
  - knowing where you are
  - knowing what you can do
  - knowing where you are going – or what will happen
  - knowing where you've been – or what you've done.

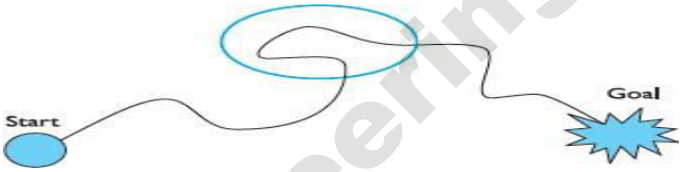
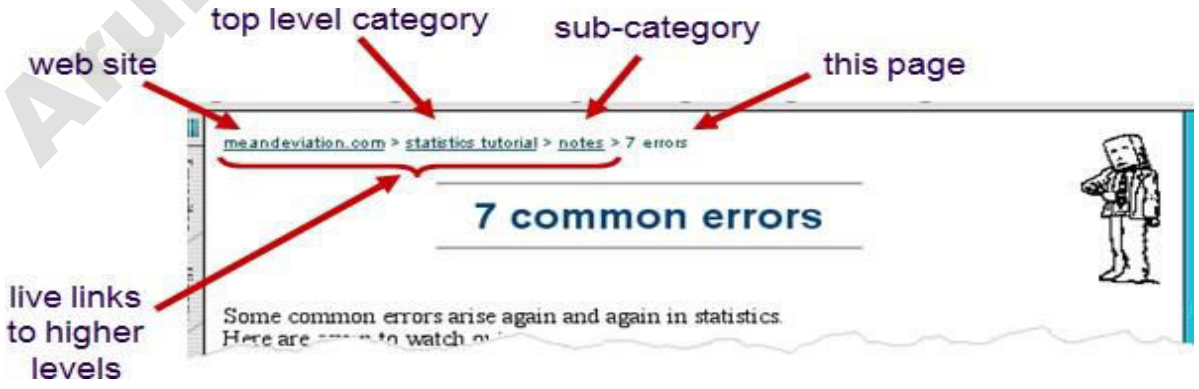


Fig.2.2 Local Structure

Four golden rules

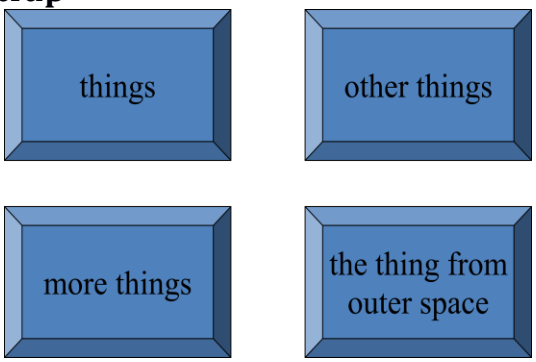
- knowing where you are
- knowing what you can do
- knowing where you are going
  - or what will happen
- knowing where you've been
  - or what you've done

Where you are



navigation system that shows a user's location in a site or web app.

beware the big button trap



Public information systems often have touch screens and so have large buttons. Watch someone using one of these and see how often they go to the wrong screen and have to use 'back' or 'home' to try again. If you look more closely you will find that each button has only one or two words on it giving the title of the next screen, and possibly some sort of icon

Modes

- lock to prevent accidental use ...
  - remove lock - 'c' + 'yes' to confirm
  - frequent practiced action
- if lock forgotten
  - in pocket 'yes' gets pressed
  - goes to phone book
  - in phone book ...
    'c' – delete entry
    'yes' – confirm
    ... oops !

Global Structure – Hierarchical Organization

- 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 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.
- 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.

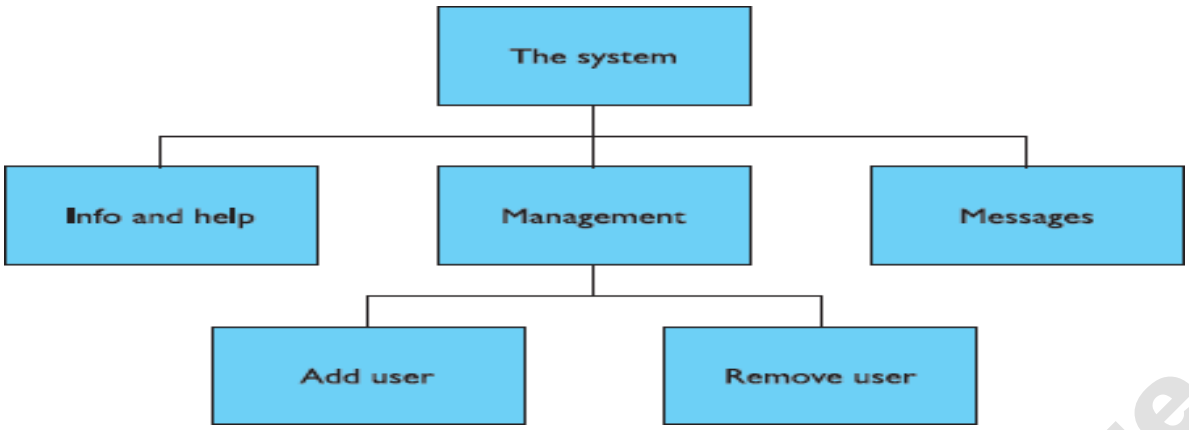


Fig 2.3: Application Functional Hierarchy

Global structure – dialog

- 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.
- A 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.
- 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. A simple way is to use a network diagram showing the principal states or screens. The linked together with arrows. This can:
  - show what leads to what
  - show what happens when
  - include branches and loops
  - be more task oriented than a hierarchy.

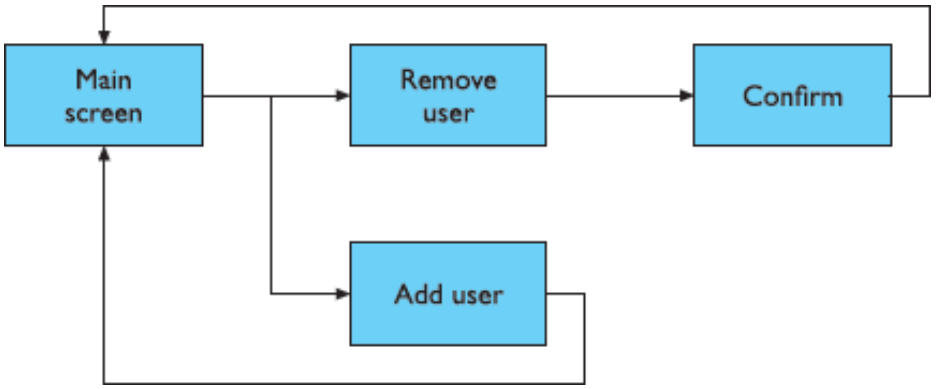


Fig 2.4: Network of screens/states

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

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:

a. 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.

b. 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.

c. 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.

SCREEN DESIGN AND LAYOUT

- A single screen image often has to present information clearly and also act as the locus for interacting with the system. This is a complex area, involving some of the psychological understanding as well as aspects of graphical design. 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

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

- Available tools
  - **Grouping of items**
  - **Order of items**
  - **Decoration - fonts, boxes etc.**
  - **Alignment of items**
  - **White space between items**

a. Grouping and structure

- If things logically belong together, then we should normally physically group them together. This may involve multiple levels of structure. For example, in Figure 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.

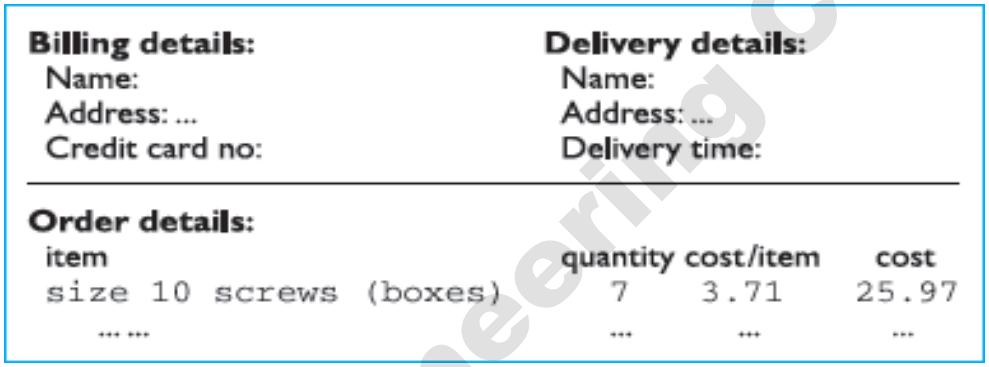


Fig 2.5: Grouping Related Items in an order screen

This reflects the following logical structure:

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

b. Order of groups and items

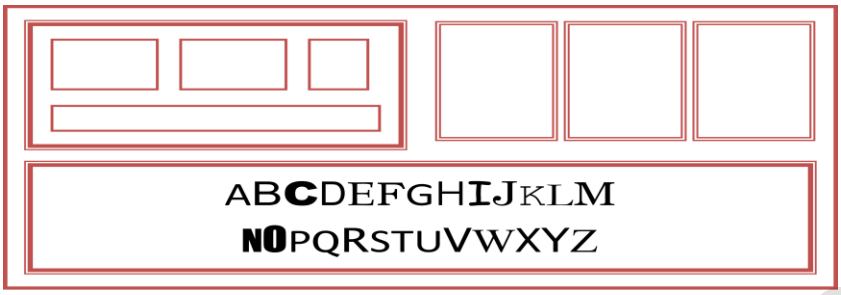
- If we look at above Figure again we can see that the screen seems to naturally suggest reading or filling in the billing details first, followed by the delivery details, followed by the individual order items. Is this the right order?
- In general we need to think: what is the natural order for the user? This should normally match the order on screen. For data entry forms or dial

c. Decoration

- 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. See how the

buttons differ in using the foreground and back-ground colors (green and gold) so that groups are associated with one another.

- use boxes to group logical items
- use fonts for emphasis, headings



d. Alignment

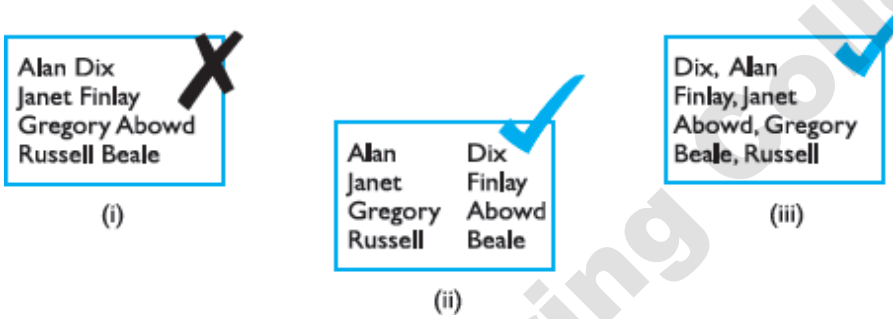


Fig 2.6: Looking up surnames

Alignment - numbers

alignment - numbers

think purpose!

which is biggest?

532.56
179.3
256.317
15
73.948
1035
3.142
497.6256

Difficult to identify the biggest number because of un aligned numbers.

visually:
long number = big number

align decimal points
or right align integers

627.865
1.005763
382.583
2502.56
432.935
2.0175
652.87
56.34

- 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. 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 mini histogram. Items like names are particularly difficult. Multiple column lists require more care.

- Text columns have to be wide enough for the largest item, which means you can get large gaps between columns shows an example of this (i), and you can see how hard this makes it for your eye to scan across the rows. There are several visual ways to deal with this including: (ii)'leaders' – lines of dots linking the columns; and (iii) using soft tone grays or colors behind rows or columns. This is also a time when it may be worth breaking other 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, but not if you needed frequently to look up names

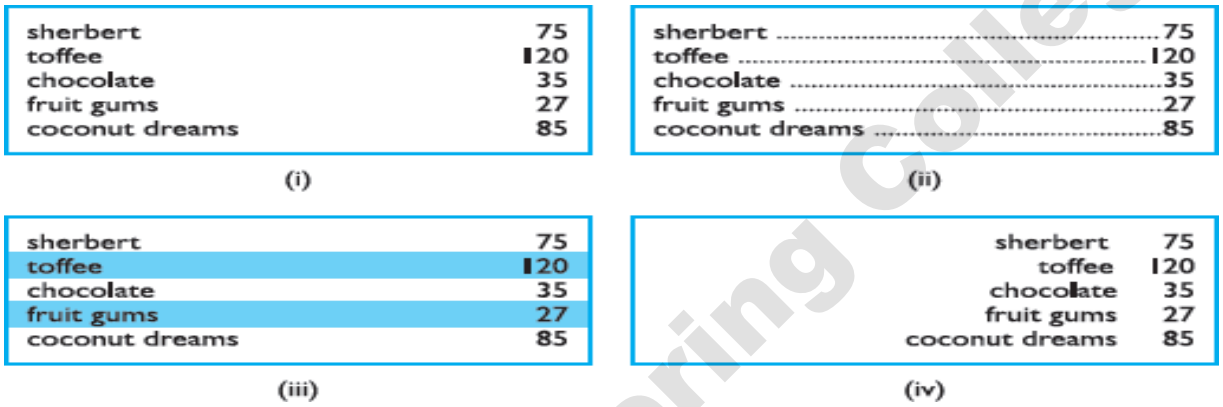


Fig 2.7: Managing Multiple Columns

e. White space

- 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.
- If one ignores the 'content' of a screen and instead concentrates on the counter – the space between the elements – one can get an overall feel for the layout. If elements that are supposed to be related look separate when you focus on the counter, then something is wrong. Screwing up your eyes so that the screen becomes slightly blurred is another good technique for taking your attention away from the content and looking instead at the broad structure.
- Space can be used in several ways. Some of these are shown in Figure The colored areas represent continuous areas of text or graphics. (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.

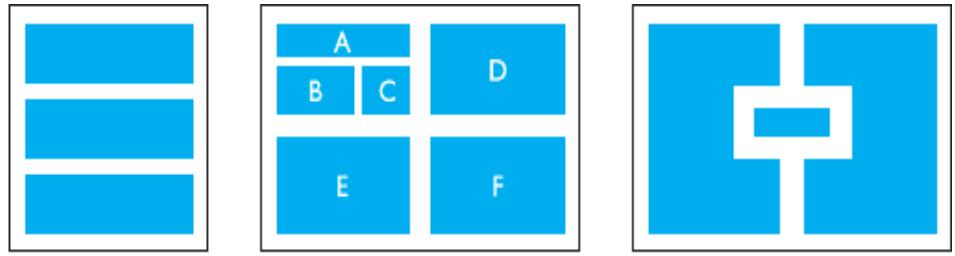
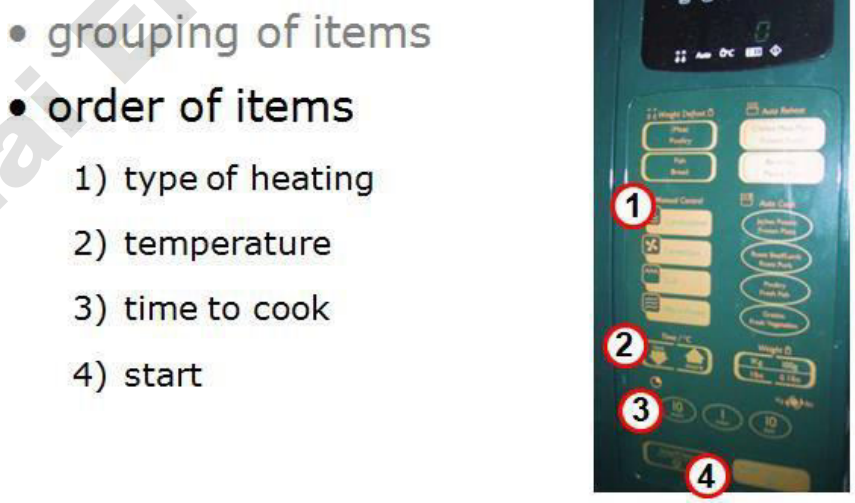


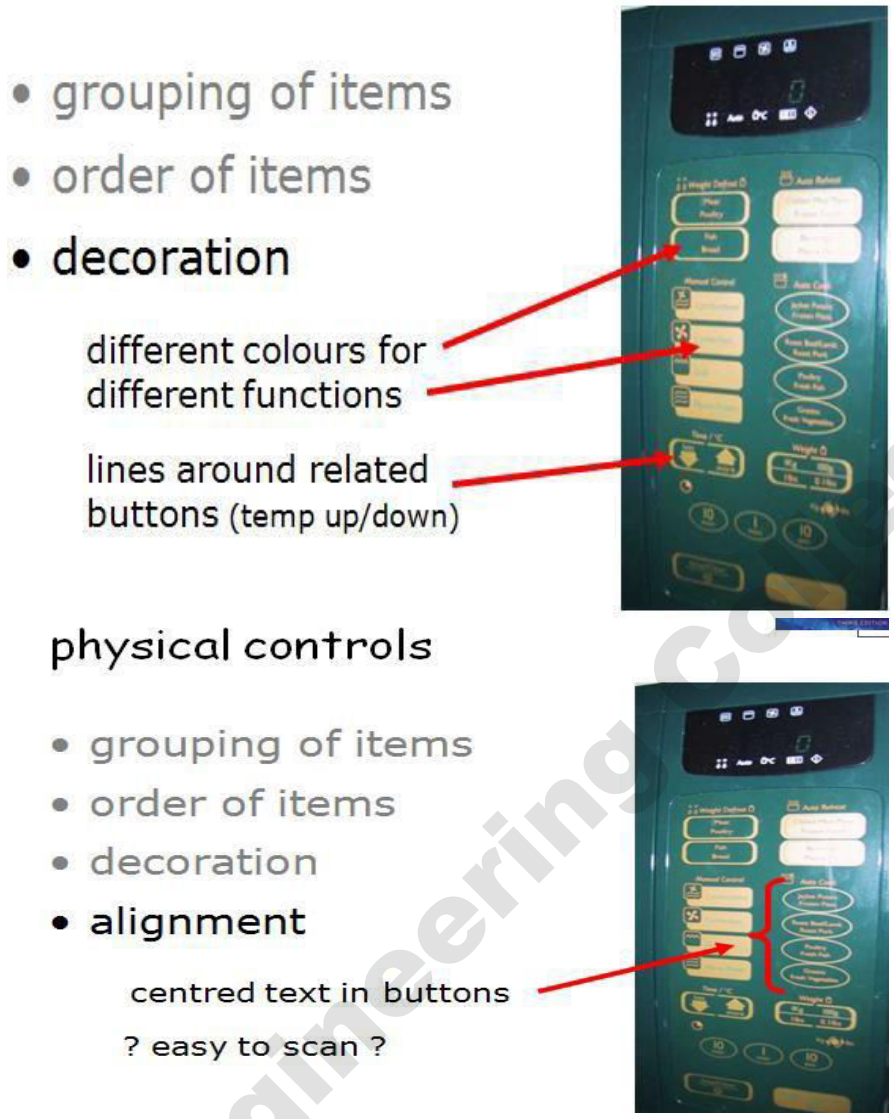
Fig 2.8: Using white space in layout

(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.

(iii) We can see space used to highlight. This is a technique used frequently in magazines to highlight a quote or graphic.

Example : physical controls in Microwave control panel





User Action And Control

a. Entering information

- Some of the most complicated and difficult screen layouts are found in forms-based interfaces and dialog boxes. In each case the screen consists not only of information presented to the user, but also of places for the userto enter information or select options. Actually, many of the same layout issues for data presentation also apply to fields for data entry.
- Alignment is still important. It is especially common to see the text entry boxes aligned in a jagged fashion because the field names are of different lengths. This is an occasion where right-justified text for the field labels may be best or, alternatively, in a graphical interface a smaller font can be used for field labels and the labels placed just above and to the left of the field they refer to.

b. Knowing what to do

- Some elements of a screen are passive, simply giving you information; others are active, expecting you to fill them in, or do something to them. It is often not even clear which elements are active, let alone what the effect is likely to be when you interact with them. This is one of the reasons for platform and company style guides.

- If everyone designs buttons to look the same and menus to look the same, then users will be able to recognize them when they see them.

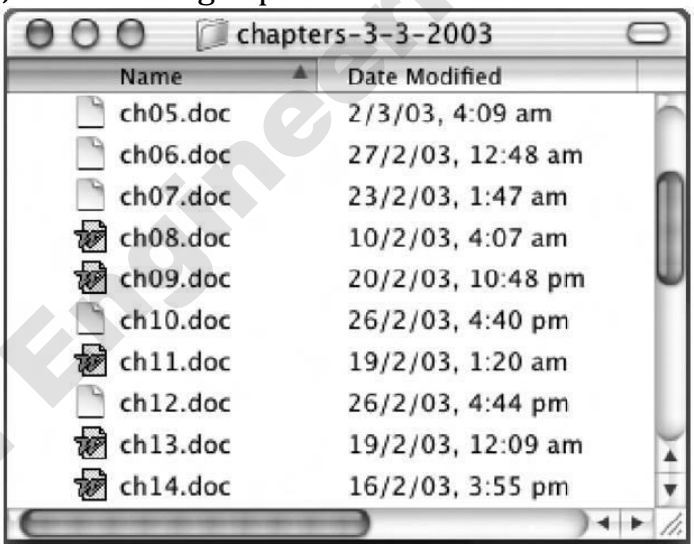
c. Affordances

- These are especially difficult problems in multimedia applications where one may deliberately adopt a non-standard and avant-garde style. The psychological idea of affordance says that things may suggest by their shape and other attributes what you can do to them: a handle affords pulling or lifting; a button affords pushing. These affordances can be used when designing novel interaction elements. One can either mimic real-world objects directly, or try to emulate the critical aspects of those objects.

Appropriate appearance

a. Presenting information

- The way of presenting information on screen depends on the kind of information: text, numbers, maps, tables; on the technology available to present it: character display, line drawing, graphics, virtual reality; and, most important of all, on the purpose for which it is being used.
- For more complex numerical data, we may be considering scatter graphs, histograms or 3D surfaces; for hierarchical structures, we may consider outlines or organization diagrams. But, no matter how complex the data, the principle of matching presentation to purpose remains.
- We have an advantage when presenting information in an interactive system in that it is easy to allow the user to choose among several representations, thus making it possible to achieve different goals.



Alphabetic file listing.

b. Aesthetics and utility

- The conflict between aesthetics and utility can also be seen in many 'well designed' posters and multimedia systems. In particular, the backdrop behind text must have low contrast in order to leave the text readable; this is often not the case and graphic designers may include excessively complex and strong backgrounds because they look good.

c. Making a mess of it: color and 3D

- One of the worst features in many interfaces is their appalling use of color. This is partly because many monitors only support a limited range of primary colors and partly because, as with the overuse of different fonts in word processors, the designer got carried away.

- Aside from issues of good taste, an overuse of color can be distracting that a significant proportion of the population is color blind, may mean that parts of the text are literally invisible to some users. In general, color should be used sparingly and not relied upon to give information, but rather to reinforce other attributes.
- The increasing use of 3D effects in interfaces has posed a whole new set of problems for text and numerical information. Whilst excellent for presenting physical information and certain sorts of graphs, text presented in perspective can be very difficult to read and the all too common 3D pie chart is all but useless.

d. Localization / internationalization

- If you are working in a different country, you might see a document being word processed where the text of the document and the file names are in the local language, but all the menus and instructions are still in English. The process of making software suitable for different languages and cultures is called localization or internationalization.
- It is clear that words have to change and many interface construction toolkits make this easy by using resources. When the program uses names of menu items, error messages and other text, it does not use the text directly, but instead uses a resource identifier, usually simply a number.

ITERATION AND PROTOTYPING

- This often starts early on with paper designs and storyboards being demonstrated to colleagues and potential users. 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. 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.
- The Figure 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.

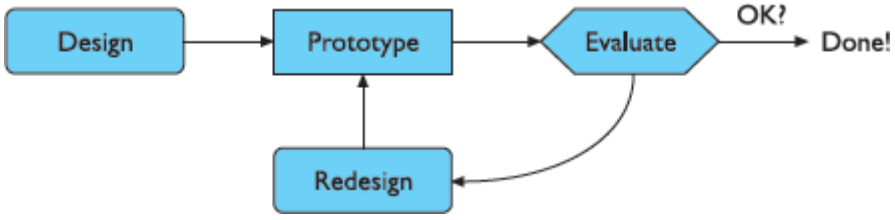
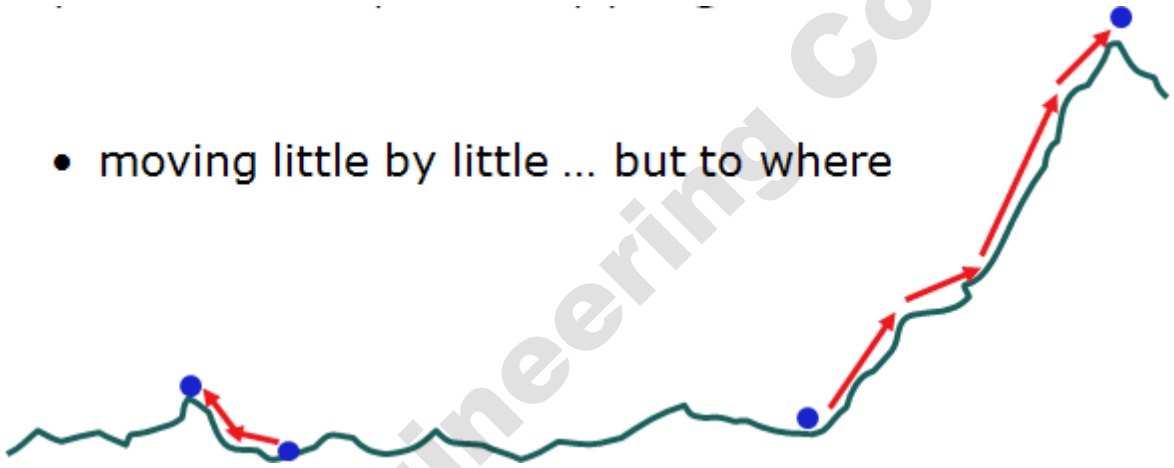


Fig 2.9: Role of Prototyping

- 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. If you wanted to climb as high as you could, you would probably book a plane to the Himalayas, not Cambridgeshire..
- A really 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. Another approach, very common in graphical design, is to have several initial design ideas and drop them one by one as they are developed further.
- pitfalls of prototyping
    - 1. need a good start point
    - 2. need to understand what is wrong



HCI IN SOFTWARE PROCESS

- The design goal is to provide reliable techniques for the repeated design of successful and usable interactive systems. It is therefore necessary that we go beyond the exercise of identifying paradigms and examine the process of interactive system design.
- Within computer science there is already a large sub discipline that addresses the management and technical issues of the development of software systems – called software engineering. One of the cornerstones of software engineering is the software life cycle, which describes the activities that take place from the initial concept formation for a software system up until its eventual phasing out and replacement.
- The important point that we would like to draw out is that issues from HCI affecting the usability of interactive systems are relevant within all the activities of the software life cycle. Therefore, software engineering for interactive system design is not simply a matter of adding one more activity that slots in nicely with the existing activities in the life cycle. Rather, it involves techniques that span the entire life cycle.

SOFTWARE LIFE CYCLE

- A fundamental feature of software engineering, therefore, is that it provides the structure for applying techniques to develop software systems. The software life cycle is an attempt to identify the activities that occur in software development. These activities must then be ordered in time in any development project and appropriate techniques must be adopted to carry them through.
- 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. Typically, the customer and the designer are groups of people and some people can be both customer and designer. It is often important to distinguish between the customer who is the client of the designing company and the customer who is the eventual user of the system.
- These two roles of customer can be played by different people. The group of people who negotiate the features of the intended system with the designer may never be actual users of the system. This is often particularly true of web applications.

Activities in the life cycle

- The graphical representation is reminiscent of a waterfall, in which each activity naturally leads into the next. The analogy of the waterfall is not completely faithful to the real relationship between these activities, but it provides a good starting point.

Requirements specification

- It involves eliciting information from the customer about the work environment, or domain, in which the final product will function. Aspects of the work domain include not only the particular functions that the software product must perform but also details about the environment in which it must operate, such as the people whom it will potentially affect and the new product's relationship to any other products which it is updating or replacing. It begins at the start of product development.
- Though the requirements are from the customer's perspective, if they are to be met by the 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).

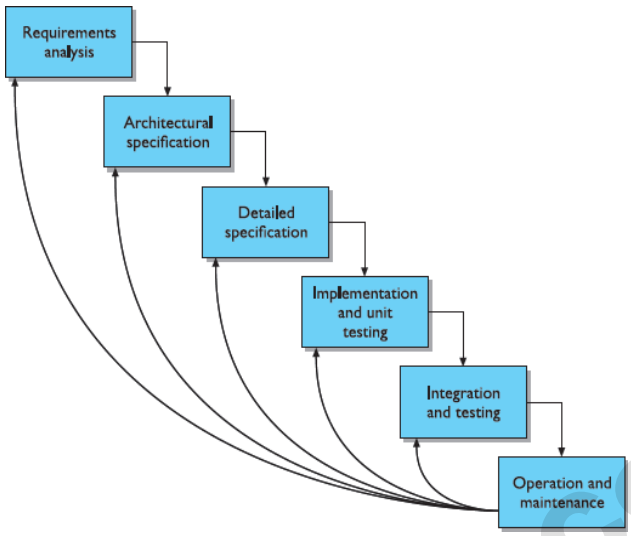


Fig 2.10: Feedback from maintenance activity to other design activity

Detailed design

- 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.Typically, there will be more than one possible refinement of the architectural component that will satisfy the behavioral constraints.
- Choosing the best refinement is often a matter of trying to satisfy as many of the non-functional requirements of the system as possible. 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

- After coding,the component can be tested to verify that it performs correctly, according to some test criteria.

Integration and testing

- Enough components have been implemented and individually tested, they must be integrated as described in the architectural design. Further testing is done to ensure correct behavior and acceptable use of any shared resources. It is also possible at this time to perform some acceptance testing with the customers to ensure that the system meets their requirements.

Maintenance

- Category of maintenance, until such time as a new version of the product demands a total redesign or the product is phased out entirely. 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.

Validation and verification

- Throughout the 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. Verification of a design will most often occur within a single life-cycle activity or between two adjacent activities.

- 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.
- The 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.
- Languages with a mathematical foundation allow reasoning and proof in the objective sense. An argument based entirely within some mathematical language can be accepted or refuted based upon universally accepted measures. A proof can be entirely justified by the rules of the mathematical language, in which case it is considered a formal proof.
- More common is a rigorous proof, which is represented within some mathematical language but which relies on the understanding of the reader to accept its correctness without appeal to the full details of the argument, which could be provided but usually are not. The difference between formality and rigour is in the amount of detail the prover leaves out while still maintaining acceptance of the proof.
- Proofs that are for verification of a design can frequently occur within one language or between two languages which both have a precise mathematical semantics.
- Validation proofs 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.

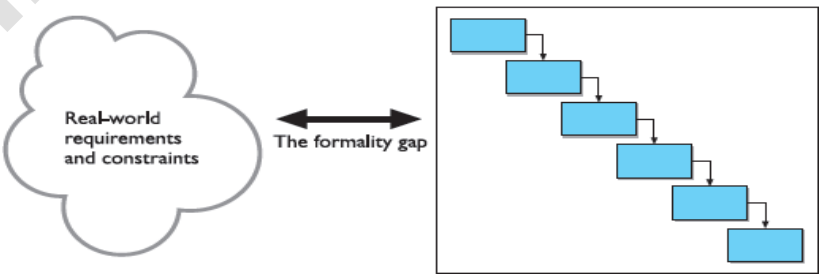


Fig 2.11: The formality gap between the real world and structured design

Interactive systems and the software life cycle

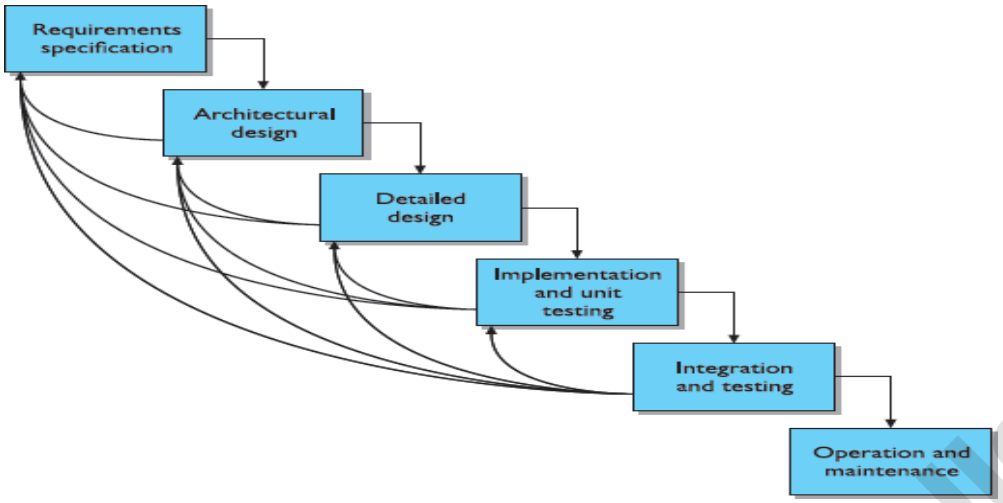


Fig 2.12: Representing Iteration in the Waterfall Model

- The traditional software life cycle suits a principled approach to design; that is, if we know what it is we want to produce from the beginning, then we can structure our approach to design in order to attain the goal.
- 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.

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 [377] and by Nielsen at Bellcore [260,261]. Engineering depends on interpretation against a shared background of meaning, agreed goals and an understanding of how satisfactory completion will be judged.
- 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.
- The danger with this limited focus is that much of the work that is accomplished in interaction involves more than just the surface features of the systems used to perform that work. 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.

- The ultimate test of usability based on measurement of user experience. Usability specification consists of
  - usability attribute/principle
  - measuring concept

- measuring method
- now level/ worst case/ planned level/ best case

- For each attribute, six items are defined to form the usability specification of that attribute. Table 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].

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

Table 2.1: Sample Usability Specification for undo with a VCR

- The below tables and 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.

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

Table 2.2: Criteria by which measuring method can be determined

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

Table2.3: Possible ways to set measurement levels in a usability specification

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. When the designer knows what the actions and situation will be, then she can set goals for measured observations. However, at early stages of design, designers do not have this information

.

ISO usability standard 9241

It adopts traditional usability categories:

- effectiveness
  - can you achieve what you want to?
- efficiency
  - can you do it without wasting effort?
- satisfaction
  - do you enjoy the process?

Some 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

PROTOTYPING IN PRACTICE

- 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
    - incremental
    - Evolutionary

a. 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. It 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.

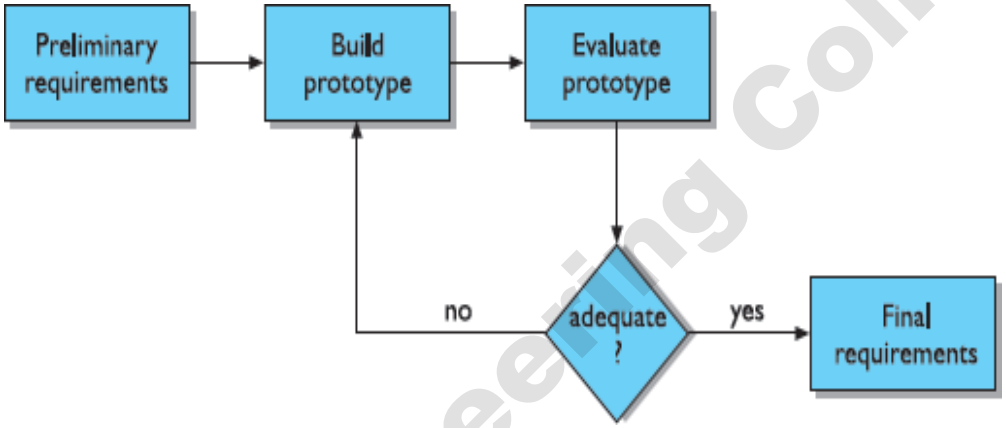


Fig 2.13: Throw-away prototyping with requirement specification

a. 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.

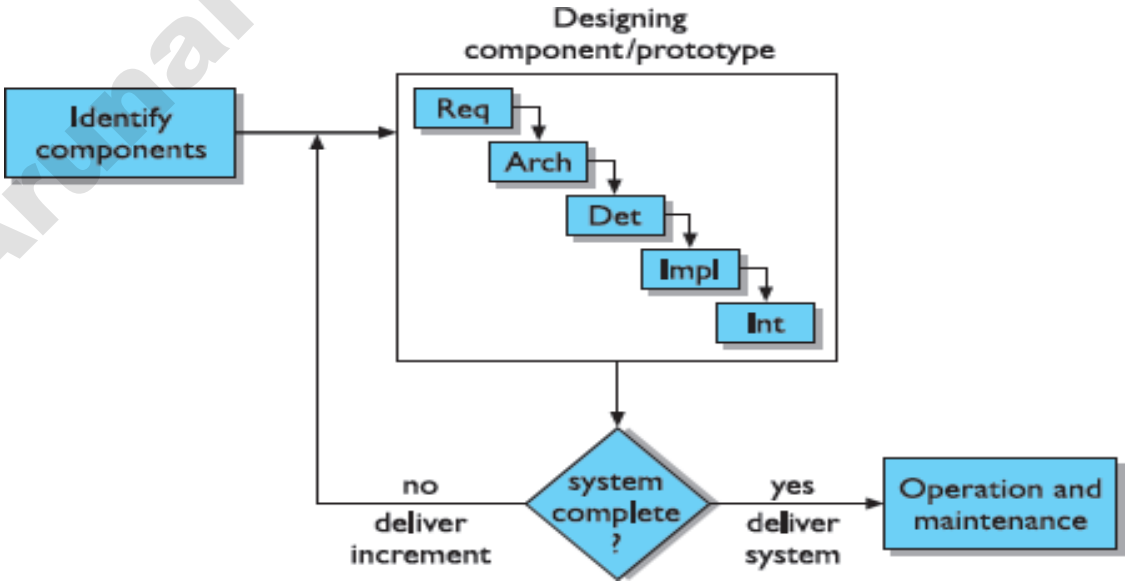


Fig 2.14: Incremental prototyping within the life cycle

b. 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.

- 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. An animation of requirements can involve no 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. Regardless of the level of functionality, the importance of a prototype lies in its projected realism.

- The prototype of an interactive system is used to test requirements by evaluating their impact with real users. An honest appraisal of the requirements of the final system can only be trusted if the evaluation conditions are similar to those anticipated for the actual operation. On the management side, there are several potential problems.

a. 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.

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

c. Non-functional features 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.

d. 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.

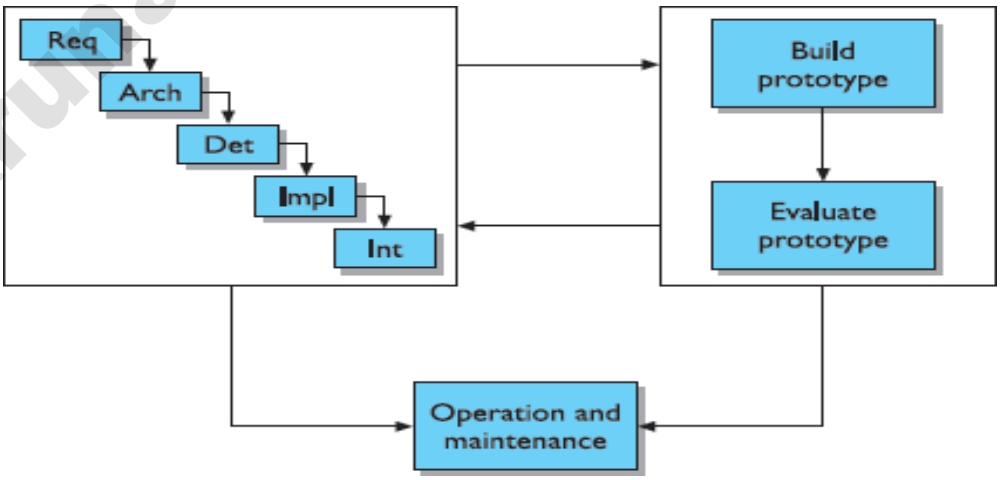


Fig 2.15: Evolutionary Prototyping throughout the life cycle

Techniques for prototyping

a. Storyboards: 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

b. Limited functionality simulations: The 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.

c. **High-level programming support:** HyperTalk was an example of a special purpose high-level programming language which makes it easy for the designer to program certain features of an interactive system at the expense of other system features like speed of response or space efficiency. These high-level programming languages allow the programmer to abstract away from the hardware specifics and think in terms that are closer to the way the input and output devices are perceived as interaction devices.

Warning about iterative design

- The ideal model of iterative design, in which a rapid prototype is designed ,evaluated and modified until the best possible design is achieved in the given project time, is appealing with two problems.

i) First, it is often the case that design decisions made at the very beginning of the prototyping process are wrong and, in practice, design inertia can be so great as never to overcome an initial bad decision

ii) Second problem is slightly more subtle, and serious. If, in the process of evaluation, a potential usability problem is diagnosed, it is important to understand the reason for the problem and not just detect the symptom

DESIGN RATIONALE

- Designing any computer system, many decisions are made as the product goes from a set of vague customer requirements to a deliverable entity. Often it is difficult to recreate the reasons, or rationale, behind various design decisions.
- 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 .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.

❖ 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.

❖ 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.

- There are some issues that distinguish the various techniques in terms of their usability within design itself. We can use these issues to sketch an

informal rationale for design rationale. One issue is the degree to which the technique impinges on the design process.

- Another issue is the cost of using the technique, both in terms of creating the design rationale and in terms of accessing it once created. A related issue is the amount of computational power the design rationale provides and the level to which this is supported by automated tools

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
- 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

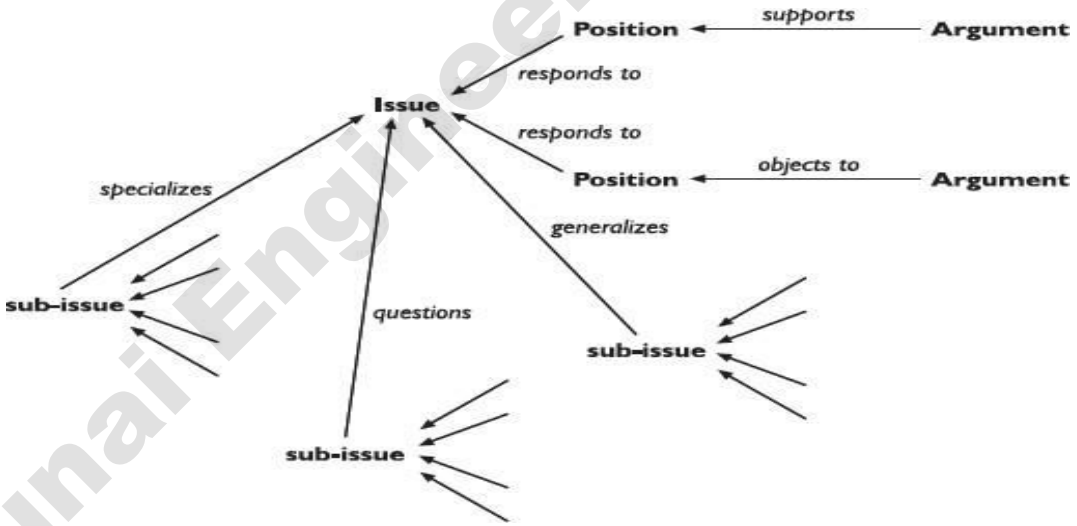


Fig 2.16: Structure of a gIBIS rationale

- The above Figure 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.

Design space analysis

- MacLean and colleagues 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.

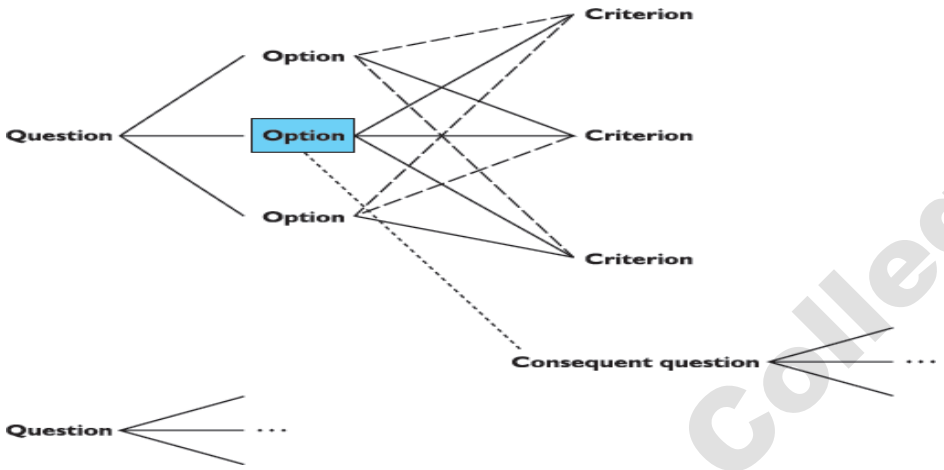


Fig 2.17: QOC notation

- 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 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. 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. It can be difficult to decide the right set of criteria with which to assess the options. 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.

Psychological design rationale

- The psychological design rationale has been introduced by Carroll and Rosson, and before we describe the application of the technique. 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.
- Carroll refers to this real-life phenomenon as the task–artifact cycle. He provides a good example of this cycle through the evolution of the electronic spreadsheet. 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.

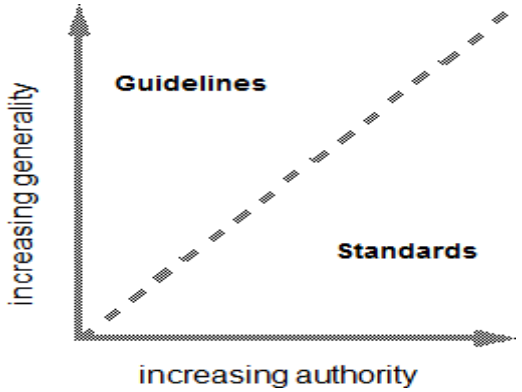
DESIGN RULES

- **Design**– the goal driven problem solving process
- **Design rules-** We require design rules, which are rules a designer can follow in order to increase the usability of the eventual software product. We can classify these rules along two dimensions, based on the rule's authority and generality.

➢ **authority** ➔ mean an indication of whether or not the rule must be followed in design or whether it is only suggested.
➢ **Generality** ➔ mean whether the rule can be applied to many design situations or whether it is focused on a more limited application situation.

- We will consider a number of different types of design rules.
  a) **Principles** are abstract design rules, with high generality and low authority.
  b) **Standards** are specific design rules, high in authority and limited in application.
  c) **Guidelines** tend to be lower in authority and more general in application.
- Design rules for interactive systems can be supported by psychological, cognitive, ergonomic, sociological, economic or computational theory, which may or may not have roots in empirical evidence.
- Designers do not always have the relevant background in psychology, cognitive science, ergonomics, sociology, and business or computer science necessary to understand the consequences of those theories in the instance

of the design they are creating. Can make another rough distinction between principles, standards and guidelines.



- Principles are derived from knowledge of the psychological, computational and sociological aspects of the problem domains and are largely independent of the technology they depend to a much greater extent on a deeper understanding of the human element in the interaction. They can therefore be applied widely but are not so useful for specific design advice.
- Guidelines are less abstract and often more technology oriented, but as they are also general, it is important for a designer to know what theoretical evidence there is to support them. A designer will have less of a need to know the underlying theory for applying a standard.
- Design rules are mechanisms for restricting the space of design options, preventing a designer from pursuing design options that would be likely to lead to an unusable system. Thus, design rules would be most effective if they could be adopted in the earliest stages of the life cycle, such as in requirements specification and architectural design, when the space of possible designs is still very large

PRINCIPLES

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

1. **Learnability** – the ease with which new users can begin effective interaction and achieve maximal performance.
2. **Flexibility** – the multiplicity of ways in which the user and system exchange information.
3. **Robustness** – the level of support provided to the user in determining successful achievement and assessment of goals.

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

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	–

Table 2.4: Summary of principles affecting learnability

a. Predictability: 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..

b. Synthesizability: It 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.

c. Familiarity: The familiarity of an interactive system measures the correlation between the user's existing knowledge and the knowledge required for effective interaction. Familiarity has to do with a user's first impression of the system.

d. Generalizability: The generalizability of an interactive system supports this activity, leading to a more complete predictive model of the system for the user. We can apply generalization to situations in which the user wants to apply knowledge that helps achieve one particular goal to another situation where the goal is in some way similar. Generalizability can be seen as a form of consistency. Generalization can occur within a single application or across a variety of applications.

e. Consistency: Consistency relates to the likeness in behavior arising from similar situations or similar task objectives. Consistency is probably the most widely mentioned principle 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.

Flexibility

- Flexibility refers to the multiplicity of ways in which the end-user and the system exchange information.

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

Table 2.5: Summary of principles affecting flexibility

a. Dialog initiative: The system can initiate all dialogs, in which case the user simply responds to requests for information. We call this type of dialog system pre-emptive.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.

b. 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.

c. 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.

d. Substitutivity: It requires that equivalent values can be substituted for each other. For example, in considering the form of an input expression to determine the margin for a letter, you may want to enter the value in either inches or centimeters.

e. Customizability: Customizability is the modifiability of the user interface by the user or the system. From the system side, we are not concerned with modifications that would be attended to by a programmer actually changing the system and its interface during system maintenance.

Robustness

- The robustness of that interaction covers features that support the successful achievement and assessment of the goals. Here, we describe principles that support 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	Task completeness, task adequacy

Table 2.6: Summary of principles affecting Robustness

a. Observability: It allows the user to evaluate the internal state of the system by means of its perceivable representation at the interface. The evaluation allows the user to compare the current observed state with his intention within the task–action plan, possibly leading to a plan revision. It can be discussed through five other principles: browsability, defaults, reachability, persistence and operation visibility.

Browsability allows the user to explore the current internal state of the system via the limited view provided at the interface.

Defaults It also reduces the number of physical actions necessary to input a value. Thus, providing default values is a kind of error prevention mechanism. There are two kinds of default values: static and dynamic. Static defaults do not evolve with the session

Reachability refers to the possibility of navigation through the observable system states. There are various levels of reachability that can be given precise mathematical definitions , but the main notion is whether the user can navigate from any given state to any other state.

Persistence deals with the duration of the effect of a communication act and the ability of the user to make use of that effect. The effect of vocal communication does not persist except in the memory of the receiver. Visual communication, on the other hand, can remain as an object which the user can subsequently manipulate long after the act of presentation.

b. Recoverability: It is the ability to reach a desired goal after recognition of some error in a previous interaction. There are two directions in which recovery can occur, forward or backward.

- Forward error recovery involves the acceptance of the current state and negotiation from that state towards the desired state. Forward error recovery may be the only possibility for recovery if the effects of interaction are not revocable.
- Backward error recovery is an attempt to undo the effects of previous interaction in order to return to a prior state before proceeding. In a text editor, a mistyped keystroke might wipe out a large section of text which you would want to retrieve by an equally simple undo button

c. Responsiveness: It 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. Significant as absolute response time is response time stability. Response time stability covers the invariance of the duration for identical or similar computational resources.

d. Task conformance: Task completeness addresses the coverage issue and task adequacy addresses the user's understanding of the tasks. Task completeness refers to the level to which the system services can be mapped onto all of the user tasks.

Standards

- set by national or international bodies to ensure compliance by a large community of designers standards require sound underlying theory and slowly changing technology
- hardware standards more common than software high authority and low level of detail
- ISO 9241 defines usability as effectiveness, efficiency and satisfaction with which users accomplish tasks (refer pg no 23)

GUIDELINES

- It concern in examining the wealth of available guidelines is in determining their applicability to the various stages of 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). 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
- 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. 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.
- 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.

RULES

- There are many sets of heuristics, but the most well used are
  - Nielsen's ten heuristics,
  - Shneiderman's eight golden rules
  - 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. Notice how they relate to the abstract principles discussed earlier.

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. Like those principles, they are not applicable to every eventuality and need to be interpreted for each new situation. However, they are broadly useful and their application will only help most design projects.

Norman's Seven Principles for Transforming Difficult Tasks into Simple Ones

- We discussed Norman's execution–evaluation cycle, in which he elaborates the seven stages of action. We 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. So systems should provide the necessary knowledge within the environment and their operation should be transparent to support the user in building an appropriate mental model of what is going on.
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. Occasionally one might switch on the indicator lights instead of the windscreen wipers, but the critical controls (accelerator, brake, clutch, steering) are always the same.

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.

HCI design patterns

An approach to reusing knowledge about successful design solutions

- Originated in architecture: Alexander
- A pattern is an invariant solution to a recurrent problem within a specific context.
- Examples
  - Light on Two Sides of Every Room (architecture)
  - Go back to a safe place (HCI)
- Patterns do not exist in isolation but are linked to other patterns in *languages* which enable complete designs to be generated
- Characteristics of patterns
  - capture design practice not theory
  - capture the essential common properties of good examples of design
  - represent design knowledge at varying levels: social, organisational, conceptual, detailed
  - are intuitive and readable and can therefore be used for communication between all stakeholders
  - a pattern language should be generative and assist in the development of complete designs.

EVALUATION TECHNIQUES – UNIVERSAL DESIGN

Evaluation

- tests usability and functionality of system
- occurs in laboratory, field and/or in collaboration with users
- evaluates both design and implementation
- should be considered at all stages in the design life cycle

- 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 the evaluation feeding back into modifications to the design.
- It is not usually possible to perform extensive experimental testing continuously throughout the design, but analytic and informal techniques can and should be used close link between evaluation and the principles and prototyping techniques.

- Such techniques help to ensure that the design is assessed continually. This has the advantage that problems can be ironed out before considerable effort and resources have been expended on the implementation itself: it is much easier to change a design in the early stages of development than in the later stages.
- Evaluation has three main goals:
    a) To assess the extent and accessibility of the system's functionality
    b) To assess users' experience of the interaction
    c) To identify any specific problems with the system.

Evaluation Techniques

1. Evaluating Designs through Expert Analysis
2. Evaluating Designs through user participation
3. Evaluating Implementations
    - Empirical or experimental methods
    - Observational methods
    - Query techniques
    - Methods that use physiological monitoring

Evaluation Through Expert Analysis

We will consider four approaches to expert analysis:

- Cognitive Walkthrough
- Heuristic Evaluation
- Model based Evaluation
- Review-based evaluation

a. Cognitive walkthrough

Cognitive -Psychological processes involved in acquisition and understanding of knowledge, formation of beliefs and attitudes, and decision making and problem solving.

Cognitive Walkthrough-usability evaluation method in which one or more evaluators work through a series of tasks and ask a set of questions from the perspective of the user. The focus of the cognitive walkthrough is on understanding the system's learnability for new or infrequent users

- Cognitive walkthrough was originally proposed and later revised by Polson and colleagues 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. 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.

b. Heuristic evaluation

Heuristic -enabling a person to discover or learn something for themselves
Heuristic Evaluation-helps to identify usability problems in the user interface (UI) design. It specifically involves evaluators examining the interface and judging its compliance with recognized usability principles

- Example heuristics
  - system behaviour is predictable
  - system behaviour is consistent
  - feedback is provided

- 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. Nielsen's ten heuristics are: **(pg no 36)**
- The evaluator assesses the severity of each usability problem, based on four factors:
  - How common is the problem
  - How easy is it for the user to overcome
  - Will it be a one-off problem or a persistent one
  - How seriously will the problem be perceived

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

Model Based Evaluation

- This evaluation is done with the use of models
- The GOMS (goals, operators, methods and selection) model predicts user performance with a particular interface and can be used to filter particular design options
- Dialog models can also be used to evaluate dialog sequences for problems, such as unreachable states, complexity.
- Models such as state transition networks are useful for evaluating dialog designs prior to implementation

Review-based evaluation

- Results from the literature used to support or refute parts of design.
- Care needed to ensure results are transferable to new design.

- *Expert review: expertise in the area is required to ensure that correct assumptions* are made

Evaluating through user Participation

Two distinct evaluation styles

- Laboratory studies
- Field Studies

Laboratory studies

In this approach, users are taken out of their normal work environment to take part in controlled tests, often in a specialist usability laboratory

Advantages:

- specialist equipment available
- uninterrupted environment

Disadvantages:

- lack of context
- difficult to observe several users cooperating

Appropriate

- if system location is dangerous or impractical for constrained single user systems to allow controlled manipulation of use

Field Studies

This approach takes the designer or evaluator out into the user's work environment in order to observe the system in action

Advantages:

- natural environment
- context retained (though observation may alter it)
- longitudinal studies possible

Disadvantages:

- distractions
- noise

Appropriate

- where context is crucial for longitudinal studies

2.18.3.Evaluating Implementations

It has 4 types.

a) Empirical or experimental methods
b) Observational methods
c) Query techniques
d) Methods that use physiological monitoring

a) Empirical or experimental methods

- Controlled evaluation of specific aspects of interactive behaviour
- Evaluator chooses hypothesis to be tested
- Number of experimental conditions are considered which differ only in the value of some controlled variable.
- Changes in behavioural measure are attributed to different conditions

Experimental factors

- Participants

  - participants should be chosen to match the expected user population as closely as possible
  - Sample Size chosen
- Variables
  - things to modify and measure
  - **Independent variable (IV)**
  - characteristic changed to produce different conditions
  - e.g. interface style, number of menu items
  - **Dependent variable (DV)**
  - characteristics measured in the experiment
  - e.g. time taken, number of errors
- Hypothesis
  - Prediction of the outcome of an experiment
  - To show that this prediction is correct
- Experimental design
  - First phase is to choose the hypothesis: to decide exactly what it is you are trying to demonstrate.
  - Next step is to decide on the *experimental method.*
  - Two main methods:
    - ***between-subjects-***each participant is assigned to a different condition. *A*t least two conditions: the experimental condition and the control
    - ***within-subjects*** **-** each user performs under each different condition

Worked exercise *Design an experiment to test whether adding color coding to an interface will improve accuracy. Identify your hypothesis, participant group, dependent and independent variables, experimental design, task and analysis approach.*

Answer The following is only an example of the type of experiment that might be devised.

Participants Taken from user population.

Hypothesis Color coding will make selection more accurate.

IV (Independent Variable) Color coding.

DV (Dependent Variable) Accuracy measured as number of errors.

Design Between-groups to ensure no transfer of learning (or within-groups with appropriate safeguards if participants are scarce).

Task The interfaces are identical in each of the conditions, except that, in the second, color is added to indicate related menu items. Participants are presented with a screen of menu choices (ordered randomly) and verbally told what they have to select. Selection must be done within a strict time limit when the screen clears. Failure to select the correct item is deemed an error. Each presentation places items in new positions. Participants perform in one of the two conditions.

Analysis *t* test.

b) Observational methods

It has the following methods

- Think Aloud & Cooperative evaluation
- Protocol analysis
- Automated analysis
- Post-task walkthroughs

Think Aloud

- User observed performing task
- User asked to describe what he is doing and why, what he thinks is happening etc.
- Advantages
  - simplicity - requires little expertise
  - can provide useful insight
  - can show how system is actually use
- Disadvantages
  - subjective
  - selective
  - act of describing may alter task performance

Cooperative evaluation

- variation on think aloud
- user collaborates in evaluation
- both user and evaluator can ask each other questions throughout
- Additional advantages
  - less constrained and easier to use
  - user is encouraged to criticize system
  - clarification possible

Protocol analysis

- Paper and pencil – cheap, limited to writing speed
- Audio – good for think aloud, difficult to match with other protocols
- Video – accurate and realistic, needs special equipment
- Computer logging – automatic and modest, large amounts of data difficult to analyze
- User notebooks – coarse and subjective, useful insights, good for longitudinal studies
- Mixed use in practice.
- audio/video transcription difficult and requires skill.
- Some automatic support tools available

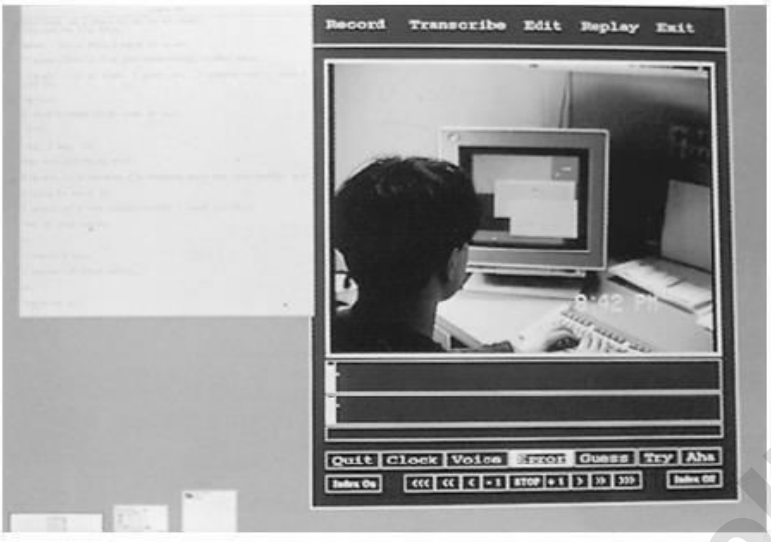
Automated analysis – EVA(*Experimental Video Annotator*)

- Workplace project - multimedia workstation

with a direct link to a video recorder

- Post task walkthrough
  - user reacts on action after the event
  - used to fill in intention
- Advantages
  - analyst has time to focus on relevant incidents
  - avoid excessive interruption of task
- Disadvantages

- lack of freshness
- may be post-hoc interpretation of events



EVA: an automatic protocol analysis tool. Source: Wendy I

Post-task walkthroughs

- The transcript, whether written or recorded, is replayed to the participant who is invited to comment, or is directly questioned by the analyst.
- In some circumstances the participant cannot be expected to talk during the actual observation.
- Necessary in cases where think aloud is not possible

c) Query techniques (Interviews , Questionnaires)

Interviews

- Analyst questions user on one-to -one basis usually based on prepared questions
- Informal, subjective and relatively cheap
  - Advantages
    - can be varied to suit context
    - issues can be explored more fully
    - can elicit user views and identify unanticipated problems
- Disadvantages
  - very subjective
  - time consuming

Questionnaires

- Set of fixed questions given to users
- Advantages
  - quick and reaches large user group
  - can be analyzed more rigorously
- Disadvantages
  - less flexible
  - less probing

- Need careful design
  - what information is required?

  - how are answers to be analyzed?
- Styles of question
  - general
  - open-ended
  - scalar
  - multi-choice
  - ranked

Worked exercise *You have been asked to compare user performance and preferences with two different learning systems, one using hypermedia (see Chapter 21), the other sequential lessons. Design a questionnaire to find out what the users think of the system. How would you go about comparing user performance with these two systems?*

Answer Assume that all users have used both systems.

Questionnaire

Consider the following questions in designing the questionnaire:

- what information is required?
- how is the questionnaire to be analyzed?

You are particularly interested in user preferences so questions should focus on different aspects of the systems and try to measure levels of satisfaction. The use of scales will make responses for each system easier to compare.

Table 9.3 shows an example questionnaire.

To test performance you would design an experiment where two groups of participants learn the same material using the two systems, and test how well they have learned (using a standard measurable test).

Participants User group

IV (Independent Variable) Style of learning system

DV (Dependent Variable) Performance (measured as test score)

Design Between-subjects design

Table 9.3 Questionnaire to compare two systems

PART I: Repeat for each system

Indicate your agreement or disagreement with the following statements. (1 indicates complete disagreement and 5 complete agreement.)

The system tells me what to do at every point.

Disagree 1 2 3 4 5 Agree

It is easy to recover from mistakes.

Disagree 1 2 3 4 5 Agree

It is easy to get help when needed.

Disagree 1 2 3 4 5 Agree

I always know what the system is doing.
Disagree 1 2 3 4 5 Agree
I always know where I am in the training material.
Disagree 1 2 3 4 5 Agree
I have learned the material well using the system.
Disagree 1 2 3 4 5 Agree
I could have learned the material more effectively using a book.
Disagree 1 2 3 4 5 Agree
I always know how well I am doing.
Disagree 1 2 3 4 5 Agree

PART II: Comparing both systems:

Which system (choose 1) was most:
Helpful to use A B
Efficient to use A B
Enjoyable to use A B

Please add any comments you have about either system:

d) Methods that use physiological monitoring

It has 2 methods

Eye tracking

Physiological measurement

Eye tracking

- Head or desk mounted equipment tracks the position of the eye
- Eye movement reflects the amount of cognitive processing a display requires
- Measurements include
  - Fixations: eye maintains stable position. Number and duration indicate level of difficulty with display
  - Scan paths: moving straight to a target with a short fixation at the target is optimal

Eye-tracking equipment.

physiological measurements

- Emotional response linked to physical changes
- These may help determine a user's reaction to an interface
- Measurements include:
  - heart activity, including blood pressure, volume and pulse.
  - activity of sweat glands: Galvanic Skin Response (GSR)
  - electrical activity in muscle: electromyogram (EMG)
  - electrical activity in brain: electroencephalogram (EEG)
- some difficulty in interpreting these physiological responses - more research needed

Data Lab Psychophysiology equipment showing some of the sensors

CHOOSING AN EVALUATION METHOD - Factors distinguishing evaluation techniques

- 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.

Universal Design Principles:

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

- 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.
- 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. Repetitive or sustained actions should be avoided.
- 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.

MULTI-MODAL INTERACTION (EXTRA)

The principle of universal design relies on multi-modal interaction.

Multi-Sensory Systems

- More than one sensory channel in interaction
    - e.g. sounds, text, hypertext, animation, video, gestures, vision
- Used in a range of applications:
    - particularly good for users with special needs, and virtual reality
- It increases the *bandwidth of the interaction* between the human and the computer
- It makes human–computer interaction more like the interaction between humans and their everyday environment

Usable Senses

The 5 senses (sight, sound, touch, taste and smell) are used by us every day

- each is important on its own
- together, they provide a fuller interaction with the natural world

Computers rarely offer such a rich interaction

Can we use all the available senses?

- ideally, yes

– practically – no

We can use • sight • sound • touch (sometimes)

We cannot (yet) use • taste • smell

Multi-modal vs. Multi-media

- Multi-modal systems
  - use more than one sense (or mode) of interaction

e.g. visual and aural senses: a text processor may speak the words as well as echoing them to the screen

- Multi-media systems
  - use a number of different media to communicate information

e.g. a computer-based teaching system:may use video, animation, text and still images: different media all using the visual mode of interaction; may also use sounds, both speech and non-speech.

Speech

Human beings have a great and natural mastery of speech

- makes it difficult to appreciate the complexities but
- it's an easy medium for communication

Structure of Speech

- Phonemes
  - English contains 40 phonemes- 24 consonants and 16 vowel sounds
  - basic atomic units
  - sound slightly different depending on the context they are in
- Allophones
  - all the sounds in the language
  - between 120 and 130 of them
- Morphemes
  - either parts of words or whole words
  - smallest unit of language that has meaning.

The Phonetic Typewriter

- Developed for Finnish (a phonetic language, written as it is said)
- Trained on one speaker, will generalise to others.
- A neural network is trained to cluster together similar sounds, which are then labelled with the corresponding character.
- When recognising speech, the sounds uttered are allocated to the closest corresponding output, and the character for that output is printed.
  - requires large dictionary of minor variations to correct general mechanism
  - noticeably poorer performance on speakers it has not been trained on

Speech Synthesis

The process of generating spoken language by machine on the basis of written input . Useful for users who are blind or partially sighted

Successful in certain constrained applications when the user:

- is particularly motivated to overcome problems
- has few alternatives

Examples:

- screen readers

- read the textual display to the user
  utilised by visually impaired people
- warning signals
  - spoken information sometimes presented to pilots whose visual and haptic skills are already fully occupied

Non-Speech Sounds

- Often used to provide transitory information, such as indications of network or system changes, or of errors
- Used to provide status information on background processes
- Commonly used for warnings and alarms
- Evidence to show they are useful
  - fewer typing mistakes with key clicks
  - video games harder without sound
- Language/culture independent, unlike speech

Touch

- Use of touch in the interface is known as *haptic interaction*
  - Cutaneous perception
    - tactile sensation(the **sensation** produced by pressure receptors in the skin); vibrations on the skin
  - kinesthetics
    - movement and position; force feedback
- Information on shape, texture, resistance, temperature, comparative spatial factors
- example technologies
  - electronic braille displays
  - force feedback devices Eg.PHANTOM
    - resistance, texture

Handwriting recognition

- Handwriting is another communication mechanism which we are used to in day-to-day life
- Interpret handwritten input and handwriting appears to offer both textual and graphical input
- Technology
  - Handwriting consists of complex strokes and spaces
  - Captured by digitising tablet
    - strokes transformed to sequence of dots
  - large tablets available
    - suitable for digitising maps and technical drawings
  - smaller devices, some incorporating thin screens to display the information
    - PDAs such as Palm Pilot
    - tablet PCs
- Problems
  - personal differences in letter formation
  - co-articulation effects
- Breakthroughs:
  - stroke not just bitmap
  - special 'alphabet' – Graffeti on PalmOS
- Current state:
  - usable – even without training
  - but many prefer keyboards!

Gesture

- Able to control the computer with certain movements of the hand
- Applications
  - gestural input - e.g. “put that there”
  - sign language
- Technology
  - data glove
  - position sensing devices
- Benefits
  - natural form of interaction - pointing
  - enhance communication between signing and non-signing users
- Problems
  - user dependent, variable and issues of coarticulation

Users with disabilities

- visual impairment
  - screen readers, SonicFinder
- hearing impairment
  - text communication, gesture, captions
- physical impairment
  - speech I/O, gesture, predictive systems (e.g. Reactive keyboard)
- speech impairment
  - speech synthesis, text communication
- age groups
  - older people e.g. disability aids, memory aids, communication tools to prevent social isolation
  - children e.g. appropriate input/output devices, involvement in design process
- cultural differences
  - influence of nationality, generation, gender, race, sexuality, class, religion, political persuasion etc. on interpretation of interface features
  - e.g. interpretation and acceptability of language, cultural symbols, gesture and colour

ARUNAI ENGINEERING COLLEGE

DEPARTMENT OF CSE

IV YEAR - VII SEMESTER

CS8079 – HUMAN COMPUTER INTERACTION (R2017)

UNIT III MODELS AND THEORIES

HCI Models: Cognitive models: Socio-Organizational issues and stakeholder requirements –Communication and collaboration models-Hypertext, Multimedia and WWW.

COGNITIVE MODELS

- One way to classify the models is in respect to how well they describe features of the competence and performance of the user. Competence models tend to be ones that can predict legal behavior sequences but generally do this without reference to whether they could actually be executed by users.
- Performance models not only describe what the necessary behavior sequences are but usually describe both what the user needs to know and how this is employed in actual task execution. The presentation of the cognitive models in this chapter follows this classification scheme, divided into the following categories:

a) hierarchical representation of the user's task and goal structure
b) linguistic and grammatical models
c) physical and device-level models.

- The first category deals directly with the issue of formulation of goals and tasks. The second deals with the grammar of the articulation translation and how it is understood by the user. The third category again deals with articulation, but at the human motor level instead of at a higher level of human understanding.

Goal and Task Hierarchies

- Many models make use of a model of mental processing in which the user achieves goals by solving sub goals in a divide-and-conquer fashion. We will consider two models, GOMS and CCT, where this is a central feature. 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>>

- Different design issues demand different levels of analysis. This most abstract task is referred to as the unit task. The unit task does not require any problem-solving skills on the part of the user, though it frequently demands quite sophisticated problem-solving skills on the part of the designer to determine them.

1. GOMS

- The GOMS model of Card, Moran and Newell is an acronym for Goals, Operators, Methods and Selection [56]. A GOMS description consists of these four elements:

a. Goals: These are the user's goals, describing what the user wants to achieve. Further, in GOMS the goals are taken to represent a 'memory point' for the user, from which he can evaluate what should be done and to which he may return should any errors occur.

b. Operators: These are the lowest level of analysis. They are the basic actions that the user must perform in order to use the system. They may affect the system (for example, press the 'X' key) or only the user's mental state (for example, read the dialog box).

c. Methods : There are typically several ways in which a goal can be split into sub goals. For instance, in a certain window manager a currently selected window can be closed to an icon either by selecting the 'CLOSE' option from a pop-up menu, or by hitting the 'L7' function key. In GOMS these two goal decompositions are referred to as methods, so we have the **CLOSE-METHOD** and the **L7-METHOD**:

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.

d. 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.

2. Cognitive complexity theory

- CCT has two parallel descriptions: one of the user's goals and the other of the computer system (called the device in CCT). The description of the user's goals is based on a GOMS-like goal hierarchy, but is expressed primarily using production rules. CCT uses generalized transition networks, a form of state transition network. The production rules are a sequence of rules:

  if condition then action

where condition is a statement about the contents of working memory. If the condition is true then the production rule is said to fire.

- An action may consist of one or more elementary actions, which may be either changes to the working memory, or external actions such as keystrokes. The production rule 'program' is written in a LISP-like language.
- Rules in CCT need not represent error-free performance. They can be used to explain error phenomena, though they cannot predict them. For instance, the rules above for inserting a space are 'buggy' – they do not check the editor's mode.
- The CCT rules are closely related to GOMS-like goal hierarchies; the rules may be generated from such a hierarchy, or alternatively, we may analyze the production rules to obtain the goal tree:

  GOAL: insert space
  . GOAL: move cursor – if not at right position
  . PRESS-KEY-I
  . PRESS-SPACE
  . PRESS-ESCAPE

Linguistic Models

- BNF grammars are frequently used to specify dialogs. The models here, although similar in form to dialog design notations, have been proposed with the intention of understanding the user's behavior and analyzing the cognitive difficulty of the interface.

a. BNF

- Representative of the linguistic approach is Reisner's use of Backus–Naur Form (BNF) rules to describe the dialog grammar . 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 name ::= expression
- The '::=' symbol is read as 'is defined as'. Only non-terminals may appear on the left of a definition. The right-hand side is built up using two operators '+' (sequence) and '|' (choice).
- For example, the first rule says that the non-terminal draw-line is defined to be select-line followed by choose-points followed by last point.All of these are non-terminals, that is they do not tell us what the basic user actions are.
- The second rule says that select-line is defined to be position mouse (intended to be over the 'line' menu entry) followed by CLICK-MOUSE. This is our first terminal and represents the actual clicking of a mouse. To see what position-mouse is, we look at the last rule.
- This tells us that there are two possibilities for position-mouse (separated by the '|' symbol). One option is that position-mouse is empty – a special symbol representing no action. That is, one option is not to move the mouse at all.
- The other option is to do a MOVE-MOUSE action followed by position-mouse. This rule is recursive, and this second position-mouse may itself either be empty or be a MOVE-MOUSE action followed by position-mouse, and so on. That is, position-mouse may be any number of MOVE-MOUSE actions whatsoever.
- Choose-points is defined recursively, but this time it does not have the option of being empty. It may be one or more of the non-terminal choose one which is itself defined to be (like select-line) position-mouse followed by CLICK-MOUSE.
- The BNF description of an interface can be analyzed in various ways. One measure is to count the number of rules. The more rules an interface requires to use it, the more complicated it is. This measure is rather sensitive to the exact way the interface is described. For example, we could have replaced the rules for choose points and choose-one with the single definition
  choose-points ::= position-mouse + CLICK-MOUSE| position-mouse + CLICK-MOUSE + choose-points
  More robust measure also counts the number of '+' and '|' operators. This would, in effect, penalize the more complex single rule. Another problem arises with the rule for select-line. This is identical to the choose-one rule.

b. Task–action grammar

- Task–action grammar (TAG) attempts to deal with some of these problems by including elements such as parameterized 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 two possible 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 first. 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'
```

PHYSICAL AND DEVICE MODELS

a. Keystroke-level model

- 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:
  a. K Keystroking, actually striking keys, including shifts and other modifier keys.
  b. B Pressing a mouse button.
  c. P Pointing, moving the mouse (or similar device) at a target.
  d. H Homing, switching the hand between mouse and keyboard.
  e. D Drawing lines using the mouse.
  f. M Mentally preparing for a physical action.
  g. 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.

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

Table3.1: Times for various operators in the keystroke level model

b. Three-state model

- 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. These different devices – mouse, trackball, light pen – feel very different.
- Buxton has developed a simple model of input devices the three-state model, which captures some of these crucial distinctions. He begins by looking at a mouse. If you move it with no buttons pushed, it normally moves the mouse cursor about. This tracking behavior is termed state 1. Depressing a button over an icon and then moving the mouse will often result in an object being dragged about. This he calls state 2.

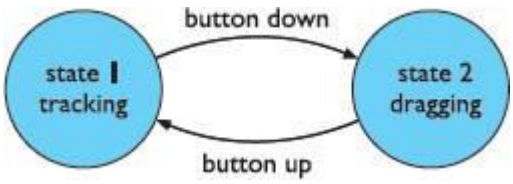


Fig3.1: Mouse Transitions : states 1 and 2

- Instead we consider a light pen with a button, it behaves just like a mouse when it is touching the screen. When its button is not depressed, it is in state 1, and when its button is down, state 2. However, the light pen has a third state, when the light pen is not touching the screen. In this state the system cannot track the light pen's position. This is called state 0.
- 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.

- The only additional complexity is if the device has several buttons, in which case we would have one state for each button: 2left, 2middle, 2right.
- At first, the model appears to characterize the states of the device by the inputs available to the system. So, from this perspective, state 0 is clearly different from states 1 and 2. However, if we look at the state 1–2 transaction, we see that it is symmetric with respect to the two states.

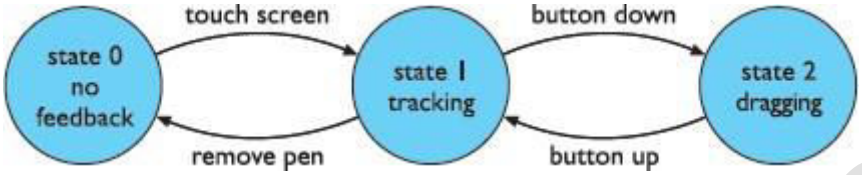


Fig.3.2: Light pen transitions: three states

- In principle, there is no reason why a program should not decide to do simple mouse tracking whilst in state 2 and drag things about in state 1 State 2 requires a button to be pressed, whereas state 1 is one of relative relaxation (whilst still requiring hand–eye coordination for mouse movement). There is a similar difference in tension between state 0 and state 1.
- It is well known that Fitts' law has different timing constants for different devices. Recall that Fitts' law says that the time taken to move to a target of size S at a distance D is: $a \quad b \log 2(D/S \quad 1)$
- The constants a and b depend on the particular pointing device used and the skill of the user with that device.

	Device	*a* (ms)	*b* (ms/bit)
Pointing (state 1)			
	Mouse	–107	223
	Trackball	75	300
Dragging (state 2)			
	Mouse	135	249
	Trackball	–349	688

Fig3.3: Fitt's law coefficients

Cognitive Architectures

a. Problem space model

- The problem space model 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 architecture of the machine only allows the definition of the search or problem space and the actions that can occur to traverse that space. Termination is also assumed to happen once the desired state is reached.

- Notice that the machine does not have the ability to formulate the problem space and its solution, mainly because it has no idea of the goal. It is the job of the programmer to understand the goal and so define the machine to achieve it.
- We can adapt the state-based computational model of a machine in order to realize the architecture of a knowledge-level system. The new computational model is the problem space model, based on the problem-solving work of Newell and Simon at Carnegie–Mellon University. A problem space consists of a set of states and a set of operations that can be performed on the states.
- Once the initial state is set, the task within the problem space is to find a sequence of operations that form a path within the state space from the initial state to one of the desired states, whereupon successful termination occurs.

b. Interacting cognitive subsystems

- Barnard has proposed a very different cognitive architecture, called interacting cognitive subsystems . 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.
- The other hand is the computational and representational approach characteristic of psycholinguistic research and AI problem-solving literature.
- The architecture of ICS is built up by the coordinated activity of nine smaller subsystems: five peripheral subsystems are in contact with the physical world and four are central, dealing with mental processes. Each subsystem has the same generic structure.
- A subsystem is described in terms of its typed inputs and outputs along with a memory store for holding typed information. It has transformation functions for processing the input and producing the output and permanently stored information.
- Each of the nine subsystems is specialized for handling some aspect of external or internal processing. For example, one peripheral subsystem is the visual system for describing what is seen in the world.

SOCIO-ORGANIZATIONAL ISSUES AND STAKE HOLDER REQUIREMENTS

Organizational Issues

- 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.

1. Cooperation or conflict?
2. Changing power structures
3. The invisible worker
4. Who benefits?
5. Free rider problem
6. Critical mass
7. Automating processes – workflow and BPR
8. Evaluating the benefits

Capturing Requirements

- 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.
- need to take account of
    - stakeholders
    - work groups and practices
    - organisational context

- We consider several approaches:
    - socio-technical modeling
    - soft systems methodology
    - participatory design
    - ethnographic methods and contextual inquiry.

- Who are the stakeholders? can be defined as anyone who is affected by the success or failure of the system. It can be useful to distinguish different categories of stakeholder, and the following categorization from the CUSTOM approach is helpful for this:

1. Primary stakeholders are people who actually use the system – the end-users.
2. 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).
3. 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).
4. Facilitating stakeholders are people who are involved with the design, development and maintenance of the system.

Example: Classifying stakeholders – an airline booking system

An international airline is considering introducing a new booking system for use by associated travel agents to sell flights directly to the public.

Primary stakeholders: travel agency staff, airline booking staff

Secondary stakeholders: customers, airline management

Tertiary stakeholders: competitors, civil aviation authorities, customers' travelling companions, airline shareholders

Facilitating stakeholders: design team, IT department staff

- All of the approaches we are considering here are concerned with understanding stakeholders within their organizational context.

1. Socio-technical models

- Technological determinism, the view that social change is primarily dictated by technology, with human and social factors being secondary concerns, was prevalent. The socio-technical systems view came about
- to counter this technology-centric position, by stressing that work systems were composed of both human and machine elements and that it was the interrelationship between these that should be central.

- Socio-technical models for interactive systems are therefore concerned with technical,social, organizational and human aspects of design. They recognize the fact that technology is not developed in isolation but as part of a wider organizational environment. It is important to consider social and technical issues side by side so that human issues are not overruled by technical considerations.
- The key focus of the socio-technical approach is to describe and document the impact of the introduction of a specific technology into an organization. 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.
- Information is gathered using methods such as interviews, observation, focus groups and document analysis. The methods guide this information-gathering process and help the analyst to make sense of what is discovered.
- By attempting to understand these issues, socio-technical approaches aim to provide a detailed view of the role technology will play and the requirements of successful deployment. We will compare two approaches to illustrate how this may work in practice.

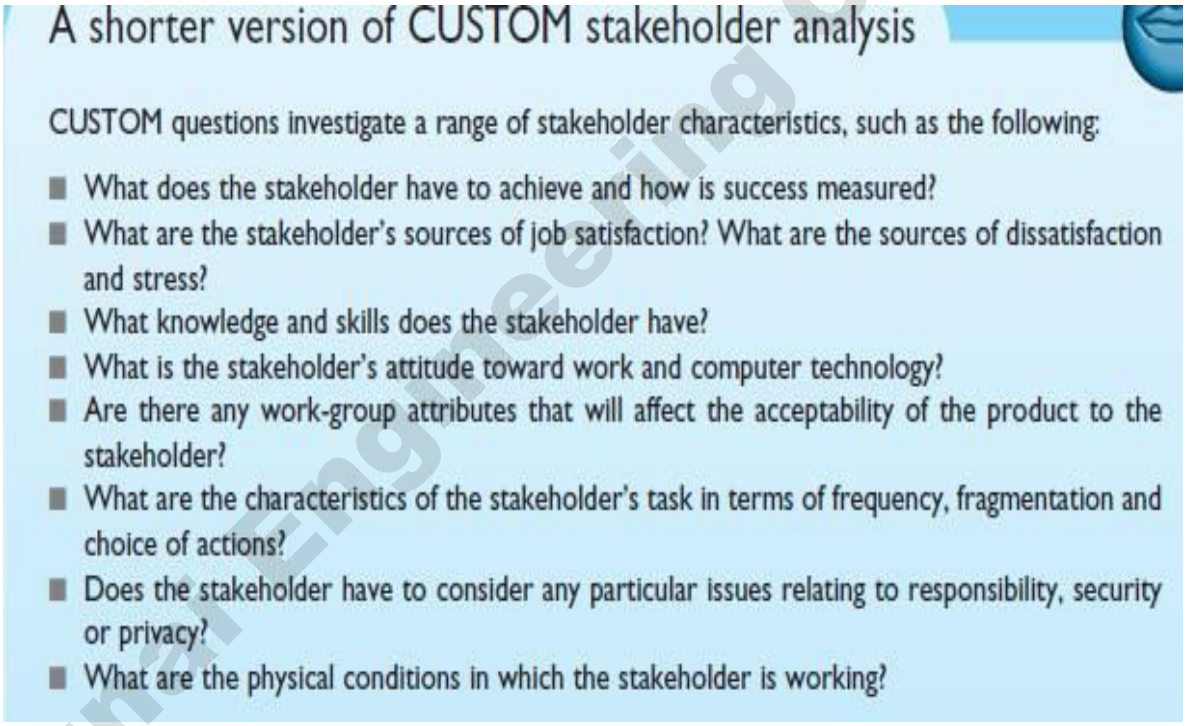
2. CUSTOM methodology

- CUSTOM is a socio-technical methodology designed to be practical to use in small organizations . It is based on the User Skills and Task Match (USTM) approach, developed to allow design teams to understand and fully document user requirements .
- CUSTOM focuses on establishing stakeholder requirements :all stakeholders are considered, not just the end-users. It is applied at the initial stage of design when a product opportunity has been identified, so the emphasis is on capturing requirements.
- It is a forms-based methodology, providing a set of questions to apply at each of its stages. There are six key stages to carry out in a CUSTOM analysis:

a. Describe the organizational context, including its primary goals, physical characteristics, political and economic background.
b. 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.
c. 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.
d. 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.
e. 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.

f. Consolidate and check stakeholder requirements. Here the stakeholder needs list is checked against the criteria determined at earlier stages.

- Stages 2 to 4 are described in terms of the current situation (before the new technology is introduced) and the proposed situation (after deployment). Stakeholders are asked to express their views not only of their current role and position but of their expectations in the light of the changes that will be made.
- The stakeholder concerns and goals are elaborated. In addition, the impact of the technology on working practices is considered (Stage 3) and the transformations that will be supported by the system specified (Stage 4). The changes from the current position to the proposed position represent the issues that need to be addressed to ensure successful deployment, and these are made explicit during Stages 5 and 6.
- CUSTOM provides a useful framework for considering stakeholder requirements and the use of forms and questions (a 'manual' for its use is available makes it relatively straightforward, if somewhat time consuming , to apply. For less complex situations, a shortened version of CUSTOM stakeholder analysis is available .This also provides a checklist for investigations for stages 2– 4.



3. 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.
- There is no assumption of a particular solution: the emphasis is rather on understanding the situation fully. SSM was developed by Checkland to help designers reach an understanding of the context of technological developments and the influences and concerns that exist within the system under consideration.
- SSM has seven stages. A distinction is made between the 'real-world' stages (1–2, 5–7) and the systems stages (3–4). The first stage of SSM is the recognition of the problem and initiation of analysis. This is followed by a detailed description of the problem situation: developing a rich picture. This will include all the stakeholders, the tasks they carry out and the groups they work in, the organizational structure and its processes and the issues raised by each stakeholder.
- Any knowledge elicitation techniques can be used to gather the information to build the rich picture, including observation (and video and audio recording), structured and unstructured interviews and questionnaires, and workshops incorporating such activities as role play, simulations and critical incident analysis.
- Less structured approaches are used initially to avoid artificially constraining the description. The rich picture can be in any style – there are no right or wrong answers – but it should be clear and informative to the designer.

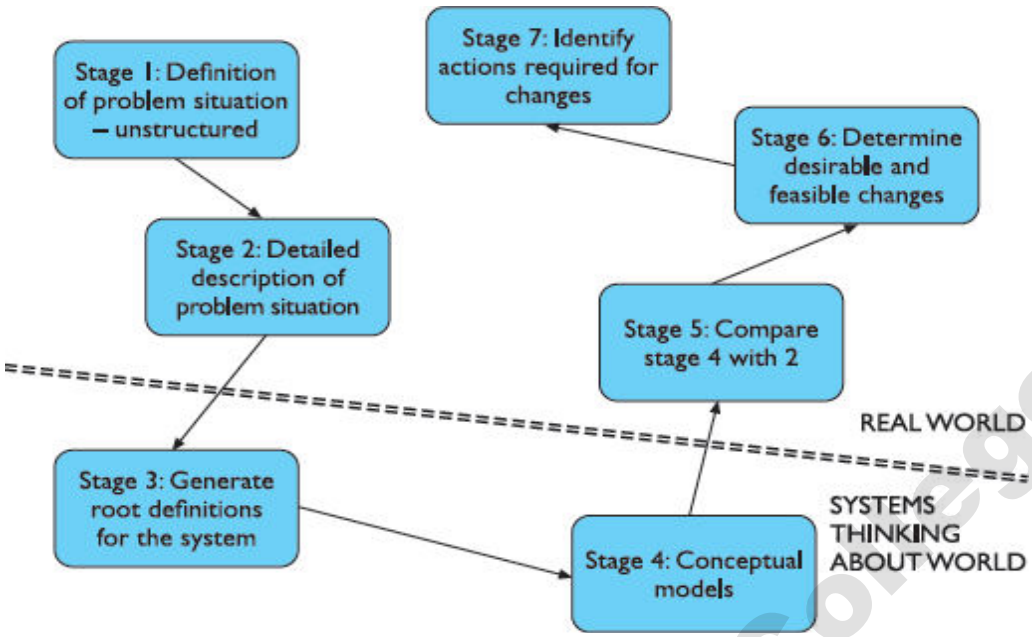


Fig3.4: Seven stages of soft systems methodology

- At the next stage in SSM we move from the real world to the systems world and attempt to generate root definitions for the system, which define the essence of what the system is about. There may be several root definitions of a system, representing different stakeholder perspectives, for example. Root definitions are described in terms of specific elements, summarized using the acronym, CATWOE:

1. Clients – those who receive output or benefit from the system.
2. Actors – those who perform activities within the system.
3. 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.
4. Weltanschauung – (from the German) meaning world view. This is how the system is perceived in a particular root definition.
5. Owner – those to whom the system belongs, to whom it is answerable and who can authorize changes to it.
6. Environment – the world in which the system operates and by which it is influenced.

Root definition for airline management: an airline booking system

- **Client:** customer
- **Actor:** travel agency staff
- **Transformation:** customer's intention and request to travel transformed into sale of seat on flight and profit for organization
- **Weltanschauung:** profits can be optimized by more efficient sales
- **Owner:** airline management
- **Environment:** Regulations of international civil aviation authorities and national contract legislation. Local agency policies worldwide

Participatory design

In participatory design:
workers enter into design context
In ethnography (as used for design):
designer enters into work context

- 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 can only be effective within that context if these experts are allowed to contribute actively to the design process.
- It therefore aims to refine system requirements iteratively through a design process in which the user is actively involved.
- Participatory design has three specific characteristics. It aims to improve the work environment and task by the introduction of the design. This makes design and evaluation context or work oriented rather than system oriented. Secondly, it is characterized by collaboration: the user is included in the design team and can contribute to every stage of the design.
- Finally, the approach is iterative: the design is subject to evaluation and revision at each stage. The participatory design process utilizes a range of methods to help convey information between the user and designer. They include

a. **Brainstorming** This involves all participants in the design pooling ideas. This is informal and relatively unstructured although the process tends to involve 'onthefly' structuring of the ideas as they materialize. All information is recorded without judgment. The session provides a range of ideas from which to work. These can be filtered using other techniques.
b. **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 the potential designs and the impact they will have.
c. **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.
d. **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.

Example:

DESIGN FOCUS

Tomorrow's hospital – using participatory design

The nurse walks around the ward to a patient's bedside. She takes her PDA from her pocket and enters the patient's name. Her PDA is connected via a wireless network to the central patient treatment database and the patient's information comes on screen including reminders of treatments needed. First is a blood pressure and heart rate check. She does these checks and enters the results. A mild painkiller is also prescribed. She gets the tablet, which is individually wrapped in a bar coded packet. Her PDA has a built-in bar code reader and she scans this across the tablet. The system registers the drug's use and removes it from the to-do list as she gives the patient the tablet with a glass of water.

A picture of tomorrow's hospital? In fact, this is exactly the day-to-day activity in Hospital da Trofa, just outside Porto, Portugal. There have been numerous attempts to use PDA-based systems in hospitals. But most have failed. So why is this one being used? The hospital is part of a European Commission funded project Team-HOS and the system was designed using a methodology that has a strong *participatory* focus [316]. From the beginning, nurses, doctors, pharmacologists and dieticians were involved in and in control of the design. It has benefited from their knowledge and experience, which is why it does the right things for the context. Furthermore, the pride with which the hospital staff describe the system shows that they really feel it is their system, not one imposed from above.

See the book website for an extended case study: /e3/casestudy/trofa/

Effective Technical and Human Implementation of Computer-based systems (ETHICS)

- ETHICS [243] is a method developed by Enid Mumford within the socio-technical tradition, but it is distinct in its view of the role of stakeholders in the process. In the ETHICS methodology, stakeholders are included as participants in the decision making process.
- ETHICS considers the process of system development as one of managing change: conflicts will occur and must be negotiated to ensure acceptance and satisfaction with the system. If any party is excluded from the decision-making process then their knowledge and contribution is not utilized and they are more likely to be dissatisfied. 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 decision making 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 focuses 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).

- The final stages of the ETHICS approach focus on the actual design and evaluation of the system. Necessary organizational changes are designed alongside the technical system. These are then specified in detail, implemented and evaluated.
- The ETHICS approach attempts to reach a solution that meets both user and task requirements by having specialist teams negotiate objectives and rank potential solutions.

Ethnographic methods

- Real action is situated action; it occurs in interaction with the materials and people of the workplace. In extremis, it is claimed that an action can only be understood in the place, in the social situation, and at the time at which it occurred.
- Such a level of contextualization is obviously useless for design, and its advocates will in practice generalize from their observations, even if they ostensibly eschew such generalization.
- Many branches of sociology and anthropology have long recognized that one cannot study people divorced from their social and cultural context. ethnography has become very influential, particularly in the study of group systems.
- Ethnography is based on very detailed recording of the interactions between people and between people and their environment. It has a special focus on social relationships and how they affect the nature of work.

- The ethnographer does not enter actively into the situation, and does not see things from a particular person's viewpoint. However, an aim is to be encultured, to understand the situation from within its own cultural framework. Culture here means that of the particular workgroup or organization, rather than that of society as a whole. Ethnographers try to take an unbiased and open-ended view of the situation

Contextual inquiry

Approach developed by Holtzblatt

- in ethnographic tradition but acknowledges and challenges investigator focus
- model of investigator being apprenticed to user to learn about work
- investigation takes place in workplace - detailed interviews, observation, analysis of communications, physical workplace, artefacts
- number of models created:
  - sequence, physical, flow, cultural, artefact
  - models consolidated across users
- output indicates task sequences, artefacts and communication channels needed and physical and cultural constraints

COMMUNICATION AND COLLABORATION MODELS

- single-user or multi-user
- We need to understand normal human–human communication:
- face-to-face communication involves eyes, face and body
- conversation can be analyzed to establish its detailed structure.

Look at several levels – minutiae to large scale context:

- face-to-face communication
- conversation
- text based communication
- group working

FACE-TO-FACE COMMUNICATION

- Face-to-face contact is the most primitive form of communication primitive, that is, in terms of technology. The first thing to note is that face-to-face communication involves not just speech and hearing, but also the subtle use of body language and eyegaze. It has a range of these phenomena, and how they influence our use of computer-mediated communications.

a. 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. the rules of face-to-face conversation are not conscious, so, when they are broken, we do not always recognize the true problem.

- Personal space also differs across cultures: North Americans get closer than Britons, and southern Europeans and Arabs closer still. This can cause considerable problems during cross-cultural meetings

b. Eye contact and gaze

- 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.
- The 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. problems with direct eye contact, many signals can be easily read through a video channel. This involves not just the eyes, but the whole facial expression, and this is apparent even on poor-quality video or very small (pocket- TV-sized) monitors

c. Gestures and body language

- 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 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.*

d. 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! 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'.
- The nods, grimaces, shrugs of the shoulder and small noises are called *backchannels*. They feed information back from the listener to the speaker at a level below the turn-taking of the conversation.

- Back channels -media effects

Restricting media restricts back channels

video – loss of body language
audio – loss of facial expression
half duplex – lose most voice back-channel responses
text based – nothing left!

e. 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. Back channels are often a crucial part of this process

Conversation

- It focuses on two-person conversations, but this can range from informal social chat over the telephone to formal courtroom cross-examination. As well as the discipline of *conversational analysis*, there are other sociological and psychological understandings of conversation.
- There are three uses for theories of conversation in CSCW.

1. First, they can be used to analyze transcripts, for example from an electronic conference. This can help us to understand how well the participants are coping with electronic communication.
2. 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.
3. Thirdly, and most controversially, they can be used to drive design structuring the system around the theory. We will concentrate mainly on the first goal, although this will have implications throughout for design

Basic conversational structure

- Imagine we have a transcript of a conversation 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. the most basic conversational structure is *turntaking*.On the whole we have an alternating pattern: Alison says something, then Brian, then Alison again.

- The speech within each turn is called an *utterance*. There can be exceptions to this turn-taking structure even within two-party conversation. Often we can group the utterances of the conversation into pairs: a question and an answer, a statement and an agreement. The answer or response will normally follow directly after the question or statement and so these are called adjacency pairs . The requirement of adjacency can be broken if the pair is interposed with other pairs for clarification, etc.:

Brian: Do you want some gateau? Alison: Is it very fattening? Brian: Yes, very. Alison: And lots of chocolate? Brian: Masses. Alison: I'll have a big slice then.

- This conversation can be denoted: B-x, A-y, B-y, A-z, B-z, A-x. Adjacency pair 'x' ('Do you want some gateau?'–'I'll have a big slice then') is split by two other pairs 'y' and 'z'. One would normally expect the interposed pairs to be relevant to the outer pair, seeking clarification or determining information needed for the response

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.
external context – dependence on the environment.

Example

Brian: (*points*) that post is leaning a bit
Alison: that's the one you put in

Two types of context:

- external context – reference to the environment
  e.g., Brian's '*that*' – the thing pointed to ← *deictic reference*
- internal context – reference to previous conversation
  e.g., Alison's '*that*' – the last thing spoken of

Common Ground

Resolving context depends on meaning
⇒ participants must share meaning
so must have shared knowledge

Conversation constantly negotiates meaning
... a process called *grounding*:

Alison: So, you turn right beside the river.
Brian: past the pub.
Alison: yeah ...

Each utterance is assumed to be:
relevant – furthers the current topic
helpful – comprehensible to listener

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!

Tracing topics is one way to analyse conversation.

- Alison begins – *topic* is roses
- Brian shifts topic to greenfly
- Alison misses shift in focus … *breakdown in communication*

- 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 in 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:

1. **substantive** directly relevant to the development of the topic;
2. **annotative** points of clarification, elaborations, etc.;
3. **procedural** talking about the process of collaboration itself.

- 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

- At a lower level, we may see breakdown due to incorrectly read gestures or eyegaze, and through missed or inappropriate back channel responses. Despite the frequency of breakdowns in normal speech, our communication is not usually significantly affected because we are so efficient at repair.
- Redundancy, frequency of turn-taking and back channels, all contribute to the detection of breakdown and its rapid repair. Electronic communications often reduce redundancy (a single channel), reduce the frequency of turn-taking and reduce back channels. The problem is thus not so much breakdowns in communication, but a reduced ability to recover from them.

Constructing a shared understanding

- The major difference between a book and conversation is that the latter is interactive. The shared knowledge used in a book is static, whereas that used during a conversation is dynamic, as the participants increase their understanding of one another and as they shift their focus from topic to topic.
- When participants come to a conversation, they may come from different backgrounds and bring different knowledge. Even close colleagues will have

different recent experiences, and as we have seen in previous examples, have different foci.

- The participants do not try to unify their knowledge and background indeed, they could not fully do so without living one another's lives. Instead, they seek to obtain a *common ground*, a shared understanding sufficient for the task in hand.
- Establishing this common ground will involve negotiating the meanings of words and constructing shared interpretations of the world. Clark and Schaefer refer to this process as *grounding*. The aim of grounding is to construct a meaning *in the conversation* which is sufficient for the task. Two guiding principles for our utterances are that they should be *relevant* and *helpful*.

Speech act theory

- A particular form of conversational analysis, *speech act theory*, has been both influential and controversial in CSCW. Not only is it an analytic technique, but it has been used as the guiding force behind the design of a commercial system, Coordinator.
- Speech act theory has origins going back over 25 years, but was popularized by Winograd and Flores in the design of Coordinator The basic premise of speech act theory is that utterances can be characterized by what they *do*. If you say 'I'm hungry', this has a certain *propositional meaning* that you are feeling hungry.
- Speech act theory concerns itself with the way utterances interact with the actions of the participants. Individual speech acts can contribute to a conversation. The basic structure of conversations can then be seen as instances of generic conversations. One example of such a generic structure is a *conversation for action* (*CfA*).
- It represents the stages two participants go through in initiating an external action that one of them should perform. There are two variants, the one shown representing a conversation where the first speaker (A) is requesting that the other participant (B) does something. The other, similar, variant is where the first speaker begins with an offer. where the first speaker begins with an offer.
- The numbered circles in Figure 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.
- The network has some nodes marked with a double circle. These are the completion nodes, and at these points neither party expects any more acts by the other as part of this conversation. So the fragment above which left Alison and Brian in state 3 must continue. Of these completion nodes only state 5 represents conclusions where the request has been satisfied.

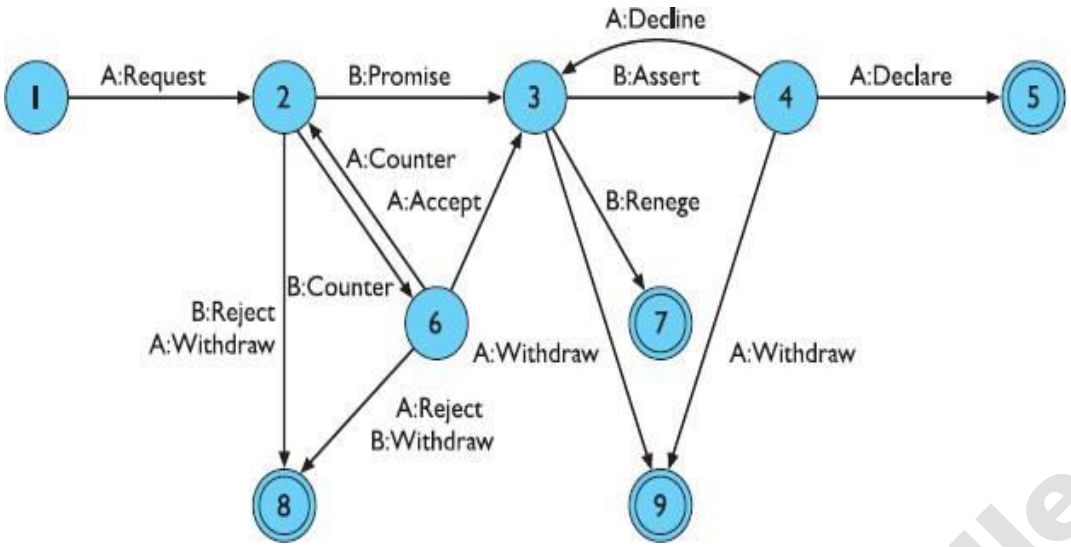


Fig.3.5: Conversation for action
Circles represent 'states' in the conversation
Arcs represent utterances (speech acts)

Stages between Conversations

• Request , • Promise , • Assert ,• Decline , • Reject , • Withdraw , • Counter , • Accept , • Renege , • Declare

There are other generic conversation forms as well as CfA. These include:

1. **conversation for clarification** usually embedded within a CfA to clarify the required action (different from countering a request);
2. **conversation for possibilities** looking toward future actions;
3. **conversation for orientation** building up a shared understanding.

CfA in action

- Simplest route 1–5:

Alison:	have you got the market survey on chocolate mousse?	*request*
Brian:	sure	*promise*
Brian:	there you are	*assert*
Alison:	thanks	*declare*

- More complex routes possible, e.g., 1–2–6–3 ...

Alison:	have you got ...	*request*
Brian:	I've only got the summary figures	*counter*
Alison:	that'll do	*accept*

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.
- Types of electronic text:

- discrete directed messages, no structure (email)
- linear messages added (in temporal order)
- non-linear hypertext linkages
- spatial two dimensional arrangement

- 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-to-face 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:

a) **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.
b) **linear** – participants' messages are added in (usually temporal) order to the end of a single transcript.
c) **non-linear** – when messages are linked to one another in a hypertext fashion.
d) **spatial** – where messages are arranged on a two-dimensional surface.

Back channels and affective state

- 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.The speaker would pause to seek back channel confirmation or to offer the floor, the text 'speaker' must either continue regardless, or finish the message, effectively passing the turn.
- One consequence of the lack of interruptions and more measured pace of interaction is that the utterances are more grammatical than speech. 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.

Problems with text

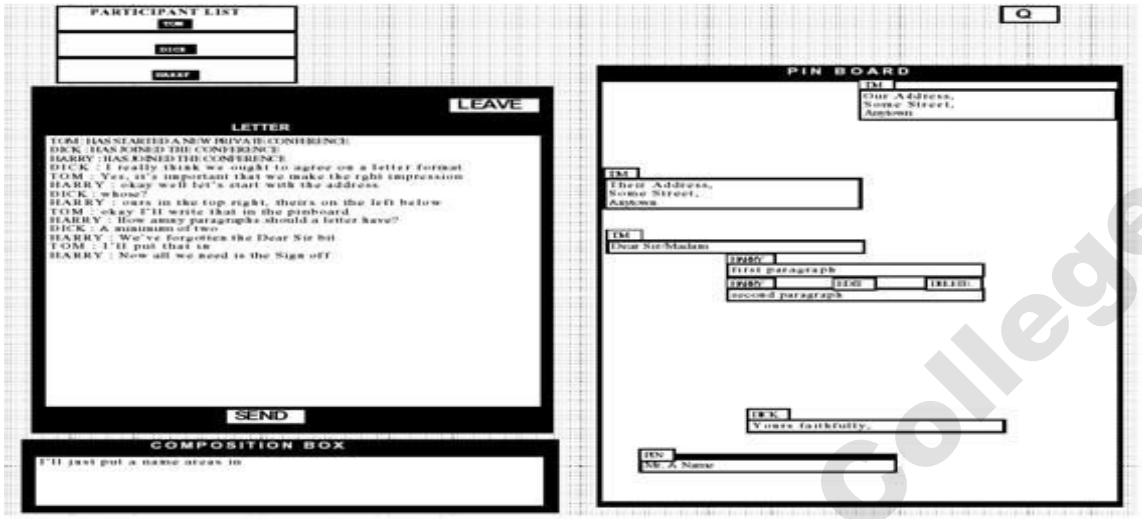
No facial expression or body language
⇒ weak *back channels*

So, difficult to convey:
affective state – happy, sad, ...
illocutionary force – urgent, important, ...

Participants compensate:
'flaming' and smilies
;-) :-(:-)

Grounding constraints

example - 'Conferencer'



linear conversation area – LHS RHS – spatial simulated pinboard

- This grounding process is linked strongly with the types of channels through which the conversants communicate. Clark and Brennan 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-face interaction. text-based system, different participants can compose simultaneously, but they lack cotemporality. 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.
- Linear transcripts obviously have some idea of sequence, but this is confused by the overlap and interleaving caused by the lack of cotemporality and imultaneity. Consider this typical interchange during the use of the York Conferencer system:

1. Bethan: How many should be in the group?
2. Rowena: Maybe this could be one of the four strongest reasons?
3. Rowena: Please clarify what you mean.
4. Bethan: I agree.
5. Rowena: Hang on.
6. Rowena: Bethan what did you mean?

- In a spoken conversation, Rowena and Bethan would have quickly corrected themselves if they began to speak at once, and the linearity would have reflected a *common* experience. The trouble is that the participants in the text-based conference each experienced the messages in a different order:

  Rowena: 2 1 3 4 5 6
  Bethan: 1 2 4 3 5 6

Turn-taking

- The fundamental structures of conversation was *turn-taking.*The last transcript was an example of a breakdown in turn-taking.Breakdowns are quite rare in two-party electronic conversations and are quickly corrected.
- In a pair of participants, turn-taking is simple; first one person says something, then the other. The only problem is deciding exactly *when* the exchange should happen. With three or more participants, turn-taking is more complex. They must decide *who* should have the next turn. This is resolved by face-to-face groups in a number of ways.
  - First, the conversation may, for a period, be focused on two of the parties, in which case normal two-party turn-taking holds.
  - Secondly, the speaker may specifically address another participant as the utterance is finished, either implicitly by body position, or explicitly: 'what do you think Alison?'
  - Finally, the next speaker may be left open, but the cotemporality of the audio channel allows the other participants to negotiate the turn.
- Basically, whoever speaks first, or most strongly, gets in. These mechanisms are aided by back channels, as one of the listeners may make it clear that she wants to speak. In this case, either the speaker will explicitly pass the turn (the second option above), or at least the other listeners are expecting her to speak.
- The movement between effective two-party conversation (the first option) and open discussion will be mediated by back channel messages from the other participants.
- In an unstructured text-based conversation the third option is not available, nor, of course, are the back channels. Paired conversation is quite common and the second option, explicitly naming the next speaker, is possible.

Context and deixis

- Utterances are highly ambiguous and are only meaningful with respect to *external context*, the state o the world, and *internal context*, the state of the conversation. Both of these are problems in text-based communication.
- The 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').
- The means of direct communication, remote participants have difficulty in using deictic reference. They cannot simply say 'that one', but must usually describe the referrant: 'the big circle in the corner'. If their displays are not WYSIWIS then they must also ensure that their colleague's display includes the object referred to and that the description is unambiguous.
- Asynchronous participants have even more problems with deixis as there is no opportunity for their colleagues there are also problems with deictic reference to internal context

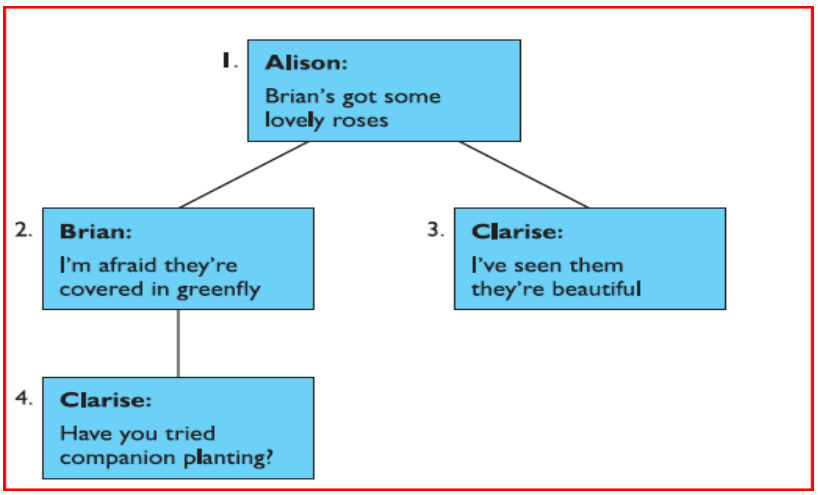


Fig3.6: Hypertext Conversation Structure

- In speech, the context is intimately connected to linear sequence and adjacency. As we have seen, even in linear text transcripts, overlap breaks the strict sequentiality of the conversation, and thus causes problems with indexicals and with context in general.

1. Alison: Brian's got some lovely roses.
2. Brian: I'm afraid they're covered in greenfly.
3. Clarise: I've seen them, they're beautiful.

- Hypertext-based systems avoid the implied sequentiality of a linear transcript. In the example, both Brian and Clarise replied to Alison's message at the same time. In a hypertext these would form parallel conversations. This is shown in above Figure , where in addition Clarise has sent a second message offering advice on Brian's greenfly. The use of 'they' in Clarise's message is now perfectly clear.

Pace and granularity

Pace of conversation – the rate of turn taking

face-to-face – every few seconds
telephone – half a minute
email – hours or days

as the pace of a conversation reduces, there is a tendency for the *granularity to increase*

- 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. Even most monologs are interactive in the sense that the speaker is constantly looking for cues of comprehension in the listener.

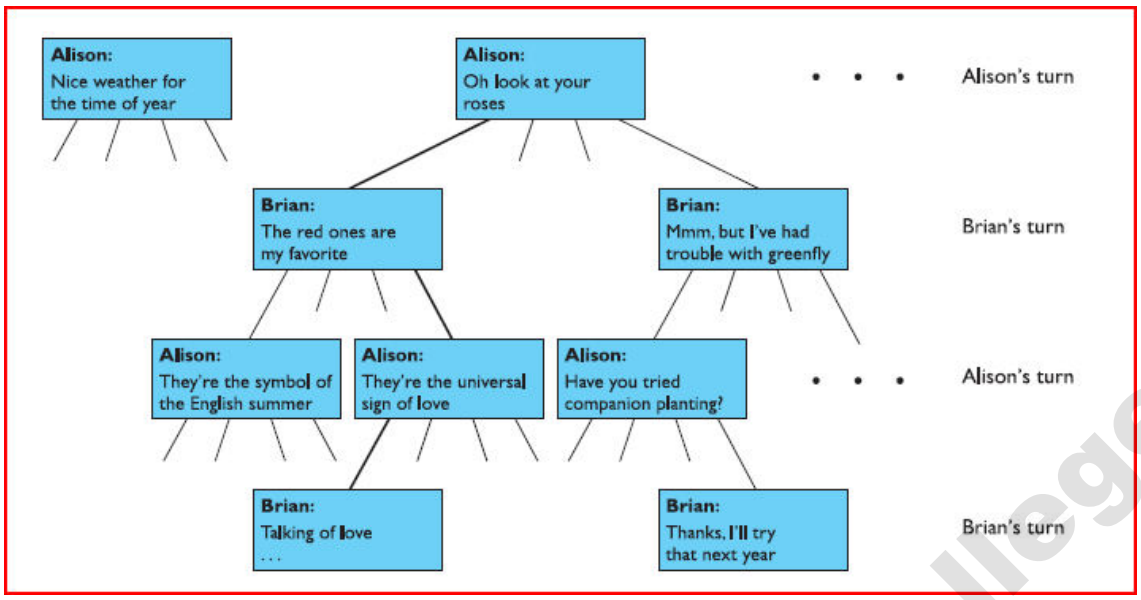


Fig3.7: Conversation Game

- Reducing the pace of a conversation reduces its *interactivity*. at the small scale of clarifying individual utterances, interactivity is important in determining the direction of a conversation. Imagine that the conversation is a little like a game, where the participants can make moves. In Figure , we can see some of the moves Alison and Brian can make whilst talking in the garden (Clarise has gone home).
- At each turn of the conversation, Alison or Brian can choose to say one thing which continues the discussion. That is, they gradually work out a path from the top of the tree downwards. A particularly promising conversation path is shown with bold lines. In a hypertext-based system one can expand several branches of a conversation tree, but in speech or in a linear text transcript the conversation follows one branch.
- Whatever medium is used, you cannot normally progress down the tree faster than the pace of the conversation. To overcome these limitations, people adopt several *coping strategies*. The simplest strategy is just to avoid conversation. This can be done by delegating parts of a task to the different participants. Each participant can then perform much of the task without communication. They must still communicate for large-scale strategic decisions, but have significantly reduced the normal communications.
- This approach reduces communication by reducing collaboration. More interesting in a cooperative work setting are two coping strategies which increase the chunk size of messages in order to reduce the number of interactions required to complete a task. These strategies are frequently seen in both text-based conferences and in letter writing.
- The first of these coping strategies is multiplexing. Basically, the conversants hold several conversations in parallel, each message referring to several topics. In terms of the conversation tree, this corresponds to going down several branches at once.

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.
- A hypertext that is created 'on the fly' is unlikely to be comprehensible to any but those involved in its creation. Conklin and Begeman, themselves associated with the hypertextbased argumentation tool gIBIS, conclude that 'traditional linear text provides a continuous, unwinding thread of context as ideas

GROUP WORKING

- 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, and then discuss the problems of studying group working.

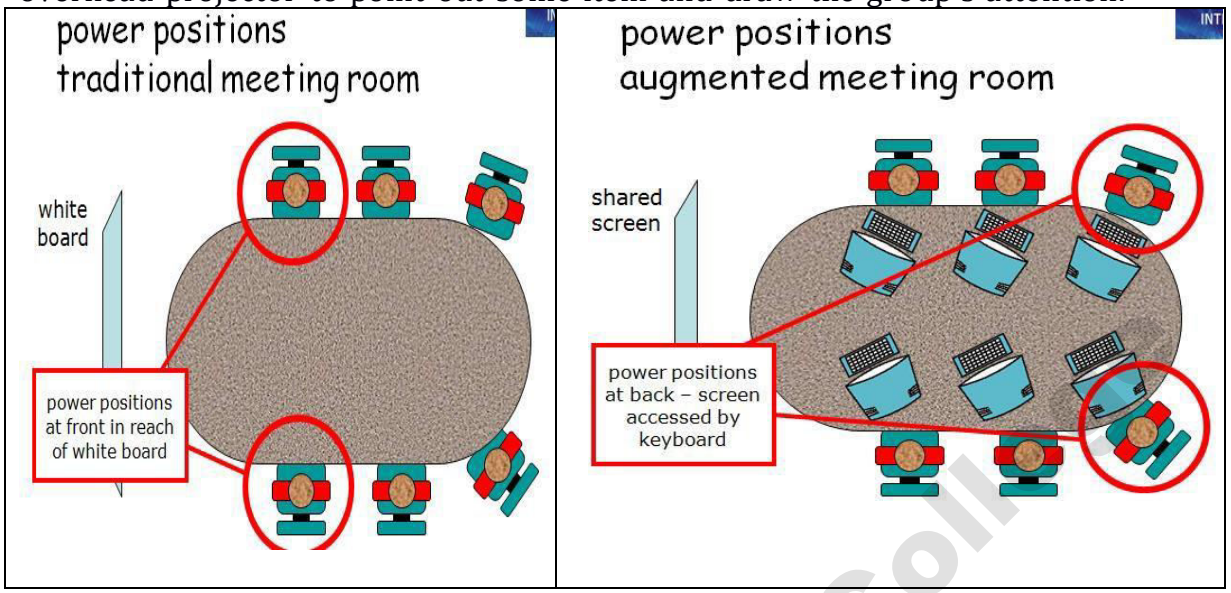
Group dynamics

- 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 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'. Not only do the social relationships *within* the group change, but the group membership and structure can change in time.
- A member leaving or a new member joining can cause dramatic changes in the behavior of the group. Groupware systems, for example *argumentation tools*, can help in that they record the history of the group. Groupware designers should in general be aware that new members can and will enter the group and should design their software accordingly. The group may also divide into subgroups for detailed discussion and then reform. Tools must be able to support this.

Physical layout

- 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.
- 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. the users still had some difficulty in adapting to the *power positions* in the electronic meeting room.
- At first sight, the electronic meeting room is not unlike a normal conference room. If the shared screen is a whiteboard or an overhead projector, then the most powerful position is toward the front of the room . Managers would

normally take this seat as they can then easily move to the whiteboard or overhead projector to point out some item and draw the group's attention.



Distributed cognition

- 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. 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.

HYPERTEXT, MULTIMEDIA AND WWW

Understanding Hypertext

- Hypertext attempts to get around these limitations of text by structuring it into a mesh rather than a line. This allows a number of different pages to be accessed from the current one, and, if the hypertext is well designed, the user should find it easier to follow his own particular idea through the mesh rather than being forced down one route.
- Typically, hypertext systems incorporate diagrams, photographs and other media as well as text. Such systems are often known as *multimedia* or *hypermedia* systems, although the three terms are often used interchangeably.
- A hypertext system comprises a number of pages and a set of *links* that are used to connect pages together. The links can join any page to any other page, and there can be more than one link per page. Thus a hypertext document does not simply start a linear progression and follow it to an end, but goes in lots of different directions, some of which terminate, while others link back into different parts of the document.
- There are many different ways of traversing the network, and so there are many different ways of reading a hypertext document – the intention is that the user is able to read it in the way that suits him best. Links can exist at the

end of pages, with the user choosing which one to follow, or can be embedded within the document itself.

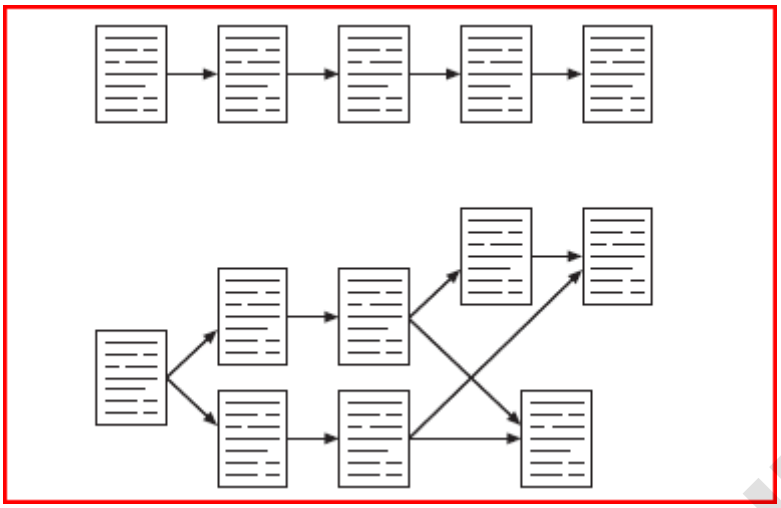


Fig3.8: Typical structures of linear text and hypertext

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

Video and audio

- 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.

Delivery technology

On the computer

- 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

- 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.

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 wireless access points or through modems linked to mobile phone networks. 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. Only when you visit the location does the note become visible. This is a bit like an image map on a web page, but rather than clicking a mouse over an image to link to information, here you need to physically move to a location! Another example is the GUIDE system, which uses various means to detect location (closest network access point or GPS) and then delivers appropriate tourist information.

Application areas

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, bitmaps and so on, and hot-spots on the cards allow movement between cards.
- Cards may also contain forward and backward buttons and a home icon, to allow the user to move sequentially and start from scratch respectively. HyperCard can be used for a range of applications including information management and teaching.
- HyperCard's simple scripting language and easy to produce graphical interfaces meant it was also used extensively as a rapid prototyping tool for generating interactive systems. In fact, HyperCard stacks for both single-user and networked applications are available from the book website.

Help and documentation

- Hypermedia systems are ideally suited to online manuals and other help system applications. 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.

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. 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.
- Establishing a sense of community can be very important on websites as it is one way to ensure loyalty and get visitors to return. This may involve explicit community features such as chat areas, or may simply be a matter of using a design, language and image that suggests a site which is open and listening to 'readers'.

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. Although the hypertext may have a hierarchical or other structure, the user may navigate by hyperlinks that move across this main structure. It is easy to lose track of where you are and where you have been.

Designing structure

- 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.

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. Another type of hypertext takes the form of 'levels of access' to a document. Different levels of access privilege 'see' different amounts of information.
- A document structured in this way may provide one level of access that gives only a brief overview of the topic. The next level of access presents a fuller description of the system, while the next level may also include information regarding the precise meaning of technical terms used in the system. The final level of access may add historical information and such like

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. Although the back button is used extensively, it is used relatively little to go back more then one step. For error correction this makes sense, but for general revisiting one might think that moving back several steps would be common.

Indices, directories and search

- 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! The words in an index are chosen because they are significant key phrases or words with a domain meaning, and not every occurrence of a word is indexed, only those deemed in some way important.
- The main difference between an electronic index and a paper one is that with the paper index you have to physically look up the page after finding the word in the index, whereas in an electronic index the links are 'live' so you can simply click to the content.

Web Technology And Issues

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. web owes its success to many factors, including the robustness and (relative) ease of use offered by popular browsers from the very first graphical browser Mosaic, and continued in commercial browsers such as Netscape Navigator, Microsoft Internet Explorer and Opera. These offer a graphical interface to the document, controlled by the mouse. Hypertext links are shown by highlighting the text that acts as the link in an alternative color, and are activated by clicking on the link. A further color is used to indicate a link that has already been visited. Hypertext links can also be embedded into regions within an image.

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.

- 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.

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'! 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.
- Latency is caused by several factors – the finite speed of electrical or optical signals (no faster than the speed of light), and delays at routers along the way that take messages from one computer network and pass them on. This latency may not always be the same, varying with the exact route through the network traveled by a message, the current load on the different routers,etc. Variability in the latency is called jitter.

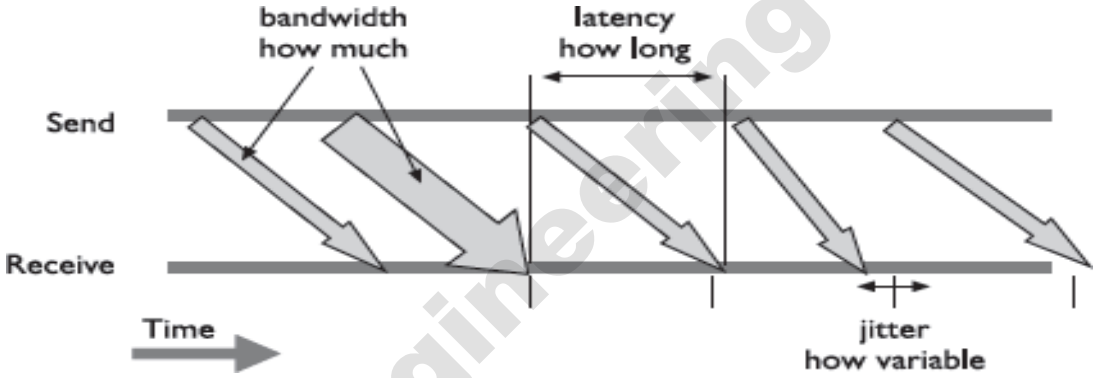


Fig.3.9: Bandwidth, Latency and Jitter

Static Web Content

The message and the medium

- Excellent page design can make useless material look attractive, but it still remains useless material. 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 and 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, and make an effort to retrieve it, which clearly must have an influence on design.

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. 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 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. There is an increasing desire to have fine control.
- Cascading style sheets (CSS) allow you to specify fonts, line spacing, size, etc., in a similar way to styles in a word processor or DTP package. The use of color is of great importance for web pages, but it is often abused.
- First, it should be remembered that a significant proportion of the potential viewers of the page will have problems with color, either because they are using older machines with a limited color palette, or because they have some form of color blindness. Color, when used, should not be the only cue available.

Graphics

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.

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.

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.
- The need to download movies and sound puts sharp limits on the length of clip that can be shown. Streaming media over the internet, such as RealVideo, RealAudio and CuSeeMe, allow potentially unlimited sources.

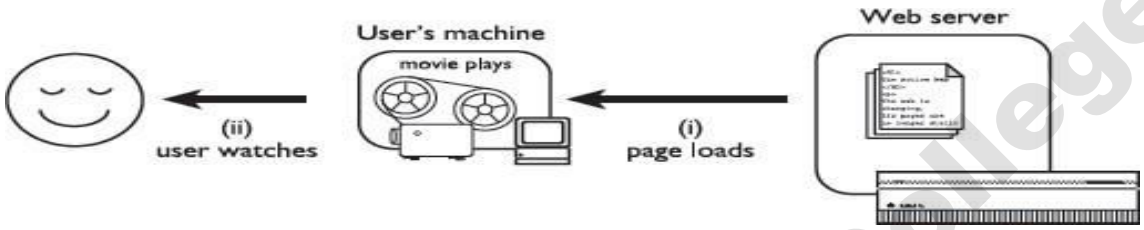


Fig.3.10: Animated GIF or movie needs to download completely

Dynamic Web Content The active web

- 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.

What happens where

- When considering dynamic material on the web we need to take the external, user's viewpoint and ask what is changing: media, presentation or actual data; by whom: by the computer automatically, by the author, by the end-user or by another user; and how often, the pace of change: seconds, days or months? From a technical standpoint, we also need to know where 'computation' is happening: in the user's web-browsing client, in the server, in some other machine or in the human system surrounding it? The 'what happens where' question is the heart of architectural design.
- 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. 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.

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 and thereafter all interaction happens on the PC, not across the network

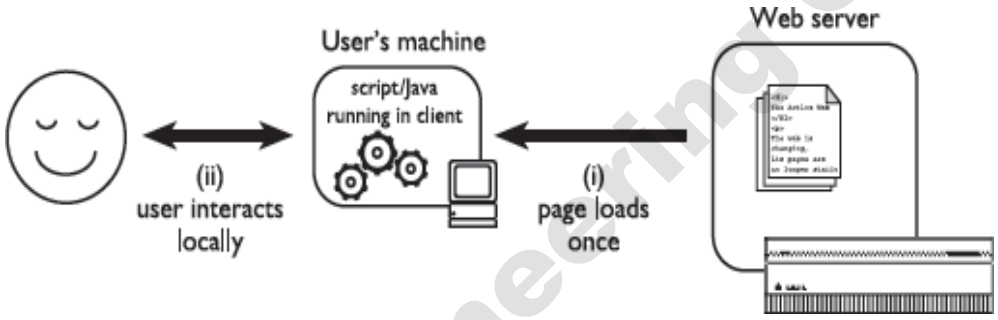


Fig.3.11: Java applet or Javascript running locally

Search

- Some user-driven interaction can be accommodated at the client end, but not all. Consider search engines. It would be foolish to download several megabytes of information so that a Java applet can search it online! Instead, all common websearch pages work by submitting forms to the server where CGI programs perform the searches and return results.
- An additional reason for this approach is that most browsers support forms, but some still do not support Java or scripting in a consistent manner. The web search engine for this book works in this way. 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

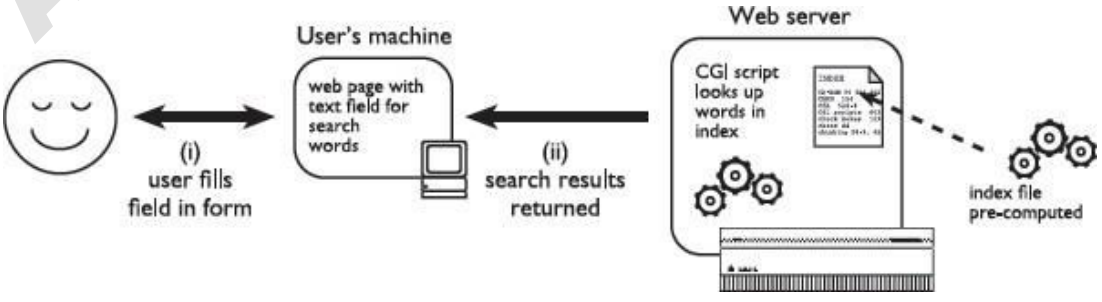


Fig.3.11: HCI Book Search

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, enter 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.

ARUNAI ENGINEERING COLLEGE

DEPARTMENT OF CSE

IV YEAR - VII SEMESTER

CS8079 – HUMAN COMPUTER INTERACTION (R2017)

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. - Case Studies

BRIEF HISTORY OF MOBILE

The Brick Era calls

- It was Portable!
- More expensive than payphones
- Enormous battery
- Stakeholders: Stockbrokers, sales people, ...
- After a while, more cellular radio towers and... it got (a little bit) smaller

The Candy Bar Era calls SMS

- 2G network : GSM, CDMA, TDMA, iDEN
- More cellular towers
  - less power needed
  - much smaller
- Better voice quality
- Added SMS
- Everyone wanted to have a mobile phone
  - economic prosperity in EU, USA, and JP

The Feature Phone Era calls SMS & MMS music & photos

- 2.5G network: GPRS
- Camera
- MMS
- Data-capable devices
- Internet on mobile (very poor)
  - high prices
  - poor marketing
  - inconsistent rendering

The Smartphone Era calls SMS & MMS music & photos

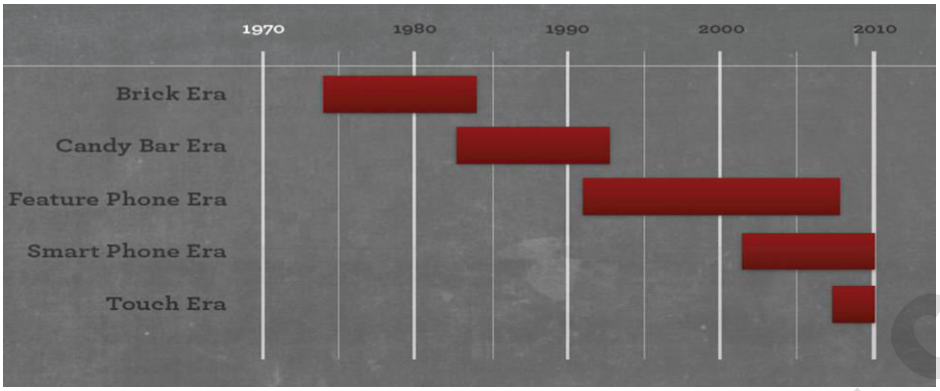
- 3G, HSDPA, WI-FI
- Like a feature phone, but simulating a PC
- Its own OS (es. Symbian)
- Larger screens, stylus
- The Mobile Platform becomes key
- (push) email as primary driver

The Touch Era calls SMS & MMS music & photos APPS

(NOT a phone - NOT a computer)

3G, 4G

- Accelerometers
- GPS/Location-based
- User-centered design



MOBILE ECOSYSTEM

Mobile is an entirely unique ecosystem and, like the Internet, it is made up of many different parts that must all work seamlessly together. However, with mobile technology, the parts are different, and because you can use mobile devices to access the Internet, that means that not only do you need to understand the facets of the Internet, but you also need to understand the mobile ecosystem. Think of the mobile ecosystem as a system of layers, as shown in Figure 4.1. Each layer is reliant on the others to create a seamless, end-to-end experience. Although not every piece of the puzzle is included in every mobile product and service, for the majority of the time, they seem to add complexity to our work, regardless of whether we expressly put them there.

Services
Applications
Application frameworks
Operating Systems
Platforms
Devices
Aggregators
Networks
Operators

Figure 4.1. The layers of the mobile ecosystem

Although the mobile ecosystem consists of many different components, probably the most recognizable and important one is the mobile phone. All phones sold today fall into one of three categories: feature phones, smart phones, or touch phones.

- **Feature Phones:** The vast majority of cell phone users in the US have feature phones. These are the basic flip phones that come free or at a low-cost with carrier contracts and pre-paid plans. Features phones get their name from the various features that come with the devices. These phones generally have camera, a handful of applications, and rudimentary web browsers. Prior to feature phones, cell phones only made calls and sent and received text messages.
- **Smart phones:** They makeup a much smaller portion of the US market than feature phones, but smartphones are also very popular devices, especially among attorneys. The most recognizable smartphone is the Blackberry. In addition to all the capabilities of feature phones, smartphones typically run more applications and an operating system, have a larger screen size, and utilize a QWERTY keyboard input (standard keyboard format).
- **Touch phones:** Since the introduction of the iPhone in 2007, the fastest growing category of phones in the US market have been touch phones. Touchphones can be thought of as the next generation of smartphones - they have larger screens, more robust web browsers, and more powerful applications. Touchphone users are also the mobile web's power users. A recent UK study showed that 93% of iPhone owners use their device to access news and information on the mobile web.
- **Other mobile devices** - iPads & tablets are also entering the mobile space and should be watched as their market share increases, especially considering that higher end models have built-in wifi capabilities.

Operators

- Operators are also referred as Mobile Network Operators (MNOs); mobile service providers, wireless carriers, or simply carriers; mobile phone operators; or cellular companies
- Essentially make the entire mobile ecosystem work (Gate keepers)
- Operator's role in the ecosystem is to create and maintain a specific set of wireless services over a reliable cellular network
- to create and maintain wireless services over a reliable cellular network
- Operators operate wireless networks

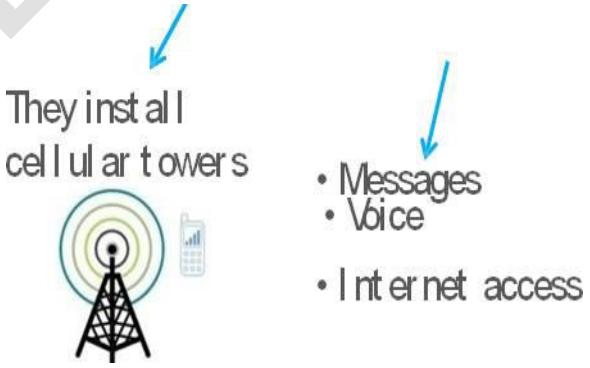
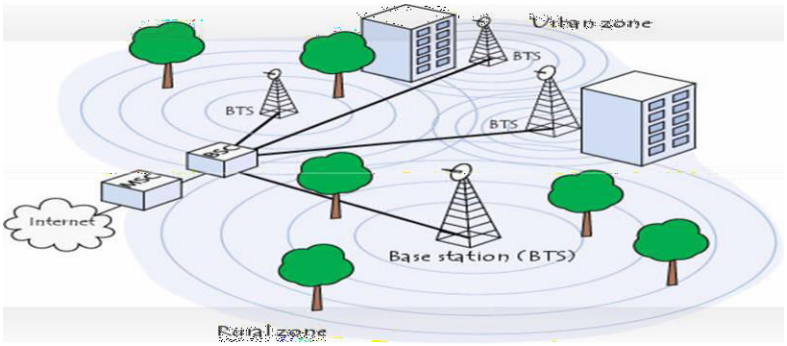


Table 2-1. World's largest mobile operators

Rank	Operator	Markets	Technology	Subscribers (in millions)
1.	China Mobile	China (including Hong Kong) and Pakistan	GSM, GPRS, EDGE, TD-SCDMA	436.12
2.	Vodafone	United Kingdom, Germany, Italy, France, Spain, Romania, Greece, Portugal, Netherlands, Czech Republic, Hungary, Ireland, Albania, Malta, Northern Cyprus, Faroe Islands, India, United States, South Africa, Australia, New Zealand, Turkey, Egypt, Ghana, Fiji, Lesotho, and Mozambique	GSM, GPRS, EDGE, UMTS, HSDPA	260.5
3.	Telefónica	Spain, Argentina, Brazil, Chile, Colombia, Ecuador, El Salvador, Guatemala, Mexico, Nicaragua, Panama, Peru, Uruguay, Venezuela, Ireland, Germany, United Kingdom, Czech Republic, Morocco, and Slovakia	CDMA, CDMA2000 1x, EV-DO, GSM, GPRS, EDGE, UMTS, HSDPA	188.9
4.	América Móvil	United States, Argentina, Chile, Colombia, Paraguay, Uruguay, Mexico, Puerto Rico, Ecuador, Jamaica, Peru, Brazil, Dominican Republic, Guatemala, Honduras, Nicaragua, Ecuador, and El Salvador	CDMA, CDMA2000 1x, EV-DO, GSM, GPRS, EDGE, UMTS, HSDPA	172.5
5.	Telenor	Norway, Sweden, Denmark, Hungary, Montenegro, Serbia, Russia, Ukraine, Thailand, Bangladesh, Pakistan, and Malaysia	GSM, GPRS, EDGE, UMTS, HSDPA	143.0
6.	China Unicom	China	GSM, GPRS	127.6
7.	T-Mobile	Germany, United States, United Kingdom, Poland, Czech Republic, Netherlands, Hungary, Austria, Croatia, Slovakia, Macedonia, Montenegro, Puerto Rico, and U.S. Virgin Islands	GSM, GPRS, EDGE, UMTS, HSDPA	126.6
8.	TeliaSonera	Norway, Sweden, Denmark, Finland, Estonia, Latvia, Lithuania, Spain, and Central Asia	GSM, GPRS, EDGE, UMTS, HSDPA	115.0

NETWORKS

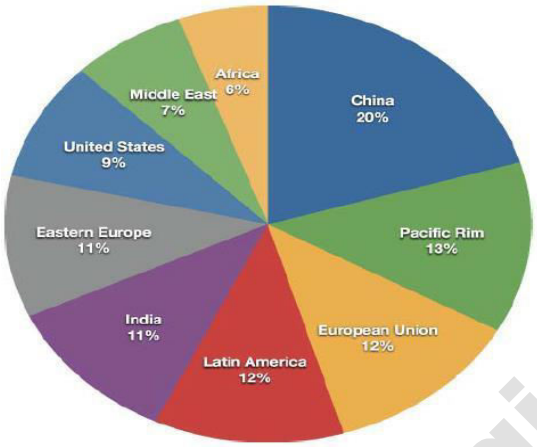
Mobile networks communicate through electromagnetic radio waves with a cell site base station, the antennas of which are usually mounted on a tower,



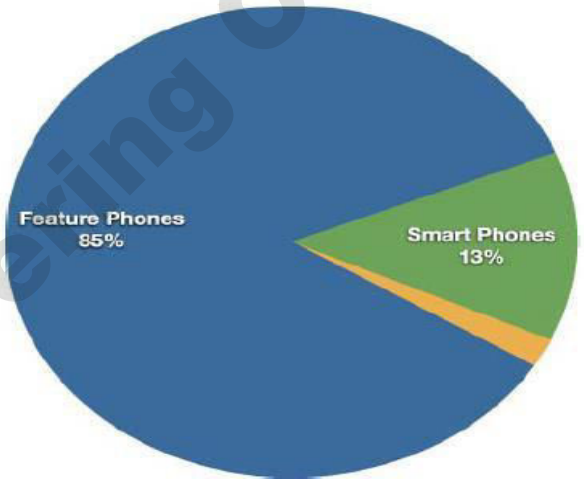
2G	Second generation of mobile phone standards and technology	Speeds
GSM	Global System for Mobile communications	12.2 kbits/s
GPRS	General Packet Radio Service	max 60 kbits/s
EDGE	Enhanced Data rates for GSM Evolution	59.2 kbits/s
HSCSD	High-Speed Circuit-Switched Data	57.6 kbits/s
3G	Third generation of mobile phone standards and technology	Speeds
W-CDMA	Wideband Code Division Multiple Access	14.4 Mbits/s
UMTS	Universal Mobile Telecommunications System	3.6 Mbits/s
UMTS-TDD	Time Division Duplexing	16 Mbits/s
TD-CDMA	Time Divided Code Division Multiple Access	16 Mbits/s
HSPA	High-Speed Packet Access	14.4 Mbits/s
HSDPA	High-Speed Downlink Packet Access	14.4 Mbits/s
HSUPA	High-Speed Uplink Packet Access	5.76 Mbit/s

DEVICES

handsets or terminals in industry , But also other devices such as tablets, ebook readers…



Mobile Devices around the world



Breakdown of devices

- **Feature Phones:** The vast majority of cell phone users in the US have feature phones. These are the basic flip phones that come free or at a low-cost with carrier contracts and pre-paid plans. Features phones get their name from the various features that come with the devices. These phones generally have camera, a handful of applications, and rudimentary web browsers. Prior to feature phones, cell phones only made calls and sent and received text messages.
- **Smart phones:** They makeup a much smaller portion of the US market than feature phones, but smartphones are also very popular devices, especially among attorneys. The most recognizable smartphone is the Blackberry. In addition to all the capabilities of feature phones, smartphones typically run more applications and an operating system, have a larger screen size, and utilize a QWERTY keyboard input (standard keyboard format).
- **Touch phones:** Since the introduction of the iPhone in 2007, the fastest growing category of phones in the US market have been touch phones. Touchphones can be thought of as the next generation of smartphones - they have larger screens, more robust web browsers, and more powerful applications. Touchphone users are also the mobile web's power users. A recent UK study showed that 93% of iPhone owners use their device to access news and information on the mobile web.

Other mobile devices - iPads & tablets are also entering the mobile space and should be watched as their market share increases, especially considering that higher end models have built-in wifi capabilities.

PLATFORMS

A mobile platform's primary duty is to provide access to the devices. To run software and services on each of these devices, you need a platform, or a core programming language in which all of your software is written. Like all software platforms, these are split into three categories:

1. Open Source: free to use and modify
   - Android
2. Proprietary: by device makers
   - iPhone, BlackBerry, Palm
3. Licensed: sold to device makers
   - JavaME, BREW, Windows Mobile

1) 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.

2) 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.

3) Open Source

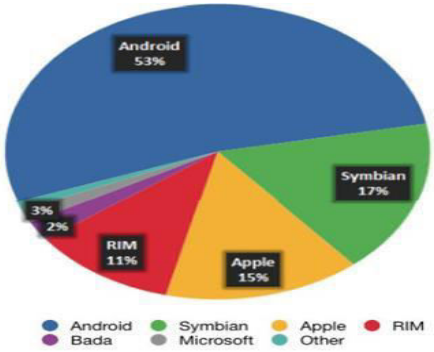
Open source platforms are mobile platforms that are freely available for users to down load, alter, and edit. Open source mobile platforms are newer and slightly controversial, but they are increasingly gaining traction with device makers and developers. Android is one of these platforms. It is developed by the Open Handset Alliance, which is spear-headed by Google. The Alliance seeks to develop an open source mobile platform based on the Java programming language.

Operating Systems

It used to be that if a mobile device ran an operating system, it was most likely considered a smartphone. But as technology gets smaller, a broader set of devices supports 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.
You might notice that many of these operating systems share the same names as the platforms on which they run. Mobile operating systems are often bundled with the platform they are designed to run on.

OPERATING SYSTEM

OS have core services or toolkits that enable apps to talk to each other and share data or services.OSs are common in Smart Phones, but rare in Feature phones.



Smart Phones by OS

- 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 : M*obile operating system that runs on top of the Windows Mobile platform.
- *Palm OS : OS* 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 : runs its own open source operating system, which can be customized by operators and device manufacturers.*

APPLICATION FRAMEWORKS

Often, the first layer the developer can access is the application framework or API released by one of the companies mentioned already. The first layer that you have any control over is the choice of application framework. 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 frame-work. 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, but have 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. And based on the number of mobile web apps written for the iPhone versus the number written for other, more full-featured WRTs. Nonetheless, it is a move in the right direction.

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. 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. Although 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

Definition: In the realm of technology, this usually refers to a computer program that runs on a website (Google Apps), a small computing device (iPad App) or a cell phone (Android App).

- Apps live between the device and the user
- They must fit with their usage context
- They must know the specific device attributes and capabilities
- FRAGMENTATION PROBLEM

SERVICES

Services are "everything the user is trying to do"
They are often available at different levels:

- Application
- Application Framework
- OS

Example services may include:

- the Internet
- sending a text message
- being able to get a location

All of these layers must be passed through before you get to the content

7th Mass MEDIA - MOBILE

1)	Printing Press
2)	Recordings
3)	Cinema
4)	Radio
5)	Television
6)	Internet
7)	MOBILE

TYPES OF MOBILE APPLICATIONS

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; this decision is determined by the impact it will have on the user experience. The technical capabilities and capacity of the publisher also factor into which approach to take. Earlier the common mobile platforms was discussed in terms of how they factor in the larger mobile ecosystem. Now let us look deeper into each of these platforms from a more tactical perspective, unpacking them, so to speak, to see what is inside. Figure 4.2 illustrates the spectrum of mobile media; it starts with the basic text-based experiences and moves on to the more immersive experiences.

Figure 4.2. Multiple mobile application medium types

MOBILE APPLICATION MEDIUM TYPES

Figure 6-1. Multiple mobile application medium types

I.SMS

2. mobile websites

3. mobile web widgets

4. mobile web applications

5. native applications, and

6. Games.

SMS:

The most basic mobile application you can create is an SMS application. Although it might seem odd to consider text messages applications, they are nonetheless a designed experience. Given the ubiquity of devices that support SMS, these applications can be useful tools when integrated with other mobile application types.

Typically, the user sends a single keyword to a five-digit short code in order to return information or a link to premium content. For example, sending the keyword "freebie" to a hypothetical short code "12345" might return a text message with a coupon code that could be redeemed at a retail location, or it could include a link to a free ringtone. SMS applications can be both "free," meaning that there is no additional charge beyond the text message fees an operator charges, or "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.

Figure 6-2. An SMS application to interact with a billboard in Manhattan

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:

Figure 4.3. An example of a mobile website

As you might expect, 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 that take you to a page a level deeper, as shown in Figure 4.3.

Mobile websites often have a simple design and are typically informational in nature, offering few—if any—of the interactive elements you might expect from a desktop site. Mobile websites have made up the majority of what was consider the mobile web for the past decade, starting with the early WML-based sites (not much more than a list of links) 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. For example, in just one year, the U.S. market went from being just barely in the top five consumers of the mobile web to number one, largely due to the impact of the iPhone alone.

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

Largely in response to the poor experience provided by the mobile web over the years, there has been a growing movement to establish mobile widget frameworks and platforms. For years the mobile web user experience was severely underutilized and failed to gain traction in the market, so several operators, device makers, and publishers began creating widget platforms (as shown in figure 4.4) to counter the mobile web's weaknesses. Initially user saw mobile web widgets as another attempt by the mobile industry to hype a technology that no one wants. So in order to define a mobile web widget: 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 HTML- based 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.

figure 4.4 : An example mobile web widget

Basically, mobile web widgets are small web applications that can't run by themselves; they need to be executed on top of something else. One reason for all the confusion around what is a mobile web widget is that this definition can also encompass any web application that runs in a browser. 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), user can create compelling user experiences that tap into device features and, in many cases, can run while the device is offline. Widgets, however, are not to be confused with the utility application context, a user experience designed around short, task-based operations.

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. By "application-like" experience, mean that they do not use the drill-down or page metaphors in which a click equals a refresh of the content in view. Web applications allow users to interact with content in real time, where a click or touch performs an action within the current view.

The history of how mobile web applications came to be so commonplace is interesting, and is one that can give an understanding of how future mobile trends can be assessed and understood. Shortly after the explosion of Web 2.0, web applications like Facebook, Flickr, and Google Reader hit desktop browsers, and there was discussion of how to bring those same web applications to mobile devices. 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.

Figure 4.5 : The Facebook mobile web app

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.

To make matters worse, the perceived market demand for mobile web applications was not seen as a priority with many operators and device makers. It

was the classic chicken-or-the-egg scenario. What had to come first, market demand to drive browser innovation or optimized content to drive the market?
With the introduction of the first iPhone, we saw 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 (as shown in figure 4.5). The combination of a high-profile device with an incredibly powerful mobile web browser and a quickly increasing catalog of nicely optimized experiences created the perfect storm the community had been waiting for.
Usage of the mobile web exploded with not just users of the iPhone, but users of other handsets, too. Because web applications being created for the iPhone were based on web standards, they actually worked reasonably well on other devices. Operators and device makers saw that consumers wanted not just the mobile web on their handsets, but the regular Web, too. In less than a year, we saw a strong unilateral move by all operators and devices makers to put better mobile web browsers in their phones that could leverage this new application medium. We have not seen such rapid innovation in mobile devices since the inclusion of cameras.
The downside, of course, like all things mobile-web-related, is that not all devices support the capability to render mobile web applications consistently. However, we do see a prevalent trend that the majority of usage of the mobile web is coming from the devices with better browsers, in some markets by a factor of 7:1. So although creating a mobile web application might not reach all devices, it will reach the devices that create the majority of traffic.
Pros : The pros of mobile web applications are:

- 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 cons of mobile web applications are:

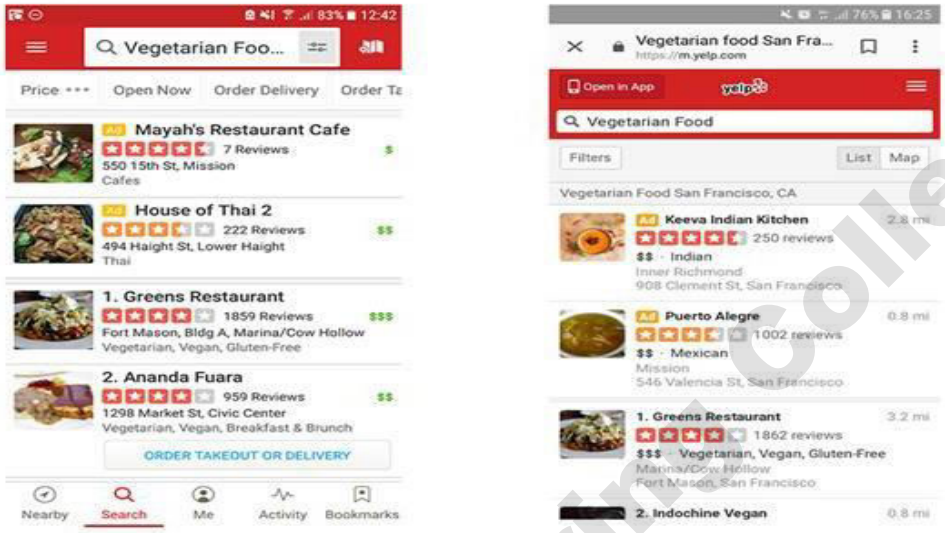
- The optimal experience might not be available on all handsets.
- They can be challenging (but not impossible) to support across multiple devices.
- They don't always support native application features, like offline mode, location lookup, file system access, camera, and so on.

NATIVE APPLICATIONS

- called "platform applications,"
- have to be developed and compiled for each mobile platform.
- native or platform applications are built specifically for devices that run the platform
- most common of all platforms is Java ME (formerly J2ME)
- In addition to Java, other smartphone programming languages include versions of C, C++, and Objective-C

- Because platform applications sit on top of the platform layer, they can tap into the majority of the device features, working online or offline, accessing the location and the filesystem
- the majority (70 %) of native applications in use today could be created with a little bit of XHTML, CSS, and JavaScript

Yelp native app vs. Yelp.com web app



Difference

Web apps

- need an active internet connection in order to run,
- will update themselves

Mobile Native apps

- may work offline.
- faster and more efficient, but they do require the user to regularly download updates.

Pros

The pros of native applications include:

- They offer a best-in-class user experience, offering a rich design and tapping into device features and offline use.
- They are relatively simple to develop for a single platform.
- You can charge for applications.

Cons

The cons of native applications include:

- They cannot be easily ported to other mobile platforms.
- Developing, testing, and supporting multiple device platforms is incredibly costly.
- They require certification and distribution from a third party that you have no control over.
- They require you to share revenue with the one or more third parties.

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 (as shown in figure 4.6). But this is treated differently from native applications for two reasons: they cannot be easily duplicated with web technologies, and porting them to multiple mobile platforms is a bit easier than typical platform-based applications.

Figure 4.6 : An example game for the iPhone

Although user can do many things with a powerful mobile web browser, creating an immersive gaming experience is not one of them—at least not yet. Seeing as how these types of gaming experiences appear on the desktop using standard web technologies, but is believed we are still a few years out from seeing them on mobile devices. Adobe's Flash and the SVG (scalable vector graphics) standard are the only way to do it on the Web now, and will likely be how it is done on mobile devices in the future, the primary obstacle being the performance of the device in dealing with vector graphics.

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.

These differences, are what make mobile games stand apart from all other application genres—their capability to be unique and difficult to duplicate in another application type, though the game itself is relatively easy to port.

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.

Table 6-1. Mobile application media matrix

	Device support	Complexity	User experience	Language	Offline support	Device features
SMS	All	Simple	Limited	N/A	No	None
Mobile websites	All	Simple	Limited	HTML	No	None
Mobile web widgets	Some	Medium	Great	HTML	Limited	Limited
Mobile web applications	Some	Medium	Great	HTML, CSS, JavaScript	Limited	Limited
Native applications	All	Complex	Excellent	Various	Yes	Yes
Games	All	Complex	Excellent	Various	Yes	Yes

MOBILE INFORMATION ARCHITECTURE

It defines not just how your information will be structured, but also how people will interact with it.

Although information architecture has become a common discipline in the web industry, unfortunately, the mobile industry has only a handful of specialized mobile information architects.

For example, if we look at the front page of http://www.nytimes.com as seen from a desktop web browser compared to how it may render in a mobile browser (as shown in figure 4.7), we see a content-heavy site that works well on the desktop, and is designed to present the maximum amount of information above the "fold" or where the screen cuts off the content. However, in the mobile browser, the text is far too small to be useful.

Figure 4.7: Comparing the New York Times website in desktop &mobile browsers

Figure 4.8 : The many mobile experiences of the New York Times

The role of a mobile information architect would be to interpret this content to the mobile context. Do you use the same structure, or sections? Do you present the same information above the fold? If so, how should that be prioritized? How does the user navigate to other areas? Do you use the same visual and interaction paradigms, or invent new ones? And if you do start to invent new paradigms, will you lose the visual characteristics of what users expect?
These are only some of the questions asked when starting to create a mobile information architecture. As you can see in figure above there are several different ways that the New York Times has been interpreted for the mobile context. Also it is needed to design our mobile information architecture to address the mobile context. Given that many devices can detect user current location, which is one of the most immediate types of context, how does the New York Times application address the user's context? For example, as a publication that serves both New York City and a larger global audience, if user is not in New York, should user still see the local New York headlines? Or should user see the headlines based on current location? This is shown in figure 4.8.

Keeping It Simple
When thinking about your mobile information architecture, you want to keep it as simple as possible.

Support your defined goals: If something doesn't support the defined goals, lose it. Go back to the user goals and needs, and identify the tasks that map to them. Find those needs and fill them.
Ask yourself: what need does my application fill? What are people trying to do here? Once it is understand that, it is a simple process of reverse-engineering the path from where they want to be to where they are starting. Cut out everything else.

Clear, simple labels: Good trigger labels, the words used to describe each link or action, are crucial in Mobile. Words like "products" or "services" aren't good trigger labels. They don't tell anything about that content or what can expect. Users have a much higher threshold of pain when clicking about on a desktop

site or application, hunting and pecking for tasty morsels. Mobile performs short, to-the-point, get-it-quick, and get-out types of tasks. What is convenient on the desktop might be a deal breaker on mobile. Keep all labels short and descriptive, and never try to be clever with the words used to evoke action.

The worst sin is to introduce branding or marketing into information architecture; this will just serve to confuse and distract users. Executives love to use the words they use internally to external communications on websites and applications, but these words have no meaning outside of your company walls. Don't try to differentiate product offering by what it is called. Create something unique by creating a usable and intuitive experience based on focusing on what users need and using the same language they use to describe those needs.

Based on what web design is, should use simple, direct terms for navigating around pages rather than overly clever terms. That latter typically result in confused visitors who struggle to find the content they are looking for. When that happens, they will go elsewhere to look for the information they want. So, if the these same mistakes is applied to a constrained device like mobile, then it ends up adding confusion to the user experience at a higher magnitude than the Web.

Site Map:

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 4.9. 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.

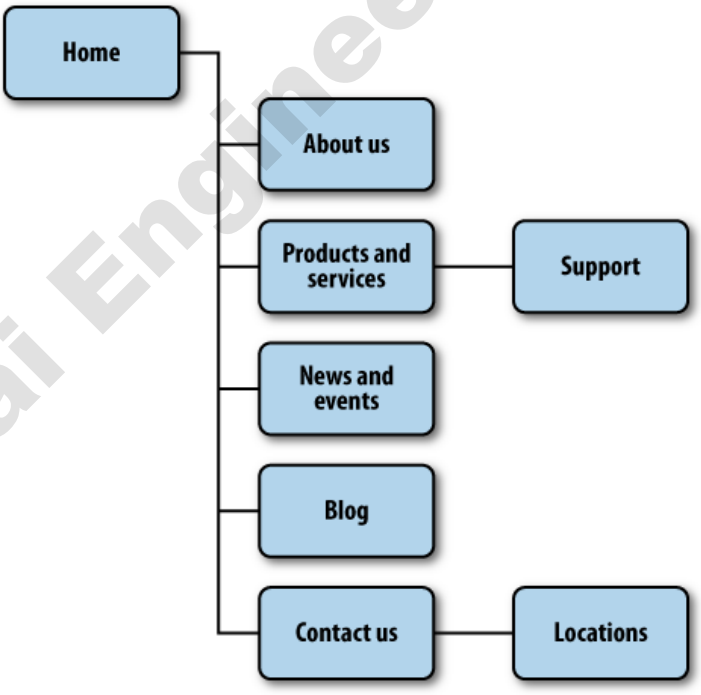


Figure 4.9 : An example mobile site map

Limit opportunities for mistakes:

Now think of your own website. How many primary navigation areas do you have? Seven? Eight? Ten? Fifteen? What risk is there to the users for making a wrong choice? If they go down the wrong path, they can immediately click back to where

they started and go down another path, eliminating the wrong choices to find the right ones. The risks for making the wrong choice are minor. In Figure 4.10, there is a 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, this assumption cannot be maked. In the mobile context, tasks are short and users have limited time to perform them. And with mobile websites, it can't be assumed that the users have access to a reliable broadband connection that allows them to quickly go back to the previous page. In addition, the users more often than not have to pay for each page view in data charges. So not only do they pay cash for viewing the wrong page by mistake, they pay to again download the page they started from. Therefore, it is to limit users' options, those forks in the road, to five or less. Anything more, and have to introduce far too much risk that the user will make a mistake and head off in the wrong direction.

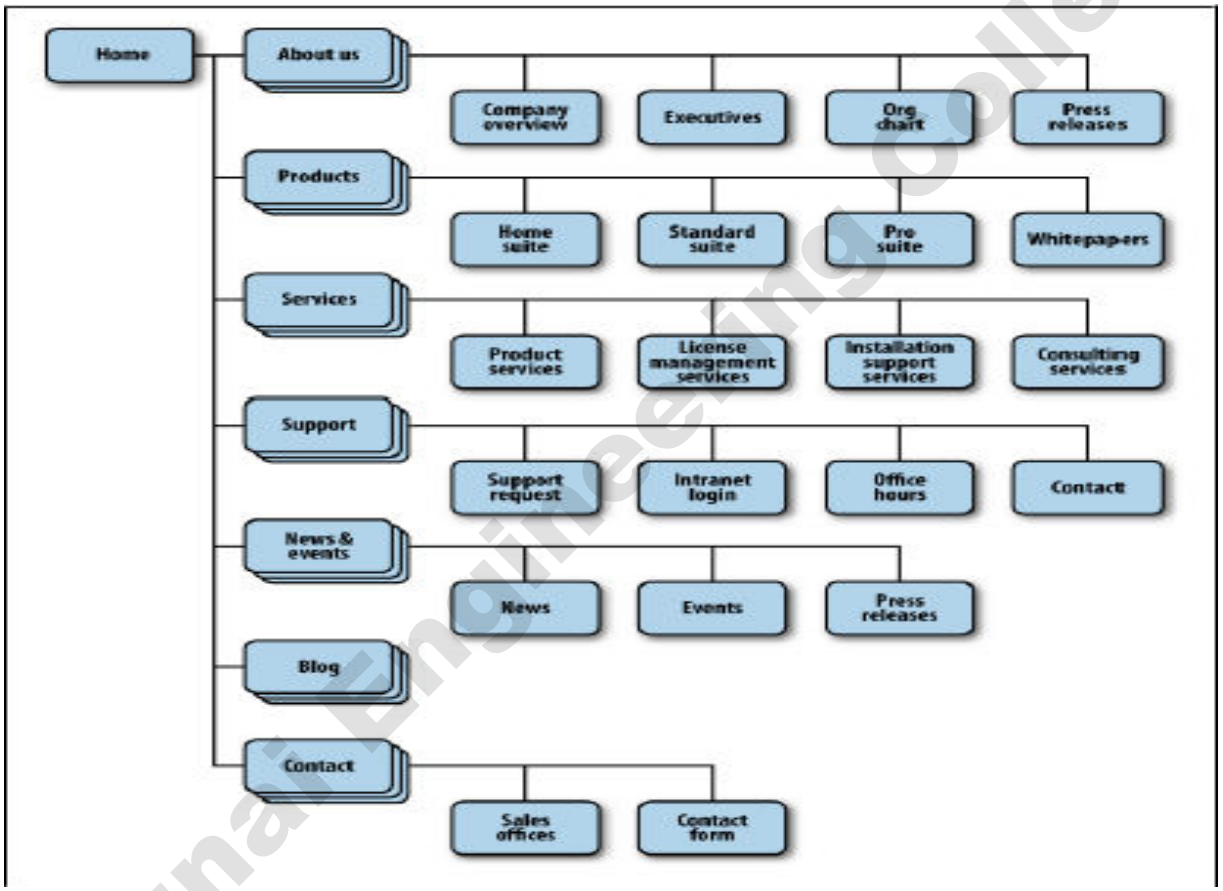


Figure 4.10: An example of a bad mobile information architecture that was designed with desktop users in mind rather than mobile users

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. To reduce risking the user's time and money, have to make sure that enough information is present for the user to wade through the information architecture successfully. On the Web, these risks are taken very lightly, but not with mobile. This is done by teasing content within each category that is, providing at least one content item per category.

The challenge with ringtone sites is there are a lot of items, grouped by artist, album, genre, and so on. The user starts with a goal like "I want a new ringtone" and finds an item that suits his taste within a catalog of tens of thousands of items.

In order to make sense of a vast inventory of content, we have to group, subgroup, and sometimes even +subgroup again, creating a drill-down path for the user to browse. Though on paper this might seem like a decent solution, once you populate an application with content, the dreaded "Page 1 of 157" appears. What user would ever sit there with a mobile device and page through 157 pages of ringtones? What user would page through five pages of content?

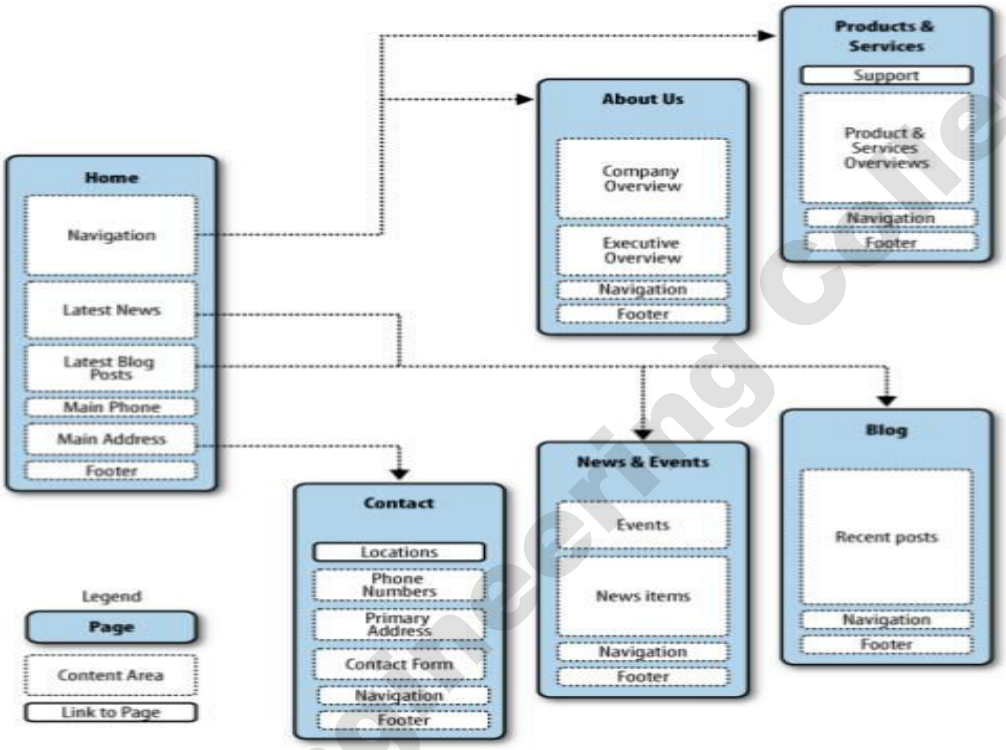


Figure 4.11. Teasing content to confirm the user's expectations of the content within

In figure4.11, we 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. We immediately saw that users were finding content more quickly, driving up our sales. It was like night and day. In Figure 4.11, 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. I've always found them to be a useful tool for rear-chitecting large websites.

However, information architecture in mobile is more like software than it is the Web, meaning that createing 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 a holistic or bird's-eye view of your mobile information architecture, just as a road map does. When all the paths are seen next to each other and take a step back, start to see shortcuts and how can get users to their goal faster or easier, as shown in figure 4.12.

Now the business analyst says, "Just create user or process flows," such as the esoteric diagram shown in figure 4.13, which is made up of boxes and diamonds that look more like circuit board diagrams than an information architecture. A good architect's job is to create a map of user goals, not map out every technical contingency or edge case.

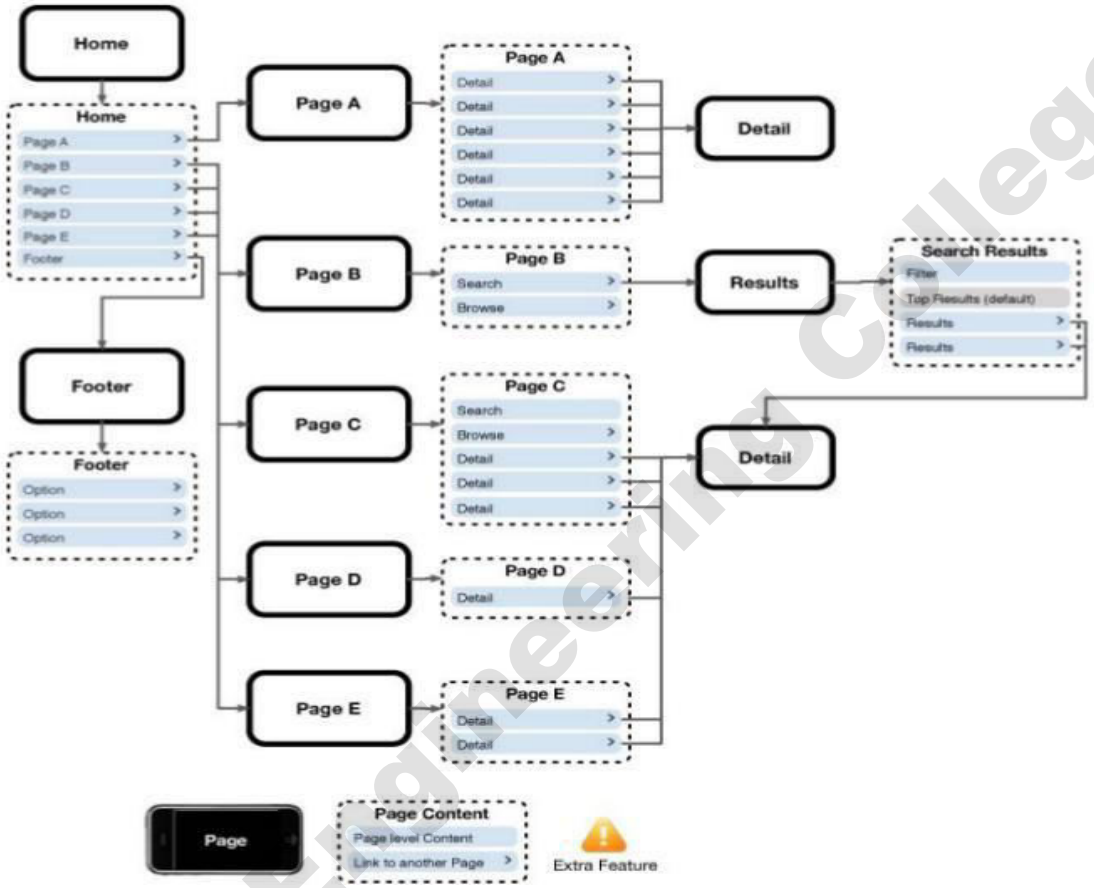


Figure 4.12. An example clickstream for an iPhone web application

Too often, process flows go down a slippery slope of adding every project requirement, bogging down the user experience with unnecessary distractions, rather than focusing on streamlining the experience. Remember, in mobile, it is to keep it as simple as possible. We need to have an unwavering focus on defining an excellent user experience first and foremost. Anything that distracts us from that goal is just a distraction.

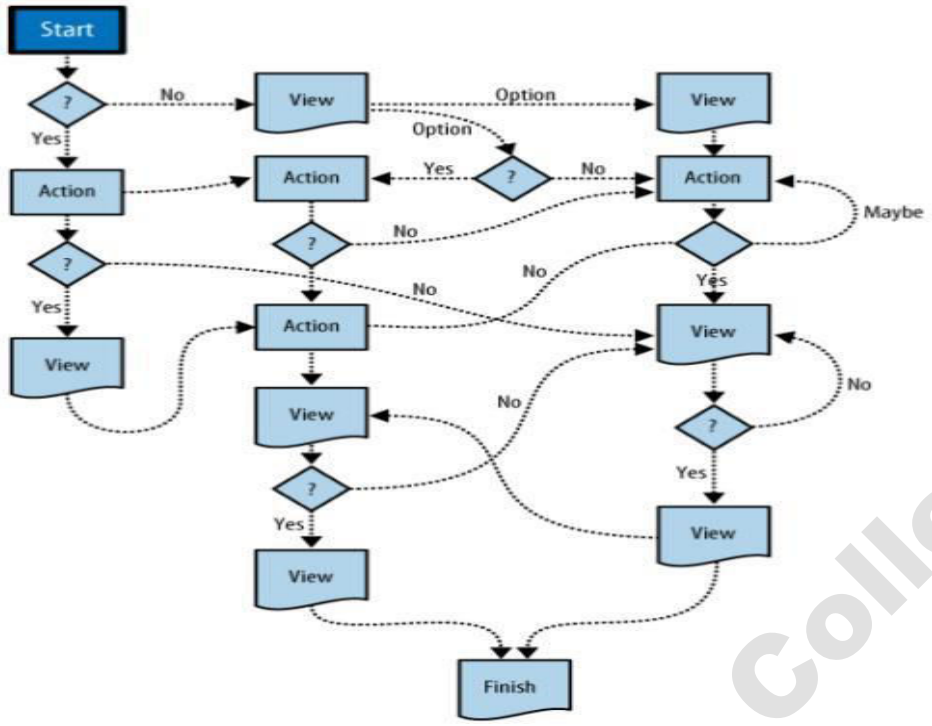


Figure 4.13. An example process flow diagram

Wireframes:

The next information architecture tool at disposal is wireframes. Wireframes are a way to lay out information on the page, also referred to as information design. Site maps show how the content is organized in our informational space; wireframes show howthe 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 figure 4.14 a. serve to make our information space tangible and useful.

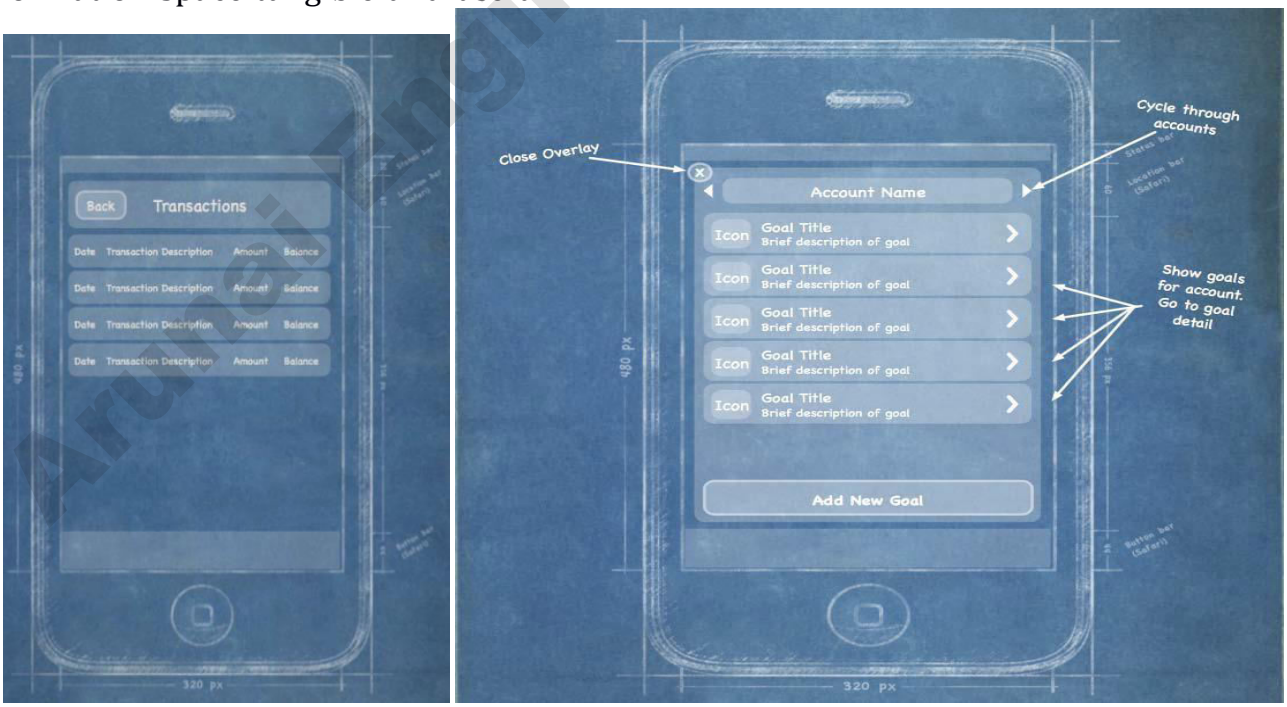


Figure 4.14.

Figure 4.14. a. An example of an iPhone web application wireframe, intended to be low fidelity to prevent confusion of visual design concepts with information

design concepts b. Using annotations to indicate the desired interactions of the site or application

But the purpose of wireframes is not just to provide a visual for our site map; they also serve to separate layout from visual design, defining how the user will interact with the experience. How do we lay out our navigation? What visual or interaction metaphors will we use to evoke action? These questions and many more are answered with wireframes.
Although wireframes is found to be one of the most valuable information deliverables to communicate the vision for how a site or app will work, the challenge is that a diagram on a piece of paper doesn't go a long way toward describing how the interactions will work. Most common are "in-place" interactions, or areas where the user can interact with an element without leaving the page. This can be done with Ajax or a little show/hide JavaScript. 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 figure 4.14 b. In mobile, using wireframes as the key deliverable, that turns good ideas into excellent mobile products.

Prototyping:
Prototypes might sound like a scary step in the process. Some view them as redundant or too time-consuming, preferring to jump in and start coding things. But as with wireframes, that each product built out some sort of prototype has saved both time and money. The following sections discuss some ways to do some simple and fast mobile prototyping.
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 4.15, and putting them in front of people. Create a basic script of tasks (hopefully based on either context or user input) and ask users to perform them, pointing to what they would do. You act as the device, changing the screens for them. The size matters and you'll learn as much from how the user manages working with small media as you will what information is actually on it.

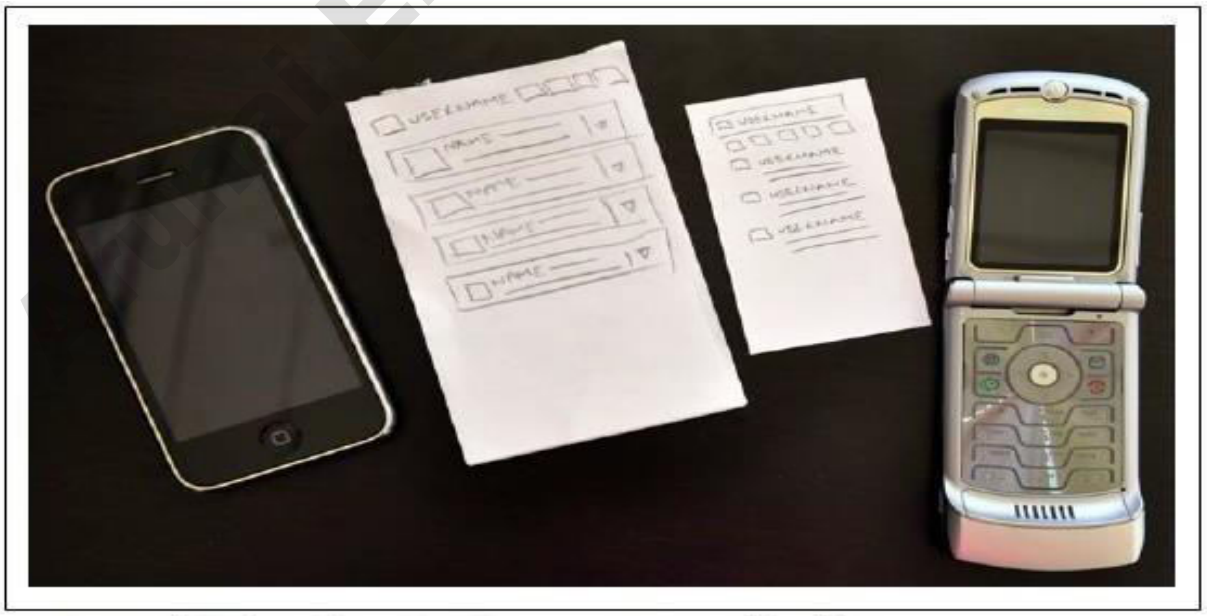


Figure 7-12. A touch interface paper prototype next to its smaller sibling

Context prototype:

The next step is creating a context prototype as in figure 4.16. 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. Leave the office. Go for a walk down to your nearest café. Or get on a bus or a train. As you are traveling about, pull out your device and start looking your interface in the various contexts you find yourself currently in. Pay particular attention to what you are thinking and your physical behaviour while you are using your interface and then write it down. If you are brave and don't have strict nondisclosure issues, ask the people around you to use it, too. I wouldn't bother with timing interactions or sessions, but try to keep an eye on a clock to determine how long the average session is.

Figure 4.15

Figure 4.16

Figure 4.17

HTML prototypes:

The third step is creating a lightweight, semi functional 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 as in figure 4.17. 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, you see how much content will really be displayed on screen (it is always less than you expect), and you have to deal with slow load times and network latency. In short, you will feel the same pains your user will go through. Whatever route you wish to take, building a mobile prototype takes you one very big leap forward to creating a better mobile experience.

Different Information Architecture for Different Devices:

Depending on which devices you need to support, mobile wireframes can range from the very basic to the complex. On the higher-end devices with larger screens, we might be inclined to add more interactions, buttons, and other clutter to the screen, but this would be a mistake. Just because the user might have a more advanced phone, that doesn't mean that it is a license to pack his screen with as much information.

The motivations, goals, and how users will interact with a mobile experience are the same at the low end as they are on a high-end device. For the latter, there are better tools to express the content. The greatest challenge in creating valuable experiences is knowing when to lose what is not needed. There is no choice on lower-end devices—it must be simple. When designing for both, it is best to try and to keep your information architecture as close to each other as possible without sacrificing the user experience. They say that simple design is the

hardest design, and this principle certainly is true when designing information architecture for mobile devices.

MOBILE 2.0

The mobile community started to discuss the idea of "Mobile 2.0," borrowing from many of the same principles behind Web 2.0. Each of these principles serves to transform the Web into a more agile and user-centered medium for delivering information to the masses. Mobile development, under the bottlenecks of device fragmentation and operator control, is sorely in need of a little reinvention as well.

Following is a recap of the original seven principles of Web 2.0:

The Web as a platform: For the mobile context, this means "write once, deploy everywhere," moving away from the costly native applications deployed over multiple frameworks and networks.

Harnessing collective intelligence: This isn't something the mobile community has done much of, but projects like WURFL is exactly what mobile needs more of.

Data is the next Intel inside: It can include the data we seek,the data we create, and the data about or around our physical locations.

End of the software release cycle: Long development and testing cycles heavily weigh on mobile projects, decreasing all hopes of profitability. Shorter agile cycles are needed to make mobile development work as a business. Releasing for one device, iterating, improving, and then releasing for another is a great way to ensure profitability in mobile.

Lightweight programming models: Because mobile technology is practically built on enterprise Java, the notion of using lightweight models is often viewed with some skepticism. But decreasing the programming overhead required means more innovation occurs faster.

Software above the level of a single device: This effectively means that software isn't just about computers anymore. We need to approach new software as though the user will demand it work in multiple contexts, from mobile phones to portable gaming consoles and e-book readers.

Rich user experiences: A great and rich user experience helps people spend less time with the software and more time living their lives. Mobile design is about enabling users to live their lives better.

Although the mobile industry has been through many more evolutions than just two, the concepts behind Web 2.0 are some of the most important ideas in not just mobile technology, but the Web as a whole.

Mobile 2.0: The Convergence of the Web and Mobile:

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. Mobile 2.0 tells us that mobile will be the primary context in which we leverage the Web in the future.

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. They see the success of small

iterative development cycles and want to apply this to mobile development, something that is not that feasible in the traditional mobile ecosystem.
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.
- 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:
If you are going to provide mobile web applications, you have to have a mobile web browser that supports Ajax, or, as it is technically known, XMLHttpRequest. 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. 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 your normal power consumption. There are number of reasons for why this happens from mobile hardware guys much smarter than myself, but to summarize, the two most prevalent are:
• 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.
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.
The Mobile User Experience Is Awful:
Device API's usually force to use their models of user experience, meaning that have to work in the constraints of the API. This is good for creating consistent experiences for that platform, but these experiences don't translate to others. For example, would you take an iPhone app design and put it on an Android device? The user experience for these devices is similar but still remains different. Modern mobile web browsers, as they come closer to their desktop counterparts,

remove this distinction, giving us the same canvas on mobile devices that we have for the desktop. It means we can have a consistent user experience across multiple mediums.

Mobile Widgets Are the Next Big Thing:

At many Mobile 2.0 events, there are 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. The concept of small network-enabled applications is very promising, but the mobile industry tends to take promising ideas like this, inflate expectations to unsustainable levels, then abandon them at the first sign of trouble or sacrifice them for the next big thing, whichever happens first.

Carrier Is the New "C" Word:

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.

Mobile Needs to Check Its Ego:

On the mobile side, there are some incredibly intelligent people who have been innovating amazing products under insane constraints for years. On the web side, there are creative amateurs who have helped build a community and ecosystem out of passion and an openness to share information.

The web guys want to get into the game and move the medium forward, partly out of desire open up a new market for themselves, but mostly out of passion for all things interactive. But, to the mobile community, they are seen as a threat to expertise. On the other hand, to the web community, the mobile guys come off as overly protective, territorial, selfish, and often snobbish or egotistical, effectively saying, "Go away." They have to deal with really hard problems that would make a web professional give up to go serve coffee.

Unless the mobile community comes together with the web community by sharing information, experience, and guidance, one day they will find that their experience has become obsolete. In return, the web guys will share their enthusiasm, willingness to learn, and passion that many in mobile development have forgot. It's that one principle of Web 2.0 that the mobile community has left out: harnessing collective intelligence. The Web and the mobile community are reaching a point where the two worlds can no longer afford not to be working together, sharing what they know and harnessing the collective intelligence of both media.

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. But before we get into that, let's back up for a minute. The web is about content. Sure, there are programming languages, APIs, and other technical underpinnings, but what do you do when you open a web browser? You read.

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. Now you can share any moment with any group of people in real time. Think about how powerful a concept that is! It could change entire cultures. 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. In the early days of the Web, I marveled at how a networked population might change our society forever. Now I realize that the change occurs wherever the device is, the context it is within. The early "Web 1.0" days clearly changed how business is done, because businesses are the primary consumer of desktop computers. 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. I see now that this is the time and medium that delivers on that initial promise of the Web: to change society forever.

MOBILE DESIGN

Interpreting Design:

Mobile design isn't that different. Precise designs might look better, but they can be brutal to implement. More flexible designs might not be much to look at, but they work for the most users, or the lowest common denominator. But more than that, our back-grounds and our training can actually get in the way of creating the best design for the medium. We like to apply the same rules to whatever the problem in front of us might be. In mobile design, you interpret what you know about good design and translate it to this new medium that is both technologically precise and at times incredibly unforgiving, and you provide the design with the flexibility to present information the way you envision on a number of different devices.

The Mobile Design Tent-Pole:

To have a successful mobile design, we have to adapt to today's changing audiences and niches. Find that emotional connection, that fundamental need that serves many audiences, many cultures, and many niches and design experiences. Too often, designers simply echo the visual trends of the day, mimicking the inspiration of others. But with mobile design, once you find that essential thing, that chewy nougat we call "context" that lives at the center of your product, then you will find ample inspiration of your own to start creating designs that translate not only across multiple devices, but across multiple media.

Sure, there are countless examples of poorly designed mobile products that are considered a success. You only need to look as far as the nearest mobile app store to find them. This is because of the sight unseen nature of mobile

commerce. Traditionally, you couldn't demo—or in some cases even see—screenshots of a game or mobile application before you bought it. You simply had to purchase it and find out if it was any good.

Apple's App Store quickly changed that. We can clearly see that the best-selling games and applications for the iPhone are the ones with the best designs (Figure 4.18).Users look at multiple screenshots, 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 most.

Figure 4.18. The app icon design greatly influences the user's expectation of quality

Designing for the Best Possible Experience:

When the first iPhone came out, there was a lot of trouble from web and mobile peers for publicly saying, "The iPhone is the only mobile device that matters right now." They would argue, "What about ABC or XYZ platforms?". The response was that those are important, but the iPhone provides the best possible experience and that is where consumers will go. Since those days, the iPhone shatter just about every record in mobile devices, becoming one of the best-selling phones ever and one of the most used mobile browsers in the world—two-thirds of mobile browsing in the U.S. comes from an iPhone or an iPod touch, not to mention that more than a billion mobile applications have been sold for these devices in under a year.

Although it may defy the business instincts to focus the product on just one device, in mobile development, the risks and costs of creating that tent-pole product are just too high. This is so easily seen through bad or just plain uninspired mobile design. Asking creative people to create uninspiring work is a fast track to mediocrity.

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. Do not get hindered by the constraints of the technology. Phrases like "lowest common denominator" cannot be part of the designer's vocabulary. Your design—no, your work of art—should serve as the shining example of what the experience should be, not what it can be. Trying to create a mobile design in the context of the device constraints isn't where you start; it is where you should end.

The greatest mistakes we in the mobile community make is being unwilling to or feeling incapable of thinking forward. The tendency to frame solutions in the past (past devices, past standards) applies only to those low-quality, something-for-everyone-but-getting-nothing tent-pole products. Great designs are not unlike great leaps forward in innovation. They come from shedding the baggage regarding how things are done and focus on giving people what they want or what they need.

4.5.1 ELEMENTS OF MOBILE DESIGN

Good mobile design requires three abilities: the first is a natural gift for being able to see visually how something should look that produces a desired emotion with the target audience. The second is the ability to manifest that vision into something for others to see, use, or participate in. The third is knowing how to utilize the medium to achieve your design goals.

Context:

As the designer, it is the job to make sure that the user can figure out how to address context using your app. Make sure 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?

The answers to these questions will greatly affect the course of your design. Treat these questions as a checklist to your design from start to finish. They can provide not only great inspiration for design challenges, but justification for your design decisions later.

Message:

Another design element is your message, or what you are trying to say about your site or application visually. One might also call it the "branding," although I see branding and messaging as two different things. The message is the overall mental impression create explicitly through visual design. If we take a step back, and look at a design from a distance, what is our impression? Or conversely, look at a design for 30 seconds, and then put it down. What words would we use to describe the experience?

Branding shouldn't be confused with messaging. Branding is the impression your company name and logo gives—essentially, your reputation. Branding serves to reinforce the message with authority, not deliver it. In mobile, the opportunities for branding are limited, but the need for messaging is great. With such limited real estate, the users don't care about the brand, but they will care about the messaging, asking themselves questions like, "What can this do for me?" or "Why is this important to me?"

The approach to the design will define that message and create expectations. A sparse, minimalist design with lots of whitespace will tell the user to expect a focus on content. A "heavy" design with use of dark colors and lots of graphics will tell the user to expect something more immersive. For example, hold the book away from you and look at each of the designs in Figure 4.19; try not to focus too heavily on the content. What do each of these designs "say" to you?

Which of the following designs provide a message? What do they say to you?

Figure 4.19. What is the message for each of these designs?

What is the message for each of these designs?

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. But I'm not exactly sure what it is saying. 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 you 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, you know 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, you can see 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.

Look and Feel:

The concept of "look and feel" is an odd one, being subjective and hard to define. Typically, look and feel is used to describe appearance, as in "I want a clean look and feel" or "I want a usable look and feel." The problem is: as a mobile designer, what does it mean? And how is that different than messaging?

Look and feel in a literal sense, as something real and tactile that the users can "look" at, then "feel"—something they can touch or interact with. Look and feel is used to evoke action—how the user will use an interface. Messaging is holistic, as the expectation the users will have about how you will address their context. It is easy to confuse the two, because "feel" can be interpreted to mean our emotional reaction to design and the role of messaging.

Figure 4.20: Pattern Tap shows a number of user interface patterns that help to establish look and feel

On large mobile projects or in companies with multiple designers, a style guide or pattern library is crucial, maintaining consistency in the look and feel and reducing the need for each design decision to be justified. For example, in Figure 4.20 you can see the site Pattern Tap, which is a visual collection of many user interface patterns meant for websites and web applications, but there is no reason why it can't serve as inspiration for your mobile projects as well. Although a lot of elements go into making Apple's App Store successful, the most important design element is how it looks and feels. Apple includes a robust user interface tool that enables developers to use prebuilt components, supported with detailed Human Interface Guidelines (or HIG) of how to use them, similar to a pattern library. This means that a developer can just sit down and create an iPhone application that looks like it came from Apple in a matter of minutes. During the App Store submission process, Apple then ensures that the developer uses these

tools correctly according to the HIG. The look and feel can either be consistent with the stock user interface elements that Apple provides; they can be customized, often retaining the "spirit" of Apple's original design; or an entirely new look and feel can be defined—this approach is often used for immersive experiences. The stock user experience that Apple provides is a great example of how look and feel works to supporting messaging.

For the end user, the design sends a clear message: by using the same visual interface metaphors that Apple uses throughout the iPhone, I can expect the action, or how this control will behave, but I can also expect the same level of quality. This invokes the message of trust and quality in the application and in the platform as a whole. Apple isn't the first to use this shared look and feel model in mobile—in fact, it is incredibly common with most smartphone platforms—but they are surely making it incredibly successful, with a massive catalog of apps and the sales to support it.
The mobile designers must be creative and remember the context. Like in the early days of the Web, people tend to be skeptical about mobile experiences. The modal context of the user—in this case, what device he is using—should be considered during the design, as it will help to establish the user's expectations of the experience.

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 your design more difficult to produce. The first time layout should rear its head is during information architecture. In fact, about 90 percent of layout decisions were during the information architecture period. We have to ask ourself questions like: where should the navigation go on the page or screen? What kind of navigation type should I use? Should I use tabs or a list? What about a sidebar for larger screens? All of these should be answered when defining the information architecture and before you begin to design. Why define the layout before the mobile design? Design is just too subjective of an issue. If you are 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. There is no right answer—only opinions and gut instincts. Plus, in corporate environments you have internal politics you have to consider, where the design opinions of the CEO or Chief Marketing Officer (CMO) might influence a design direction more than, say, the Creative Director or Design Director.
By defining design elements like layout prior to actually applying the look and feel, we can separate the discussion. As a self-taught designer, we started out in this business making designs for my own projects. I could just put pen to paper and tweak it to my heart's content. If I wanted to radically change the layout, I could. When I started my mobile design career with my first mobile company more than a decade ago, I realized that this approach didn't work. The majority of comments that reviewers would make were about the layout. They focused on the headers, the navigation, the footer, or how content blocks are laid out, and so on. But their feedback got muddied with the "look and feel, the colors, and other design elements." Reviewers do make remarks like "I like the navigation list, but can you make it look more raised?" Most designers don't hear that; they hear "The navigation isn't right, do it again." But, with this kind of feedback, there are

two important pieces of information about different types of design. First, there is confirmation that the navigation and layout are correct. Second, there is a question about the "look and feel." Because designers hear "Do it again," they typically redo the layout, even though it was actually fine.

Creating mobile designs in an environment with multiple reviewers is all about getting the right feedback at the right time. Your job is to create a manifestation of a shared vision. Layout is one of the elements you can present early on and discuss in-dependently. People confuse the quality and fidelity of your deliverables as design. By keeping it basic, you don't risk having reviewers confuse professionalism with design.

The irony is that as I become more adept at defining layouts, I make them of increasingly lower fidelity. For example, when I show my mobile design layouts as wireframes during the information architecture phase, I intentionally present them on blueprint paper, using handwriting fonts for my annotations (Figure 4.21). It also helps to say that this is not a design, it is a layout, so please give me feedback on the layout.

Figure 4.21. Using a low-fidelity wireframe to define the layout design element before visual design begins

Different layouts for different devices: The second part of layout design is how to visually represent content. In mobile design, the primary content element you deal with the is navigation. Whether you are designing a site or app, you 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 you support. There are two distinct types of navigation layouts for mobile devices: touch and scroll. With touch, you literally point to where you want to go; 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 Figure 4.22.

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 4.23, you can actually see by the bold outline that the first item selected on the screen is the link around the logo.

Figure 4.22. iPhone HIG, showing the layout dimensions of Safari on the iPhone.

Figure 4.23. Example layout of a scroll-based application, where the user had to press the D-pad past each link to scroll the page

When dealing with scroll navigation, you also have to make the choice of whether to display navigation horizontally or vertically. Visually, horizontally makes a bit more sense, but when you consider 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. 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. Different devices have different color depths.

Color characteristics

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, evil, death (in Western cultures), 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

Color	Represents
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

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.

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. I've found that I use three basic ways to define a color palette:

i)Sequential: In this case, 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 that I select using a color wheel.

ii) 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, I use an adaptive palette to make sure that my colors are consistent with the target mobile platform.

iii) Inspired: This is a design that is created from the great pieces of design you 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 size.

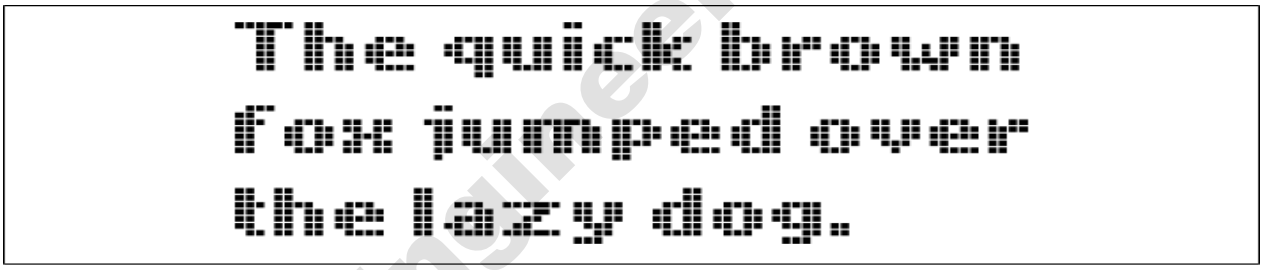


Figure 8-12. What most mobile designers think of when it comes to mobile typography

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 4.24, 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.

The quick brown fox jumps over the lazy dog.
The quick brown fox jumps over the lazy dog.
The quick brown fox jumps over the lazy dog.

Figure 4.24. Different ways text can render on mobile screens

Figure 8-14. Microsoft ClearType using subpixels to display sharp text

The Microsoft Windows Mobile platform uses the subpixel technique with its Clear-Type technology.

Table 8-3. Dimensions and PPI for some mobile devices

Mobile device	Diagonal	Pixels	PPI
Nokia N95	2.6"	240×320	153
Apple iPhone 3G	3.5"	320×480	163
Amazon Kindle	6.0"	600×800	167
HTC Dream	3.2"	320×480	181
Sony Ericsson W880i	1.8"	240×320	222
Nokia N80	2.1"	352×416	256

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 1440×900 pixels"). 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. As this applies to mobile devices, the higher the density of pixels, the sharper the screen appears to the naked eye. This guideline especially applies to type, meaning that as text is antialiased on a screen with a high density of tiny pixels, the glyph appears sharper to the eye. Some mobile

screens have both a high PPI and subpixel technology, though these are unnecessary together.

Fortunately, today's mobile devices have a few more options than a single typeface, but the options are still fairly limited. Coming from web design, where we have a dozen or so type options, the limited choices available in mobile design won't come as a big surprise. Essentially, you have a few variations of serif, sans-serif, and monospace fonts, and depending on the platform, maybe a few custom fonts.

In researching this book, I scoured the Web and tapped my mobile community resources to find a list of the typefaces that are included in each of the major device platforms, but I could only come up with a few—nothing close to a complete list. This goes to show how far behind mobile typography is, that designers don't even have a basic list to work from.

Therefore, when creating mobile designs for either web or native experiences, my advice is to stick with either the default device font, or web-safe fonts—your basic serif variants like Times New Roman and Georgia or sans-serif typefaces like Helvetica, Arial, or Verdana.

Font replacement: The ability to use typefaces that are not already loaded on the device varies from model to model and your chosen platform. Some device APIs will allow you to load a typeface into your native application. Some mobile web browsers support various forms of font replacement; the two most common are sIFR and Cufon. sIFR uses Flash to replace HTML text with a Flash representation of the text, but the device of course has to support Flash. Cufon uses JavaScript and the canvas element draws the glyphs in the browser, but the device of course needs to support both JavaScript and the canvas element.

In addition, the @font-face CSS rule allows for a typeface file to be referenced and loaded into the browser, but a license for web use is usually not granted by type foundries.

Readability: The most important role of typography in mobile design is to provide the user with excellent readability, or the ability to clearly follow lines of text with the eye and not lose one's place or become disoriented. This can be done by following these six simple rules:

Figure 8-16. Classics, an iPhone application designed with readability and typography in mind

Use a high-contrast typeface: Remember that mobile devices are usually used outside. Having a high-contrast typeface with regard to the background will increase visibility and readability.

Use the right typeface: The type of typeface you use tells the user what to expect. For example, a sans-serif font is common in navigation or compact areas, whereas serif typefaces come in handy for lengthy or dense content areas.

Provide decent leading (rhymes with "heading") or line spacing Mobile screens are often held 10–12" away from the eye, which can make tracking each line difficult. Increase the leading to avoid having the users lose their place. Leave space on the right and left of each line; don't crowd the screen. Most mobile frameworks give you full access to the screen, meaning that you normally need to provide some spacing between the right and left side of the screen's edge and your text—not much, typically about three to four character widths.

Generously utilize headings: Break the content up in the screen, using text-based headings to indicate to the user what is to come. Using different typefaces, color, and emphasis in headings can also help create a readable page.

Use short paragraphs: Like on the Web, keep paragraphs short, using no more than two to three sentences per paragraph.

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. 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. We can have some helpful icons that clearly communicate an idea and some perplexing icons that leave you scratching your head.

***Glyphish** provides free iPhone icons*

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 4.25, you can see how the pig graphic is designed to be positioned to the right regardless of the device orientation.

Figure 4.25. Using graphics in multiple device orientations

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. 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.

In Table below, you can see each of the design tools and what interface toolkits are available for it.

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
Mobile web	Photoshop, HTML, CSS, and	W3C Mobile Web Best

	JavaScript	Practices
Mobile widgets	Photoshop, HTML, CSS, and JavaScript	Opera Widget SDK, Nokia Web Runtime
Mobile web apps	Photoshop, HTML, CSS, and JavaScript	iUI, jQTouch, W3C Mobile Web App Best Practices

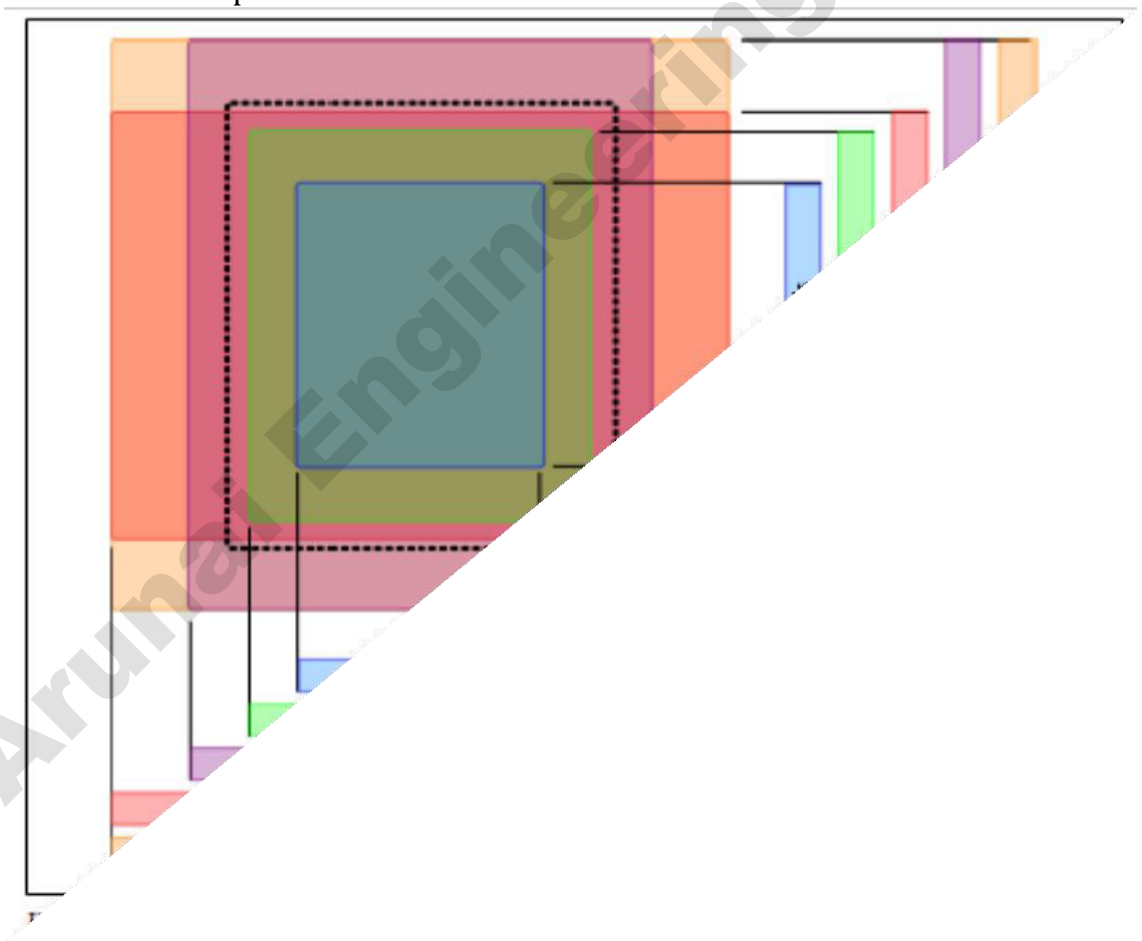
1. DESIGNING FOR THE RIGHT DEVICE

"What device suits this design best?
What market niche would appreciate it most?
What devices are the most popular within that niche?"

This knowledge will helps to develop porting and/or adaptation strategy, the most expensive and riskiest part of the mobile application.

2. 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 surplus of different screen sizes.



For example, your typical feature phone might only be 140 pixels wide, whereas your higher-end smartphone might be three to four times wider Landscape or portrait? Fixed width or fluid? Do you use one column or two?. 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 18.20
With vertical. designs, the goal is to think of your design as a cascade of content from top to bottom (Figure 18.21), similar to a newspaper.

- 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.

CASE STUDY 1:

For a pioneering distributor of petroleum products and related services develop an official app which, leveraging on location services, helps find the nearby filling station to the user. User can click a station on the map and navigate to it.

In addition, the mobile app must give information about the products, services and facilities available at their filling stations. The upcoming events & ongoing promotions should also be available on the app to view and participate. Customers should be able to provide their feedback to help improve the products and services.

Discuss about the challenge and design to develop a mobile App.

Challenge

- The client app, upon registration, stores users' driver's license.
- To notify a user when his driver's license is about to expire.
- The challenge augmented is that the license expiry date tends to differ for user to user.
- The app upon registration needs to store vehicle and mileage data.
- However, with so much data on the fly, it is evident to think about storing the data.
- Filtering results to a region, location and sub-filtering them to services/features/facilities.

Design

- The colour palette was chosen such that it includes various shades of grey with light colours here and there.
- To make sure the users receive a notification when their driver's license is about to expire, Android Local Push Notifications—Alarm Manager to set the expiry notification was employed.
- To store the vehicle and mileage data, created the local SQLite DB and to manage complex filter, SOL query with app code was used.

- **Nearby stations**
- An app user can not only find a filling station near him, he can also navigate to its location and take a look at the services, products and facilities available at the station.

- **Mileage Calculator**
- The app calculates mileage of a vehicle based on the number of miles it travelled. It calculates number of miles it travelled and how much fuel it was refilled with an The app **filling station**.

- **License Expiry Notifications**
- The app notifies a user when the license is about to expire.

- **Rating and Feedback**
- Customers' feedback matters more than anything be it a grocery store or a filling station. The app has many station and each time an app user visits one, the app prompts him to rate the experience.

Results

- The app reduced time and fuel people waste while searching for a fuel station. The app users became safer and more considerate drivers. Driver's License expiry notification assured they renew their licenses on time.

CASE STUDY 2:

The client wants a mobile application to acts as a platform for its users to enroll themselves and their friends in the training center. The Training centerhas a total of six branches that provides a variety of courses and training to their students.

The application is bilingual and is available in English and Hindi. This application has many things to offer to its users. It gives a comprehensive information about all the courses whether they are ongoing or upcoming. Any user can avail these courses by making the payment through the app itself.

Discuss about the challenges and solution for developing the Mobile APP.

Challenges

The client is a reputed training institute. It's well-known for the variety of courses and quality of teaching. However, they were facing problems in reaching out to more students. They wanted to develop a mobile application that can work like a platform to enroll in their institute and also to guide their students.

Solution :

- To overcome the challenges of the client an app has to be developed to fulfil all the clients' requirements.
- With this application, all their users can see the number of different courses, its eligibility, timings, instructor, and many more.
- The app also provides all the guidance to the students related to the institute.
- **Explore courses with various filters**
- The institute has many courses and that's why this feature comes in handy as it allows its users to search for a course by using different filters such as category, age, location, gender, start date, and many more.
- **Bilingual**
- The client's institute is a reputed one and they wanted to attract students of all demographics. That's why we need to develop an app in both English and Hindi language.
- **Online enrollment& payment**
- This feature allows the user to enroll for various courses of their choices. Along with that they can also make payments online via the mobile app.
- **Gallery**
- Users can always have a look at the gallery that features pictures from all six branches of the training institute.
- **Notify me**
- This feature is a crucial one as it lets its user to set a reminder to any of the upcoming course. Whenever, the date of that course approaches near, the app reminds the user to enroll for it.
- **News**
- This feature shows all the news related to academics. It also shows the details about examination taken for various vacancies.

Result

- This app will turn out to be a huge success for clients. It will play a pivotal role in attracting many new students into the institute.

CASE STUDY 3:

With medical technologies taking huge leaps every year, most of our health-related woes are taken care of. Doctors are successfully treating some conditions which were once incurable. However, with the advancement, the number of tasks involving the medical procedure has also increased to an overwhelming level.

An app that helps to manage all those procedures and tasks in an easy and systematic manner need to be developed. This application brings together all the components of health report which includes the medical history, symptoms, medication, appointments, immunizations, allergies, and fitness records.

Discuss about the challenges and solution for developing the APP

Challenges

- First need to fetch the thehealthkit& Fitbit data on the application.
- Integration of both will be a challenge as it would increase the complexity of the app.
- Moreover, to fetch data in real-time from the healthkit& Fitbit to the application will be a tough job.
- Other challenge is about the medicine reminders. Client would need this feature in which the user would get notifications for his/her medical dose.
- Apart from that, need to provide encryption and decryption of data which would ensure foolproof privacy and security of user's data.
- Need to provide multiple languages in the app.

Solution:

- To solve the first challenge can useHealthkit's& Fitbits' API to fetch all the data to the app.
- For the second challenge which was of medicine reminders, can synchronize with the calendars. Can add reminders according to the frequency of the dosage.
- For encryption and decryption, can use AES encryption algorithm that works parallelly across every platform such as iOS, Android, and PHP.
- To make this app available in all the languages can create a master in which all the language translations for all the modules can be stored.
- **Record symptoms**
- In this feature the user can record all his/her symptoms with details such as date of first appearance, severity, and other miscellaneous information.
- **Immunizations**
- The user can record history of all the vaccines, any other immunizations taken in a single app which will give the user and the doctor a clear idea about your immunization history.
- **Chronic conditions**
- This feature allows the user to record all the necessary details of his/her chronic disease which would help the doctor to take further action rapidly.
- **Allergies**

- In this section the user can fill all the details of the allergies if he/she has any. This will prevent cases in which doctor prescribes medication to which the patient is allergic, causing harm.
- **Fitness data**
- This is a crucial feature in which all the data from fitbit or healthkit is fetched to the app. Now the user can monitor his/her fitness stats and manage medication simultaneously**.**
- **Medicine dosage**
- The user can enter all the details such as their name, amount of dosage to be taken, and the frequency of a dosage of all the medicines that the user have been prescribed with**.**
- **Medical appointments synced with calendar**
- This is a critical feature as the user can sync all the appointments with the calendar. User will receive a reminder whenever the user have a medical appointment fixed that day.
- **Voice commands**
- This feature is useful for elderly users who may face problems in typing. They can simply give voice commands to access functions and to write notes.
- **Share PDFs, video, and audio files with the doctor**
- This feature gives the user a golden opportunity to present the medical history with a great amount of details to the doctor. In this feature the user can include PDFs, videos, and audio files forthe doctor.
- **Create multiple dependent accounts**
- The user don't need to create separate accounts for each member of his family. By this feature , the user can create and manage multiples account of his family members on the same application

Result:

- The application will get a tremendous amount of appreciation from all; especially, from those who had to manage a large number of medical tasks for themselves or their family members