HCI Human Computer Interaction

ITec4122

Yadeta G.

Outlines

1. **Introduction**
2. **Human in HCI**
3. **Computer in HCI**
4. **Interaction**
5. **Interaction Design & HCI Software process**
6. **Design Rules and Implementation Support**
7. **Evaluation Techniques and Universal Design**
8. **User Support**

Evaluation Methods:

60% Continuous Assessments and 40% final Exam

What is this course about?

- How to make a good (**usable**) user interface for an interactive application
- Why are usable user interfaces important?
  - Nearly all applications have a user interface
  - **Bad interfaces are frustrating** for the user and will influence the productivity
  - Competitors may have better systems
- **Good** user **interface** are **hardly noticed**, bad ones are!
- It is easier to make a bad interface than a good one.

Introduction

What is Human Computer Interaction (HCI)? As defined by the Special Interest Group on Human-Computer Interaction (SIGCHI) of the Association for Computing Machinery (ACM)

"Human Computer Interaction is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of the major phenomena surrounding them."

(HCI) is a cross-disciplinary area (e.g., **engineering**, **psychology**, **ergonomics**, and **design**) that deals with the theory, design, implementation, and evaluation of the ways that humans use and interact with computing devices.

Introduction ...cont'd

- HCI is the study of how people interact with computers and to what extent computers are or are not developed for successful interaction with human beings

HCI concerns:

- **process**: design, evaluation and implementation
- **on**: interactive computing systems for human use
- **plus**: the study of major phenomena surrounding them (ergonometric).

Humans, Computers and Interaction

- The **H**umans good at: Sensing low level stimuli, pattern recognition, inductive reasoning, multiple strategies, adapting "Hard and fuzzy things".
- The **C**omputers good at: Counting and measuring, accurate storage and recall, rapid and consistent responses, data processing/calculation, repetitive actions, performance over time, "Simple and sharply defined things".
- The **I**nteraction focused on the list of skills is somewhat complementary. Let humans do what humans do best and computers do what computers do best.

A major shift

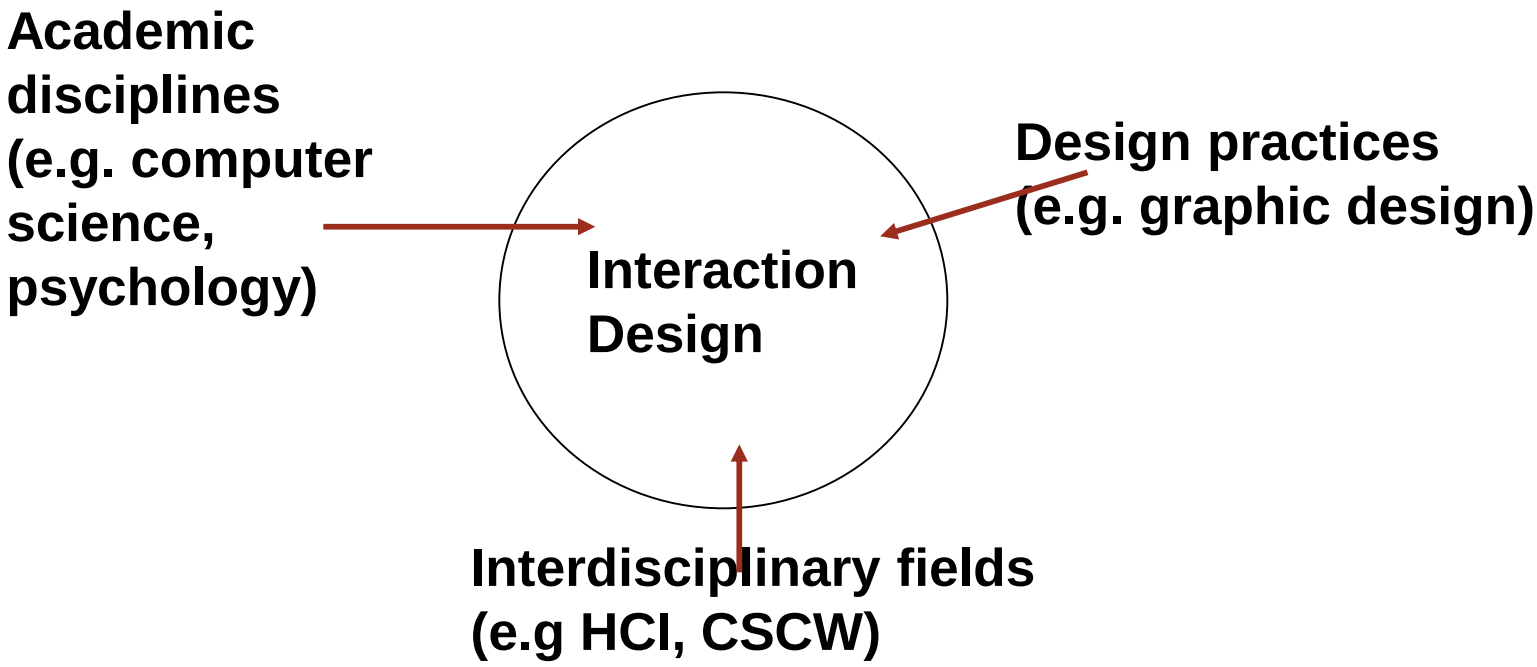
- 50s - Interface at the hardware level for engineers - switch panels
- 60-70s - interface at the programming level - COBOL, FORTRAN
- 70-90s - Interface at the terminal level - command languages
- 80s - Interface at the interaction dialogue level - GUIs, multimedia
- 90s - Interface at the work setting - networked systems, groupware

The current technologies were the product of previous HCI concept.

From HCI to Interaction Design

- **Human-computer interaction (HCI)** is: "concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them"
- **Interaction design (ID)** is: "the design of spaces for human communication and interaction"
- One distinction: more application areas, more technologies and more issues to consider when designing 'interfaces'

Relationship between ID, HCI and other fields



Goals of HCI

To develop or improve the

- Safety
- Utility
- Effectiveness
- Efficiency
- Usability
- Appeal (application)

. . . of systems that include computers

The goals of HCI

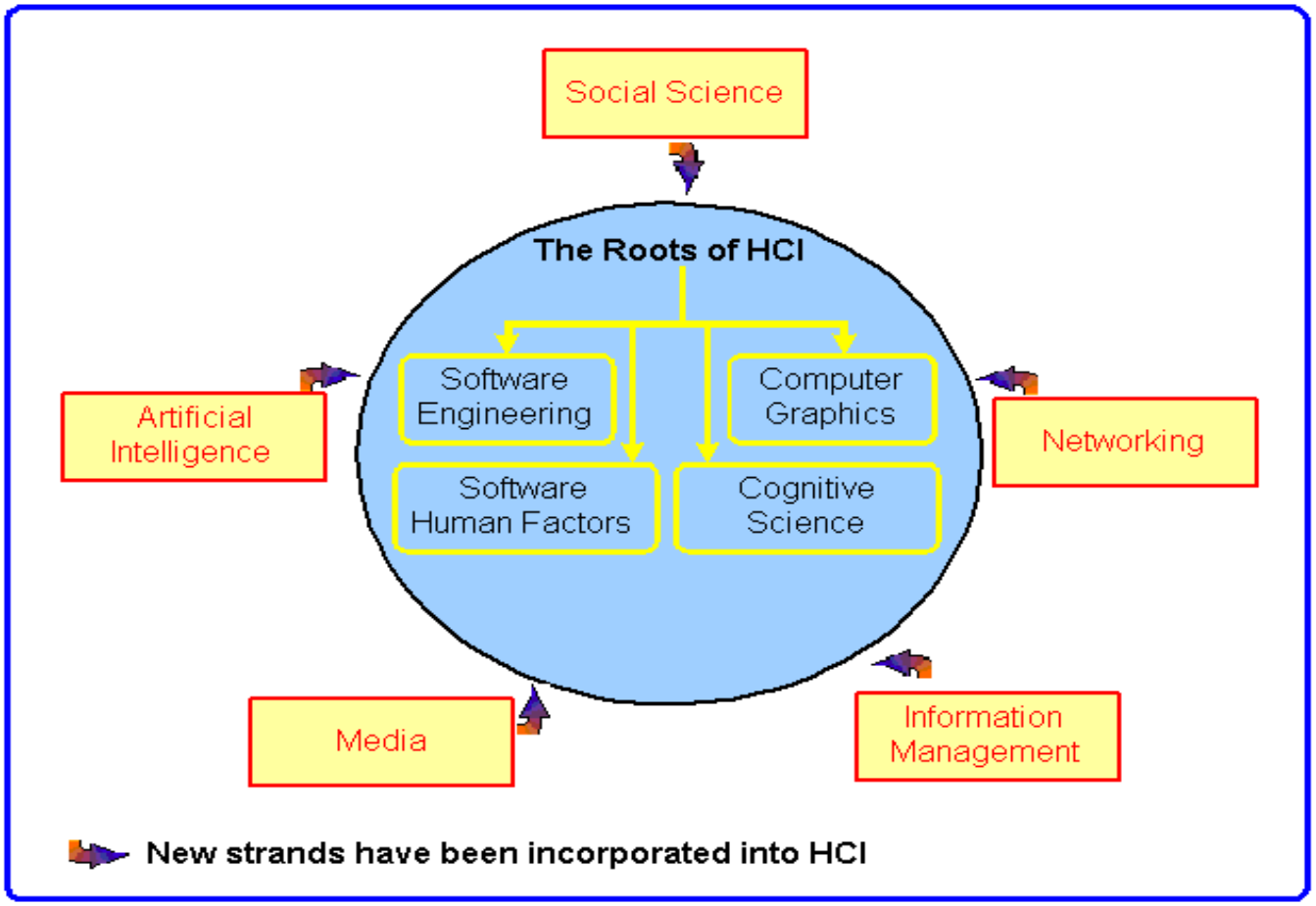
To achieve usability, the design of the user interface to any interactive product, needs to take into account and be tailored around a number of factors, including:

- Cognitive, perceptual, and motor capabilities and constraints of people in general
- Special and unique characteristics of the intended user population in particular
- Unique characteristics of the users' physical and social work environment
- Unique characteristics and requirements of the users' tasks, which are being supported by the software
- Unique capabilities and constraints of the chosen software and/or hardware and platform for the product

Safety

- Safety of Users—think of
  - Air traffic control
  - Hospital intensive care
- Safety of Data—think of
  - Protection of files from tampering
  - Privacy and security

Relationship of HCI wither other Disciplines



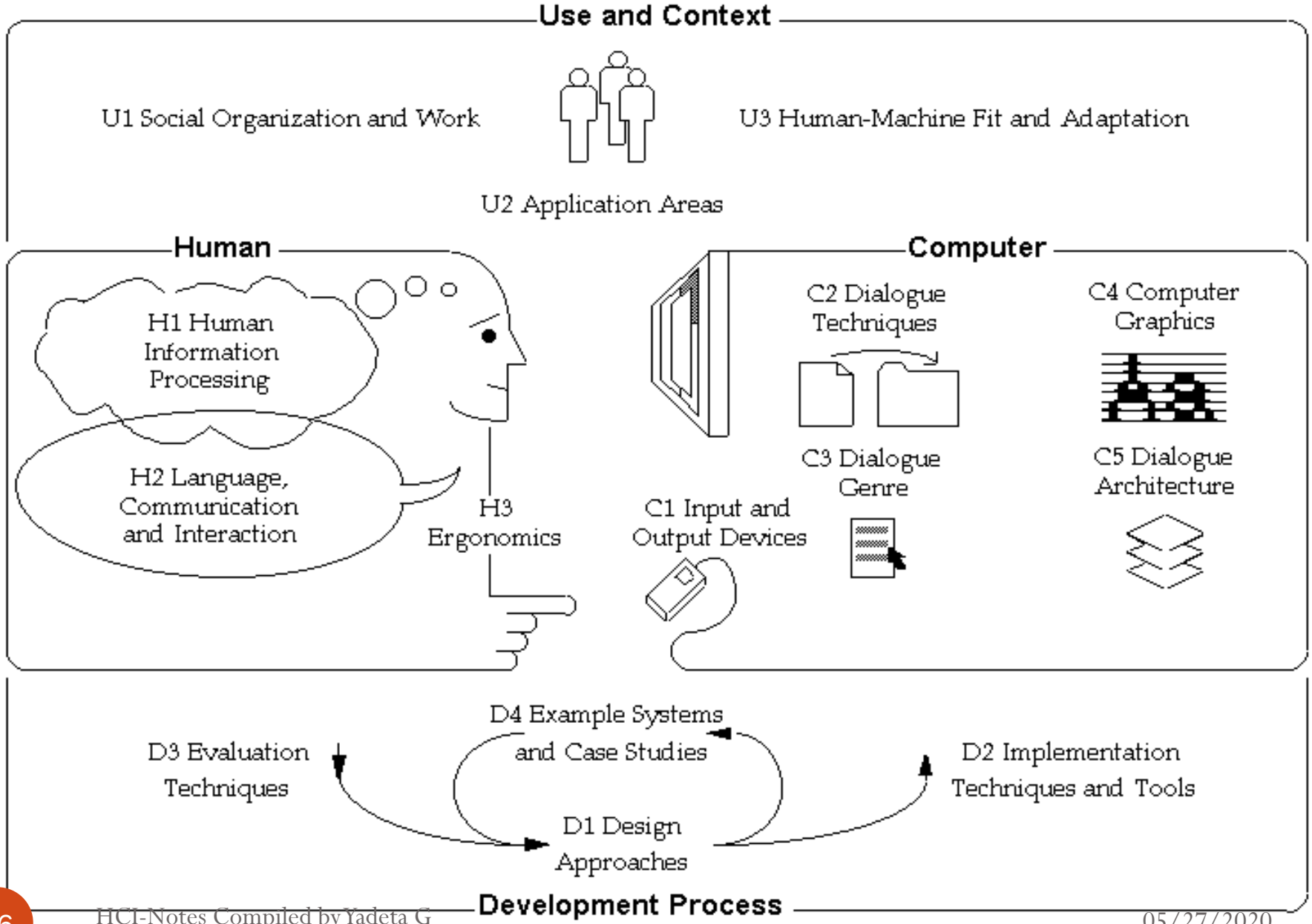
Fields that HCI builds on

- Computer Science
  - Implementation of software
- Engineering
  - Faster, cheaper equipment
- Ergonomics
  - Design for human factors
- Graphic design
  - Visual communication
- Technical writing
  - Textual communication

Fields that HCI builds on

- Linguistics, artificial intelligence
  - Speech recognition, natural language processing
- Cognitive psychology
  - Perception, memory, mental models
- Sociology
  - How people interact in groups
- Anthropology
  - Study of people in their work settings

Topics in HCI



Topics in HCI

Computer systems exist within a larger social, organizational and work milieu (U1).

Within this context there are applications for which we wish to employ computer systems (U2).

But the process of putting computers to work means that the human, technical, and work aspects of the application situation must be brought into fit with each other through human learning, system tailorability, or other strategies (U3).

Topics in HCI

In addition to the use and social context of computers, on the human side we must also take into account:

the human information processing (H1)

communication (H2)

and physical (H3) characteristics of users

Topics in HCI

On the computer side, a variety of technologies have been developed for supporting interaction with humans:

Input and output devices connect the human and the machine (C1).

These are used in a number of techniques for organizing a dialogue (C2).

These techniques are used in turn to implement larger design elements, such as the metaphor of the interface (C3).

Getting deeper into the machine substrata supporting the dialogue, the dialogue may make extensive use of computer graphics techniques (C4).

Topics in HCI

Complex dialogues lead into considerations of the systems architecture necessary to support such features as interconnect-able application programs, windowing, real-time response, network communications, multi-user and cooperative interfaces, and multi-tasking of dialogue objects (C5).

Finally, there is the process of development which incorporates design (D1) for human-computer dialogues, techniques and tools (D2) for implementing them (D2), techniques for evaluating (D3) them, and a number of classic designs for study (D4).

Human in HCI

Humans are limited in their capacity to process information. But it has important implications for design.

Information is received and responses given via a number of input and output channels.

These channels are:

- Visual channel
- Auditory channel
- Haptic channel
- Movement

Human Input-Output channels

- A person's interaction with the outside world occurs through information being received and sent (input and output).
- In interaction, the user **receive** the information displayed by the computer and sent information for machine.
- In other speaking output info for human is input for machine and input info for human is output for computer.

Information is stored in memory

The human memories are categorized as:

- Sensory memory
- Short-term memory
- Long-term memory

- It sentiments about feeling.
- Users share common capabilities but are individuals with differences, which should not be ignored.
- We should have to appreciate with the diversity

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.

Short-term memory

- Short-term memory acts as a 'scratch-pad' for temporary recall of information.
- It is used to store information which is only required briefly.
- For example, calculate the multiplication 35 × 6 in your head.
- Short-term memory also has a limited capacity

Long-term memory

- If short-term memory is our working memory or **'scratch-pad'**, long-term memory is our main resource.
- Here we store real information, experimental knowledge, and procedural rules of behavior in fact,
- 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.

Human in description HCI

- We start with the human, the **central character** in any discussion of interactive systems.
- The human (*user)*, is after all, the one whom computer systems are designed to assist.
- The requirements of the user should therefore be our first **priority**.
- In this chapter we will look at areas of human psychology coming under the general exceptional of ***cognitive psychology***.
- One way to structure this discussion is to think of the user in a way that highlights these aspects. In other words, to think of a simplified *model* of what is actually going on?

.... Cont'd

- There are five major senses: **sight**, **hearing**, **touch**, **taste** and **smell**.
- Of these, the first **three** are the most important to **HCI**.

The human input/output channels are:

- Visual channel for Seeing
- Auditory for Hearing
- Haptic or sense for touching
- Movement, and the like

Information Processing in Human

Information is processed and applied in human bay:

- Reasoning
- Problem Solving
- Skill acquisition
- Error understanding

Emotion can influences human capabilities

THINKING: REASONING AND PROBLEM SOLVING

- We have considered how information finds its way into and out of the human system and how it is stored.
- Finally, we come to look at how it is processed and manipulated.
- This is perhaps the area which is most complex and which separates humans from other **information-processing** systems, both artificial and natural.
- Human thought is conscious/ mindful 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 … cont'd

- In addition, we are able to think about things of which we have no experience, and solve problems which we have never seen before.
- Thinking can require different amounts of knowledge.
- Some thinking activities are directed and the knowledge required is constrained.
- Others require vast amounts of knowledge from different areas.

For instance: the knowledge we used for solving problem by calculation and understanding the newspaper headlines.

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. For example, it goes from general to specific idea.

  If it is Friday then she will go to work

  It is Friday

  Therefore she will go to work.

- It is important to note that this is the logical conclusion from the premises; it does not necessarily have to correspond to our notion of truth. So, for example,

  If it is raining then the ground is dry

  It is raining

  Therefore the ground is dry.

Inductive reasoning

- Induction is generalizing from cases we have seen to conclude information about cases we have not seen.
- For example, if every elephant we have ever seen has a trunk, we infer that all elephants have trunks.
- The best that we can do is gather evidence to support our inductive inference.

Abductive reasoning

- The type of reasoning whereby one seeks to explain relevant evidence by beginning with some commonly well known facts that are already accepted and then working towards an explanation.
- The third type of reasoning is abduction. Abduction reasons from a fact to the action or state that caused it.
- This is the method we use to derive explanations for the events we observe.
- For example: suppose we know that **Sami** always drives too fast when he has been drinking. If we see **Sami** driving too fast we may infer that he has been drinking.

Problem solving

- Reasoning is a means of concluding 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.

Three views of how human to solve problems

- **Gestalt view:** the earliest, dating back to the first half of the twentieth century is that problem solving involves both reuse of knowledge and understanding.
- Problem Space Theory: this takes the view that mind is limited information processor.
- It generating both problem states (initial state and goal state)
- Analogy in Problem Solving: this is use the analogy while solving the problem
- Implemented by mapping knowledge related to the similar domain (analog mapping) old knowledge is used to solve new problem

Skill acquisition

- The entire problem solving that we have considered so far has concentrated on handling unfamiliar problems.
- However, for much of the time, the problems that we face are not completely new.
- Instead, we gradually acquire skill in a particular domain area.
- We can gain insight into how skilled behavior works, and how skills are acquired, by considering the difference between novice and expert behavior in given domains.

Discuss: If we have no prior knowledge how we solve problem?

Error and mental model

- 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 frustration.
- 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 escape.
- In order to answer the latter part of the question we must first look at what is going on when we make an error.

Mental model

- People build their own theories to understand the causal behavior of systems. This is known as **mental model**.
- Mental model characteristics
  - Emotion
  - Individual difference
  - Psychology
  - Designing of Interactive systems

Computer in HCI

- A computer system contains 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:
- text entry: traditional keyboard, phone text entry, speech and handwriting pointing:
- principally the mouse, but also touchpad, stylus and others 3D interaction devices.

From hardware component of Computer

- **Input devices**
- **Output devices**
- **Storage devices**
- **Processing**

- Output display devices for interactive use:
- different types of screen mostly using some form of bitmap display
- large displays and situated displays for shared and public use
- Virtual reality systems and 3D visualization which 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 and Information Processing in Computer

- **Memory**
  - Short-term memory: RAM
  - Long-term memory: magnetic and optical disks
  - 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 (since all machine doesn't have same speed)

Networks and their impact on system performance.

INTERACTION

- Interaction is a kind of action which occurs when two or more object have an affect on each other
- Interaction models help us to understand what is going on in the interaction between user and system.
- They address the translations between what the user wants and what the system does.
- Simply, interaction design is a process of connecting the digital world to the human one.

Interaction Design

- **Interaction design** is a process in which designers focus on creating attractive web interfaces with logical and thought out behaviors and actions.
- Successful **interactive design** uses technology and principles of good communication to create desired user experiences.
- In design, human–computer interaction, software development, and interaction design, is "about shaping digital things for people's use"

MODELS OF INTERACTION

- So far, we have seen the usefulness of models to help us to understand complex behavior and complex systems.
- Interaction involves at least two participants: the user and the system.
- Both are complex, as we have seen, and are very different from each other in the way that they communicate and view the domain and the task.
- The 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.

Model of interaction

- We consider the most influential model of interaction, Norman's execution evaluation cycle; then we look at another model which extends the ideas of Norman's cycle.

The terms of interaction

- Traditionally, 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.
- Some examples of domains are graphic design, authoring and process control in a factory.

Model of interaction

The execution evaluation cycle

- Norman's model of interaction is perhaps the most influential in Human–Computer interaction.
- 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 plan, or part of plan, has been executed, the user observes the computer interface to evaluate the result of the executed plan, and to determine further actions.

Stages in Norman's model of interaction

- The interactive cycle can be divided into two major phases: **execution** and **evaluation**. These can then be subdivided into further stages, seven in all.

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

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.

Arrangement of controls and displays

The physical environment of the interaction

Health issues: Lighting the lighting level will again depend on the work environment. **Considering temperature**

Noise Excessive noise can be harmful to health, causing the user pain, and in acute cases, loss of hearing.

Ergonomics

- It looks at the physical characteristics of the interaction and how these influence its effectiveness.
- It deals with the relation workers with their work environments.
- The dialog between user and system is influenced by the style of the interface.
- The interaction takes place within a social and organizational context that affects both user and system.

INTERACTION STYLES

- We have said an Interaction is a dialog between the computer and the user.
- The choice of interface style can have a deep effect on the nature of this dialog.

From the number of interaction style following are the most common:

- Command line interface,
- Menus,
- Natural language
- Question/answer and query dialog,
- Form-fills
- WIMP (Windows, Icons, Menus, and Pointer)
- Point and click,
- Three-dimensional interfaces.

Interaction Design and HCI in the Software Process

Design process

- The design process is a series of steps that engineers or computer scientists follow to come up with a solution to solve a problem.
- Many times the solution involves designing a product (like a machine or computer code) that meets certain criteria and/or accomplishes a certain task.
- Interaction design is about creating involvements in often complex situations using technology of many kinds including PC software, the web and physical devices.

Design involves:

- Achieving goals within constraints
- Understanding the raw materials: computer and human
- Accepting limitations of humans and of design.
- The design process has several stages and is iterative and never complete.
- Interaction starts with getting to know the users and their context:
- Finding out who they are and what they are like . . . probably not like you!
- Talking to them, watching them.

Design is the outcomes of the following fields

- Computer Science (User interface Management Systems - Techniques)
- Cognitive psychology (Perception, attention, learning, thinking)
- Social and organizational psychology (attitude, behavior, group work)
- Ergonomics and human factors (safety, performance, adaptation)
- Linguistics (terminology, multilingual systems)
- Artificial intelligence (automated behavior)
- Philosophy (knowledge)
- Sociology (group behavior)
- Anthropology (ethnomethodology)
- Engineering (tools, techniques, equipment)
- Design (graphic design, layout, colors)

What are the components of design?

- This does not capture everything about design, but helps to focus us on certain things:
- **Goals** What is the purpose of the design we are intending to produce? Who is it for? Why do they want it?

For example, if we are designing a wireless personal movie player, we may think about young affluent users wanting to watch the latest movies whilst on the move and download free copies, and perhaps wanting to share the experience with a few friends.

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

In the case of the personal movie player: does it have to withstand rain? Must we use existing video standards to download movies? Do we need to build in copyright protection? Of course, we cannot always achieve all our goals within the constraints. So perhaps one of the most important things about design is:

..... components of design

Trade-off 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 whilst 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.

Scenarios

- **Scenarios** are rich design stories, which can be used and reused throughout design:

– they help us see what users will want to do

– they give a step-by-step walkthrough of users' interactions: including what they see, do and are thinking.

- Users need to find their way around a system. This involves:

– helping users know where they are, where they have been and what they can do next

– creating overall structures that are easy to understand and fit the users' needs

– designing comprehensible screens and control panels

Complexity of design means we don't get it right first time:

– so we need iteration and prototypes to try out and evaluate

– but iteration can get trapped in local maxima, designs that have no simple improvements, but are not good

– theory and models can help give good start points.

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 users
  - Who are they?
  - Probably not like you!
  - Talk to them.
  - Watch them.
  - .....

The golden rule of design

- Part of the understanding we need is about the **circumstances** and **context** of the particular design problem.
- We will return to this later in the chapter.
- However, there are also more generic concepts to understand.
- 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

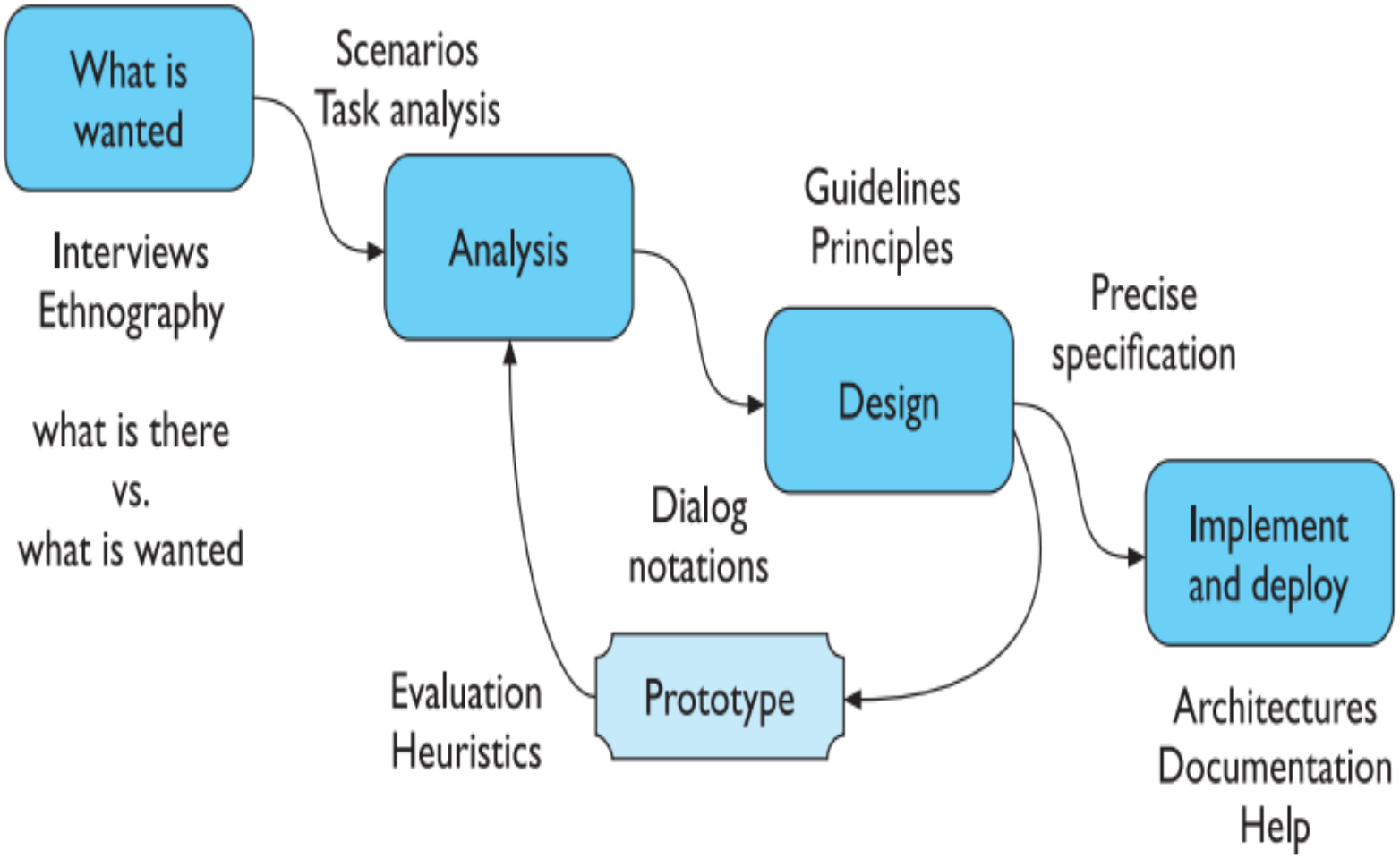
– psychological, social aspects, human error.

This is equally important in other design areas.

- For example, the way you fit seats and windows into an airplane's hull affects the safety and strength of the aircraft as a whole.

Process of designing and models

- Often HCI professionals complain that they are called in too late.
- 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.
- In the best companies, however, usability is designed in from the start.
- The next figure shows Interaction design process


What is wanted
Scenarios
Task analysis
Interviews
Ethnography
Analysis
Guidelines
Principles
Precise specification
Design
what is there
vs.
what is wanted
Dialog notations
Implement and deploy
Evaluation
Heuristics
Prototype
Architectures
Documentation
Help

From the figure

Requirements- deals with what is wanted.

- The first stage is establishing what exactly is needed.

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.

- **System annalists** Categorizes requirements, Organizes them into related sub-sets explores relationships between requirements

Design Well, this is all about design, but 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 with this.

From the figure

- **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.
- **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.

Navigation Design: design is the planning of:

- Artifacts and systems , interactions and interventions which achieve goals within constraints, select appropriate trade-offs, improve on existing solutions.
- Reusing the materials: Physical design requires understanding basic materials.

The "materials" for HCI are:

- **People**: human capabilities and psychology, human error, social context, cultural experience, . . .
- **Computers**: Processing capability of computer and limitations

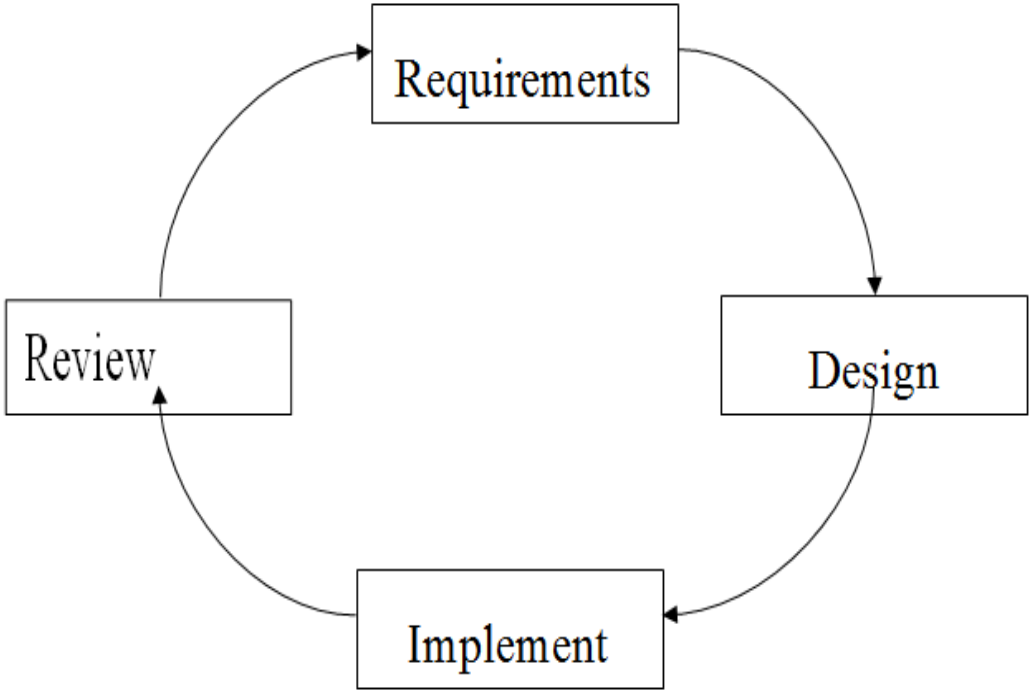
• Interaction Facilities such as WIMP, Audio, video etc…

• Platform conventions and design rules and

• Interface building blocks, toolkits

The prototype of interaction design process is:

- We said that the process of designing interaction is iterative (non terminate)



HCI in the Software Process

- In software development, designers use models that help them understand, organize and represent the system's architecture and functionality.
- In object-oriented development, the UML provides us with a large set of models to represent complementary aspects of the system.
- Use cases, class diagrams, sequence diagrams and activity diagrams are perhaps the most widely used UML models in the initial design stages.

- Software **engineering provides** a means of understanding the structure of the design process, and that process can be assessed for its effectiveness in interactive system design.
- **Usability** engineering promotes the use of explicit criteria to judge the success of a product in terms of its usability.
- **Iterative** design practices work to incorporate crucial customer feedback early in the design process to inform critical decisions which affect usability.
- **Design** involves making many decisions among numerous alternatives.
- **Design rationale** provides an explicit means of recording those design decisions and the context in which the decisions were made.

The Software Life Cycle

- Software engineering is the discipline for understanding the software design process, or life cycle.
- Designing for usability occurs at all stages of the life cycle, not as a single isolated activity.

Activities in the life cycle

Requirements specification: Designer and customer try capture what the system is expected to provide can be expressed in natural language or more precise languages, such as a task analysis would provide.

Architectural design: High-level description of how the system will provide the services required factor system into major components of the system and how they are interrelated needs to satisfy both functional and nonfunctional requirements. **We should:**

- Concentrate on how the system provides the services expected from it.
- In the case, 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 he interdependencies between separate components and the sharing of resources that will arise between components.

Detailed design: refinement of architectural components and interrelations to identify modules to be implemented separately the modification is governed by the nonfunctional requirements.

- The architectural design provides a decomposition of the system description that allows for isolated development of separate components which will later be integrated.
- For those components that are not already available for immediate integration, the designer must provide a sufficiently detailed description so that they may be implemented in some programming language.
- 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.

Coding and unit testing: The detailed design for a component of the system should be in such a form that it is possible to implement it in some executable programming language.

- After coding, the component can be tested to verify that it performs correctly, according to some test criteria that were determined in earlier activities.
- Research on this activity within the life cycle has concentrated on two areas.
- There is plenty of research that is expressed towards the automation of this coding activity directly from a low-level detailed design.

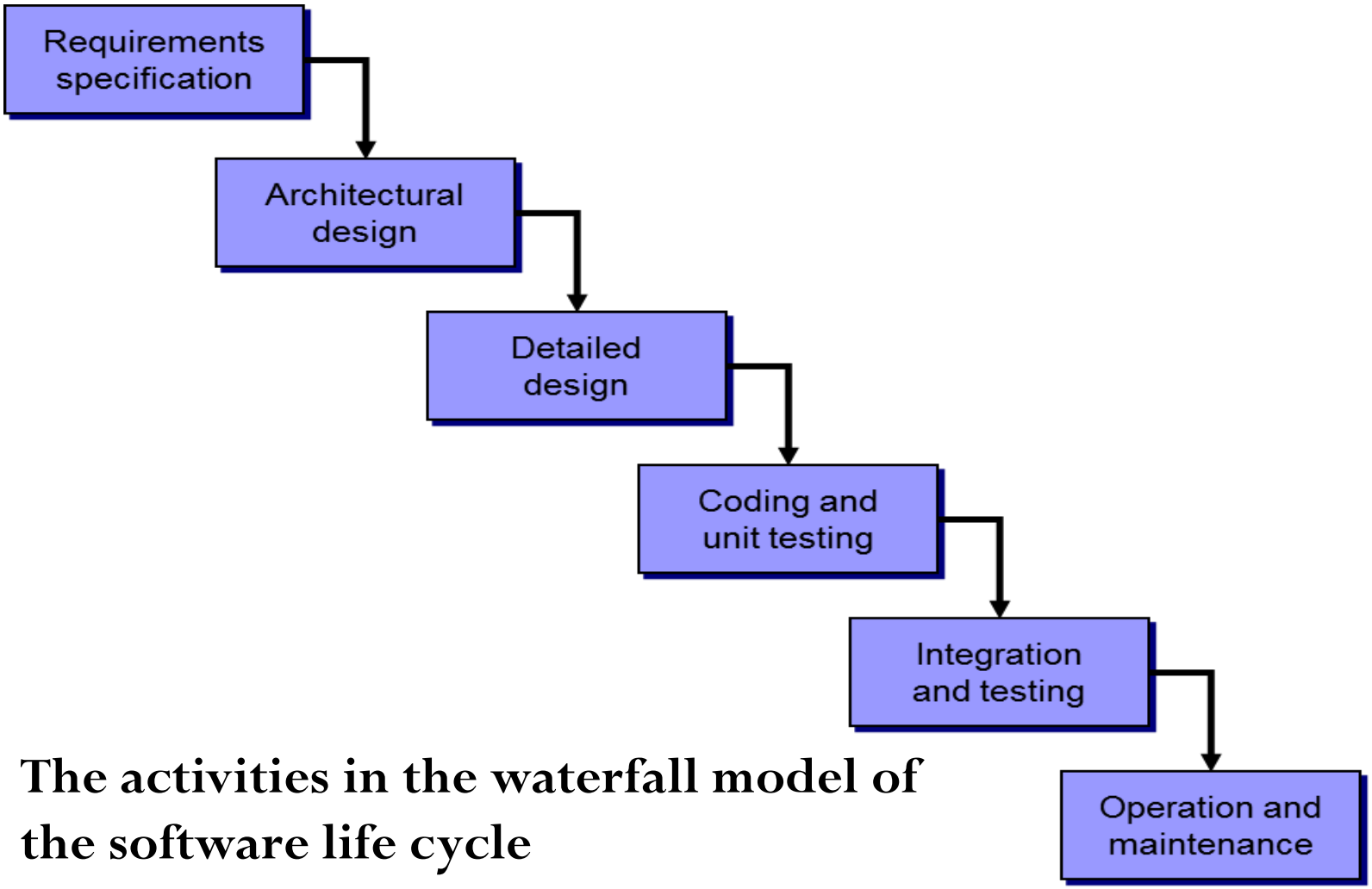
Integration and testing: Once enough components have been implemented and individually tested, they must be integrated as described in the **architectural design**.

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

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

- Consequently, the majority of the lifetime of a product is spent in the maintenance activity.
- Maintenance involves the correction of errors in the system which is discovered after release and the revision of the system services to satisfy requirements that were not realized during previous development.
- Therefore, maintenance provides feedback to all of the other activities in the life cycle.

- **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, respectively.
- Verification of a design will most often occur within a single life-cycle activity or between two adjacent activities.
- Validation is a much more subjective exercise than verification,
- In interactive system design, the validation against HCI requirements is often referred to as evaluation and can be performed by the designer in isolation or in cooperation with the customer.



The activities in the waterfall model of the software life cycle

Usability Engineering

- One approach to user-centered design has been the introduction of explicit usability engineering goals into the design process.
- 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.
- But, the problem with usability metrics is that they rely on measurements of very specific user actions in very specific situations.

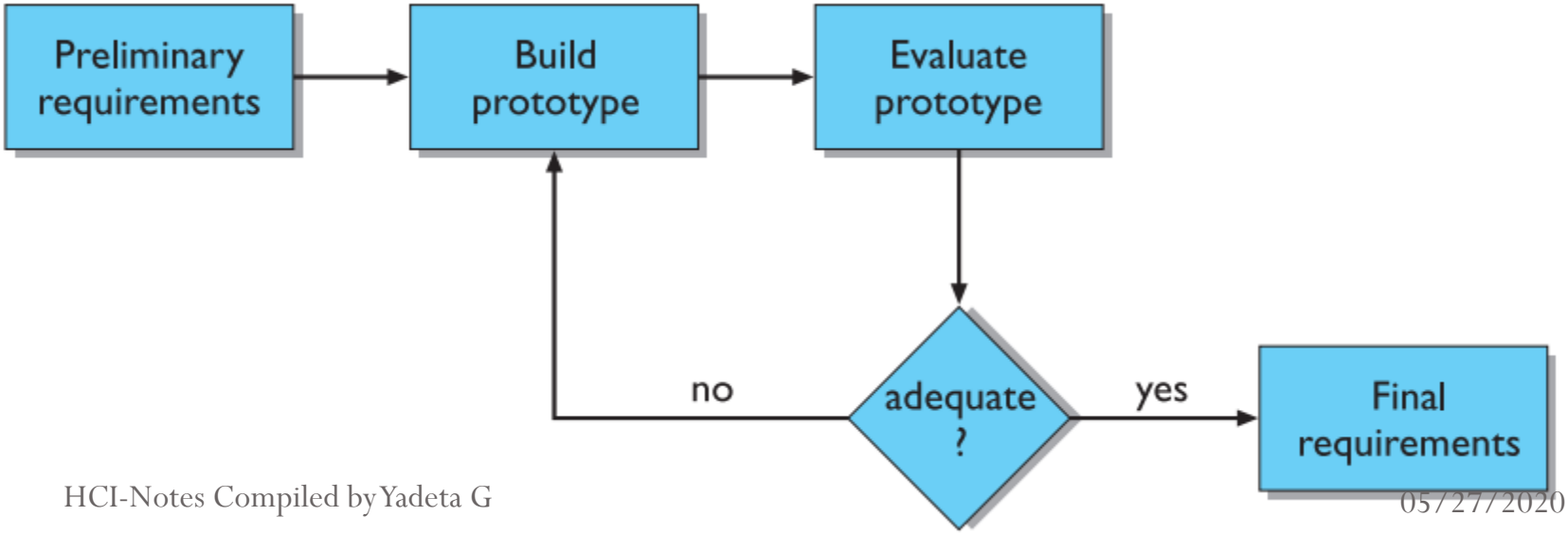
Iterative Design and Prototyping

- The design can then be modified to correct any false assumptions that were revealed in the testing.
- This is the **essence of iterative design**, a purposeful design process which tries to overcome the inherent problems of incomplete requirements specification by cycling through several designs, incrementally improving upon the final product with each pass.
- On the technical side, iterative design is described by the use of prototypes, artifacts that simulate or animate some but not all features of the intended system.
- There are three main approaches to prototyping:

Three Main Approaches To Prototyping

Throw-away the prototype is built and tested. The design knowledge gained from this exercise is used to build the final product, but the actual prototype is discarded.

- Next figure 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.



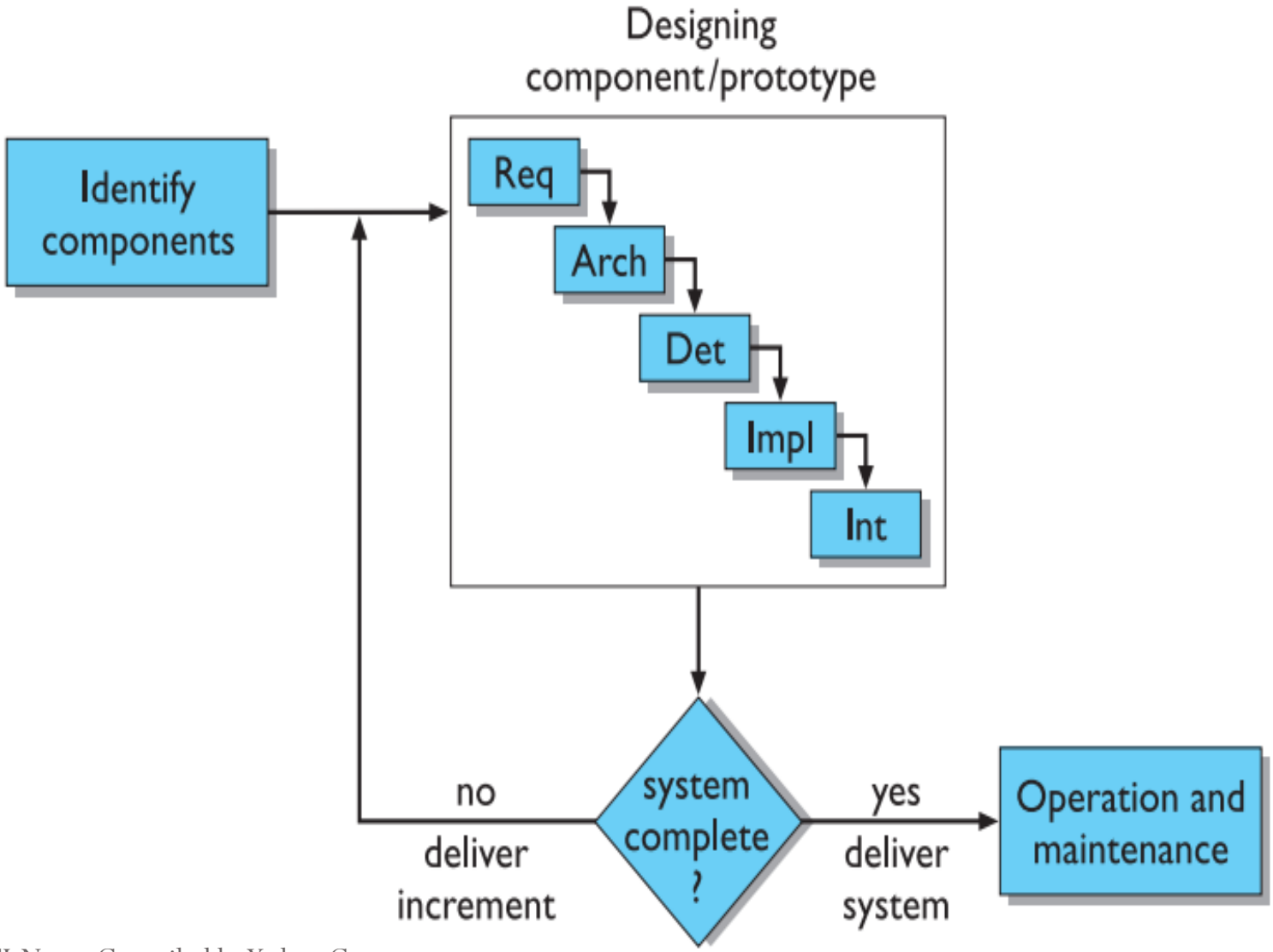
Incremental

- 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.

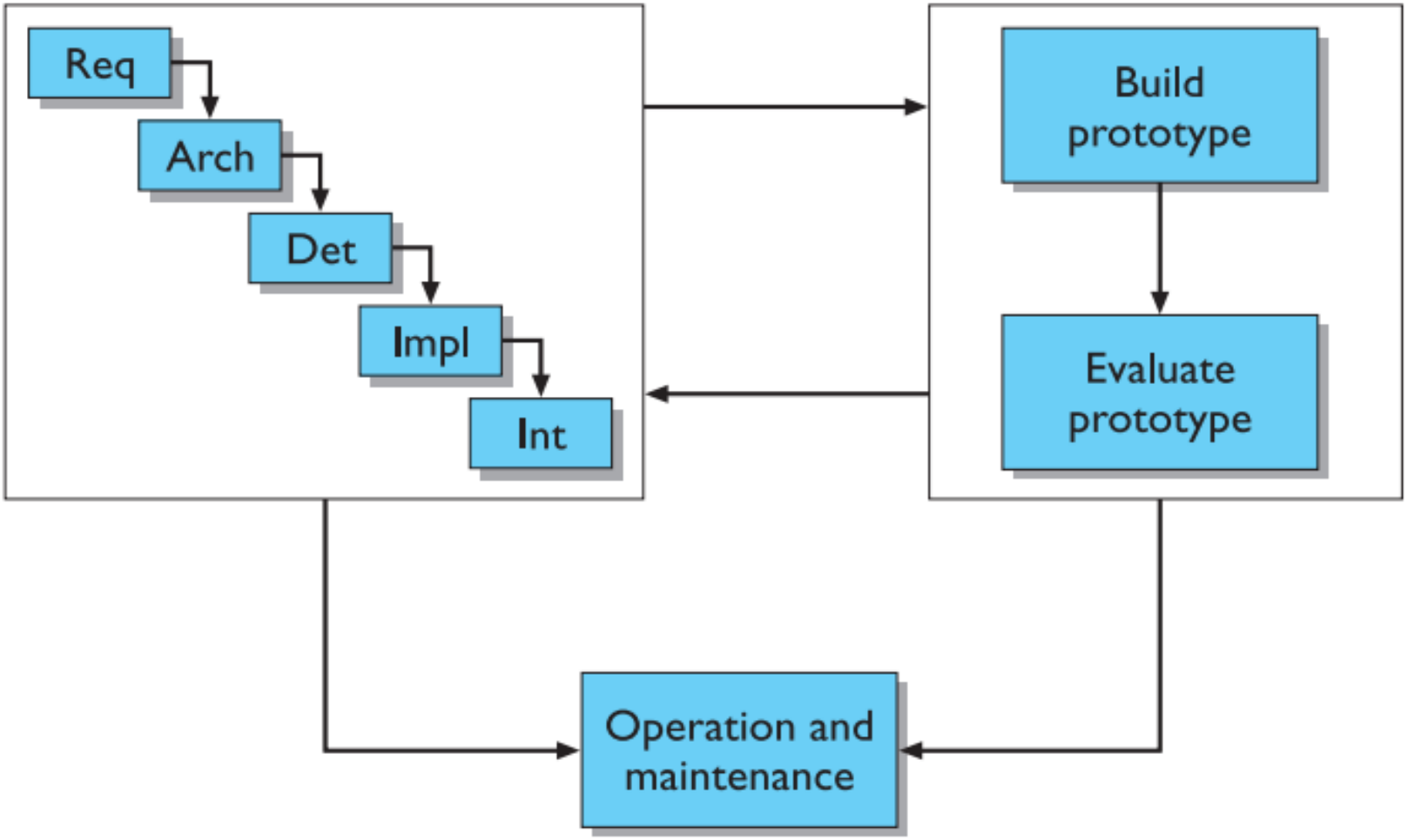
Evolutionary

- 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.
- 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.

Incremental prototyping within the life cycle



Evolutionary prototyping throughout the life cycle



Design Rationale

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

Summary

Interaction design is about creating interventions in often

- complex situations using technology of many kinds including PC software, the web and physical devices.

Design involves:

- achieving goals within constraints and trade-off between these
- understanding the raw materials: computer and human
- accepting limitations of humans and of design.

- The design process has several stages and is iterative and never complete.
- Interaction starts with getting to know the users and their context:

– finding out who they are and what they are like… probably not like you!

– talking to them, watching them.

CHAPTER SIX

Design Rules and Implementation support

Design Rules

Introduction

- One of the central problems that must be solved in a user-centered design process is how to provide designers with the ability to determine the usability consequences of their design decisions.
- We require design rules, which are rules a designer can follow in order to increase the usability of the eventual software product.
- 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.

- Designing for maximum usability is the goal of **interactive systems** design.
- Abstract principles offer a way of understanding usability in a more general sense, especially if we can express them within some coherent catalog.
- Design rules in the form of **standards** and **guidelines** provide direction for design, in both general and more concrete terms, in order to enhance the interactive properties of the system.
- The essential characteristics of good design are often summarized through '**golden rules**' or heuristics.
- Design patterns provide a potentially generative approach to capturing and reusing design knowledge.

Concept of Usability Engineering

- Usability Engineering is a method in the progress of software and systems, which includes user contribution from the inception of the process and assures the effectiveness of the product through the use of a usability requirement and metrics.
- It refers to the Usability Function features of the entire process of abstracting, implementing & testing hardware and software products.
- Requirements gathering stage to installation, marketing and testing of products, all fall in this process.

Goals of Usability Engineering

- Effective to use – Functional
- Efficient to use – Efficient
- Error free in use – Safe
- Easy to use – Friendly
- Enjoyable in use – Delightful Experience

Guidelines for designing (Normans 7 principles)

- Use both knowledge on the world and knowledge in head
- Simplify the structure of tasks
- Make things visible execute and evaluate
- Get the mappings right
- Exploit the power of constraints both natural and artificial.
- Design for error
- When all else, standardize

Guidelines for Designing Conceptual Models:

- Reflect the user's mental model, not the designer's.
- Minimal training requirements (easy to learn)
- High transfer of training (easy to remember)
- People perform real tasks well (efficient to use)
- Easy to recover from errors
- Invisible
- People like it (subjectively pleasing)
- Draw physical analogies or present metaphors.
- Comply with expectancies, habits, routines.
- Few fonts and colors (5 to 7 colors)

- Provide action-response compatibility.
- Make invisible parts and process of a system visible.
- Provide proper and correct feedback.
- Avoid anything unnecessary or irrelevant.
- Provide design consistency.
- Provide documentation and a help system that will reinforce the conceptual model.
- Promote the development of both novice and expert mental models
- Good Graphic Design and Color Choice
- Minimize User Memory Load
- (Pervasive, generic rules (cut/paste) Prompts provide format, Cancel buttons, exit, dialog box etc….)

Principles to Support Usability

- The most abstract design rules are general principles, which can be applied to the design of an interactive system in order to promote its usability.
- Derivation of principles for interaction has usually arisen out of a need to explain why a paradigm is successful and when it might not be.
- Principles can provide the repeatability which paradigms in themselves cannot provide.
- In this section we present a collection of usability principles.

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

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

In the next, we will subdivide these main categories into more specific principles that support them. By looking Summary of principles affecting learnability

Summary of principles affecting learnability

- **Predictability:** Support for the user to determine the effect of future action based on past interaction history
- **Synthesizability:** Support for the user to assess the effect of past operations on the current state
- **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
- **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

Standards

- Standards for interactive system design are usually set by national or international bodies to ensure compliance with a set of design rules by a large community.
- Standards can apply specifically to either the hardware or the software used to build the interactive system.
- Smith points out the differing characteristics between
- hardware and software, which affect the utility of design standards applied to them:
  - Underlying theory
  - Change

For example, the UK Ministry of Defense has published an Interim Defense

Standard 00–25 on Human Factors for Designers of Equipment, produced in 12 parts:

Part 1 Introduction
Part 2 Body Size
Part 3 Body Strength and Stamina
Part 4 Workplace Design
Part 5 Stresses and Hazards
Part 6 Vision and Lighting
Part 7 Visual Displays
Part 8 Auditory Information
Part 9 Voice Communication
Part 10 Controls
Part 11 Design for Maintainability
Part 12 Systems

Guidelines

- We have observed that the incompleteness of theories underlying the design of interactive software makes it difficult to produce authoritative and specific standards.
- As a result, the majority of design rules for interactive systems are suggestive and more general guidelines.
- Our concern in examining the wealth of available guidelines is in determining their applicability to the various stages of design.
- The more abstract guideline, which would be most suited to requirements specification; and the more specific the guideline, is the more suited it is to detailed design.
- The guidelines can also be automated to some extent, providing a direct means for translating detailed design specifications into actual implementation.

Golden Rules and Heuristics

- So far we have considered a range of abstract principles and detailed guidelines, which can be used to help designers produce more usable systems.
- But all of these rules require a certain amount of commitment on the part of the designer, either to track down appropriate guidelines or to interpret principles.

Is there a simpler way?

Shneiderman's Eight Golden Rules of Interface Design

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

1. Strive for consistency in action sequences, layout, terminology, command use and so on.

2. Enable frequent users to use shortcuts, such as abbreviations, special key sequences and macros, to perform regular, familiar actions more quickly.

3. Offer informative feedback for every user action, at a level appropriate to the magnitude of the action.

4. Design dialogs to yield closure so that the user knows when they have completed a task.

5. Offer error prevention and simple error handling so that, ideally, users are prevented from making mistakes and, if they do, they are offered clear and informative instructions to enable them to recover.

6. Permit easy reversal of actions in order to relieve anxiety and encourage exploration, since the user knows that he can always return to the previous state.

7. Support internal locus of control so that the user is in control of the system, which responds to his actions.

8. Reduce short-term memory load by keeping displays simple, consolidating multiple page displays and providing time for learning action sequences.

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

- In previous Chapter we discussed Norman's execution–evaluation cycle, in which he elaborates the seven stages of action.

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

1. Use both knowledge in the world and knowledge in the head.
2. Simplify the structure of tasks.

3. Make things visible: bridge the gulfs of execution and evaluation.

4. Get the mappings right.

5. Exploit the power of constraints, both natural and artificial.

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.

HCI Patterns

- As we observed in prior, one way to approach design is to learn from examples that have proven to be successful in the past: to reuse the knowledge of what made a system or paradigm successful.
- Patterns are an approach to **capturing** and **reusing** this knowledge of abstracting the essential details of successful design so that these can be applied again and again in new situations.
- Patterns originated in architecture, where they have been used successfully, and they are also used widely in software development to capture solutions to common programming problems.
- Patterns address the problems that designers face by providing a 'solution statement'.

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

- They capture design practice and embody knowledge about successful solutions: they come from practice rather than psychological theory.
- They capture the essential common properties of good design: they do not tell the designer how to do something but what needs to be done and why.
- They represent design knowledge at varying levels, ranging from social and organizational issues through conceptual design to detailed widget design.
- They are not neutral but embody values within their rationale. Alexander's language clearly expresses his values about architecture. HCI patterns can express values about what is humane in interface design.
- The concept of a pattern language is generative and can therefore assist in the development of complete designs.
- They are generally intuitive and readable and can therefore be used for communication between all stakeholders.

Implementation support

- Programming tools for interactive systems provide a means of effectively translating abstract designs and usability principles into an executable form.
- These tools provide different levels of services for the programmer.
- Windowing systems are a central environment for both the programmer and user of an interactive system, allowing a single workstation to support separate user system threads of action simultaneously.

- Interaction toolkits abstract away from the physical separation of input and output devices, allowing the programmer to describe behaviors of objects at a level similar to how the user perceives them.

- User interface management systems are the final level of programming support tools, allowing the **designer** and **programmer** to control the relationship between the presentation objects of a toolkit with their functional semantics in the actual application.

User Interface Management System (UIMS)

The main concerns of a UIMS, for our purposes, are:

- a conceptual architecture for the structure of an interactive system which concentrates on a separation between application semantics and presentation;
- techniques for implementing a separated application and presentation whilst preserving the intended connection between them;
- support techniques for managing, implementing and evaluating a run-time interactive environment.

UIMS as a conceptual architecture

- A major issue in this area of research is one of separation between the semantics of the application and the interface provided for the user to make use of that semantics.

There are many good arguments to support this separation of concerns:

- **Portability** To allow the same application to be used on different systems it is best to consider its development separate from its device-dependent interface.
- **Reusability Separation** increases the likelihood that components can be reused in order to cut development costs.
- **Multiple interfaces** To enhance the interactive flexibility of an application, several different interfaces can be developed to access the same functionality.
- **Customization** The user interface can be customized by both the designer and the user to increase its effectiveness without having to alter the underlying application.

The logical components of a UIMS were identified as:

- **Presentation** The component responsible for the appearance of the interface, including what output and input is available to the user.
- **Dialog control** The component which regulates the communication between the presentation and the application.
- **Application** interface The view of the application semantics that is provided as the interface.

Chapter 7 Evaluation Techniques and Universal Design

Evaluation tests the usability, functionality and acceptability of an interactive system.

Evaluation may take place:– in the laboratory and in the field.

Some approaches are based on expert evaluation:

- analytic methods
- review methods
- model-based methods.

Some approaches involve users:

- experimental methods
- observational methods
- query methods.

An evaluation method must be chosen carefully and must be suitable for the job.

What is Evaluation?

- In previous chapters we have discussed a design process to support the design of usable interactive systems.
- However, even if such a process is used, we still need to assess our designs and **test our systems to ensure** that they actually behave as we expect and meet user requirements. This is the role of evaluation.
- Evaluation should not be supposed 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.

Goals of Evaluation

Evaluation has three main goals:

- to assess the extent and accessibility of the system's functionality,
- to assess users' experience of the interaction, and
- to identify any specific problems with the system.

- We will consider evaluation techniques under **two broad** headings: expert analysis and user participation.
- However, we will consider why we do evaluation and what we are trying to achieve (**goals of evaluation**).

- The system's functionality is important in that it must accord with the user's requirements.
- In other words, the design of the system should enable users to perform their intended tasks more easily.
- In addition to evaluating the system design in terms of its functional capabilities, it is important to assess the user's experience of the interaction and its impact upon him.
- This includes considering aspects such as how easy the system is to learn, its usability and the user's satisfaction with it.
- The final goal of evaluation is to identify specific problems with the design.
- These may be aspects of the design which, when used in their intended context, cause unexpected results, or confusion amongst users.

Ease of Learning & Ease of Use

Ease of learning

- determine naturalness of voice commands and quality of feedback
- determine effectiveness of buttons' images and text in conveying function
- understanding how the shopping list and recipe function works

Ease of use

- After initial exploration, given a task, can the user quickly accomplish it
- Do they take the direct path or press some wrong buttons along the way
- Evaluation will be based on time, errors, and frustration

Evaluation Through Expert Analysis

Evaluation should occur throughout the design process.

- In particular, the first evaluation of a system should ideally be performed before any implementation work has started.
- If the design itself can be evaluated, expensive mistakes can be avoided, since the design can be altered prior to any major resource commitments.
- Typically, the later in the design process that an error is discovered,

The four approaches to expert analysis:

- cognitive walkthrough,
- heuristic evaluation,
- the use of models and
- use of previous work.

Evaluation Through User Participation

- The techniques we have considered so far concentrate on evaluating a design or system through analysis by the designer, or an expert evaluator, rather than testing with actual users.
- However, useful as these techniques are for filtering and refining the design, they are not a replacement for actual usability testing with the people for whom the system is intended: the users.
- Different approaches to evaluation through user participation includes: **empirical or experimental methods**, **observational methods, query techniques,** and methods that use physiological monitoring, such as eye tracking and measures of heart rate and skin conductance.

Choosing an Evaluation Method

- As we have seen, a range of techniques is available for evaluating an interactive system at all stages in the design process.

So how do we decide which methods **are most appropriate** for our needs?

- There are no hard and fast rules in this – each method has its particular strengths and weaknesses and each is useful if applied appropriately.
- However, there are a number of factors that should be taken into account when selecting evaluation techniques.
- These also provide a way of categorizing the different methods so that we can compare and choose between them.

Factors Distinguishing Evaluation Techniques

We can identify at **least eight factors** that distinguish different evaluation techniques and therefore help us to make an appropriate choice. **These are:**

1. the stage in the cycle at which the evaluation is carried out
2. the style of evaluation
3. the level of subjectivity or objectivity of the technique
4. the type of measures provided
5. the information provided
6. the immediacy of the response
7. the level of interference implied
8. the resources required.

Universal Design

Universal design is **about designing systems so that they can be used** by anyone in any circumstance.

- Multi-modal systems are those that use more than one human input channel in the interaction.
- These systems may, for example, use:
  - speech
  - non-speech sound
  - touch
  - handwriting
  - gestures.
- Universal design means designing for diversity, including:

– people with sensory, physical or cognitive impairment

– people of different ages

– people from different cultures and backgrounds.

Introduction

- Universal design is the process of designing products so that they can be used by as many people as possible in as many situations as possible.
- In our case, this means particularly designing interactive systems that are usable by anyone, with any range of abilities, using any technology platform.
- This can be achieved by designing systems either to have built in redundancy or to be compatible with assistive technologies.

Universal Design Principles

- We have defined 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'.
- But what does that mean in practice?
- Is it possible to design anything so that anyone can use
- it and if we could, how practical would it be?
- Wouldn't the cost be prohibitive? In reality, we may not be able to design everything to be accessible to everyone, and we certainly cannot ensure that everyone has the same experience of using a product, but we can work toward the aim of universal design and try to provide an equivalent experience.

Seven General Principles of Universal Design

- These were intended to cover all areas of design and are equally applicable to the design of interactive systems.
- These principles give us a framework in which to develop universal designs.
- Principle one is **equitable use**: the design is useful to people with a range of abilities and appealing to all. No user is excluded or stigmatized.
- Wherever possible, access should be the same for all; where identical use is not possible, equivalent use should be supported.
- Where appropriate, security, privacy and safety provision should be available to all.

- Principle two is **flexibility in use**: the design allows for a range of ability and preference, through choice of methods of use and adaptivity to the user's pace, precision and custom.
- Principle three is that the **system be simple and intuitive to use**, regardless of the knowledge, experience, language or level of concentration of the user.
- The design needs to support the user's expectations and accommodate different language and literacy skills.
- It should not be unnecessarily complex and should be organized to facilitate access to the most important areas.
- It should provide prompting and feedback as far as possible.

- Principle four is **perceptible information**: the design should provide effective communication of information regardless of the environmental conditions or the user's abilities.
- Principle five is **tolerance for error:** minimizing the impact and damage caused by mistakes or unintended behavior.
- Principle six is **low physical effort:** systems should be designed to be comfortable to use, minimizing physical effort and fatigue.
- 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.

Multi-Modal Interaction

- As we have seen in the previous section, providing access to information through more than one mode of interaction is an important principle of universal design.
- Such design relies on multi-modal interaction.
- We also discussed like there are five senses: sight, sound, touch, taste and smell.
- Therefore, our design of interaction system should consider the variety of information Transmission/accessed like:
- Sound in the interface (sound non-speech and speech)

In Multimodal Interaction:

- Sound in the Interface includes: speech sound, non-speech sounds --> we also do for speech recognition in our interaction.
- Touch in the Interface includes: handwriting recognition

Designing for Diversity

- We can make general observations about human capabilities, users in fact have different needs and limitations.
- Interfaces are usually designed to cater for the 'average' user, but unfortunately this may exclude people who are not 'average'.
- As we saw in the introduction to this chapter, people are **diverse and there are many factors that must** be taken into account if we are to come close to universal design.
- We consider briefly some of these factors and the particular challenges that each raises.
- We will consider three key areas: **disability, age and culture**.

Designing for users with disabilities

- It is estimated that at least 10% of the population of every country has a disability that will affect interaction with computers.
- Employers and manufacturers of computing equipment have not only a moral responsibility to provide accessible products, but often also a legal responsibility.
- In many countries, legislation now demands that the workplace must be designed to be accessible or at least adaptable to all the design of software and hardware should not unnecessarily restrict the job prospects of people with disabilities.

We should consider the disabilities:

- **Visual impairment:** The sensory impairment that has attracted the most attention from researchers, perhaps because it is potentially also one of the most debilitating as far as interaction is concerned, is visual impairment.
- **Hearing impairment:** Compared with a visual disability where the impact on interacting with a graphical interface is immediately obvious, a hearing impairment may appear to have little impact on the use of an interface.
- **Physical impairment:** Users with physical disabilities vary in the amount of control and movement that they have over their hands, but many find the precision required in mouse control difficult.

Speech input and output is an option for those without speech difficulties.

We should consider the disabilities:

- **Speech impairment:** For users with speech and hearing impairments, multimedia systems provide a number of tools for communication, including synthetic speech and text-based communication and conferencing systems.

Textual communication is slow, which can lower the effectiveness of the communication.

- **Dyslexia:** Users with cognitive disabilities such as dyslexia can find textual information difficult.

In severe cases, speech input and output can alleviate the need to read and write and allow more accurate input and output.

Autism: mental condition

Autism affects a person's ability to communicate and interact with people around them and to make sense of their environment.

This manifests itself in a range of ways but is characterized by the triad of impairments:

1. **Social interaction** – problems in relating to others in a meaningful way or responding appropriately to social situations.

2. **Communication** – problems in understanding verbal and textual language including the use of gestures and expressions.

3. **Imagination** – problems with rigidity of thought processes, which may lead to repetitive behavior and inflexibility.

Designing for Different Age Groups

- We have considered how people differ along a range of sensory, physical and cognitive abilities.
- However, there are other areas of diversity that impact upon the way we design interfaces. One of these is age.
- In particular, older people and children have specific needs when it comes to interactive technology.

Designing for cultural differences

- The final area of diversity we will consider is cultural difference.
- Cultural difference is often used synonymously with national differences but this is too simplistic.

Summary

- Universal design is about designing systems that are accessible by all users in all circumstances, taking account of human diversity in disabilities, age and culture.
- Universal design helps everyone – for example, designing a system so that it can be used by someone who is deaf or hard of hearing will benefit other people working in noisy environments or without audio facilities.
- Designing to be accessible to screen reading systems will make websites better for mobile users and older browsers.
- Multi-modal systems provide access to system information and functionality through a range of different input and output channels, exploiting redundancy.
- Such systems will enable users with sensory, physical or cognitive impairments to make use of the channels that they can use most effectively.
- But all users benefit from multi-modal systems that utilize more of our senses in an involving interactive experience.
- For any design choice we should ask ourselves whether our decision is excluding someone and whether there are any potential confusions or misunderstandings in our choice.

Chapter 8: User Support

Users have different requirements for support at different times.

- **User support should be:**
  - available but unobtrusive
  - accurate and robust
  - consistent and flexible.
- **User support comes in a number of styles:**
  - command-based methods
  - context-sensitive help
  - tutorial help
  - online documentation
  - wizards and assistants
  - adaptive help.
- **Design of user support must take account of:**
  - presentation issues
  - implementation issues.

Introduction

- There is often an **implicit assumption** that if an interactive system is properly designed it will be completely intuitive to use and the user will require little or no help or training.
- This may be a grand ideal but it is far from true with even the best designed systems currently available
- The type of assistance users require varies and is dependent on many factors: their familiarity with the system, the job they are trying to do, and so on.

There are four main types of assistance that users require:

- **quick reference**
- **task-specific help**
- **full explanation**
- **tutorial..**

Requirements of User Support

- If we were to design the ideal help system, what would it look like?
- This is a difficult question to answer, but we can point to some features that we might like our help system to have.
- Not every help system will have all of these features, sometimes for good reason, but they are useful as benchmarks against which we can test the support tools we design.

Availability

- The user needs to be able to access help at any time during his interaction with the system.
- In particular, he should not have to quit the application he is working on in order to open the help application.
- Ideally, it should run concurrently with any other application.
- This is obviously a problem for non-windowed systems if the help system is independent of the application that is running.
- However, in windowed systems there is no reason why a help facility should not be available constantly, at the press of a button. Availability may include: **Accuracy and completeness, Consistency, Robustness, Flexibility**

Approaches to User Support

- As we noted in the previous section, there are a number of different approaches to providing help, each of which meets a particular need.
- These vary from simple captions to full adaptive help and tutoring systems.
- In this section we will concentrate on the styles of help provided rather than any particular help system (although we will use real help systems for illustration).
- Next we will then go on to look at adaptive help in more detail.

Approaches to User Support

Command assistance

- Perhaps the most basic approach to user support is to provide assistance at the command level – the user requests help on a particular command and is presented with a help screen or manual page describing it.

Command prompts

- In command line interfaces, command prompts provide help when the user encounters an error, usually in the form of correct usage prompts.
- Such prompts are useful if the error is a simple one, such as incorrect syntax, but again they assume knowledge of the command.

… cont'd

Context-sensitive help

- Some help systems are context sensitive.
- These range from those that have specific knowledge of the particular user (which we will consider under adaptive help) to those that provide a simple help key or function that is interpreted according to the context in which it is called and will present help accordingly.

Online tutorials

- Online tutorials allow the user to work through the basics of an application within a test environment.
- The user can progress at his own speed and can repeat parts of the tutorial if needed.
- He will also get a feel for how the application works by experimenting with examples, albeit small ones, or by watching an automated demonstration of how to perform a task.

… cont'd

Online documentation

- Online documentation effectively makes the existing paper documentation available on computer.
- This makes the material available continually (assuming the machine is running!) in the same medium as the user's work and, potentially, to a large number of users concurrently.

Wizards and assistants

- A wizard is a task-specific tool that leads the user through the task, step by step, using information supplied by the user in response to questions along the way.
- They are distinct from demonstrations in that they allow the user actually to complete the task.

Adaptive Help Systems

- In any large or complex computer system, users will be familiar with a subset of the available functionality, demonstrating expertise in some applications and having no experience with others, even to the point of being unaware of their existence.
- In addition, different users will have different needs and levels of under standing.
- Adaptive help systems attempt to address these problems by adapting the help that they provide to the individual user who is making the request and by actively suggesting alternative courses of action of which the user may not be aware.

- Adaptive help is a special case of a general class of interactive systems, known as intelligent systems.
- These include domain-specific expert systems, intelligent tutoring systems and general adaptive interfaces.
- Adaptive help systems operate by monitoring the activity of the user and constructing a model of him.
- We should consider some of the developments and solutions, concentrating, in particular, on the knowledge requirements.
- Knowledge representation: user modeling
- Knowledge representation: domain and task modeling
- Knowledge representation: modeling advisory strategy

Techniques for knowledge representation

- **Rule-based techniques**

Knowledge is represented as a set of rules and facts, which are interpreted using some inference mechanism.

- **Frame-based techniques**

Frame-based systems are used to represent commonly occurring situations and default knowledge.

- **Network-based techniques**

Networks represent knowledge about the user and system in terms of relationships between facts.

- **Example-based techniques**

Example-based techniques represent knowledge implicitly within a decision structure of a classification system.

Designing User Support Systems

- There are many ways of providing user support and it is up to the designer to decide which is most appropriate for any given system.
- However, there are a number of things which the designer should take into account.
- First, the design of user support should not be seen as an 'add-on' to system design.
- Ideally, the help system should be designed integrally with the rest of the system.
- If this is done, the help system will be relevant and consistent with the rest of the system.

- The same modeling and analytic used to design the system can guide the design of support material as well.
- Secondly, the designer should consider the content of the help and the context in which it will be used before the technology that will be required.

Presentation issues

How is help requested?

- The first decision the designer must make is how the user will access help.
- There are a number of choices.
- Help may be a command, a button, a function which can be switched on or off, or a separate application.
- A command (usually) requires the user to specify a topic, and therefore assumes some knowledge, but may fit most consistently within the rest of the interface.

- How is help displayed?
- Effective presentation of help

Implementation issues

- Alongside the presentation issues the designer must make implementation decisions.
- Some of these may be forced by physical constraints, others by the choices made regarding the user's requirements for help. We have already considered how help may be requested and how it appears to the user.

Summary

- This chapter has been concerned with user support in the form of help and documentation.
- No interactive system of any complexity is so intuitive that the user never requires help. Help should therefore be an integral part of the design.
- Users require different types of help, depending on the context and circumstances, and the user support facilities should support these.
- Different styles of help support different requirements and different types of user.
- We have considered several types of help system, including adaptive user support.
- It is important to select a support style and design user support with the user in mind, just as the design of the system is user centered.
- In particular, the presentation of help should take into account usability principles, and the language should be clear and instructional.