**UNIT- II:**

**Interaction Design Process: Iterative Design, User-centred Design, Interaction Design Models, Overview of Interaction Design Models**

**Discovery: Discovery Phase Framework, Collection, Interpretation, Documentation**

**Design: Conceptual Design, Physical Design, Evaluation, Interface Design Standards, Designing the Facets of the Interface.**

## Interaction Design Process:

### Iterative Design:

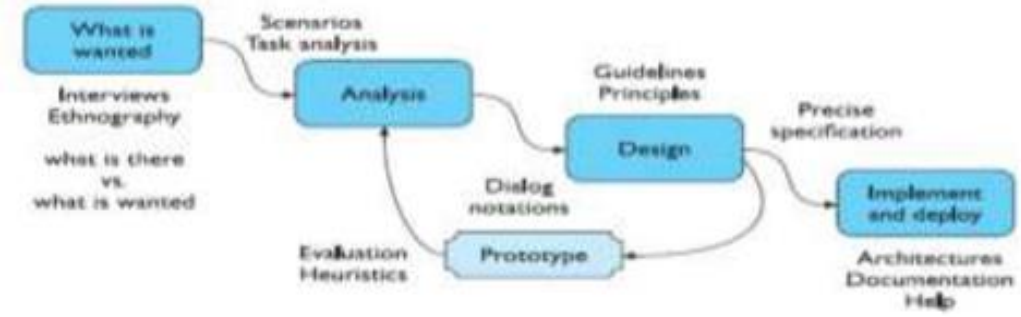



**Figure: Interaction design process**

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

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.

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.

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.

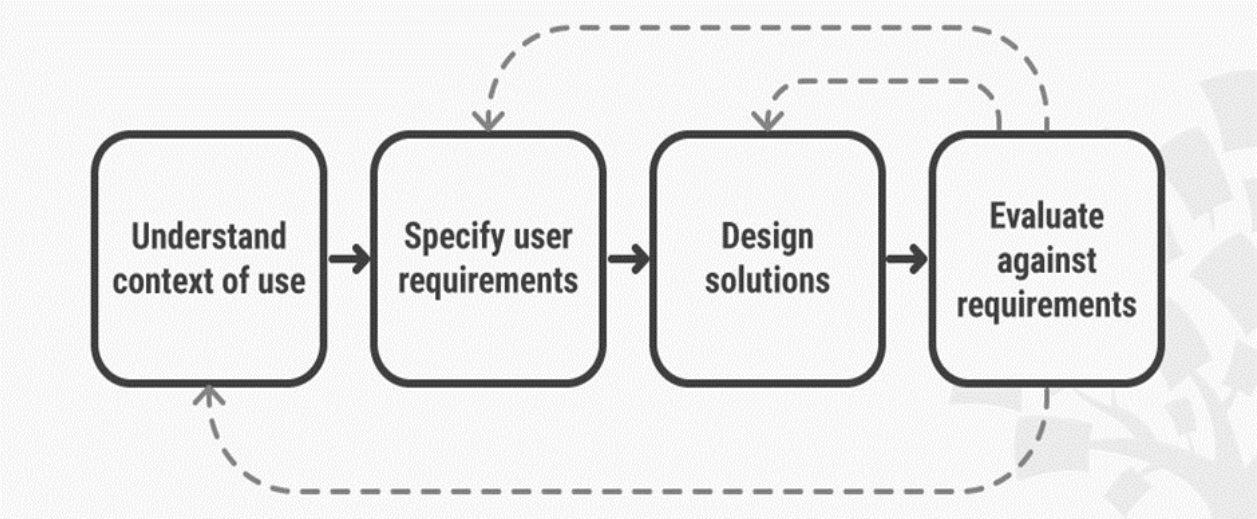
Iteration and prototyping: Humans are complex and we cannot expect to get designs right first time. We therefore need to evaluate a design to see how well it is working and where there can be improvements.

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.

## User-centred Design:

User-centered design (UCD) is an iterative design process in which designers focus on the users and their needs in each phase of the design process. In UCD, design teams involve users throughout the design process via a variety of research and design techniques, to create highly usable and accessible products for them.

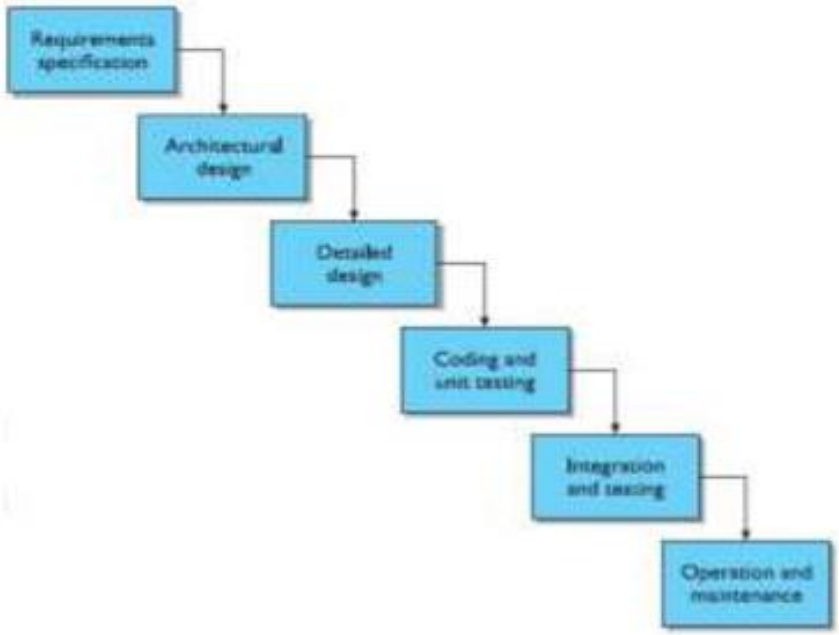
Generally, each iteration of the UCD approach involves four distinct phases. First, as designers working in teams, we try to understand the *context* in which users may use a system. Then, we identify and specify the users' *requirements*. A *design* phase follows, in which the design team develops solutions. The team then proceeds to an *evaluation* phase. Here, you assess the outcomes of the evaluation against the users' context and requirements, to check how well a design is performing. More specifically, you see how close it is to a level that matches the users' specific context and satisfies all of their relevant needs. From here, your team makes further iterations of these four phases, and you continue until the evaluation results are satisfactory..




*User-centered design is an iterative process that focuses on an understanding of the users and their context in all stages of design and development.*

# Interaction Design Models:

## Waterfall Model:



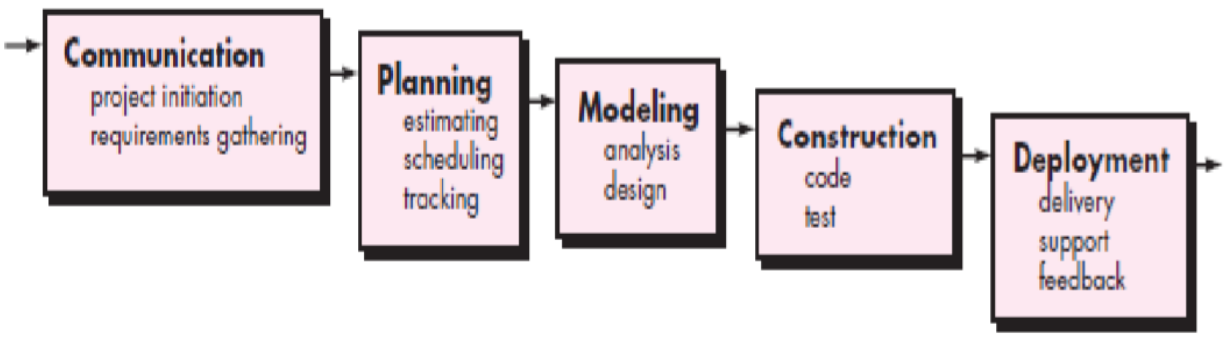

Requirements specification: Requirements specification begins at the start of product development. Though the requirements are from the customer's perspective, if they are to be met by the software product they must be formulated in a language suitable for implementation. Requirements are usually initially 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. The transformation from the expressive but relatively ambiguous natural language of requirements to the more precise but less expressive executable languages is one key to successful development. Task analysis techniques, which are used to express work domain requirements in a form that is both expressive and precise.

Architectural design: The requirements specification concentrates on what the system is supposed to do. 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.

Detailed design: The architectural design provides a decomposition of the system description that allows for isolated development of separate components which will later be integrated. 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. 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: The detailed design for a component of the system should be in such a form that it is possible to implement it in some executable programming language. After coding, the component can be tested to verify that it performs correctly, according to some test criteria that were determined in earlier activities. Research on this activity within the life cycle has concentrated on two areas. There is plenty of research that is geared towards the automation of this coding activity directly from a low-level detailed design. Most of the work in formal methods operates under the hypothesis that, in theory, the transformation from the detailed design to the implementation is from one mathematical representation to another and so should be able to be entirely automated. Other, more practical work concentrates on the automatic generation of tests from output of earlier activities which can be performed on a piece of code to verify that it behaves correctly

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.




Waterfall Model

## THE SPIRAL MODEL:

The Spiral model is proposed by “Boehm”. This model was developed to encompass the best features of waterfall model and prototyping. It is a risk-driven process model with the risk analysis feature.

Features:

1) Cyclic Approach for increasing system’s degree of definition and implementation while decreasing degree of risk-Risk is considered as each revolution is made.

2) Anchor Point Milestone for ensuring stakeholders commitment to feasible and mutually satisfactory systems solution. (Milestone is a combination of work products and conditions).

Spiral model may be viewed as a Meta model, as it can accommodate any process development model. Software is developed as a series of evolutionary releases. Project manager adjusts planned number of iterations to complete the software. During early iterations prototype is generated and during later iterations complete version is developed.

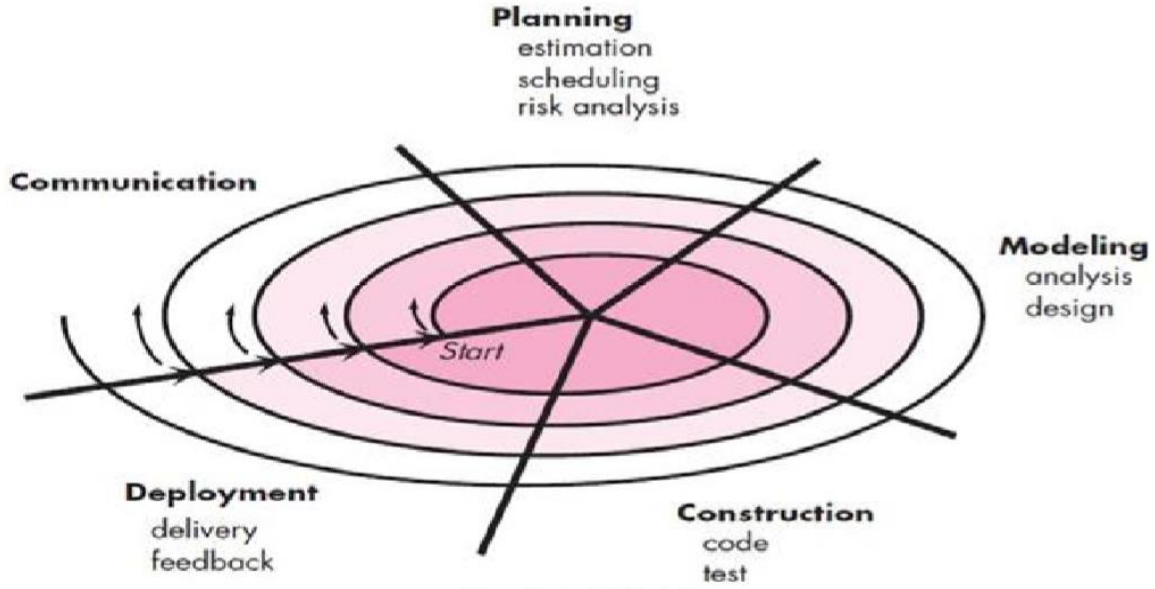



**Fig:** Spiral Model

Advantages:

 It is a realistic approach to the development of large scale systems and software. (Software evolves as the process progresses).

 It uses and enables the developer to apply the prototyping approach to any stage in evolution of product.

 Considers technical risks at all the stages of the project, and reduces risks before they become problematic.

 Like other paradigms, spiral model is not a panacea (Medicine).It demands considerable risk assessment expertise for success. If a major risk is not covered and managed, problems will occur.

## Prototype-Based Models:

Prototyping model is used when customer defines a set of objectives, but does not identify detailed input, processing output requirements, developer is unsure of efficiency of algorithm, adaptability of operating system etc, where phased model is inappropriate. Prototyping model can be used as a standalone process model. Prototyping paradigm assists the software engineer and customer to better understand what is to be built when requirements are fuzzy. Prototype helps to identify software requirements. Prototyping paradigm begins with communication, then quickly planning the prototyping iteration, modelling quick design, construction of prototype, and the prototype is deployed and then evaluated by the customer/user. Feedback is used to refine requirements for the software. Prototype can serve as "the first system", where users get a feel of actual system and developers get to build something immediately.

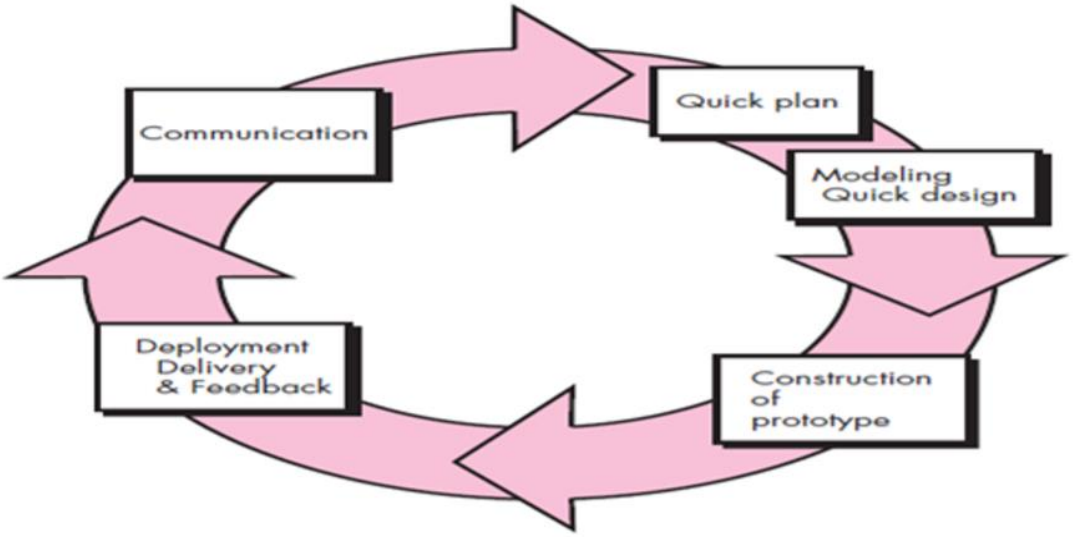



**Fig:** Prototyping Model

Prototyping can be problematic for following reasons:

 Developers may not consider overall software quality. (Few "fixes" are applied to satisfy customer).

 The developer often compromises in the implementation to get a prototype working quickly. The key is to define the rules of the game at the beginning; i.e., customer and developer must both agree that prototype is built to serve as a mechanism for defining requirements.

## Dynamic Systems Development Method:

This approach "provides a framework for building and maintaining systems which meet tight time constraints through the use of incremental prototyping in a controlled project environment". Ex: 80% of an application can be delivered in 20% of time it takes to deliver complete application. Like XP, ASD, DSDM also suggests an iterative software process. DSDM approach to each iteration follows 80% rule, where much of the detail can be completed when more business requirement/changes are known. DSDM Consortium is worldwide group of member companies, which uses DSDM approach. DSDM lifecycle defines 3 different iterative cycles, preceded by 2 additional life cycle activities.

1. Feasibility Study: Establishes business requirements and application constraints and then assesses whether the application is viable candidate for DSDM process.

2. Business Study: Establishes functional information requirements that allow the application to provide business value. Defines basic application architecture and identifies maintainability requirements for the application.

3. Functional Model Iteration: Produces a set of incremental prototypes that demonstrate functionality for the customer. It helps in gathering additional requirements from user feedback who exercises the prototype.

4. Design and Build Iteration: Revisits prototypes built during functional model iteration to ensure that they provide business value for end-users. Often occurs concurrently with Functional Model Iteration.

5. Implementation: Places latest software increment into operational environment. It should be noted that a) Increment may not be 100% complete. b) Changes may be requested as increment is put in place. In both cases, DSDM development work continues by returning to Functional Model Iteration activity.

 DSDM can be combined with XP to provide a combination approach that defines a solid process model.

# Discovery:

### Requirements Discovery phase:

Requirements discovery phase includes different techniques to be used by HCI designers to identify interaction requirements from the users of the HCI. As part of this requirements discovery, it is important to learn more about the user in the environment in which he or she would be using the interface. In order to learn about the user, it is important to identify the data collection methods that will be used to gather data that would reveal user's behaviours and preferences.

This data collected needs to be organized and transformed into requirements that can be used for the HCI design. This takes different techniques such as task analysis, use cases, primary stakeholder profile and storyboarding which will be covered in this lecture. The result of this phase is a requirements definition document that balances the needs of the user, the business, and the technical necessities.

**Methods of Collection:**

Data gathering is an important part of the requirements discovery process in interaction design. Data collection includes observation and elicitation methods. Observation methods allow the designer to immerse themselves in users' environment in their day-to-day activity by watching them but users don't participate directly with the HCI designer. On the other hand, elicitation methods require user's participation and include direct and indirect methods such as interviews, focus groups, and questionnaires.

**Direct Observation :**

This occurs when a field visit is conducted during the interaction design. Observation is a requirements discovery technique wherein the HCI designers either participates in or watches a person perform activities to learn about the interaction.

Before observation can be used in discovery, three minimum conditions set out by Tull and Hawkins (1993) need to be met:  The data has to be available for observation  The behaviour has to be repetitive, frequent, or otherwise predictable  An event has to cover a reasonably short time span.

Through observation, it is possible to describe what goes on, who or what is involved, when and where things happen, how they occur, and why things happen as they do in particular situations (Jorgensen 1989). A great deal of time is spent on paying attention, watching and listening carefully (Neuman 1994). The observer uses all the senses, noticing what is seen, heard, smelled, tasted and touched (Neuman 1994; Spradley 1979).

According to Neuman (1994), there are four possible research stances for the participant observer:

 Complete participant: the researcher operates under conditions of secret observation and full participation.

 Complete observer: the researcher is behind a one-way mirror or in an invisible role that permits undetected and unnoticed observation and eavesdropping.  Participant as observer: the researcher and members are aware of the research role, but the researcher is an intimate friend who is a pseudo member.

 Observer as participant: the researcher is a known, overt observer from the beginning, who has more limited or formal contact with members.

Whitten et. al (2000) suggested the following points when observing in the requirements discovery.

 Determine the who, what, where, when, why, and how of the observation.

 Obtain permission from appropriate supervisors or managers.

 Inform those who will be observed of the purpose of the observation.

 Keep a low profile.

 Take notes during or immediately following the observation.

 Review observation notes with appropriate individuals.

 Don't interrupt the individuals at work.

 Don't focus heavily on trivial activities.

 Don't make assumptions.

**Interviews:**

Interviews are a requirements discovery technique whereby the interaction designer collects information from individuals through face-to-face interaction.

Unstructured and open-ended interviews are with only a general goal or subject in mind and with few, if any, specific questions. The interviewer counts on the interviewee to provide a framework and direct the conversation.

The goal is to elicit the respondent‘s views and experiences in his or her own terms, rather than to collect data that are simply a choice among pre-established response categories (Anderson et al. 1994). Secondly, the interview is not bound to a rigid interview format or set of questions that would be difficult to establish given the nature of the research and will limit the results (Anderson et al. 1994).

In structured and close-ended interviews the interviewer has a specific set of questions to ask of the interviewee. Closed-ended questions restrict answers to either specific choices or short, direct responses.

Whitten et. al (2000) suggested the following list of activities that could be used when preparing interviews:

1. Select Interviewees

2. Prepare for the Interview with specific questions the interviewer will ask the interviewee.

3. Conduct the Interview

4. Follow Up on the Interview

Questions should not be leading or loaded. It is important to use consice and clear language and avoid any bias as an interviewer. Keep the questions short and to the point and avoid any intimidating questions.

**Focus Groups:**

Focus groups are a process whereby highly structured group meetings are conducted for the purpose of analyzing problems and defining requirements. Focus groups are a subset of a more comprehensive joint application development or JAD technique that encompasses the entire systems development process.

Focus groups require a facilitator role. Facilitators encourage user and management participation without allowing individuals to dominate the session. They make sure that attendees abide by the established ground rules for the session, encourage group consensus and keep the session on schedule.

Focus groups actively involve users in the interaction design and this improves their acceptance and reduces the risk of resistance at the implementation stage. They reduce the amount of time required to design interactions.

**Brainstorming:**

It is similar to focus group but more informal and use for generating ideas during group meetings. Participants are encouraged to generate as many ideas as possible in a short period of time without any analysis until all the ideas have been exhausted.

**Questionnaires :**

Questionnaires are special-purpose documents that allow the interaction designer to collect information and opinions from respondents. Questionnaires can be in a free or fixed format. Free-format questionnaires offer the respondent greater latitude in the answer. A question is asked, and the respondent records the answer in the space provided after the question. Fixed-format questionnaires contain questions that require selection of predefined responses from individuals and are normally composed of multiple-choice, rating or ranking questions.

Whitten et. al (2000) proposed the following activities when performing data collection with the use of questionnaires.

1. Determine what facts and opinions must be collected and from whom you should get them.

2. Based on the needed facts and opinions, determine whether free- or fixed-format questions will produce the best answers.

3. Write the questions.

4. Test the questions on a small sample of respondents.

5. Duplicate and distribute the questionnaire.

Data collected needs to be interpreted in order to identify the requirements for the design of the HCI. The following tools will be covered for data interpretation:

1. Task analysis

2. Ishikawa diagram

3. Use cases

4. Story boarding

5. Primary stakeholder profiles

**Task Analysis:**

A task is as a set of activities that change the system from an initial state to a specified goal or desired outcome state. The outcome may involve a significant change to the current state, so we split tasks into a sequence of subtasks, each more simple than the parent task. This process continues until the most primitive subtask is reached. This lowest level subtask is variously referred to as an action, simple task, or unit task. Task descriptions are often used to envision new systems or devices

Task analysis is used mainly to investigate an existing situation. It is used to determine functionality by distinguishing the tasks and subtasks performed. Particular attention is paid to frequent tasks, occasional tasks, exceptional tasks, and errors. Identifying goals and the strategies (combinations of tasks) used to reach those goals is also part of a good task analysis. By conducting a task analysis, the designer learns about the sequences of events that a user may experience in reaching a goal (Diaper 1989).

Rees et al. (2001) propped a list of activities in order to conduct task analysis and are described below:

1. Gathering information from observation of and/or consulting with users

2. Representing tasks in a task description notation

3. Performing an analysis of the task descriptions to achieve an optimum description

4. Using the task representation to produce a new user interface design or improve an existing one.

The most popular technique used for this type of analysis is the Hierarchical Task Analysis (HTA) tool. It involves breaking a task down into subtasks, then sub-sub-tasks and so on. These are grouped as plans which specify how the tasks might be performed in practice. It focuses on physical and observable actions and includes looking at actions not related to software or an interaction device. Start with a user goal which is examined and the main tasks for achieving it are identified.

In order to demonstrate the use of HTA, let‘s use an example of a task analysis for borrowing a book from the library. The set of tasks and subtasks for borrowing the book from the library is presented in figure 1.

1. Walk to the library

2. Search for the book

2.1 Access the library‘s catalogue computer system

2.2 Access the search book screen

2.3 Enter the in the search criteria the book title and author

2.4 Locate required book

2.5 Take note of the book‘s location

3. Walk to the book‘s location

4. Take the book and walk to the checkout counter

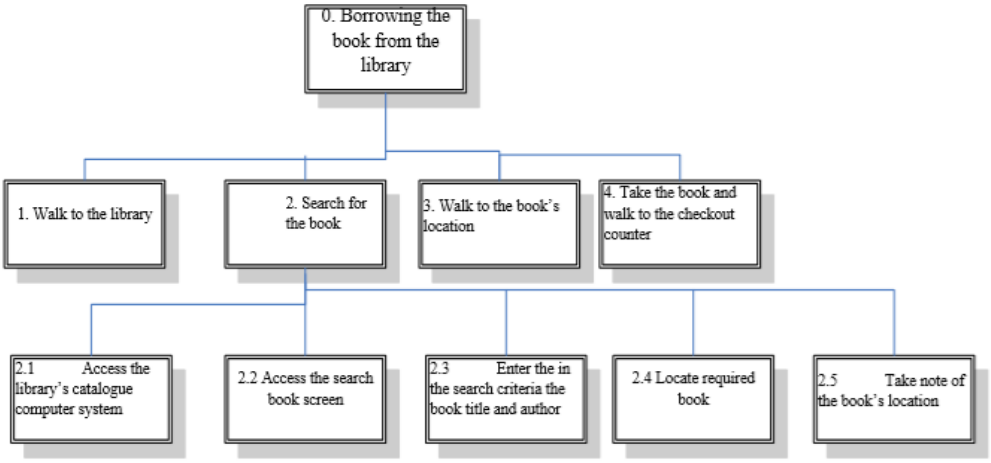



Figure 2 Graphical HTA for borrowing the book from the library

**Ishikawa diagram:**

The Ishikawa diagram is a graphical tool used to identify, explore, and depict problems and the causes and effects of those problems. It is often referred to as a cause-and-effect diagram or a fishbone diagram. This tool is used by designers in order to identify problems with the interaction that could be tackled with the new design.

**Use Cases:**

One of the most popular and successful approaches for documenting business processes, events and responses is a technique called use cases developed by Dr. Ivar Jacobson (Jacobson et al. 1993). Use cases describe the business process, and document how the business works and the business goals of each interaction with the system. These use cases are then extended to show how the human interaction will support the business goals.

The interactions within the use case should be contained, initiated and seen through to completion by an actor. The use case should further result in achieving a business goal and leaving the system in a stable state (Reed 2002). The nature of a use case is to define the "what" of a system.

An actor represents anything that needs to interact with the system to exchange information. An actor is a user, a role, which could be an external system as well as a person.

Benefits of use cases are highlighted by Witthen et. al (2000):

• Facilitates user involvement.

• A view of the desired human interaction‘s functionality from an external person‘s viewpoint.

• An effective tool for validating requirements.

• An effective communication tool.

**Storyboards:**

Movies studios create storyboards that show the various scenes of a potential film, particularly an animated film. A storyboard puts ideas on paper and then puts the papers in a certain sequence to provide a concept of how the film will play out. It also gives the production team an opportunity to look at the concept and make suggestions for improving the film before it takes its final form.

In user interface design, there is a more interactive version of storyboarding called paper prototyping. Paper prototyping involves creating a paper version of a software program, hardware product, or Web site so you can learn how users interact with the design before

you develop the product. This paper prototype involves using a series of images that can be animated, used to describe a work flow for a human computer interaction. They can facilitate the process of task decomposition and the identification of interface requirements since they help to visualize existing work flows.

**Primary Stakeholder Profiles:**

The HCI design includes four distinct stakeholder groups (Donoghue, 2002):

- Users
- Engineers and designers
- Sales and marketing personnel
- Managers

Users are the primary stakeholders since the use the design directly. Engineers and designers are secondary stakeholders since they supply input or receive output for the design. Managers are facilitators since they help to maintain the design. Sales and marketing personal are indirect stakeholders since they are affected by the design but they don‘t have direct contact with it. Users expect to have a successful experience with the user interface (UI) the first time around. Because the users are the people who determine whether something is useful, the characteristics of your users will go a long way toward determining what is actually usable. However, users look for some general goals when they use an interface (Donoghue, 2002):

- The UI must be easy to learn.
- The UI must solve the user's needs.
- The UI help must be both easily accessible and effective in resolving the user's problem quickly.

Defining a user‘s profile is an essential prerequisite for designing a HCI. The user profile will influence the design and evaluation of an interface. Badre (2002) suggested the following activities to generate a user profile:

1. Identify the relevant individual differences.

2. Identify and specialize the cognitive processing capabilities and limits.

3. Generate audience definition and categorization.

Individual differences can be grouped into four categories (Badre 2002):

1. Knowledge, experience, and skill: Individual users may differ in level and type of education as well as in their knowledge, experience, and skill levels. There are several key

factors in this category that are determinants of a user's performance. These factors describe the cognitive abilities and styles of projected users as well as their knowledge and experience of the projected HCI‘s domain.

2. Personality factors: Affect the ease of user acceptance for interacting with and navigating a HCI. Such attributes as tolerance level and motivation should indicate how much time users will spend trying to use the new HCI to perform a transaction before giving up.

3. Physical and demographic attributes: Demographic attributes with implications for the design of a HCI are age, gender, salary, and mobility. An audience definition statement should take into account factors related to physical capabilities and limitations. Issues that might affect design include the use of glasses (near-sightedness, bifocals), left- and right-handedness, auditory devices, and other visual and motor aids.

4. User levels: The designer should take into consideration the users' varying levels of expertise with the computing environment used for the HCI. Level can range from novice to master level.

The usability effectiveness of designs depends in great part on their compatibility with the user's information-processing capabilities and limitations. Designers must take into account the users' cognitive and perceptual limits and how people are likely to process information in an interactive environment. There are some basic universal human information-processing characteristics, which affect the way people store, remember, and manipulate information, which in turn have implications for HCI design. For example, if the user has selective attention, this means both creating designs that draw user attention to a particular screen location and optimizing the ease of locating displayed information such as using a unique bright color to draw attention to a displayed link can increase the chances that it will be noticed before other links.

Generating an audience profile means generating a document that specifies the relevant characteristics, the range and frequency values of the identified characteristics, and how this specified information might impact design decisions.

User profiles can be also seen from the marketing point of view. Eisenberg and Eisenberg (2006) discussed the creation of primary stakeholder profiles in terms of marketing, which essentially means to persuade the user that the interface is worth using. Profiles connect three different dimensions of information:

 Demographics: This segments some of the persona features. For example, demographic data shows such data as the user's gender, location, and income.

 Psychographics: This segments some of the persona needs and determines questions that each persona may ask. For example, a spontaneous type and a competitive type will ask different questions and will want different types of information.

 Topology: This allows you to segment by determining how complex the persuasion process is; that complexity is based on a customer's perceptions and experiences.

Regarding the topology dimension, Eisenberg and Eisenberg mapped a four-dimension model for the process of persuasion in sales:

 Need: This is the urgency that a user feels for a product or service.

 Risk: This is the amount of risk the user is willing to accept regarding such features as a career or self-esteem.

 Knowledge: This is how much knowledge the user has about the product, which can affect need and risk. For example, if someone feels he doesn't have enough information about a product or service, the risk factor for that user is higher.

 Consensus: This is the understanding during the persuasion process of how many people need to be convinced and when.

**Marketing analysis and existing competition:**

Marketing plays an important role when defining HCI requirements. Designers need to take into consideration competition and existing technologies when defining requirements in order to build competitive interactions. Users are driving the marketing and acceptance of user interfaces therefore it is important to make every effort to find out market demographics. A marketing analysis for defining requirements of the HCI can be done by using the techniques suggested by Cooper and Reimann (2003):

 Reviews of competing products.

 Reviews of market research, such as computing media Web sites and technology white papers.

 Researching market demographics in the area that the HCI will be used. That research can include analyzing demographic, geographic, or behavioral variables to see if any patterns emerge.

**Documenting HCI requirements:**

Requirements need to be documented after they are discovered by the HCI designer. A requirements definition document should consist of the following according to Whitten et al. (2000):

1. The functions and services the system should provide.

2. Nonfunctional requirements including the system‘s features, characteristics, and attributes.

3. The constraints that restrict the development of the system or under which the system must operate.

4. Information about other systems the system must interface with.

Heim (2007) proposes that the formal discovery documentation should include:

1. Mission Statement

2. Requirements Document

3. Project Management Document

4. Usability Guidelines.

# Design:

## Conceptual Design:

After the requirements discovery phase is completed, the design phase uses the requirements in order to create conceptual and physical designs.

Conceptual design is the creation of alternative design ideas. At this stage, a description of the proposed system is presented in terms of a set of integrated ideas and concepts about what it should do, behave and look like.

The conceptual design follows requirements analysis but precedes the physical design. It helps to determine functions, operations and features and the workflow of the interface. It also allows the designer to identify inputs and outputs and the style of the interaction.

Several tools for conceptual design will be introduced here including brainstorming, card sort, personas, scenarios, semantic networks and cognitive walkthroughs.

## Brainstorming:

These are sessions with your project team that will help to uncover ideas that your team can implement now or sometime later. Brainstorming sessions should be centered on a topic. For example, a brainstorming session can center on what user interface elements will meet specific goals. (Cooper & Reimann 2003)

Participants are encouraged to generate as many ideas as possible in a short period of time without any analysis until all the ideas have been exhausted.

## Card sort:

Card sorting is normally performed with the use of small paper cards that are usually lined. Each card represents one screen and these cards can be shown easily on a table or the wall in order to represent multiple screens. Thread or lines within the cards indicate sequences

or hyperlinks. They are used often in web design but can also be used in any design that involves multiple screens.

Card sorting is an excellent way to explore work flow without having to create a detailed screen design. However, they have limitations since it is hard to explore elements that are not included in the cards such as icons or other interface elements.

**Scenarios:**

Scenarios can be used to express proposed or imagined situations that describe the different tasks a person would need to do in order to accomplish a specific goal with the design. They are used throughout the design process as scripts for user evaluation of prototypes and for co-operation among designers involved in an interaction project.

In creating such scenarios, a designer specifies interface objects and actions for given contexts from the perspective of the user (Badre 203).

To create successful scenarios, Badre (2002) suggested to answer the following questions.

1. Where and under what conditions will the system be used?

2. For what purpose will the system be used?

3. Who will use the system (the target audience)?

4. How will the system be used?

Scenarios provide a fast and effective way to imagine the design concepts in use. They are simple stories about what it would be like to use the interface once it has been made and the protagonists of these stories are the personas (Saffer 2006).

**Semantic Networks:**

Semantic networks are an excellent way to represent concepts associations in interaction design. Rees et. al (2001) mentioned that the human thought is basically nonlinear and that the human learning and perceptual process is essentially organized as a semantic network in which concepts are linked together by associations. In few words, human learn and remember through nonlinear associations. The fundamental components of a semantic network are:

 Nodes: Representing concepts

 Links: Representing the relationships between nodes.

In a semantic network a node is defined as a single concept or idea. Nodes can be virtually any kind of information such as text, graphics, animation, audio, video, images, programs, and so on. Nodes can be "typed," indicating how they are used. For example, if a node in a

Web-based hypertext system is designated as the "home page," there is the implication that that node will be used in a specific way to traverse that system (i.e., it is the node where readers will begin).

Nodes are connected to other nodes with links. The role of a link is to connect related concepts or nodes. Links are bidirectional, meaning that a reader can go backwards and forwards.

Like nodes, links can also be "typed," illustrating features of the relationship of the nodes they connect. For example, a link might reflect some relationship between nodes such as parent–child. (Rees et. al 2001).

**Cognitive walkthrough:**

Cognitive walkthrough is a technique that involves a group of users that evaluates a human interaction by going through a set of tasks. The user interface is presented as a paper prototype to the evaluators and they follow the various scenarios of the interaction. The input to the walkthrough includes the user profile, especially the users' knowledge of the task domain and of the interface, and the task cases. The evaluators may include human factors engineers, software developers, or people from marketing, documentation, etc. . Zhang (2008) identifies the following questions required to design the walkthrough:

 Who will be the users of the system?

 What task(s) will be analyzed?

 What is the correct action sequence for each task?

 How is the interface defined?

Dix et. al (1998) mention that for each task in the walkthrough the evaluator should consider:

 What impact will interaction have on user?

 What cognitive processes are required?

 What learning problems may occur?

 Does the design lead the user to generate the correct goals?

**Personas:**

Personas are profiles of the users that interact with a HCI These profiles describe user characteristics such as user expertise, user motivation, user job functions and the impact of the interaction in the user‘s job. They help the designer to understand who will be using the interaction in order meet user expectations.

Personas can be created by observing and talking to users. Personas don‘t have to be for a specific individual but for a set of people that share the same goals. Personas should identify each persona's desires and the expectations, behaviours, attitudes, biases, and other factors that affect them.

A set questions that could be helpful to create a persona are identified by Hackos and Redish (1998) as follows:

 Do you like your work? Why or why not?

 What motivates you in your job?

 Is the database product related directly to your primary job? That is, do you need to use this product every day to get your job done?

 How much do you know about the subject matter you're designing for?

 What technical skills and job knowledge do you bring to the job?

 How do you approach technology? Do you love it or put up with it?

 Do you prefer learning from written documentation, or do you prefer online help? What do you think of the documentation of the current system?

 Do you like new experiences, or do you think if it isn't broken you shouldn't fix it?

Personas might use construct context scenarios that describe personas and their activities in a typical day using the new and improved system, which includes the new user interface. The scenarios don't discuss the form and function, but only the behaviours of the user and the interface (Buttow 2007).

**Interaction design models:**

Interaction design models are useful for analyzing and understanding interface design. They are used to test designs that might be hard to test with real users and prototypes. They can also be used to document designs. Interaction design models can be predictive or descriptive. Predictive models can be used to simulate user actions in order to test a design. Descriptive models are mainly use to document designs and to visualize its logic and behaviour.

**GOMS:**

The model of goals, operators, methods, and selection rules (GOMS) (Card, Moran, and Newell 1983) allows to predict how long an experienced worker will take to perform a particular operation when using a given interface design.

The significance of these model components is:

 Goals: Symbolic structures that define states to be achieved, and determine a set of possible methods.

 Operators: Elementary motor or information processing acts, whose execution is necessary to change any aspect of the user's memory or to affect the task environment.

 Methods: Descriptions of procedures for accomplishing a goal, cast as a continual sequence of sub goals and operators, with conditional tests on the user's immediate memory and on the task state.

 Selection rules: Elements that allow a choice between two or more methods that can be used to accomplish the goal.

The GOMS models describes the interaction as a sequence of small, discrete subtasks or unit tasks.

**Keystroke-Level Model:**

The Keystroke-Level Model is a simplified version of GOMS and was proposed by Card, Moran & Newell (1980) as a method for predicting user performance. The model is based on the concept that the time that it takes the user-computer system to perform a task is the sum of the times it takes for the system to perform the serial elementary gestures that the task comprises. Although different users might have widely varying times, the researchers found that for many comparative analyses of tasks involving use of a keyboard and a graphical input device, it is possible to use a set of typical times rather than measuring the times of individuals. By means of careful laboratory experiments, Card, Moran and Newell (1983) developed a set of timings for different operations. The model includes six operations including keystroking, pointing, homing, mental preparation and response.