Introduction to software design

BY:Yibeltal A.

2012 E.C

introduction

- Software design is an iterative process through which requirements are translated into a blueprint for constructing the software.
- Initially, the blueprint depicts a holistic view of software. During the design process the software specifications are transformed into design models.
- Models describe the details of the data structures, system architecture, interface, and components.
- Each design product is reviewed for quality before moving to the next phase of software development. At the end of the design process a design model and specification document is produced.

Cont….

- Software design is both a process and a model.
- The design process is a sequence of steps that enables the designer to describe all aspects of the software for building.
- Creative skill, past experience, a sense of what makes "good" software and an overall commitment to quality are examples of critical success factors for a competent design.
- It begins by representing the totality of the thing that is to be built (e.g., a three-dimensional rendering of the house); slowly, the thing is refined to provide guidance for constructing each detail

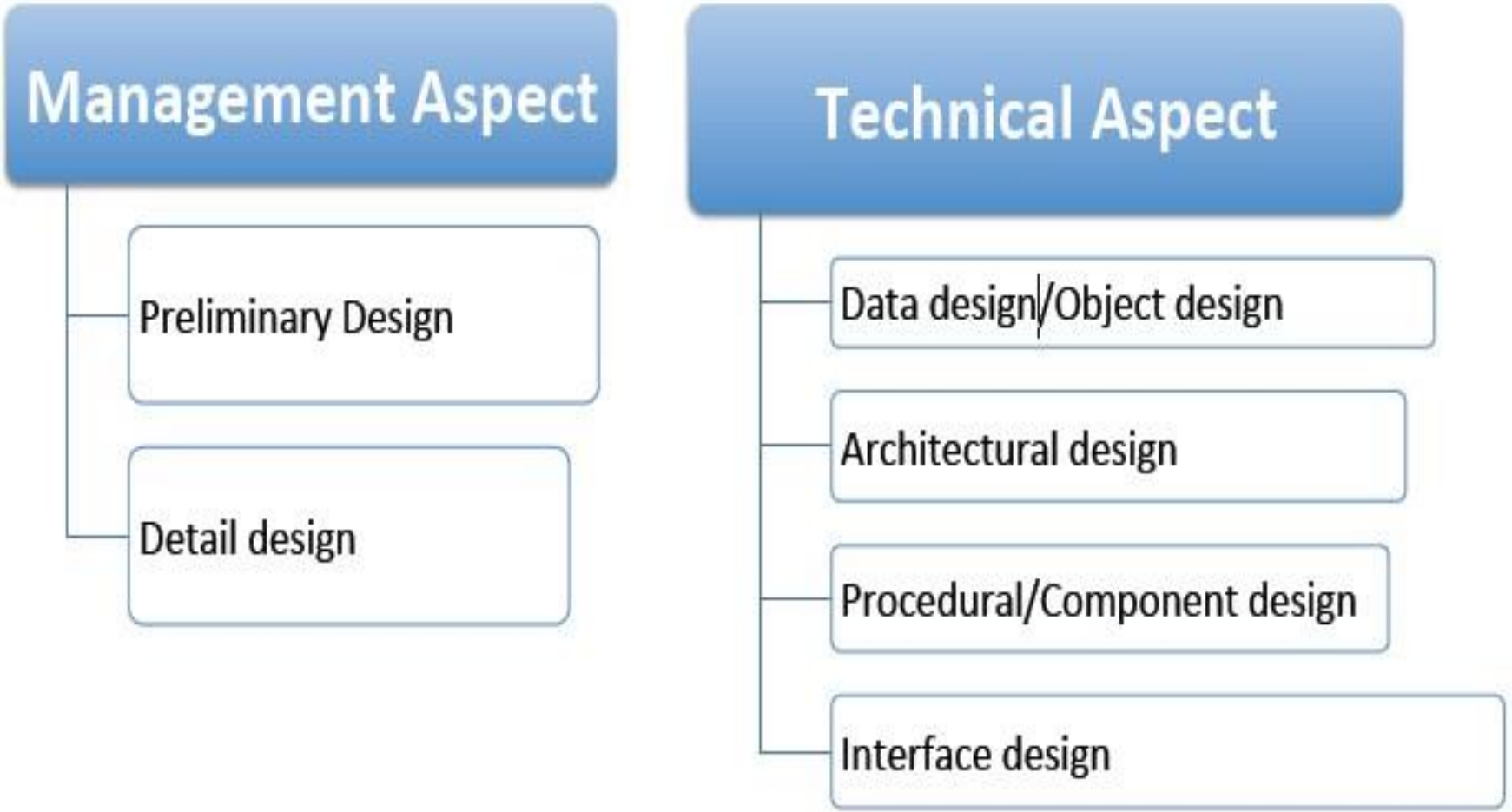
Cont…

- Generally, design is about **how** to build a system (answers **“HOW?”**)- Specifying different parts of the software, and how they will fit together.
- From a project management point of view, software design can be conducted in two main steps:
  - **Preliminary Design**: Concerned with the transformation of requirements into data and software architecture.
  - **Detail Design**: Focuses on refining the architectural representation, and lead to detailed data structure and algorithmic representations of software.
- Three main design activities in the design phase are:
  - Data design,
  - Architectural design and
  - Procedural design. In addition, many modern applications have a distinct **interface design** activity. Interface design establishes the layout and interaction mechanisms for human-machine interaction.

Cont….

- **Data Design /Class Design** : Objective of Data Design is to transform the information domain into data structures. Eg: Entity Relationship diagrams, Data dictionary, abstract data types and Data structures.
- **Architectural Design**: Identify the overall structure of the system, the principal components (sometimes called sub-systems or modules), their relationships and how they are distributed.
- **Procedural/Component Design**: Objective is to transform structural components into a procedural/component description of the software. This step occurs after the data and program structures have been established, i.e. after architectural design

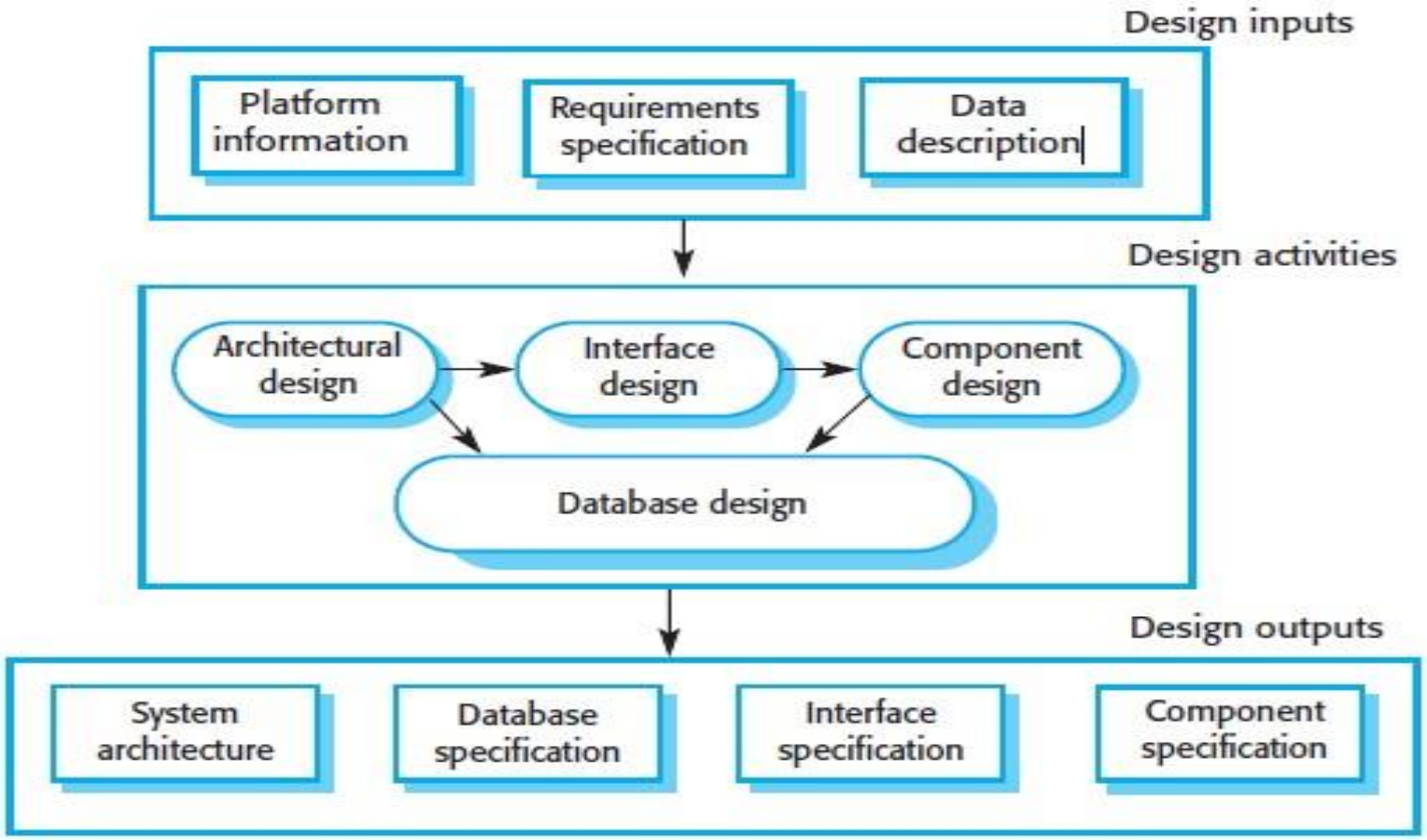
Cont…


Management Aspect
Preliminary Design
Detail design
Technical Aspect
Data design/Object design
Architectural design
Procedural/Component design
Interface design

Design process

- Any design may be modeled as a directed graph made up of entities with attributes which participate in relationships.
- The system should be described at several different levels of abstraction. Design takes place in overlapping stages.
- It is artificial to separate it into distinct phases but some separation is usually necessary.

Cont..


Design inputs
Platform information
Requirements specification
Data description
Design activities
Architectural design
Interface design
Component design
Database design
Design outputs
System architecture
Database specification
Interface specification
Component specification

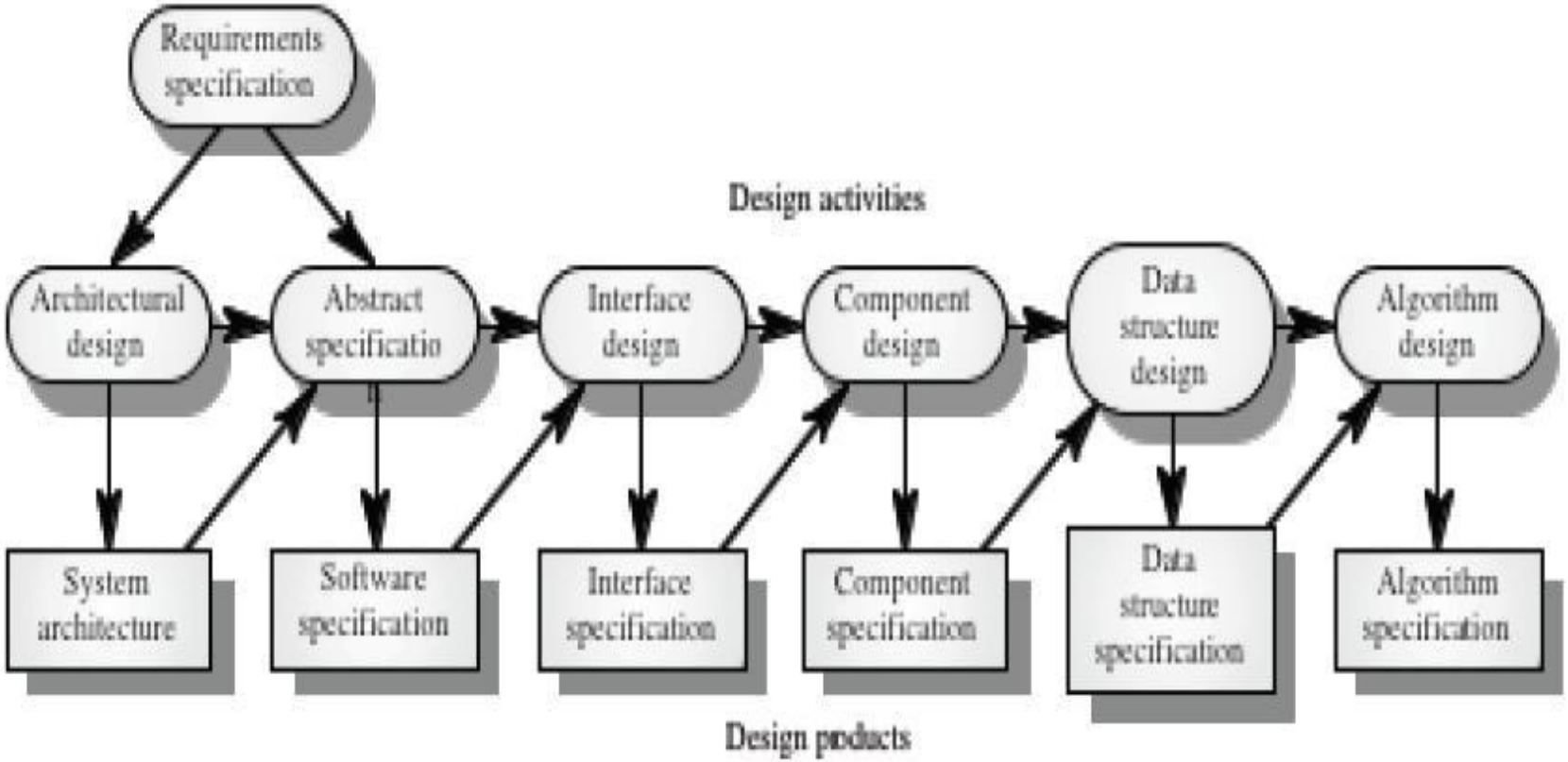
Cont…

Phases in the Design Process

- **Architectural design**- Identify sub-systems
- **Abstract specification** -Specify sub-systems
- **Interface design** -describe sub-system interfaces
- **Component design** -decompose sub-systems into components
- **Data structure design** -design data structures to hold problem data
- **Algorithm design** -design algorithms for problem functions

Cont….

- Design activities and product



Stages of Design

- **Problem understanding**–Look at the problem from different angles to discover the design requirements.
- **Identify one or more solutions**–Evaluate possible solutions and choose the most appropriate depending on the designer's experience and available resources.
- **Describe solution abstractions**– Use graphical, formal or other descriptive notations to describe the components of the design.
- Repeat process for each identified abstraction until the design is expressed in primitive terms.

Design strategies

- **Functional Design:** The system is designed from a functional viewpoint.
  - The system state is centralized and shared between the functions operating on that state.
- **Object-Oriented Design:** The system is viewed as a collection of interacting objects.
  - The system state is decentralized and each object manages its own state.
  - Objects may be instances of an object class and communicate by exchanging messages.

Objective of design

- The design must implement all of the explicit requirements contained in the analysis model, and it must accommodate all of the implicit requirements desired by the customer.
- It must be a readable, understandable guide for those who generate code and for those who test and subsequently support the software.
- The design should also provide a complete picture of the software, addressing the data, functional, and behavioral domains from an implementation perspective.

Cont.…

- Good design leads to software that is:
  - **Correct**–does what it should.
  - **Robust**– tolerant of misuse, e.g. faulty data,
  - **Flexible**–adaptable to shifting requirements,
  - **Reusable**–cut production costs for code,
  - **Efficient**–good use of processor and memory,
  - also should be **Reliable** and **Usable.**

Design Principles

- Design principles enable the software engineer to navigate the design process. which have been adapted and extended in the following list:
  - **The design process should not suffer from "tunnel vision."** A good designer should consider alternative approaches, judging each based on the requirements of the problem, the resources available to do the job.
  - **The design should be traceable to the analysis model.** Because a single element of the design model can often be traced back to multiple requirements, it is necessary to have a means for tracking how requirements have been satisfied by the design model.
  - **The design should not reinvent the wheel.** Systems are constructed using a set of design patterns, many of which have likely been encountered before. These patterns should always be chosen as an alternative to reinvention.

Cont…

- **The design should "minimize the intellectual distance" between the software and the problem as it exists in the real world.** That is, the structure of the software design should, whenever possible, mimic the structure of the problem domain.
- **The design should exhibit uniformity and integration.** A design is uniform if it appears fully coherent. In order to achieve this outcome, rules of style and format should be defined for a design team before design work begins
- **The design should be structured to accommodate change.** The design concepts discussed in the next section enable a design to achieve this principle.

Cont….

- **The design should be structured to degrade gently, even when aberrant data, events, or operating conditions are encountered.** Well- designed software should never "bomb"; it should be designed to accommodate unusual circumstances, and if it must terminate processing, it should do so in a graceful manner.
- **Design is not coding, coding is not design.** Even when detailed procedural designs are created for program components, the level of abstraction of the design model is higher than the source code. The only design decisions made at the coding level should address the small implementation details that enable the procedural design to be coded.

Cont…

- **The design should be assessed for quality as it is being created, not after the fact.** A variety of design concepts and design measures are available to assist the designer in assessing quality throughout the development process.
- **The design should be reviewed to minimize conceptual (semantic) errors.** There is sometimes a tendency to focus on minutiae when the design is reviewed, missing the forest for the trees. A design team should ensure that major conceptual elements of the design (omissions, ambiguity, in consistency) have been addressed before worrying about the syntax of the design model.

Design Considerations

- There are many aspects to consider in the design of a piece of software. The importance of each consideration should reflect the goals and expectations that the software is being created to meet. Some of these aspects are:
- **Compatibility** - The software is able to operate with other products that are designed for interoperability with another product. For example, a piece of software may be backward compatible with an older version of itself.
- **Extensibility** - New capabilities can be added to the software without major changes to the underlying architecture.

Cont….

- **Modularity** - the resulting software comprises a well defined, independent component which leads to better maintainability. The components could be then implemented and tested in isolation before being integrated to form a desired software system. This allows division of work in a software development project.
- **Fault-tolerance** - The software is resistant to and able to recover from component failure.
- **Maintainability** - A measure of how easily bug fixes or functional modifications can be accomplished. High maintainability can be the product of modularity and extensibility.
- **Reliability (Software durability)** - The software is able to perform a required function under stated conditions for a specified period of time.

Cont…

- **Reusability** - The ability to use some or all of the aspects of the preexisting software in other projects with little to no modification.
- **Robustness** - The software is able to operate under stress or tolerate unpredictable or invalid input. For example, it can be designed with a resilience to low memory conditions.
- **Security** - The software is able to withstand and resist hostile acts and influences.
- **Usability** - The software user interface must be usable for its target user/audience. Default values for the parameters must be chosen so that they are a good choice for the majority of the users.

Cont..

- **Portability** - The software should be usable across a number of different conditions and environments.
- **Scalability** - The software adapts well to increasing data or number of users.
- **Performance** - The software performs its tasks within a time-frame that is acceptable for the user, and does not require too much memory.

User Interface Design

- User interface is the first impression of a software system from the user's point of view. Therefore any software system must satisfy the requirement of user.
- **Functions and Features of UI**
  - User Interface (UI) mainly performs two functions −
    - Accepting the user's input
    - Displaying the output

Cont…

- User interface plays a crucial role in any software system. It is possibly the only visible aspect of a software system as:
  - Users will initially see the architecture of software system's external user interface without considering its internal architecture.
  - A good user interface must attract the user to use the software system without mistakes. It should help the user to understand the software system easily without misleading information. A bad UI may cause market failure against the competition of software system.
  - UI has its syntax and semantics. The syntax comprises component types such as textual, icon, button etc. and usability summarizes the semantics of UI. There are basically two major kinds of user interface – a) **Textual** b) **Graphical**.
  - Software in different domains may require different style of its user interface

Graphical User Interface

- A graphical user interface is the most common type of user interface available today.
- It is a very user friendly because it makes use of pictures, graphics, and icons hence why it is called 'graphical'.
- It is also known as a WIMP interface because it makes use of −
  - **Windows** − A rectangular area on the screen where the commonly used applications run.
  - **Icons** − A picture or symbol which is used to represent a software application or hardware device.
  - **Menus** − A list of options from which the user can choose what they require.
  - **Pointers** − A symbol such as an arrow which moves around the screen as user moves the mouse. It helps user to select objects.

Elements of UI

- To perform user interface analysis, the practitioner needs to study and understand four elements:
  - The users who will interact with the system through the interface
  - The tasks that end users must perform to do their work
  - The content that is presented as part of the interface
  - The work environment in which these tasks will be conducted

Levels of UI Design

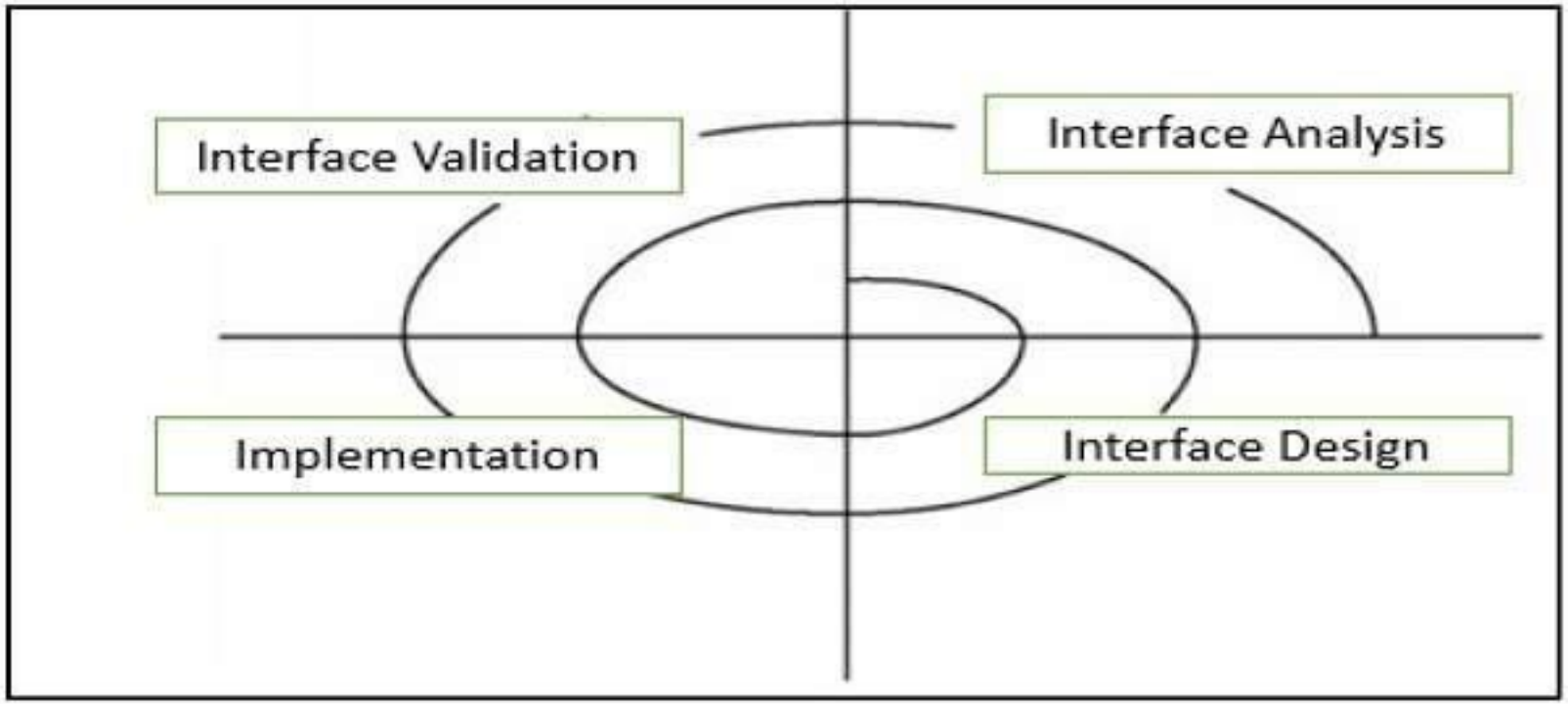
- The task to be performed and then can be divided, which are assigned to the user or machine, based on knowledge of the capabilities and limitations of each.
- The design of a user interface is often divided into four different levels.
  - **The conceptual level** – It describes the basic entities considering the user's view of the system and the actions possible upon them.
  - **The semantic level** – It describes the functions performed by the system i.e. description of the functional requirements of the system, but does not address how the user will invoke the functions.
  - **The syntactic level** – It describes the sequences of inputs and outputs required to invoke the functions described.
  - **The lexical level** – It determines how the inputs and outputs are actually formed from primitive hardware operations.

Steps of UI Design

- User interface design is an iterative process, where all the iteration explains and refines the information developed in the preceding steps.
- General steps for user interface design –
  - Defines user interface objects and actions (operations).
  - Defines events (user actions) that will cause the state of the user interface to change.
  - Indicates how the user interprets the state of the system from information provided through the interface.
  - Describe each interface state as it will actually look to the end user.

User Interface Development Process

- It follows a spiral process as shown in the following diagram:



Cont…

- **Interface Analysis:** It concentrates or focuses on users, tasks, content, and work environment who will interact with the system. It defines the human - and computer-oriented tasks that are required to achieve system function.
- **Interface Design:** It defines a set of interface objects, actions, and their screen representations that enable a user to perform all defined tasks in a manner that meets every usability objective defined for the system.
- **Interface Construction:** It starts with a prototype that enables usage scenarios to be evaluated and continues with development tools to complete the construction.

Cont…

- **Interface Validation:** It focuses on the ability of the interface to implement every user task correctly, accommodate all task variations, to achieve all general user requirements, and the degree to which the interface is easy to use and easy to learn.

User Interface Models

- When a user interface is analyzed and designed following four models are used:
  - **User Profile Model:** Created by a user or software engineer, which establishes the profile of the end-users of the system based on age, gender, physical abilities, education, motivation, goals, and personality. It Considers syntactic and semantic knowledge of the user and classifies users as novices, knowledgeable intermittent and knowledgeable frequent users.
  - **Design Model:** Created by a software engineer which incorporates data, architectural, interface, and procedural representations of the software. It is derived from the analysis model of the requirements and controlled by the information in the requirements specification which helps in defining the user of the system.

Cont…

- **Implementation Model:** Created by the software implementers who work on look and feel of the interface combined with all supporting information (books, videos, help files) that describes system syntax and semantics.
- It Serves as a translation of the design model and attempts to agree with the user's mental model so that users then feel comfortable with the software and use it effectively.
- **User's Mental Model:** Created by the user when interacting with the application. It contains the image of the system that users carry in their heads.
- Often called the user's system perception and correctness of the description depends upon the user’s profile and overall familiarity with the software in the application domain.

Design Considerations of User Interface

- **User Centered:** A user interface must be a user-centered product which involves users throughout a product's development lifecycle. The prototype of a user interface should be available to users and feedback from users, should be incorporated into the final product.
- **Simple and Intuitive:** UI provides simplicity and intuitiveness so that it can be used quickly and effectively without instructions. GUI is better than textual UI, as GUI consists of menus, windows, and buttons and is operated by simply using mouse.
- **Place Users in Control:** Do not force users to complete predefined sequences. Give them options—to cancel or to save and return to where they left off. It use terms throughout the interface that users can understand, rather than system or developer terms.

Cont..

- **Transparency:** UI must be transparent that helps users to feel like they are reaching right through computer and directly manipulating the objects they are working with. The interface can be made transparent by giving users work objects rather than system objects
- **Use Progressive Disclosure:** Always provide easy access to common features and frequently used actions. Hide less common features and actions and allow users to navigate them. Do not try to put every piece of information in one main window. Use secondary window for information that is not key information.
- **Consistency:** UI maintains the consistency within and across product, keep interaction results the same, UI commands and menus should have the same format, command punctuations should be similar and parameters should be passed to all commands in the same way.

Cont….

- **Integration:** The software system should integrate smoothly with other applications such as MS notepad and MS-Office. It can use Clipboard commands directly to perform data interchange.
- **Component Oriented:** UI design must be modular and incorporate component oriented architecture so that the design of UI will have the same requirements as the design of the main body of the software system.
- **Customizable:** The architecture of whole software system incorporates plug-in modules, which allow many different people independently extend the software. It allows individual users to select from various available forms in order to suit personal preferences and needs.
- **Reduce Users' Memory Load:** Do not force users to have to remember and repeat what the computer should be doing for them.
- **Separation:** UI must be separated from the logic of the system through its implementation for increasing reusability and maintainability.

Thanks