Information System Design

IT60105

Lecture 19

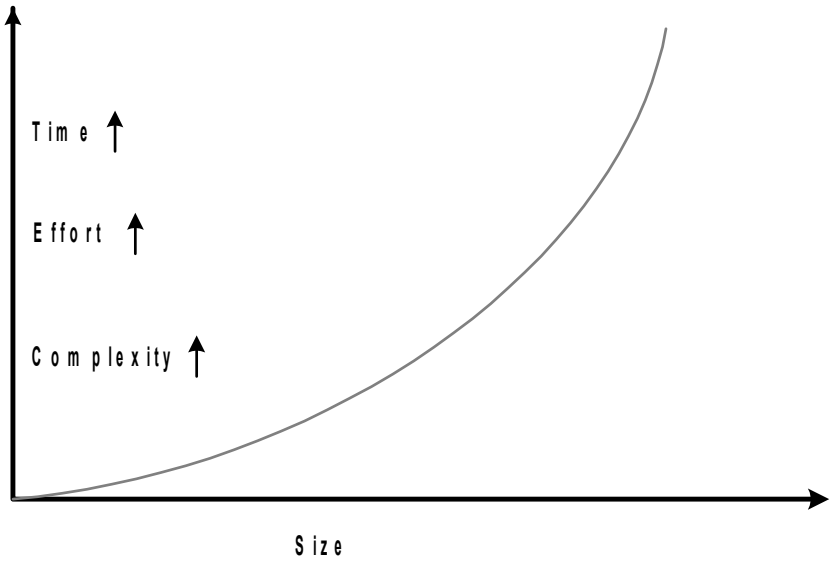
Project Planning

Lecture #19

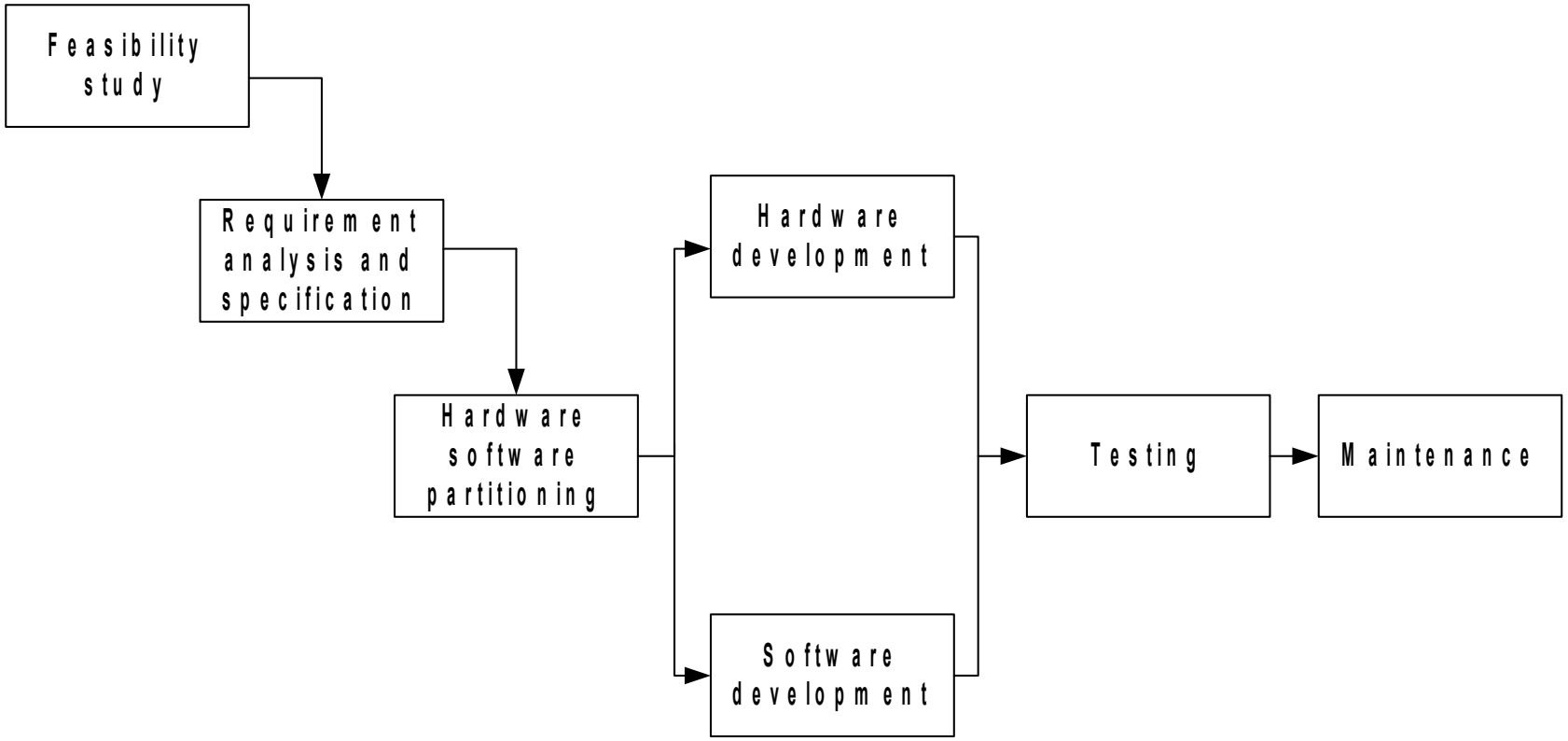
- ISD Project Planning
- SPMP Documentation
- System Design Models

Why Planning Information System Design?

- Program vs. Software
  - Size (small/big)
  - Effort (Individual/Team)
  - Application and user
  - Reliability (testing)
  - Documentation (manual)
- Technological growth
  - Hardware
  - Software
  - Communication
- IT based society



ISD Engineering



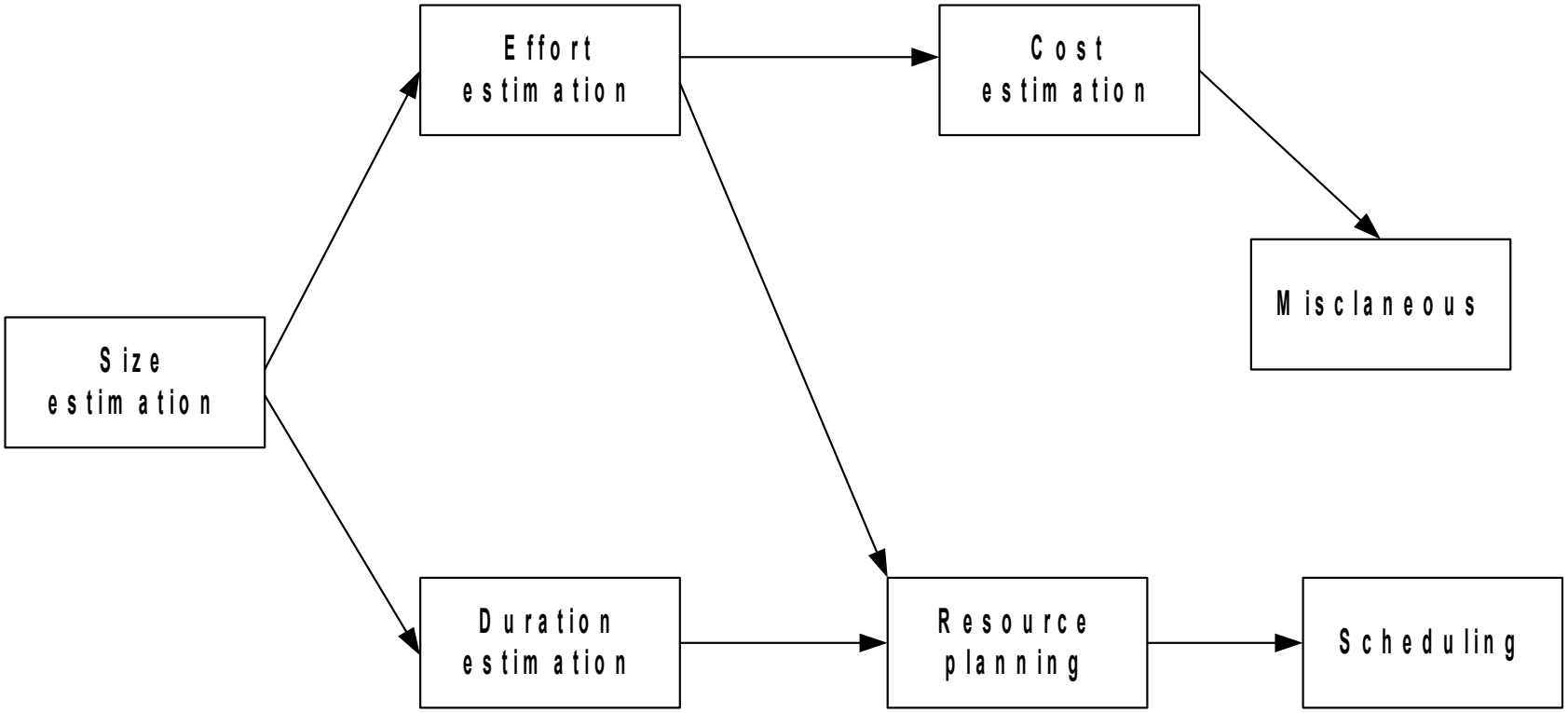
Project Management

- Crucial things to the success of any project
  - Enables a team to work efficiently
  - Distribute work load
  - Pipelining the productivity
- Project planning
  - After the feasibility study (FS) and requirement analysis and specification (RAS)
- Responsibility of the Project Manager

Activities in Project Planning

- Estimating some basic attributes of project
  - Cost
  - Duration
  - Effort
- Scheduling manpower and other resources
- Staff organization and staffing plans
- Risk identification, analysis, and abatement planning
- Quality assurance plan, configuration management planning

Activities in Project Planning



Software Project Management Plan

- SPMP documentation
- Items to be documented:
  1. Introduction
  2. Project Estimates
  3. Schedule
  4. Project Resources
  5. Staff organization
  6. Risk Management Plan
  7. Project Tracking and Control Plan
  8. Miscellaneous plan

SPMP Document Template

1. Introduction
   (a) Objective
   (b) Major Functions
   (c) Performance Issues
   (d) Management and Technical Constraint

2. Project Estimates
   (a) Historical Data Used
   (b) Estimation Techniques Used
   (c) Effort, Resource Cost, and Duration Estimation

SPMP Document Template

3. Schedule
   (a) Work Breakdown Structure
   (b) Task Network Representation
   (c) Gantt Chart Representation
   (d) PERT Chart Representation

4. Project Resources
   (a) People
   (b) Hardware
   (c) Software
   (d) Communication Support

SPMP Document Template

5. Staff organization
   (a) Team Structure
   (b) Management Reporting

6. Risk Management Plan
   (a) Risk Analysis
   (b) Risk Identification
   (c) Risk Estimation
   (d) Risk Abatement Procedures

7. Project Tracking and Control Plan

SPMP Document Template

8. Miscellaneous plan
   (a) Process Tailoring
   (b) Quality Assurance Plan
   (c) Configuration Management Plan
   (d) Validation and Verification
   (e) System Testing Plan
   (f) Delivery, Installation and Maintenance Plan

System Design Models

System Design Models

- **System Design Life Cycles**
  - Waterfall Models
    - Classical Waterfall Model
    - Iterative Waterfall Model
  - Prototyping Model
  - Evolutionary Model
  - Spiral Model
- **Rapid Application Development**
- **Component-based Software Engineering**

? System Design Life Cycle

- A system design life cycle is a **process** consists of set of **methodologies** (**activities**) beginning from system inception through delivery and retirement
- Basic activities in a system design life cycle are:
  - Feasibility study
  - Requirement analysis and specification
  - Design
  - Coding
  - Testing
  - Maintenance
- An *activity* is also alternatively termed as a *life cycle phase*

? System Design Life Cycle

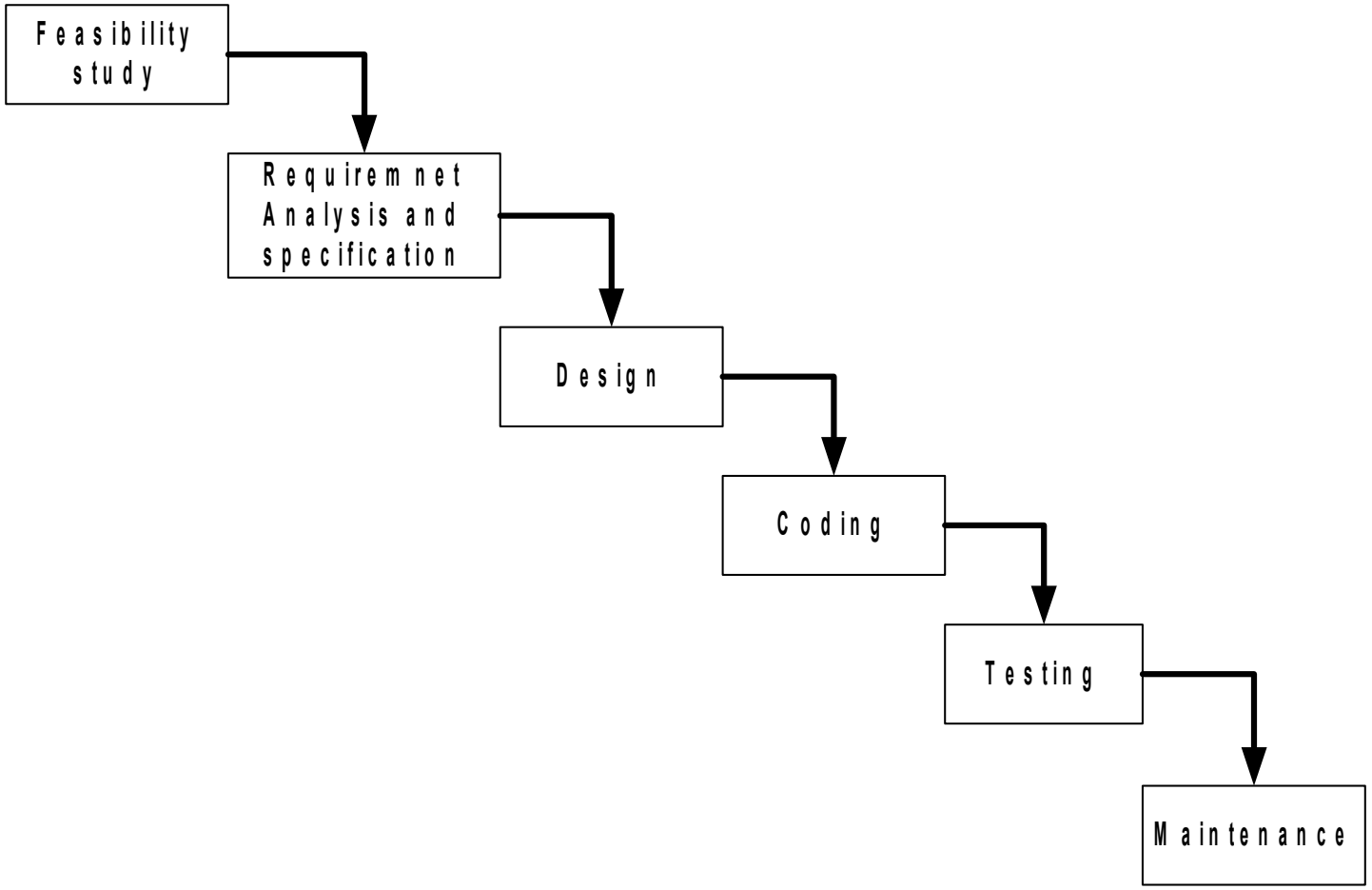
- A system design life cycle can be modeled with descriptive and diagrammatic representation representing:
  - All activities required to make a software product transit through its life cycle phases
  - Order in which activities are to be undertaken

- Number of life cycle models are known to system designers
  - All models are **same** so far the **basic activities** are concerned
  - Models are **different** so far the **order of activities** are concerned

Why a Life Cycle Model?

- It encourages development of system in systematic and disciplined manner
- Provides better understanding and communication among team members
- Pipelining the activities to carried out number of dissimilar projects together
- Better utilization of resources (human + system)

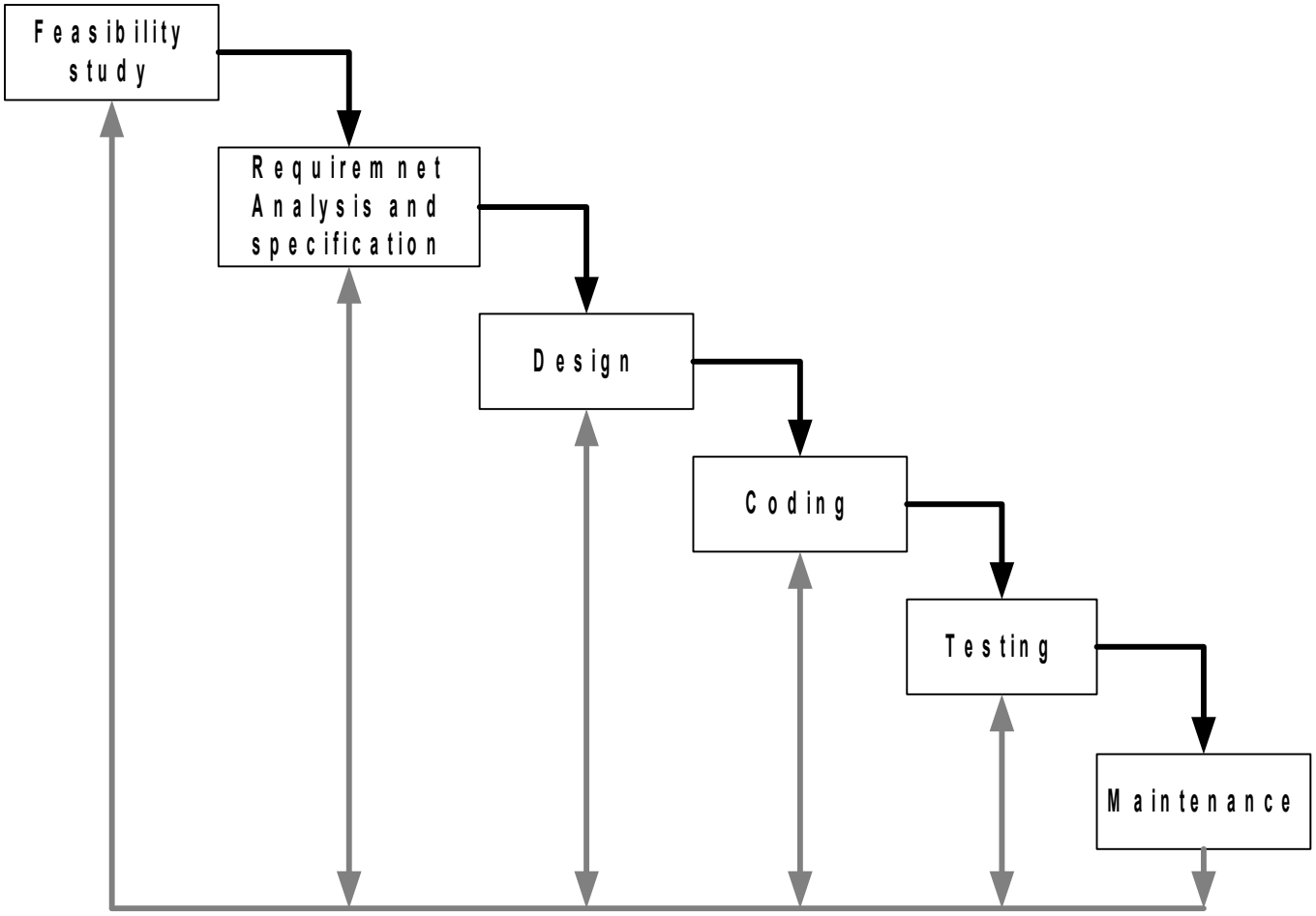
Classical Waterfall Model



Classical Waterfall Model

- The classical waterfall model is elegant and intuitively the most obvious way to develop system
- The model is considered to be *theoretical way of developing* system
- Rather the model is *not practical*
- Other life cycle models are essentially derived from the classical waterfall model

Iterative Waterfall Model

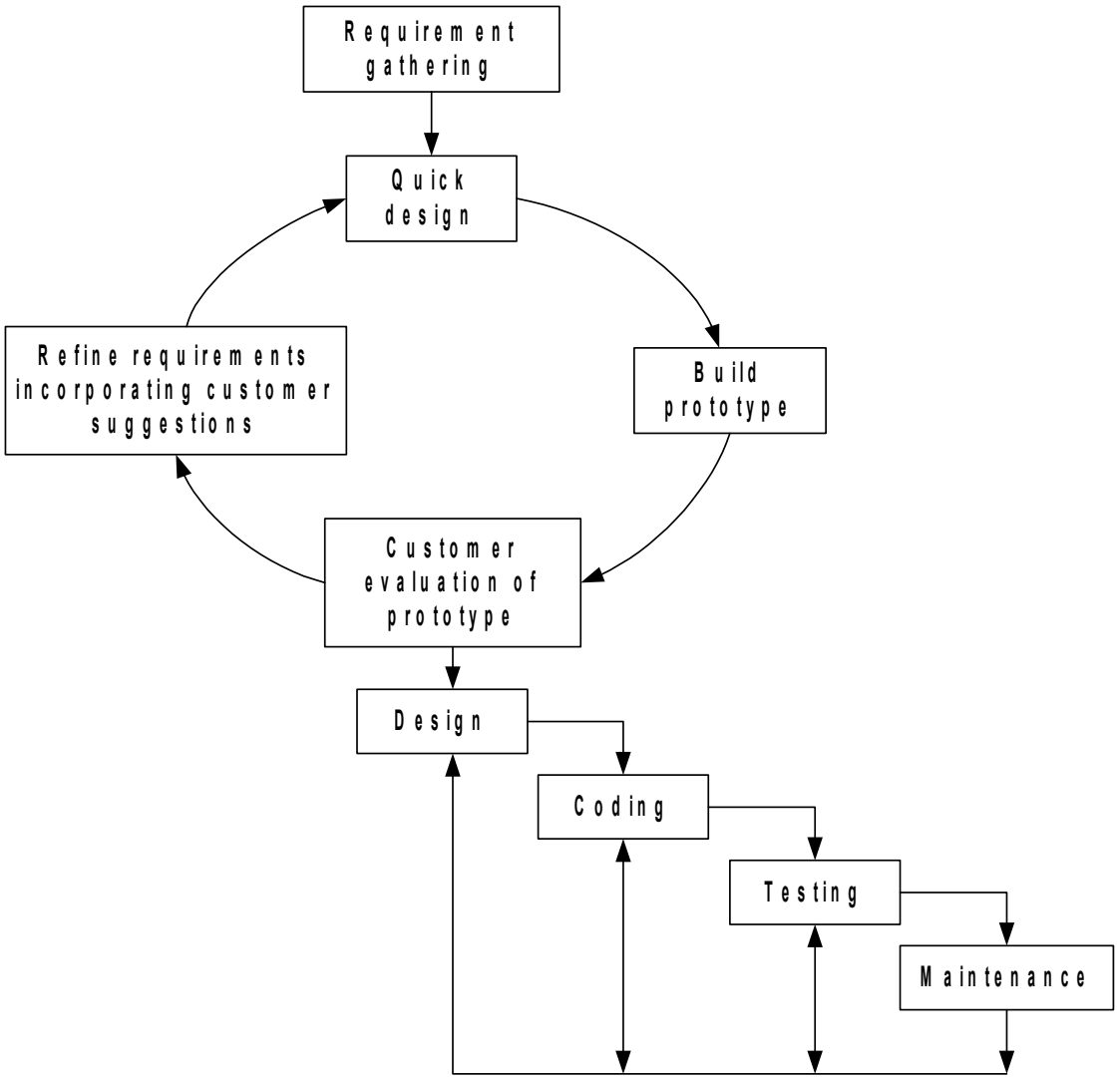


Iterative Waterfall Model

- Advantages:
  - Iterative waterfall model is practical compared to the classical waterfall model since errors are taken into consideration in it

- Drawbacks
  - It unable to handle different risks that a real-life software project is subjected to
  - It follows rigid phase sequence, but for higher productivity and better efficiency it may not be desirable at all
  - The model is suitable for well-understood problem, not suitable for very large projects

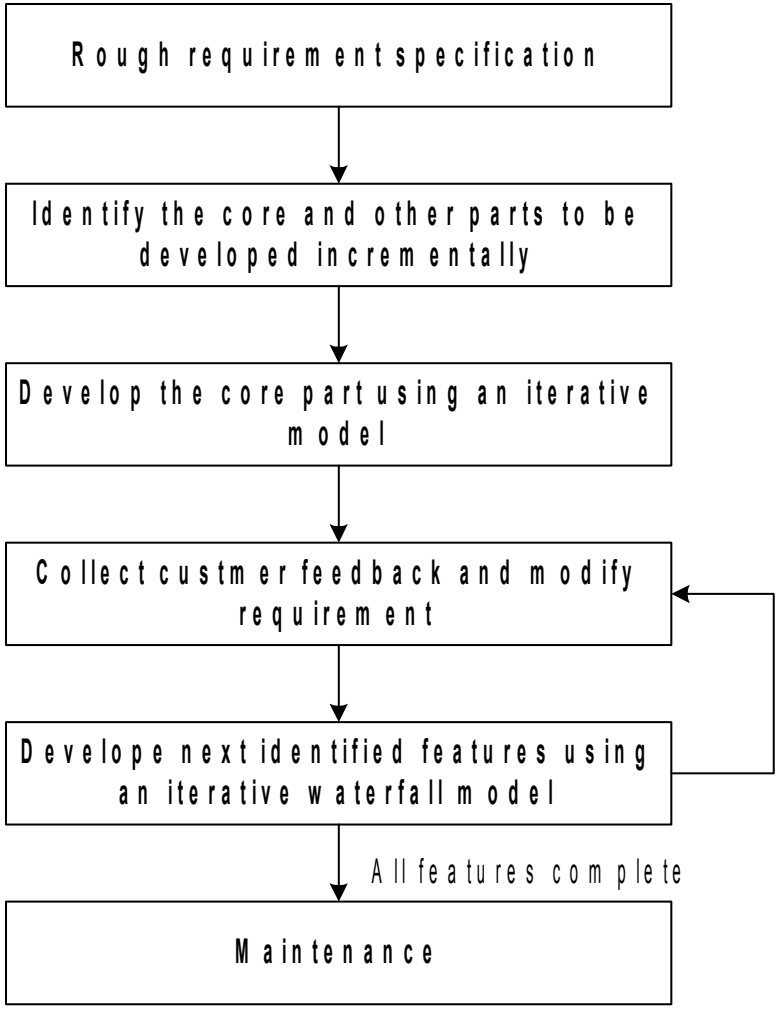
Prototyping Model



Prototyping Model

- Expensive but deal with more complex and new system development (unlike waterfall model)
- Many customer requirements get properly defined
- Technical issues get resolved by experimenting with the prototype
- Minimizes the redesign costs

Evolutionary Model



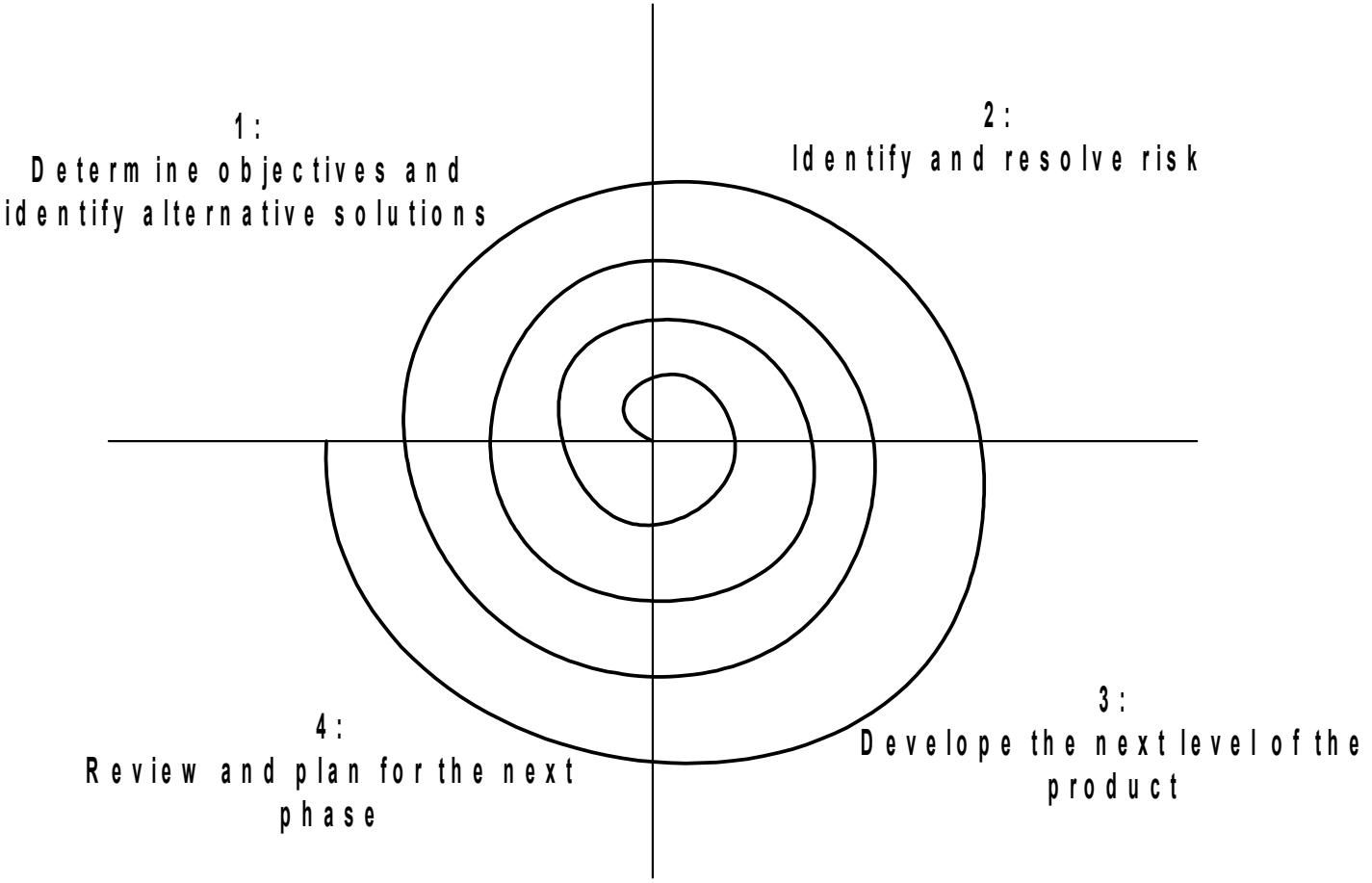
Evolutionary Model

- This model is based on the incremental approach
- User gets a chance to experiment with a partially developed system much before the complete version
- Minimum redesign effort
- Obviate the need to commit large resources in one go
- Suitable for very large system development

Spiral Model

- Spiral model is also termed as meta model, since it subsumes all the previously discussed models
- Unlike other models, it provides good insight to the project risks
- Spiral model is much more flexible compared to other models

Spiral Model



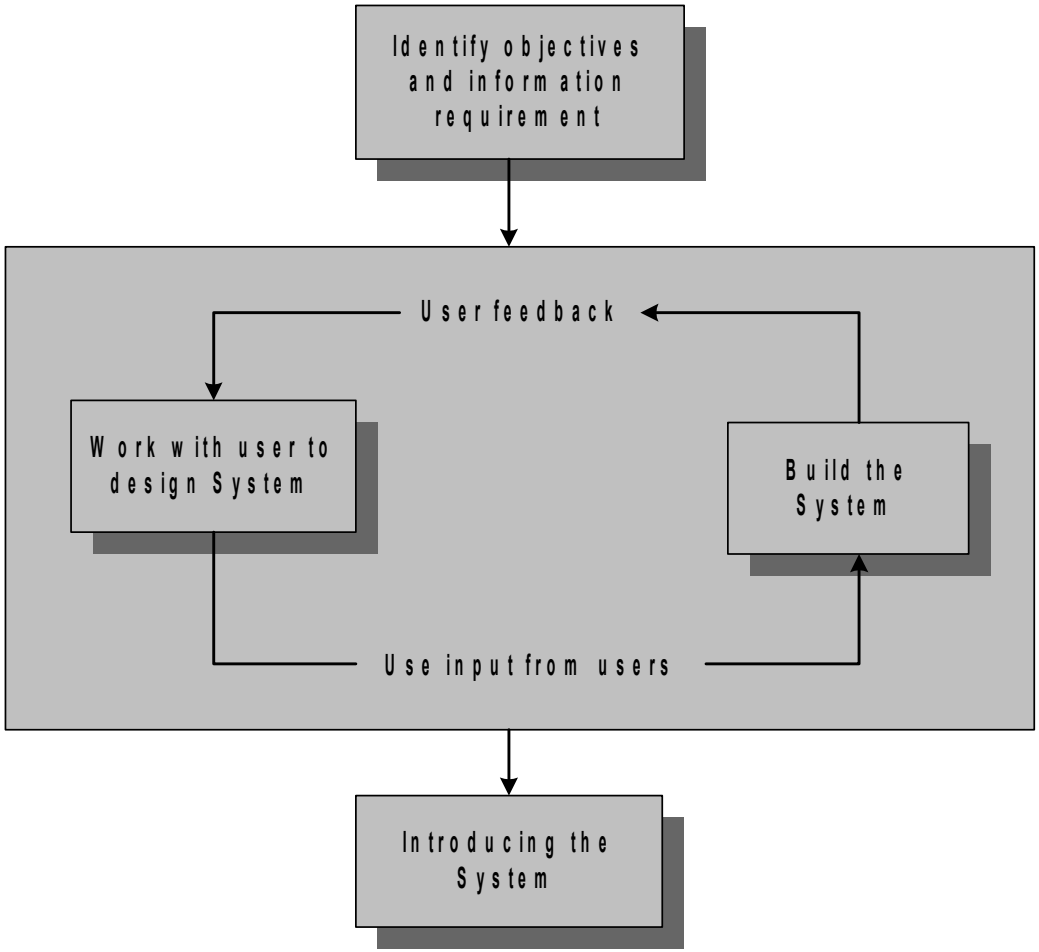
Spiral Model: Salient Features

- Models looks like a spiral with many loops
- The number of loops is not fixed rather decided by
  - Number of phases
  - Iteration of the phases
- Exact number of phases and order of phases is not fixed hence the model is more flexible
- There are four stages of each phase, which is depicted by four sectors in the model
- As a loop proceeds, a more complete version of the system gets build
  - Radius of spiral at any point represents the cost incurred in the project till then
  - Angular dimension represents the progress made in the current phase

Rapid Application Development

- It is an objective-oriented approach to system development
- RAD and Prototyping are conceptually very close
- Goal of RAD is to shortening the time typically needed in a traditional SDLC
- There are three phases in RAD
  - Requirement planning
  - Design
  - Implementation

Rapid Application Development



Rapid Application Development

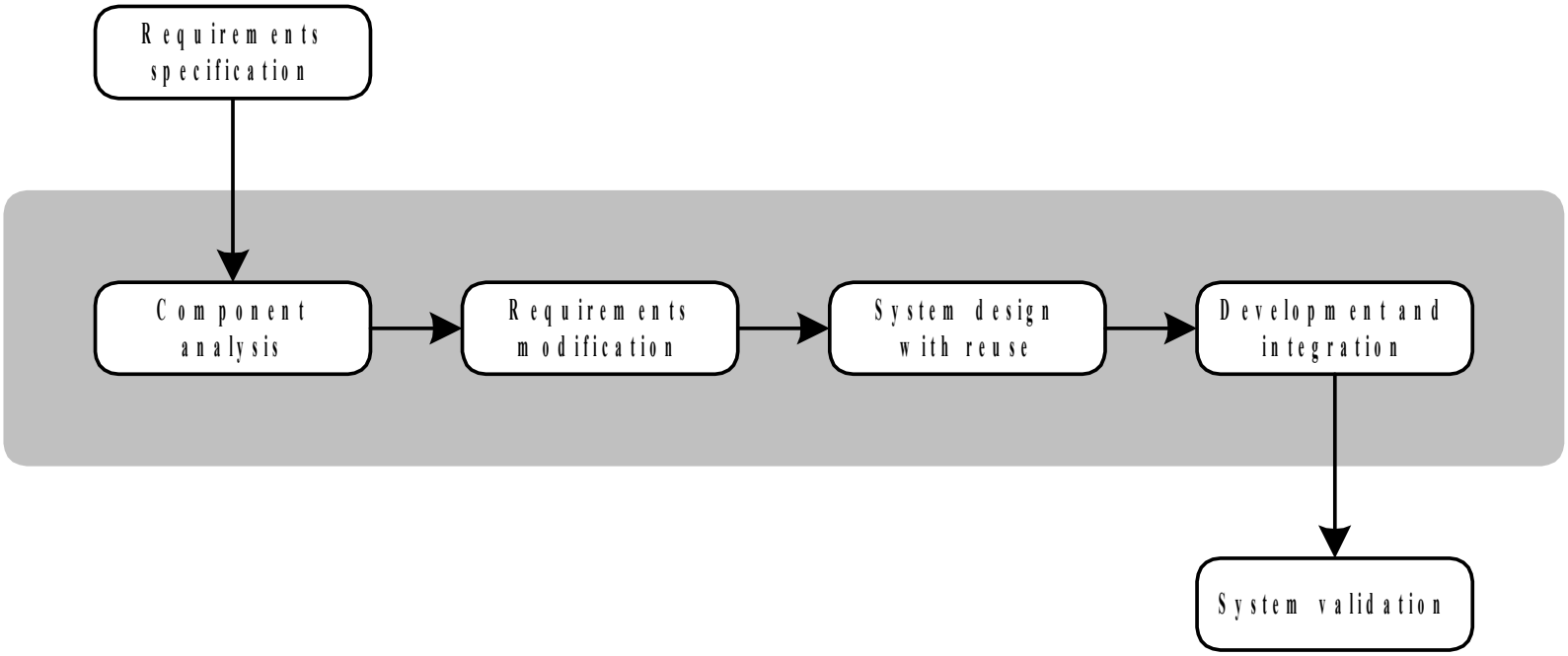
- Advantages
  - Useful for inexperienced team
  - Pressing business for speeding up the system development
  - Working with novel system development
  - Developing system for sophisticated users
  - RAD can be merged with SDLC

- Disadvantages
  - Not a systematic approach
  - Hardly bother about system documentation
  - Time vs. Attention trade-off

Component-Based Software Engineering

- In majority of the software project there is some software reuse
  - Components which are readily available to a team can be modified them as needed and incorporated them into their system
  - Reuse is often essential for rapid system development
- CBSE is a reuse-oriented approach
  - It follows a large base of reusable software components and some integrating framework for these components
  - These components are called COTS (Commercial Off-The-Shelf)

Generic Process Model of CBSE



Component-Based Software Engineering

- **Advantages**
  - Reduces the amount of software to be developed
  - Reduces the cost of development
  - Reduces risk
- **Disadvantages**
  - Requirement compromises are inevitable
  - May lead to a system that does not meet the real needs of users
  - Control over the system evolution is lost as the new versions of the reusable components are not under the control of the project group

Problems to Ponder

- Which model is suitable for which system? List a few with justification.
- Arrange the life cycle models in order according to
  - Development time
  - Redesign time (customer satisfaction)
  - Problem size
  - Cost
- Why RAD is suitable for object-oriented system development?