Lecture 3: Software Process - I

Software Engineering (CSM 31212)

MM. Mohamed Mufassirin

Lecturer in Computer Science

Dept. of Computer Science

FAS/SEUSL

Software Process

Goal	Activities	Assess
Appreciate the need for a defined software process • Be able to describe in detail the main software process models • Be able to compare software process models and choose between them	3. Differentiate SW process and SW process model	Analysis
	2. List Software process characteristics	Knowledge
	3. List the different phases in a system engineering process	Analysis
	4. Detail the differences in the SW process models and compare them	Knowledge, Analysis
	5. Suggest the most appropriate SW process model for the different systems (CASE studies)	Knowledge, comprehension, Analysis ,Synthesis, Evaluation

Objectives

- To introduce the general phases of the **software (development) life cycle**
- To describe various generic **software process models** and their pros and cons

What is a Software Process?

- A set of ordered tasks involving activities, constraints and resources that produce a software system
- A process is important because it require consistency and structure on a set of activities
- It guides our actions by allowing us to examine, understand, control and improve the activities that comprise the process
- The process of building a product is sometime called a ***System Development lifecycle*** because it describes the life of that product from conception through to its implementation,delivery,use and maintenance

Life Cycle...?

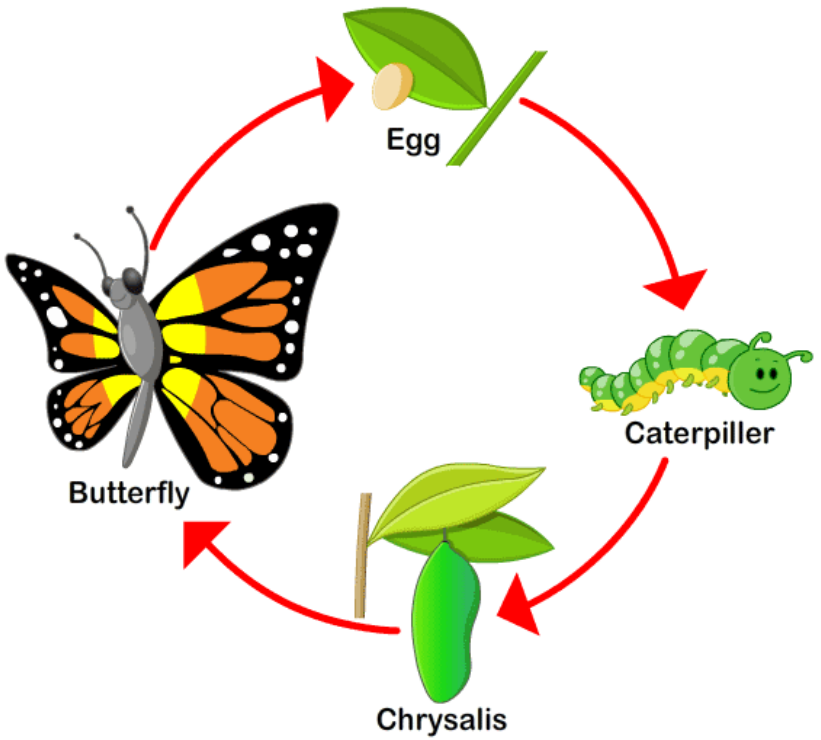
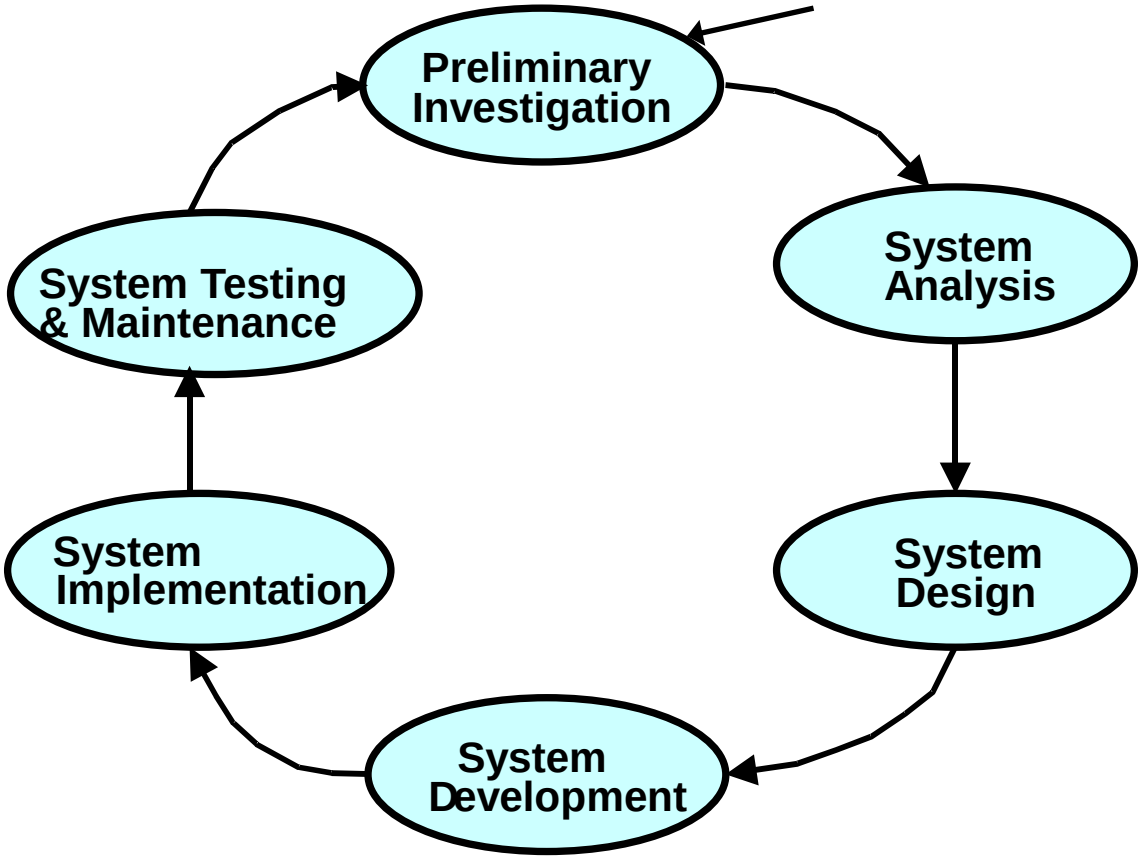


Image from: www.cliparthut.com

Generic Process Model/ SDLC Phases



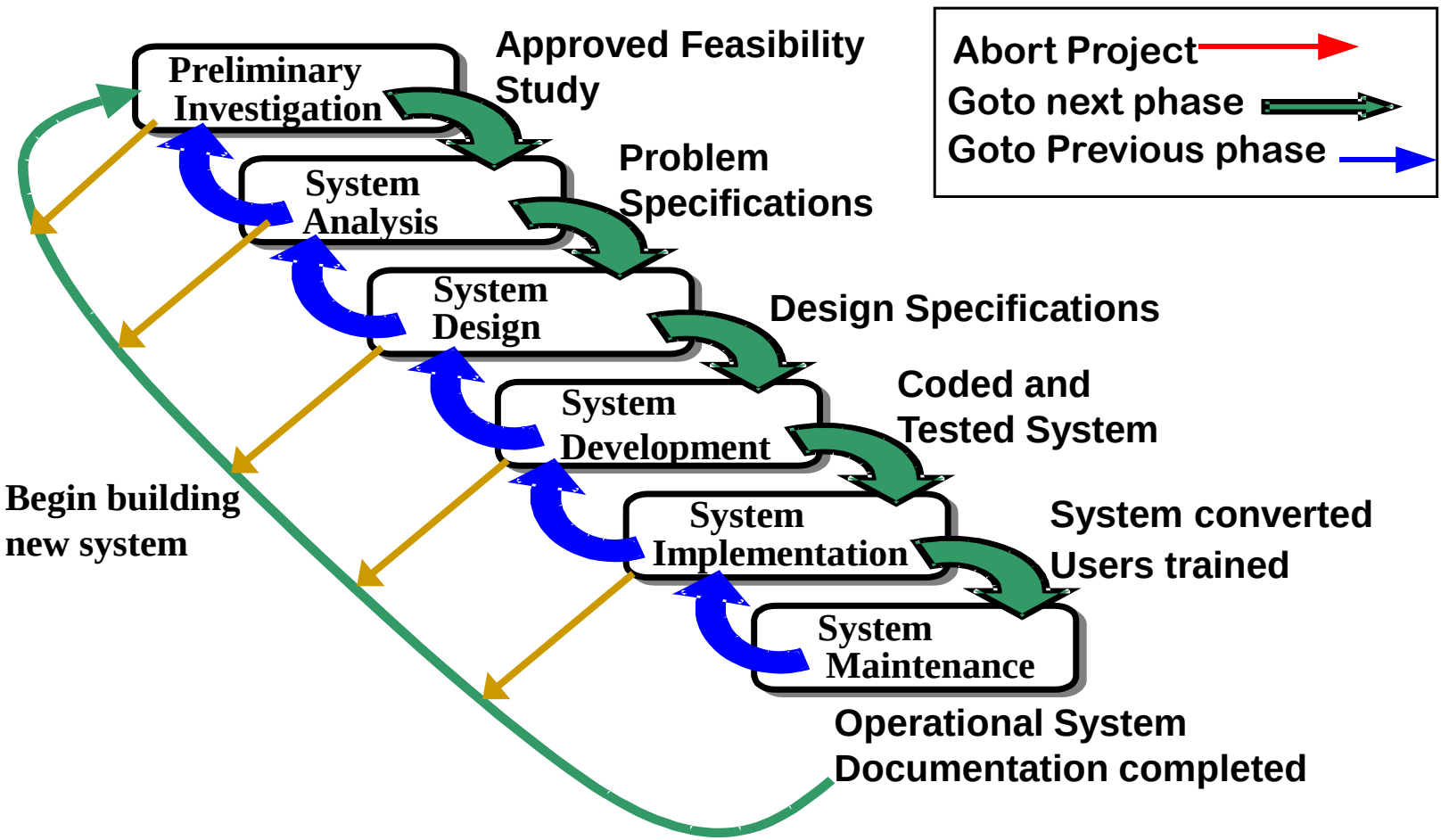
Six Phases of SDLC

- **Preliminary Investigation**
  - Assesses feasibility and practicality of system
- **System Analysis**
  - Study old system and identify new requirements
  - Defines system from user's view
- **System Design**
  - Design new/alternative system
  - Defines system from technical view

Six Phases of SDLC (Cont..)

- **System Development**
  - New hardware and software is acquired, developed, and tested
- **System Implementation**
  - System installation and training
- **System Testing & Maintenance**
  - User acceptance testing (Validation and Verification)
  - Periodic evaluation and updating

Deliverables of the SDLC



The software process

- A Structured set of activities required

  Eg. Specification, Design, Validation, Verification, Evolution

- Activities vary depending on the organisation and the type of system being developed
- Must be explicitly modelled if it is to be managed
- Process characteristics

  - Understandability
  - Visibility
  - Acceptability
  - Reliability
  - Rapidity
  - Robustness
  - Maintainability
  - Supportability

Process Characteristics

Understandability

To what extent is the process explicitly defined and how easy is it to understand the process definition…?

Visibility

Do the process activities terminate in clear results so that the progress of the process is externally visible?

Acceptability

Is the defined process acceptable to and usable by the engineers responsible for producing the software project?

Reliability

Is the process is designed in such a way that process errors are avoided or trapped before they result in product errors?

Process Characteristics…

- **Rapidity**

How fast can the process of delivering a system from a given specification be completed?

- **Robustness**

Can the process continue in spite of unexpected problems?

- **Maintainability**

Can the process evolve to reflect changing organizational requirements or identified process improvements?

- **Supportability**

To what extent can the process activities be supported by CASE tools?

Software Process Models

- **You need to model the process because:**

✓ When a team writes down a description of its development process it forms a common understanding of the activities, resources and constraints involved in software development

✓ Creating a process model helps the team find **inconsistencies**, **redundancies** and **commissions** in the process, as these problems are noted and corrected the process becomes more effective

Software Process Models (continued)

- ✓ The model reflects the goals of development and shows explicitly how the product characteristics are to be achieved

- ✓ Each development is different and a process has to be tailored for different situations, the model helps people to understand these differences

Software Process Models

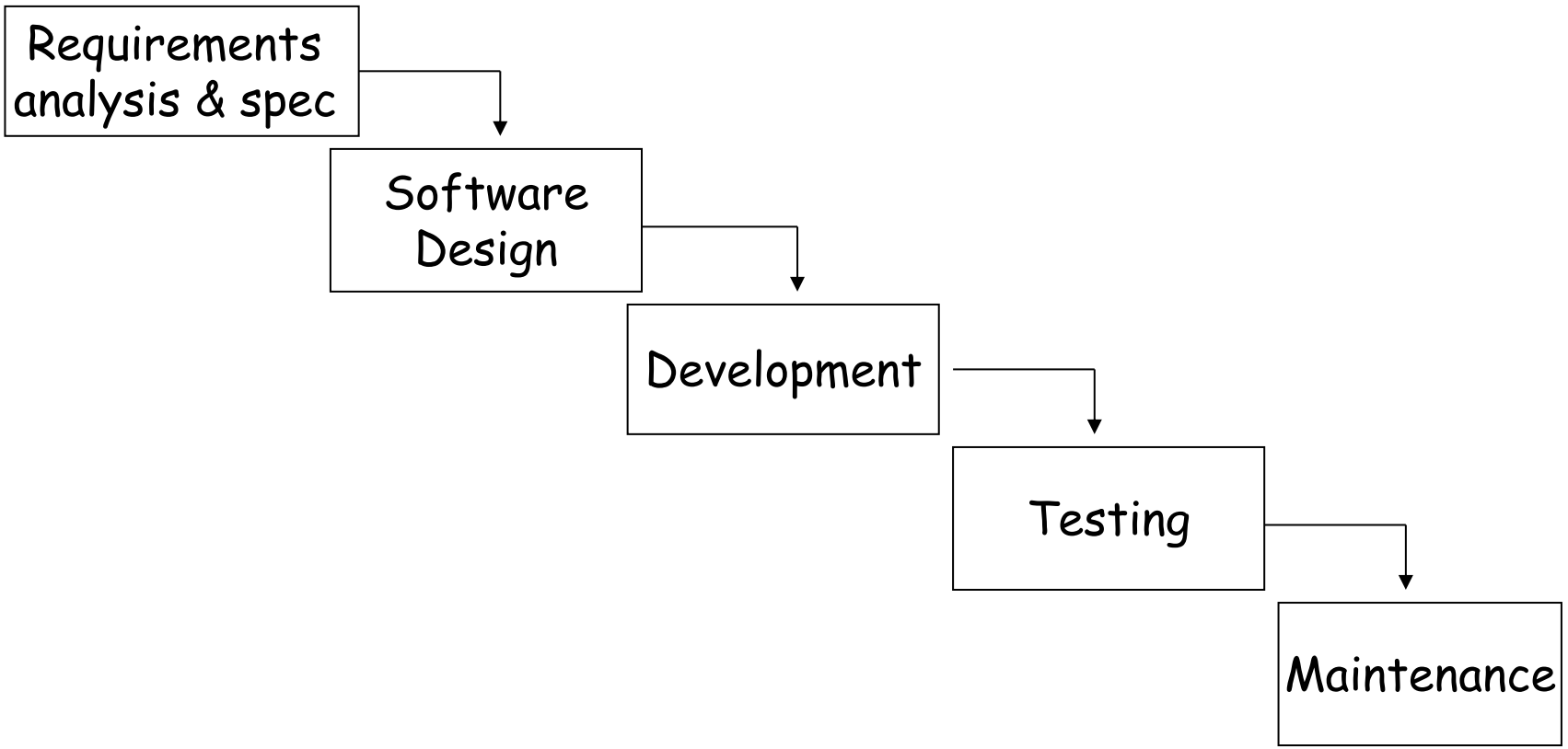
Examples

- Waterfall model
- Prototyping models
- Evolutionary models
- The spiral model
- Formal development
- Incremental development
- Rapid Application Development
- Unified Process
- Agile Process
- Extreme Programming (XP)

The Waterfall model

- Separate and distinct phases of specification and development
- A linear sequential model
- Easy to work and understand

Waterfall model…



The Waterfall Model

Software Requirement Analysis and Specification

The system's services, constraints and goals are established with the consultation with the users.

This would include the understanding of the information domain for the software, functionality, behavior, performance, interface, security and exceptional requirements. This requirements are then specified in a manner which is understandable by both users and the development staff.

Software design

The design process translates requirements into a representation of the software that can be implemented using software tools. The major objectives of the design process are the identification of the software components, the software architecture, interfaces, data structures and algorithms.

Coding (Development)

The design must be translated to a machine readable form. During this stage the software design is realized as a set of programs or program units. Programming languages or CASE tools can be used to develop software.

Testing

The testing process must ensure that the system works correctly and satisfies the requirements specified. After testing, the software system is delivered to the customer.

Maintenance

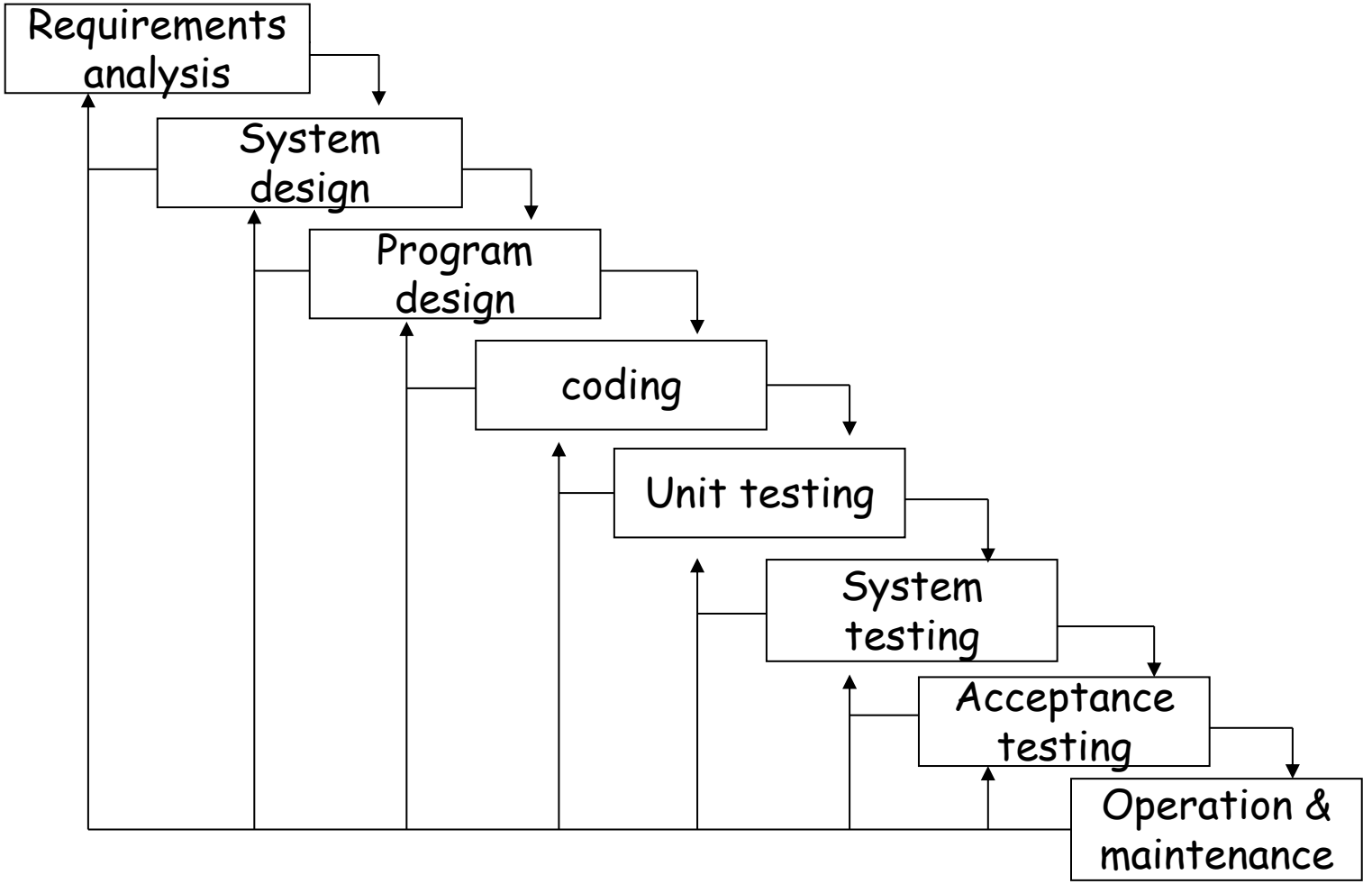
Software will undoubtedly undergo changes after it is delivered to the customer. Errors in the system should corrected and the system should be modified and updated to suit new user requirements.

Problems with the Waterfall Model

1. Real projects rarely follow the sequential flow that the model proposes. Although the Waterfall model can accommodate iteration, it does so indirectly.
2. It is often very difficult for the customer to state all requirements explicitly.
3. The Waterfall model has the difficulty of accommodating the natural uncertainty that exists at the beginning of many projects.
4. The customers must have patience. A working version of the program(s) will not be available until late in the project time-span. A major blunder, if undetected until the working program is reviewed, can be disastrous.

Solution…?

Waterfall Model II (Iterative Waterfall Model)



Comment : The Waterfall model is suitable for projects which have clear and stable requirements.

Explain when to use the waterfall model.

One should use the waterfall model only when:

- Requirements are very clear and fixed.
- There are no ambiguous requirements.
- The client has high confidence in the organization.
- The organization has experience of similar projects.
- The project is short.
- It is good to use this model when the technology is well understood.

Prototyping

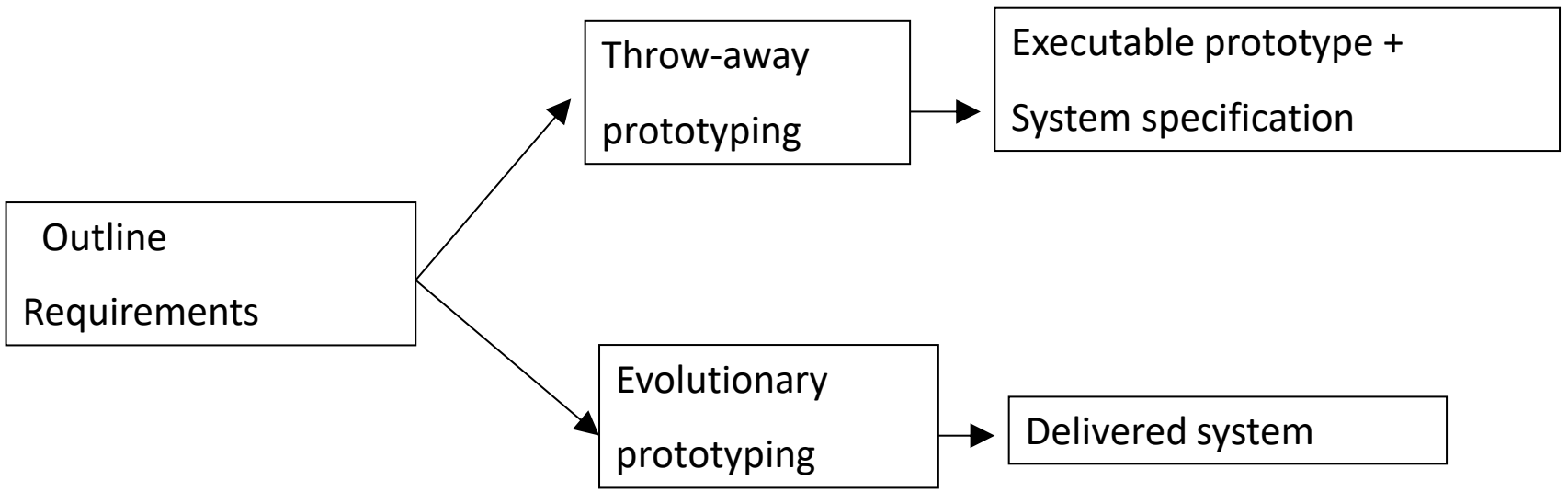
- It is very difficult for end-users to anticipate how they will use new software systems to support their work.
- If the system is large and complex, it is probably impossible to make this assessment before the system is built and put into use.
- A prototype (a small version of the system) can be used to clear the vague requirements. A prototype should be evaluated with the user participation.

There are two types of Prototyping techniques

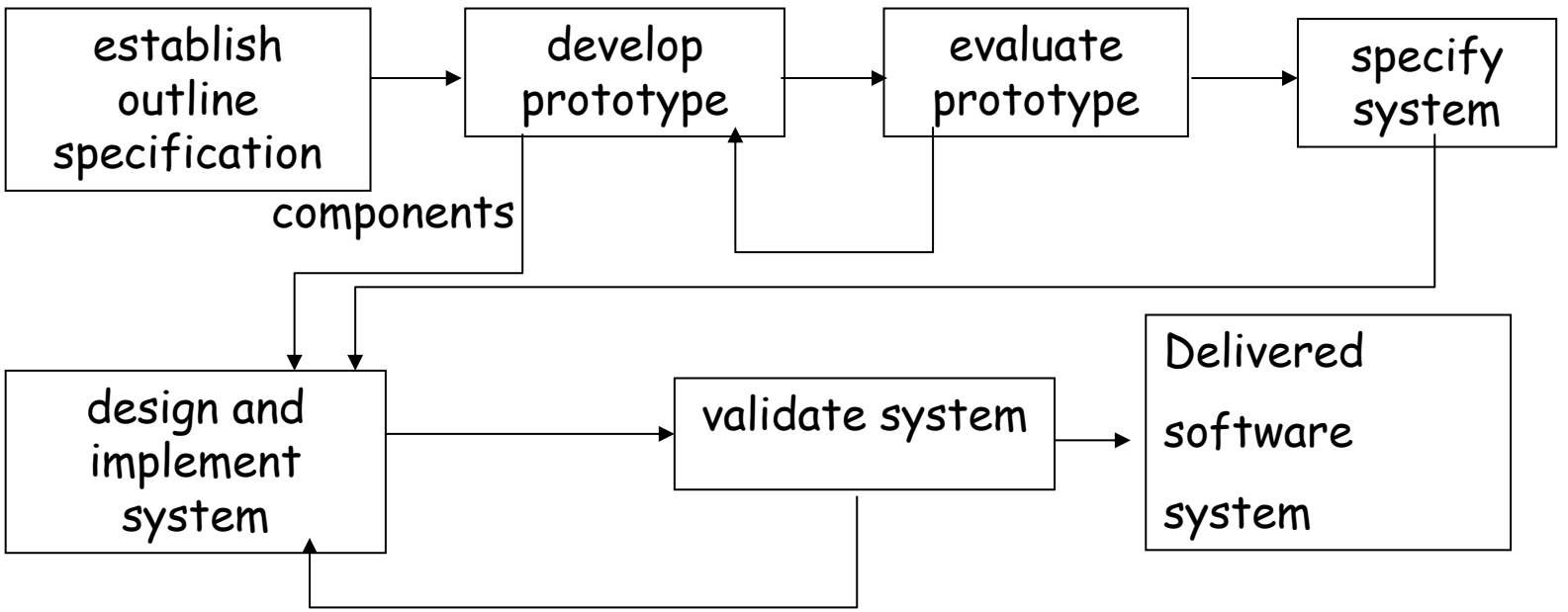
* Throw-away Prototyping

* Evolutionary Prototyping

Throw-away and Evolutionary Prototyping



Throw-away Prototyping



Throw-away Prototyping

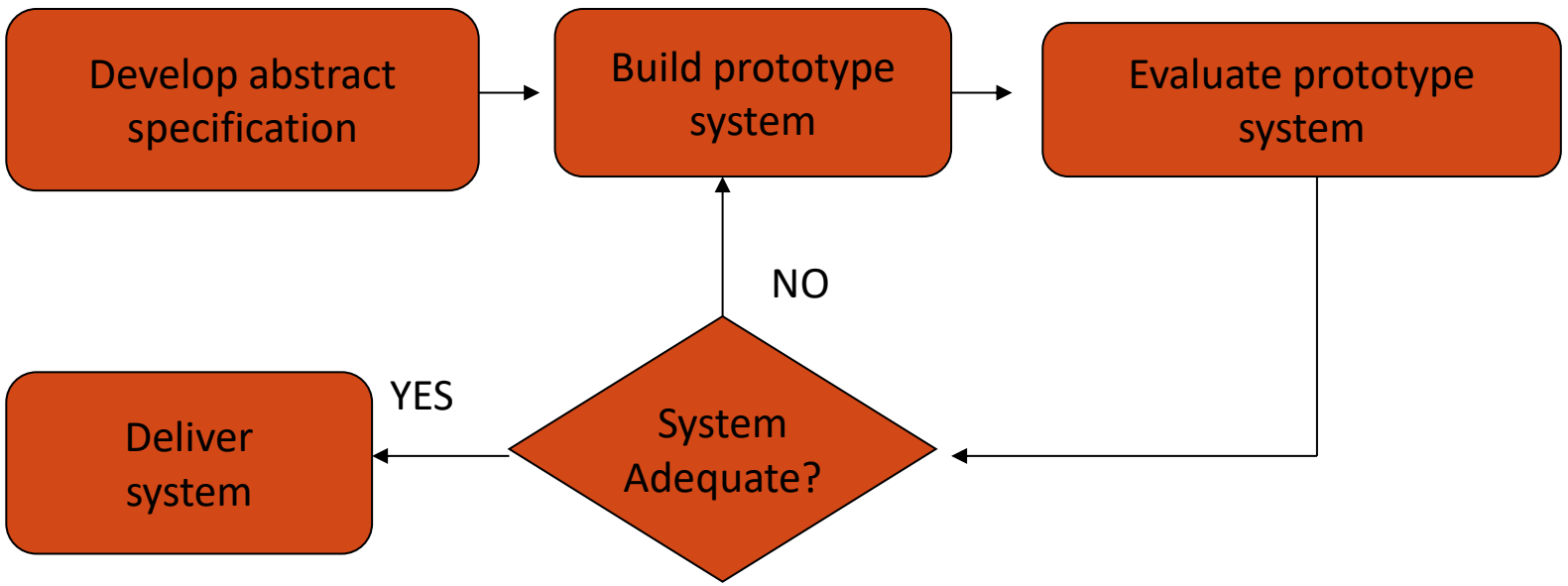
The objective is to understand the system requirements clearly. Starts with poorly understood requirements. Once the requirements are cleared, the system will be developed from the beginning.

This model is suitable if the requirements are vague but stable.

Problems with Throw-away Prototyping

1. Important features may have been left out of the prototype to simplify rapid implementation.
    1. In fact, it may not be possible to prototype some of the most important parts of the system such as safety-critical functions.
2. An implementation has no legal standing as a contract between customer and contractor.
3. Non-functional requirements such as those concerning reliability, robustness and safety cannot be adequately tested in a prototype implementation.

Evolutionary Prototyping



Evolutionary prototyping

Advantages

* Effort of prototype is not wasted
* Faster than the Waterfall model
* High level of user involvement from the start
* Technical or other problems discovered early – risk reduced
* Mainly suitable for projects with vague and unstable requirements.

Evolutionary Prototyping (continued)

Disadvantages

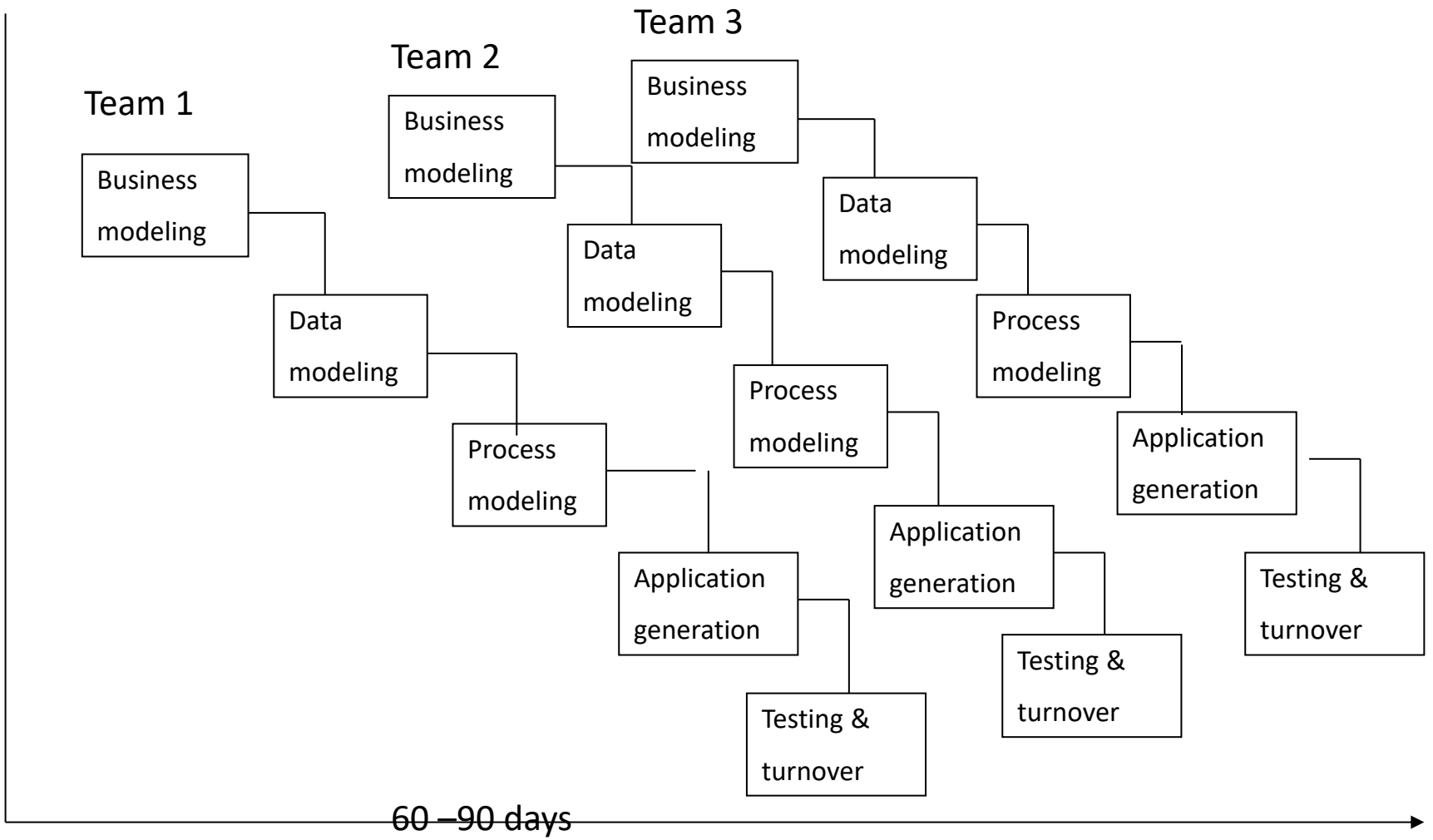
- Prototype usually evolve so quickly that it is not **cost-effective** to produce great deal of documentation
- Continual change tends to corrupt the structure of the prototype system. Maintenance is difficult and costly.
- It is not clear how the range of skills which is normal in software engineering teams can be used effectively for this mode of development.
- Languages which are good for prototyping not always best for final product.

The RAD Model

Rapid Application Development (RAD) is an incremental software development process model that emphasizes an extremely short development cycle.

If requirements are **well understood** and project **scope is constrained**, the RAD process enables a development team to create a 'fully functional system' within very short time periods (eg. 60 to 90 days)

The RAD Model



Processes in the RAD model

Business modelling

The information flow in a business system considering its functionality.

Data Modelling

The information flow defined as part of the business modelling phase is refined into a set of data objects that are needed to support the business

Process Modelling

The data objects defined in the data modelling phase are transformed to achieve the information flow necessary to implement business functions.

Application generation

RAD assumes the use of 4GL or visual tools to generate the system using reusable components.

Testing and turnover

New components must be tested and all interfaces must be fully exercised

Problems with the RAD model

- RAD requires sufficient human resources to create right number of RAD teams
- RAD requires developers and customers who are committed to the rapid-fire activities necessary to get a system completed in a much abbreviated time frame.
- If a system cannot be properly modularized, building the components necessary for RAD will be problematic.
- RAD is not applicable when technical risks are high. This occurs when a new application makes heavy use of new technology or when the new software requires a high degree of interoperability with existing computer programs.

Questions

Review Questions

1. The following are initial requirements specified by some customers. Identify the most suitable process models for these software projects.

   (a) " I need to develop a simple library system for my school. I know the requirements very well"

   (b) "I need to develop a management information system for my organisation,. I may use it for all the branches in several location in Sri Lanka. I might use it on the Internet".

   (c) " I need to develop a software system to control a space shuttle"

Thank You

?