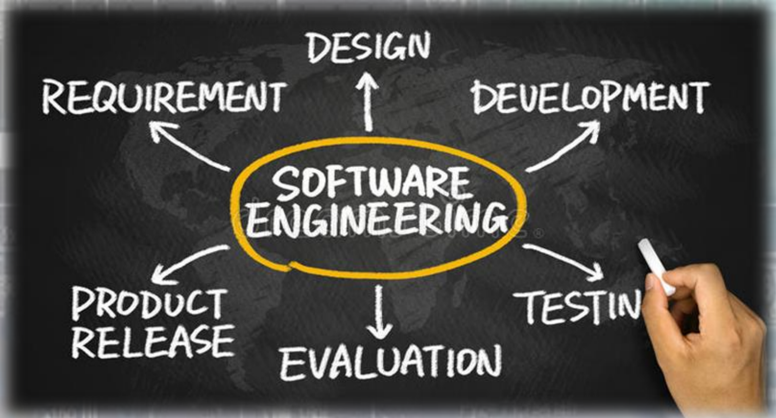
Topic 2 (Chapter 2): Software Processes



CSE241/CMM341
Foundations of Software Engineering

Photo credit: Dreamstime.com

CSE241 Course Content

Foundations of Software Engineering

Learning outcomes

- Understand the concepts of software processes and software process models
- Know about the fundamental process activities of software requirements engineering, software development, testing and evolutions
- Understand why processes should be organised to cope with changes in the software requirements and design
- Understand the notion of software process improvement and the factors that affect software process quality.

Topic 2: Software Processes

Contents

The software process

A structured set of activities required to develop a software system.

Software specification
- what the system should do
- constraints on system

Software development
- Meeting the specifications
- Implement the system

Software validation
- ensure that system does what the customer wants

Software evolution
- Modify the system to meet changes in customers needs

The four fundamental SE activities

THE SOFTWARE PROCESS

The four fundamental SE activities

SOFTWARE SPECIFICATION

- What the system should do
- Constraints on system

SOFTWARE DEVELOPMENT

- Meeting the system specifications
- Implement the system

SOFTWARE VALIDATION

- Ensure that system does what the customer wants

SOFTWARE EVOLUTION

- Modify the system to meet changes in customers needs

SE is intended to support PROFESSIONAL SOFTWARE DEVELOPMENT rather than individual programming.

SOFTWARE PROCESS DESCRIPTIONS

- When describing processes, it's also important to describe who is involved, what is produced and conditions that influence the sequence of activities.

Products or artefacts	the outcomes of a process activity
Roles	the responsibilities of the people involved in the process
Pre- and post-conditions	statements that are true before and after a process activity has been enacted or a product produced.

Categorization of Software Processes

Plan-driven processes

- all of the process activities are planned in advance
- progress is measured against the plan

Agile processes

- planning is incremental
- easier to change the process to reflect changing customer requirements

In practice, most practical processes include elements of both plan-driven and agile approaches.

There are no right or wrong software processes

Topic 2: Software Processes

Contents

SOFTWARE PROCESS MODELS

PROCESS ACTIVITIES

COPING WITH CHANGE

PROCESS IMPROVEMENTS

GENERIC SOFTWARE PROCESS MODELS

The waterfall model

Plan-driven model.

Separate and distinct phases of specification and development.

Incremental development

Specification, development and validation are interleaved.

May be plan-driven or agile.

Integration and configuration

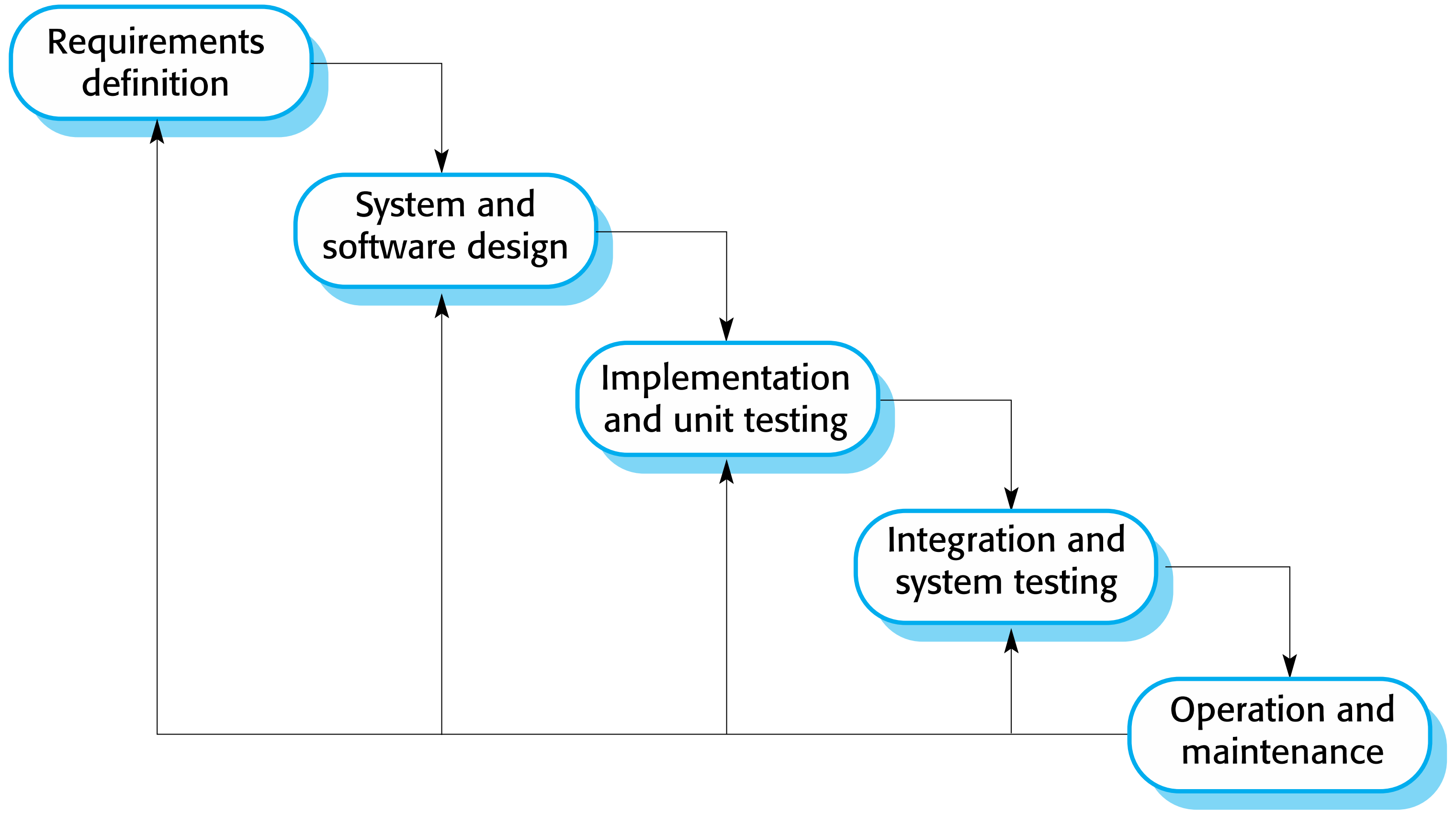
The system is assembled from existing configurable components.

May be plan-driven or agile.

Large systems

Process that incorporates elements from all of these models

The waterfall model


Requirements definition
System and software design
Implementation and unit testing
Integration and system testing
Operation and maintenance

difficulty of accommodating to changes

appropriate when requirements are well-understood

Suitable for large systems engineering project

Incremental development

cost of changing requirements reduced

easier to get customer feedback

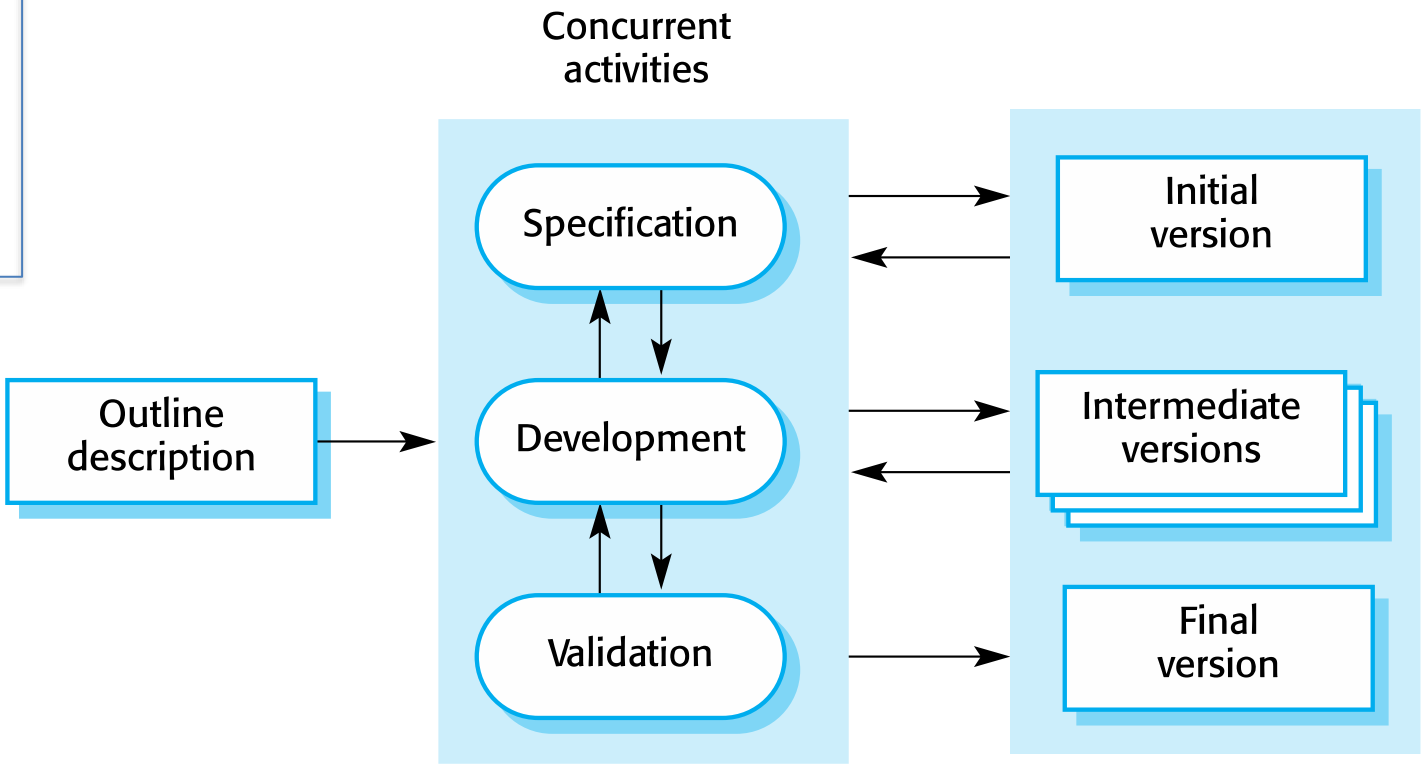
More rapid delivery and deployment

Benefit

Problem not visible

System structure can degrade as new increments added

Problem

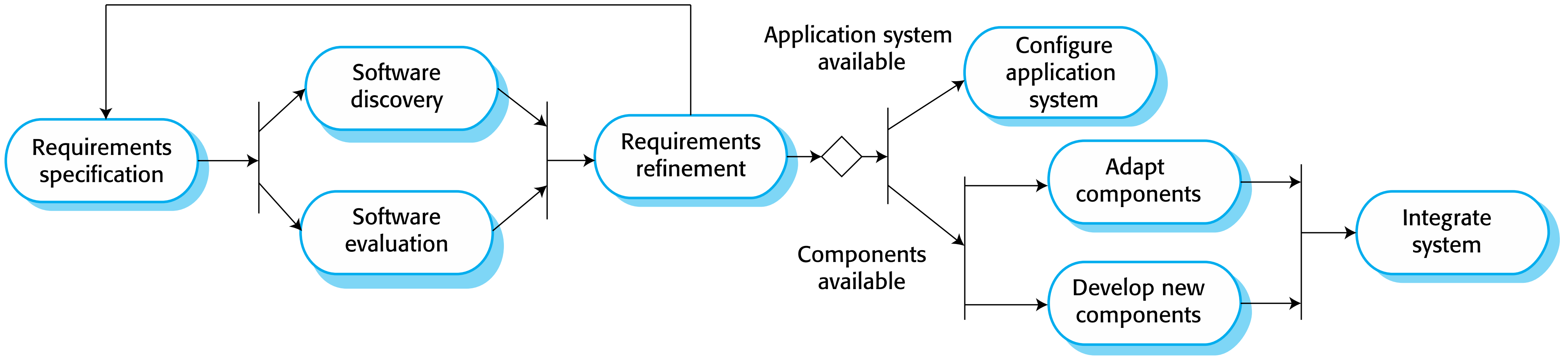


Integration and configuration: i.e. Reuse-oriented software engineering

- Based on software reuse where systems are integrated from existing components or application systems
- Reused elements may be configured to adapt their behaviour and functionality to a user's requirements

i.e. types

- Stand-alone application systems (sometimes called COTS)
- Collections of objects that are developed as a package
- Web services that are developed according to service standards and



Topic 2: Software Processes

Contents

SOFTWARE PROCESS MODELS

PROCESS ACTIVITIES

COPING WITH CHANGE

PROCESS IMPROVEMENTS

Process activities

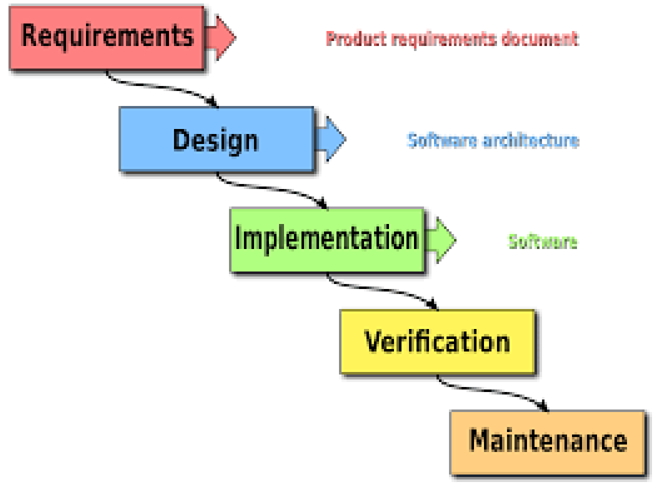
Software specification

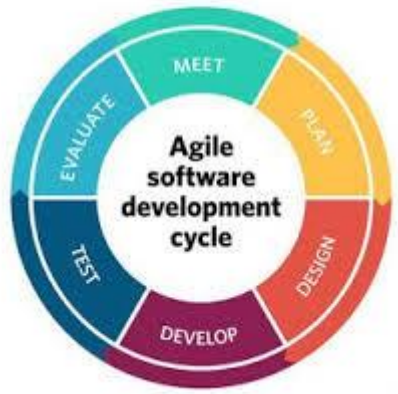
Software design & implementation

Software validation

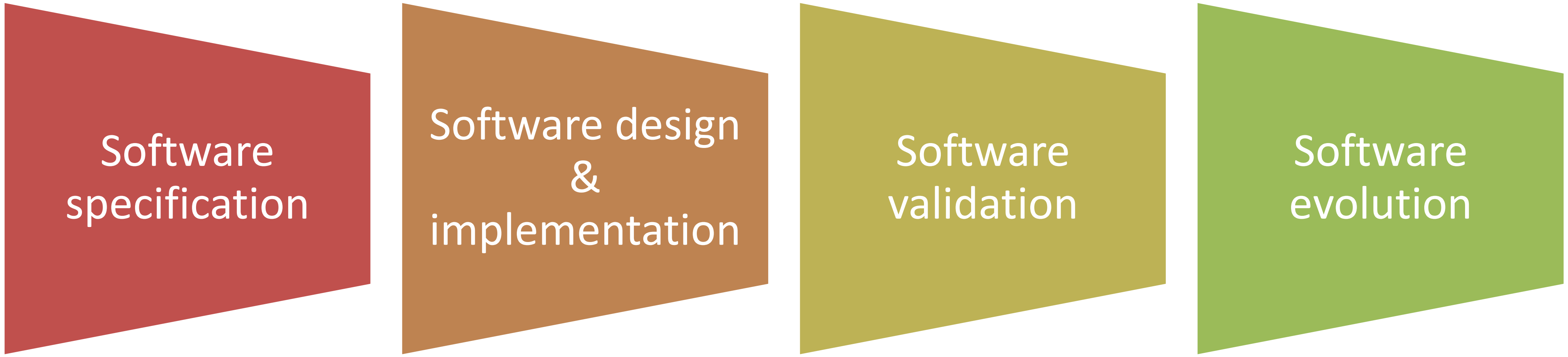
Software evolution

- **The four fundamental SE activities**
- Different development processes organized the processes differently

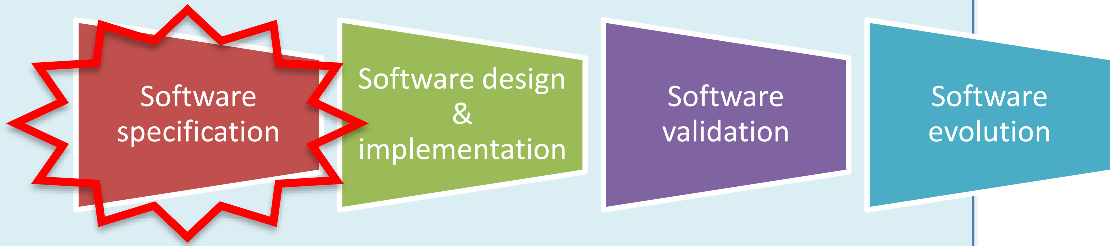




Process activities



The four fundamental SE activities

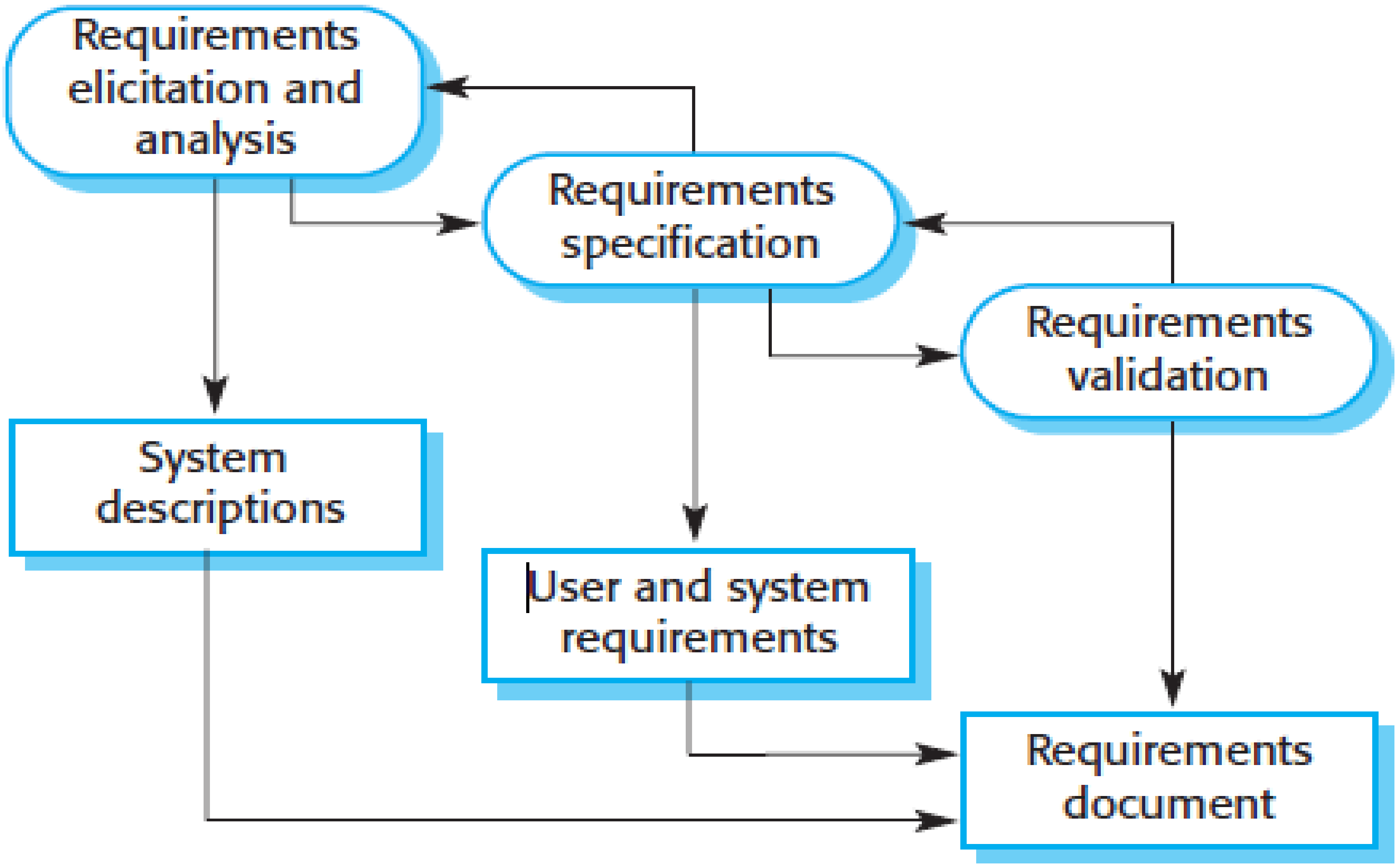


Software specification

- The process of establishing **what services are required** and the **constraints on the system's operation and development**.

- **Requirements engineering (RE)** process :
  – to produce an agreed requirements document that specifies a system satisfying stakeholder requirements.
  – presented at two levels of detail:
    1. End-users and customers - high-level statement of the requirements
    2. system developers - detailed system specification.

The RE process


Requirements elicitation and analysis
Requirements specification
Requirements validation
System descriptions
User and system requirements
Requirements document

Four main activities in RE process

Feasibility study

Can the identified user needs may be satisfied using current software and hardware technologies?

Will the proposed system will be cost-effective from a business point of view?

Can the proposed system develop within existing budgetary constraints?

Should we develop the proposed system?

Requirements elicitation and analysis

Deriving the system requirements

Development of one or more system models and prototypes

Requirements specification

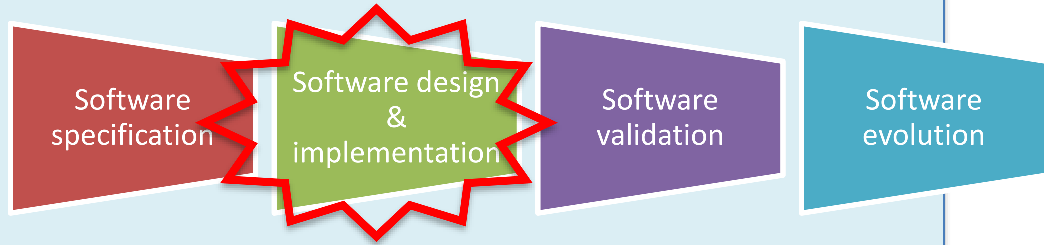
Translating the information gathered during the analysis activity into set of requirements

Two types of requirements :User requirements & system requirements

Requirements validation

checks the requirements for realism, consistency, and completeness.

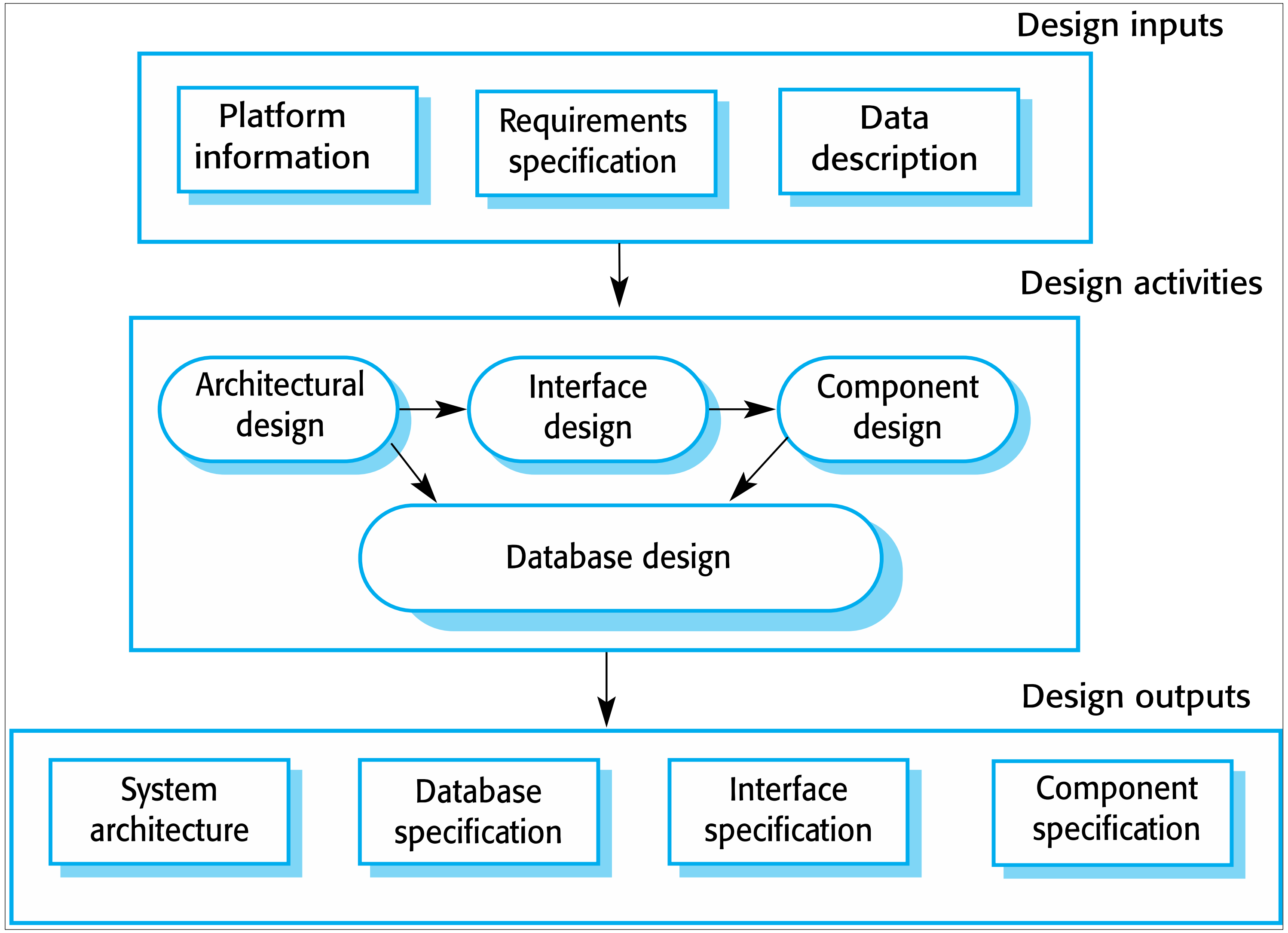
errors in the requirements document discovered and corrected



Software design & implementation

- The process of converting the system specification into an executable system.
- Software design
  - Design a software structure that realizes the specification;
- Implementation
  - Translate this structure into an executable program;
- The activities of design and implementation are closely related and may be inter-leaved.

A general model of the design process



- Platform – web based/ mobile apps/ IOT system/ embedded system/ stand alone system/ Information system

- Website that manage student registration for an activity
  - Requirement specs :
    - Student should be able to register in a particular activity

  - Data :
    - Input – student particulars : name, ID , Program, contact no, email
    - Output: notification of registration

Design activities

Software design
&
implementation

Architectual design

- identify the overall structure of the system, the principal components (subsystems or modules), their relationships and how they are distributed

Database design

- design the system data structures and how these are to be represented in a database.

Interface design

- define the interfaces between system components.

Component selection and design

- search for reusable components.
- if unavailable, you design how it will operate.

System Implementation

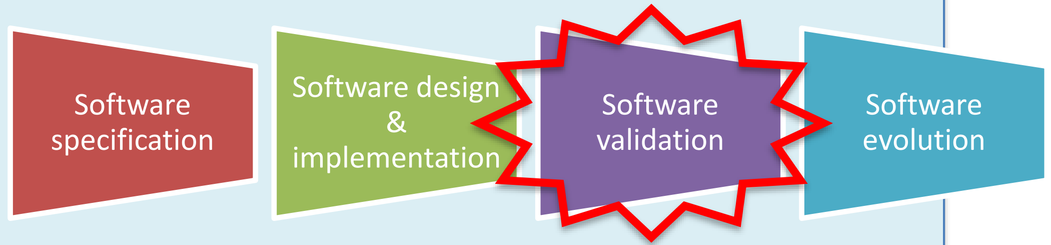
- The software is implemented either by **developing a program or programs** or by **configuring an application system**.
- Design and implementation are interleaved activities for most types of software system.
- Programming is an individual activity with no standard process.
- Debugging is the activity of finding program faults and correcting these faults.

Software design
&
implementation

System Implementation

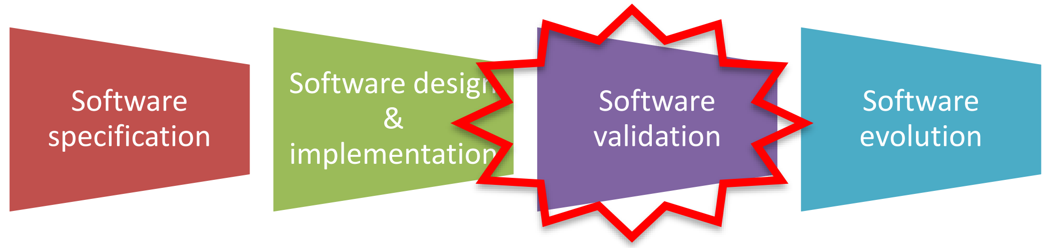
- **configuring an application system**.
- Programming →individual activity, no standard process.
- Debugging →finding & correcting program faults

Software design & implementation

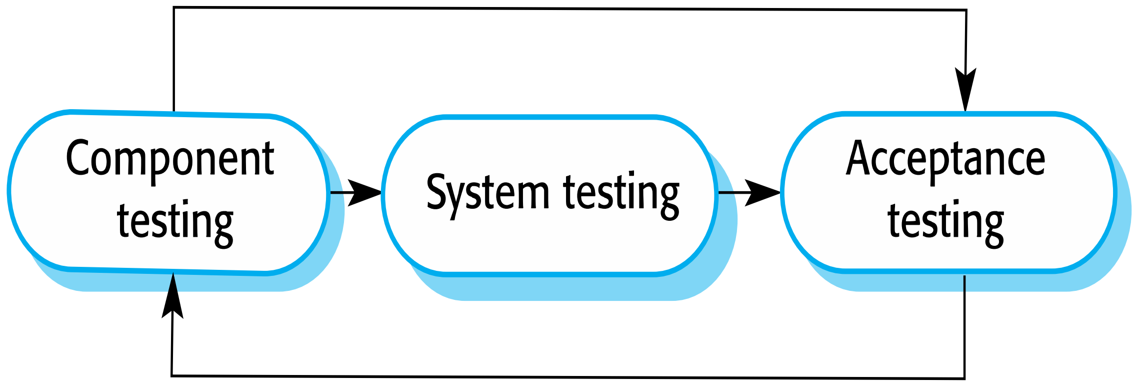


Software validation

- Verification and validation (V & V) : to show that a system **conforms to its specification** and **meets the requirements** of the system customer.
- Involves:
  - checking and review processes and system testing.
  - executing the system with test cases that are derived from the specification of the real data to be processed by the system.
- Testing is the most commonly used V & V activity.



Stages of testing



Component testing

- Individual components are tested independently;
- Components may be functions or objects or coherent groupings of these entities.

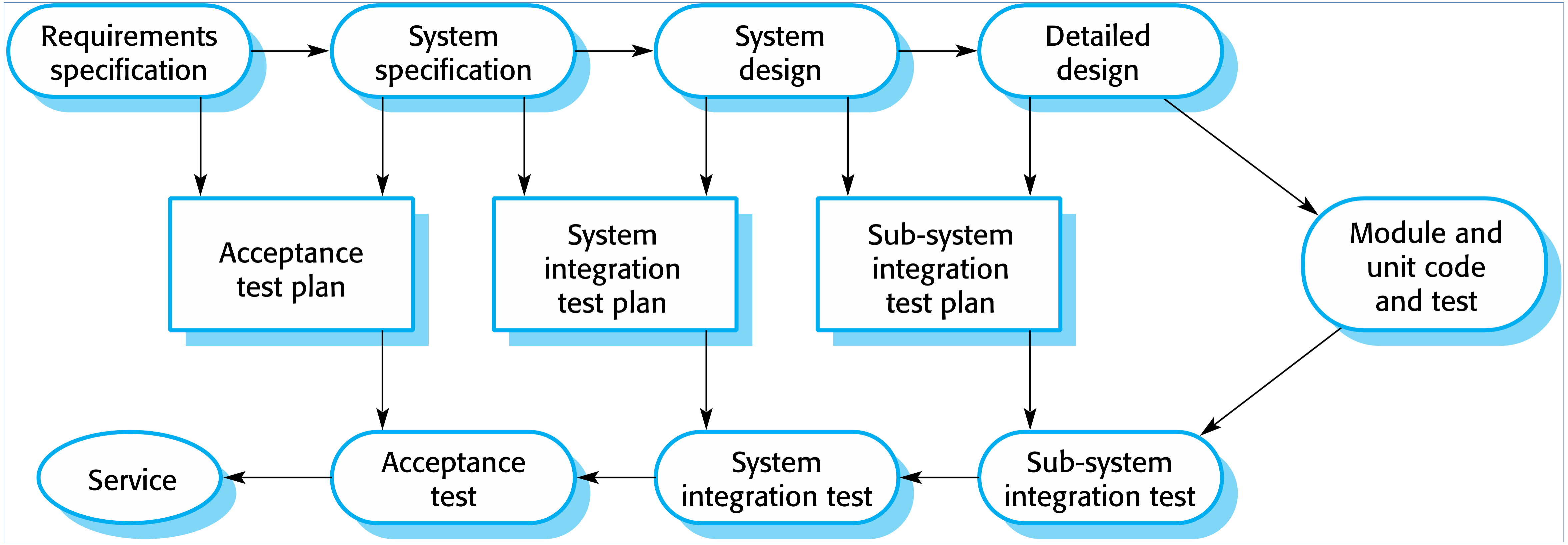
System testing

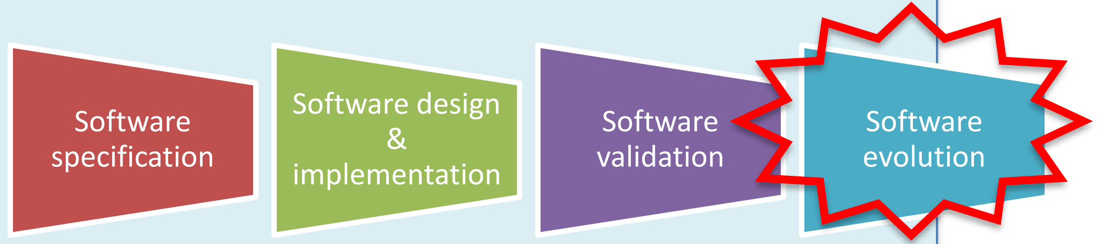
- Testing of the system as a whole.

Acceptance/ Customer testing

- Testing with customer data to check that the system meets the customer's needs.

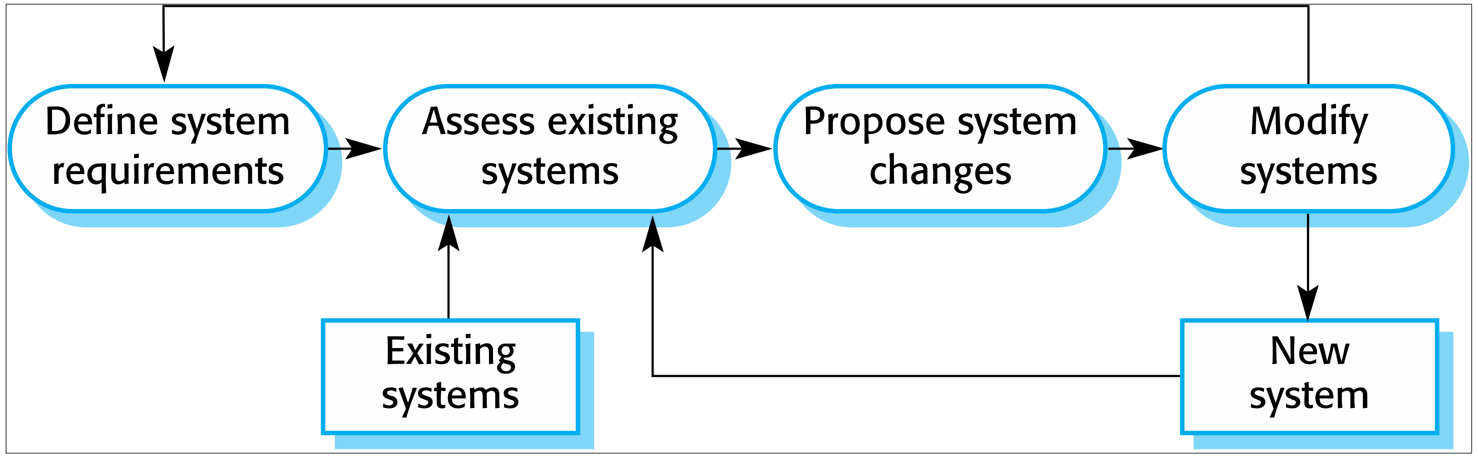
Testing phases in a plan-driven software process (V-model)


Requirements specification
System specification
System design
Detailed design
Acceptance test plan
System integration test plan
Sub-system integration test plan
Module and unit code and test
Service
Acceptance test
System integration test
Sub-system integration test



Software evolution

- Software is inherently flexible and can change.
- As requirements change through changing business circumstances, the software that supports the business must also evolve and change.



Topic 2: Software Processes

Contents

COPING WITH CHANGE

Change is inevitable in all large software projects.

- Business changes lead to new and changed system requirements
- New technologies open up new possibilities for improving implementations
- Changing platforms require application changes

Change leads to rework

Costs of change = cost of rework (e.g. re-analysing requirements) + costs of implementing new functionality

Approaches to reduce the costs of rework

- **Change anticipation**
  - software process includes activities that can anticipate possible changes before significant rework is required.
  - Example : Prototyping

- **Change tolerance**
  - the process is designed so changes can be accommodated at relatively low cost.
  - Example : incremental development.

Two ways of coping with change and changing system requirements

System prototyping

- a version of the system or part of the system is developed quickly to check the customer's requirements and the feasibility of design decisions.
- supports change anticipation.

Incremental delivery

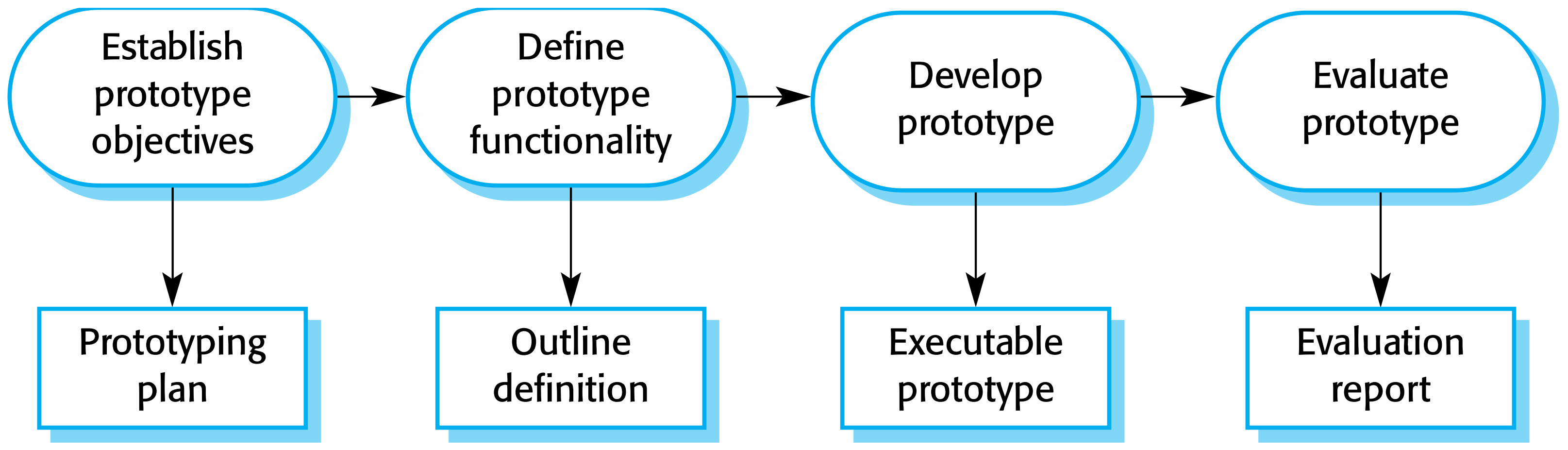
- system increments are delivered to the customer for comment and experimentation.
- supports both change avoidance and change tolerance.

Software prototyping

- A prototype is an initial version of a system used to demonstrate concepts and try out design options.
- A prototype can be used in:
  - The requirements engineering process to help with requirements elicitation and validation;
  - In design processes to explore options and develop a UI design;
  - In the testing process to run back-to-back tests.

- ✓ Improved system usability.
- ✓ A closer match to users' real needs.
- ✓ Improved design quality.
- ✓ Improved maintainability.
- ✓ Reduced development effort.

The process of prototype development



Rapid prototyping languages or tools

May involve leaving out functionality

focus on areas that are not well-understood

Error checking and recovery – Optional!

functional rather than non-functional requirements

Incremental development and delivery

development and delivery is broken down into increments

highest priority requirements are included in early increments

- Incremental development
  - ➢ Develop the system in increments and evaluate each increment before proceeding to the development of the next increment;
  - ➢ Normal approach used in agile methods;
  - ➢ Evaluation done by user/customer proxy.
- Incremental delivery
  - ➢ Deploy an increment for use by end-users;
  - ➢ More realistic evaluation about practical use of software;
  - ➢ Difficult to implement for replacement systems as increments have less functionality than the system being replaced.

Topic 2: Software Processes

Contents

SOFTWARE PROCESS MODELS

PROCESS ACTIVITIES

COPING WITH CHANGE

PROCESS IMPROVEMENTS

Process Improvement

- a way of enhancing the quality of their software, reducing costs or accelerating their development processes.
- understands existing processes and changing these processes to increase product quality and/or reduce costs and development time.

Approaches to improvement

The process maturity approach

Focuses:

- improving process and project management
- introducing good software engineering practice.

The level of process maturity reflects the extent to which good technical and management practice has been adopted in organizational software development processes.

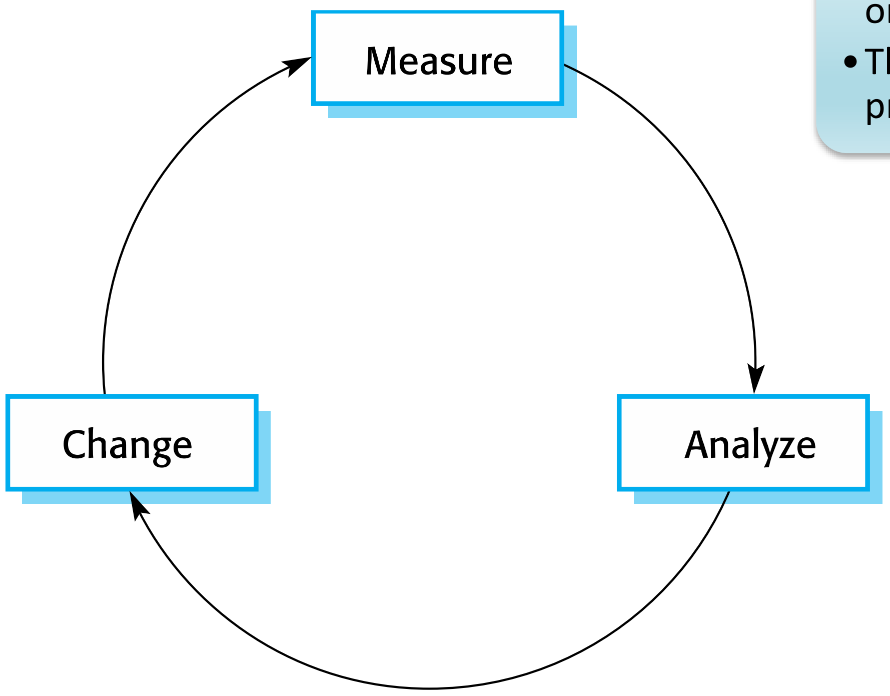
The agile approach

Focuses :

- iterative development
- the reduction of overheads in the software process.

The primary characteristics of agile methods are rapid delivery of functionality and responsiveness to changing customer requirements.

The process improvement cycle



Process measurement

- Measure one or more attributes of the software process or product.
- These measurements forms a baseline that to decide if process improvements have been effective.

Process metrics

- *Time taken*
- *Resources required*
- *Occurrences Number*

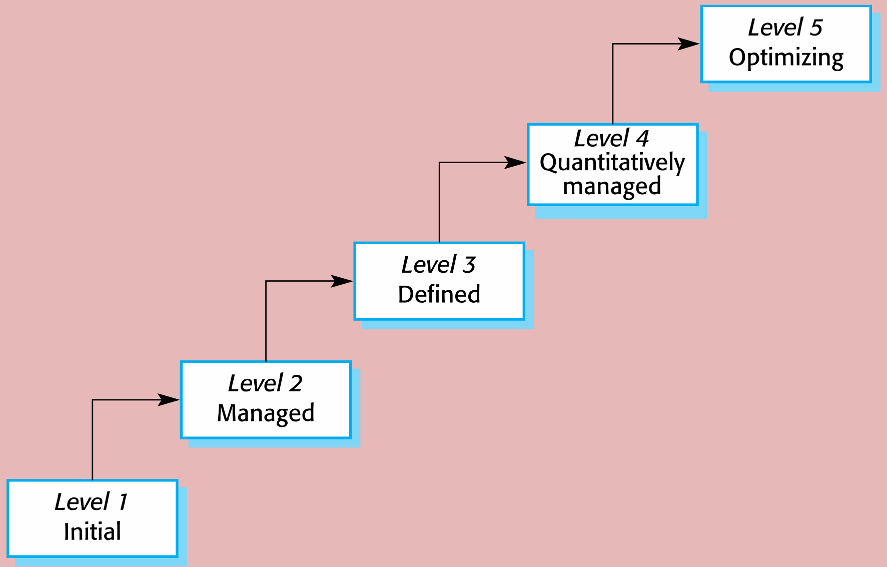
Process analysis

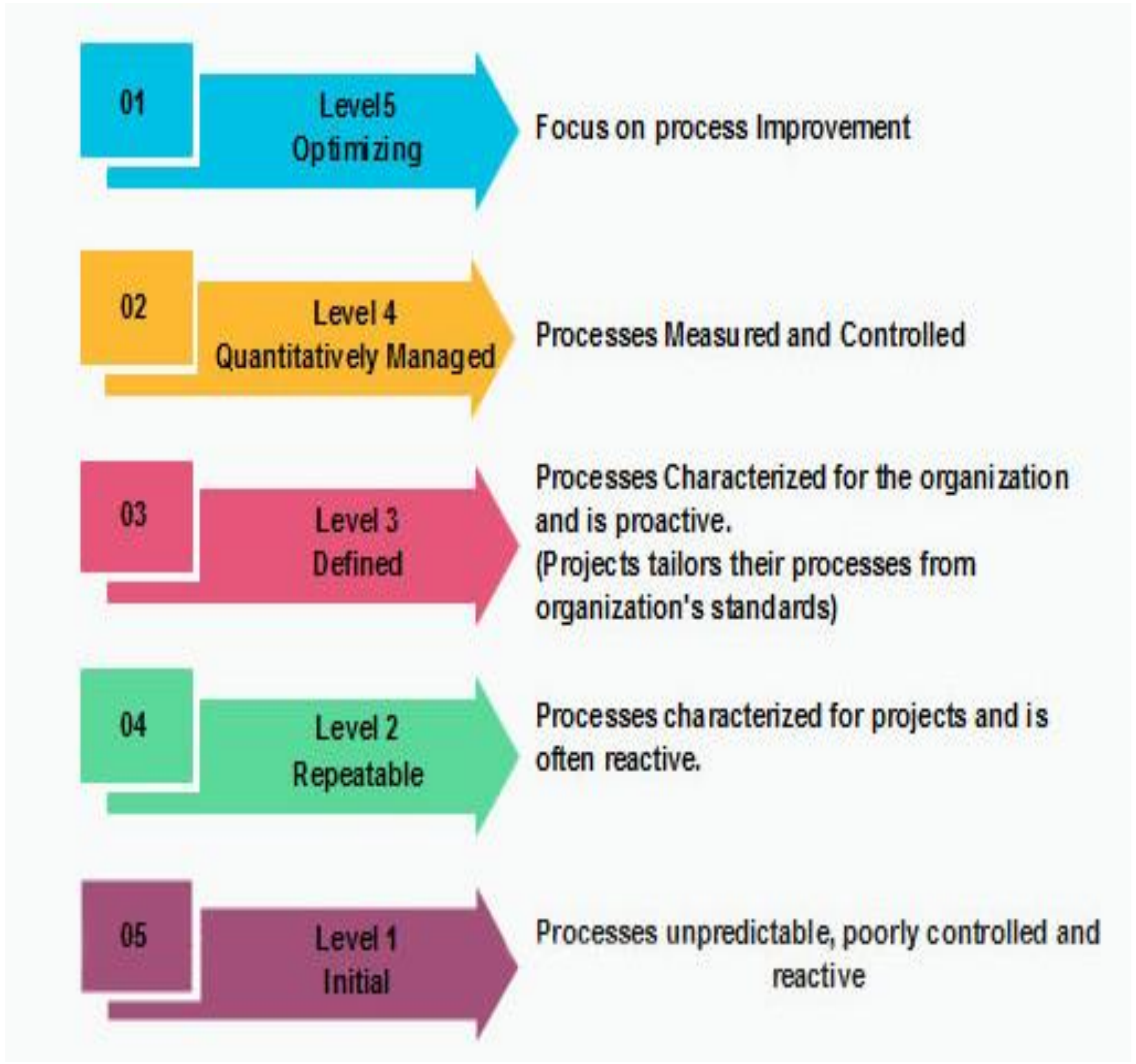
- The current process is assessed, and process weaknesses and bottlenecks are identified.
- Develop the process models (sometimes called process maps) to describe the process.

Process change

- Proposed to address some of the identified process weaknesses.
- When the changes are introduced, the cycle resumes to collect data about the effectiveness of the changes.

Process maturity models





01
Level5
Optimizing
Focus on process Improvement
02
Level 4
Quantitatively Managed
Processes Measured and Controlled
03
Level 3
Defined
Processes Characterized for the organization and is proactive.
(Projects tailors their processes from organization's standards)
04
Level 2
Repeatable
Processes characterized for projects and is often reactive.
05
Level 1
Initial
Processes unpredictable, poorly controlled and reactive

Key points (1/2)

- Software processes are the activities involved in producing a software system.
- Software process models are abstract representations of these processes.
- General process models describe the organization of software processes.
  - Examples of these general models include the ‘waterfall’ model and incremental development
- Requirements engineering (RE) is the process of developing a software specification.
- Design and implementation processes are concerned with transforming a requirements specification into an executable software system.
- Software validation is the process of checking that the system conforms to its specification and that it meets the real needs of the users of the system.
- Software evolution takes place when you change existing software systems to meet new requirements. The software must evolve to remain useful.
- Processes should include activities such as prototyping and incremental delivery to cope with change.

Key points (2/2)

- Processes may be structured for iterative development and delivery so that changes may be made without disrupting the system as a whole.
- The principal approaches to process improvement are agile approaches, geared to reducing process overheads, and maturity-based approaches based on better process management and the use of good software engineering practice.
- The SEI process maturity framework identifies maturity levels that essentially correspond to the use of good software engineering practice.