

Software Process & Process Models

Lecture # 9, 10
11, 12 Feb

Rubab Jaffar
rubab.jaffar@nu.edu.pk

Software Engineering CS-303



Today's Outline

- Integration and Configuration
- Software prototyping
- Incremental development process model
- Software process Activities

DFD Example level 0 and level 1

- <https://www.visual-paradigm.com/tutorials/data-flow-diagram-example-food-ordering-system.jsp>

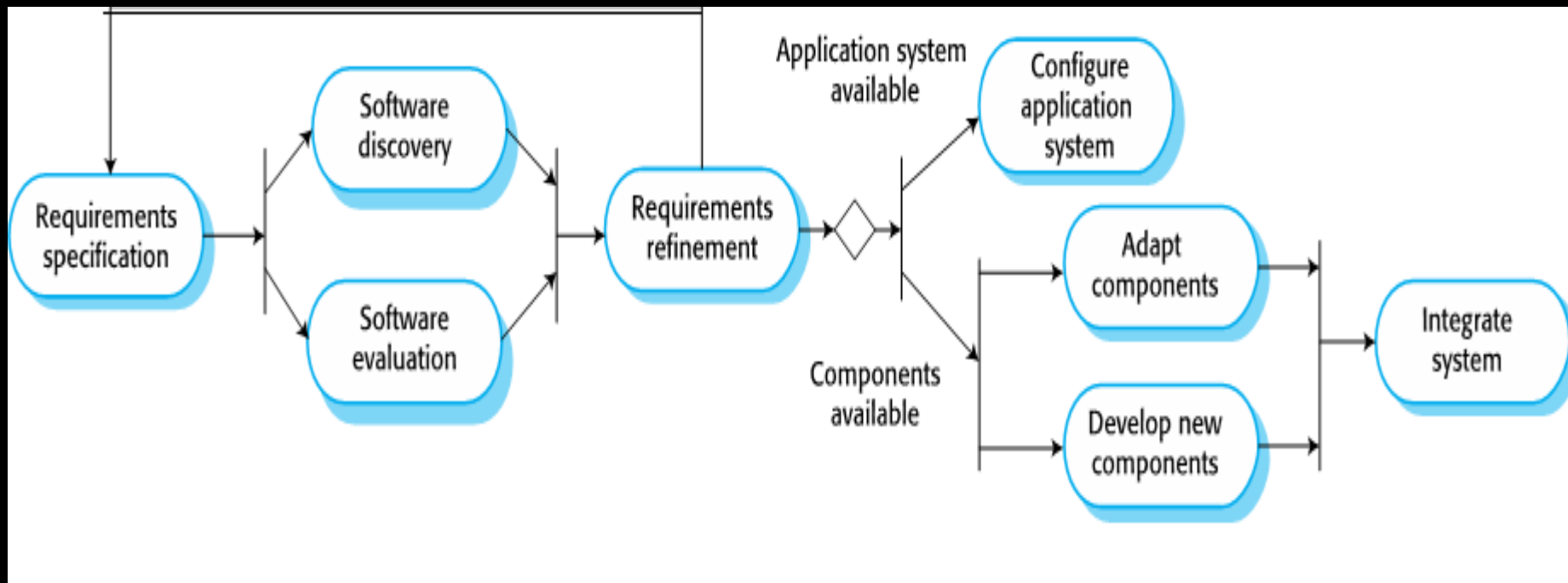
Integration and Configuration

- Based on software reuse where systems are integrated from existing components or application systems (sometimes called COTS -Commercial-off-the-shelf- systems).
- Reused elements may be configured to adapt their behaviour and functionality to a user's requirements
- Reuse is now the standard approach for building many types of business system
 - Reuse covered in more depth in Chapter 15.

Types of Reusable Software

- Stand-alone application systems (sometimes called COTS) that are configured for use in a particular environment.
- Collections of objects that are developed as a package to be integrated with a component framework such as .NET or J2EE.
- Web services that are developed according to service standards and which are available for remote invocation.

Reuse-oriented Software Engineering



Key Process Stages

- Requirements specification
- Software discovery and evaluation
- Requirements refinement
- Application system configuration
- Component adaptation and integration

Advantages and Disadvantages

- Reduced costs and risks as less software is developed from scratch
- Faster delivery and deployment of system
- But requirements compromises are inevitable so system may not meet real needs of users
- Loss of control over evolution of reused system elements

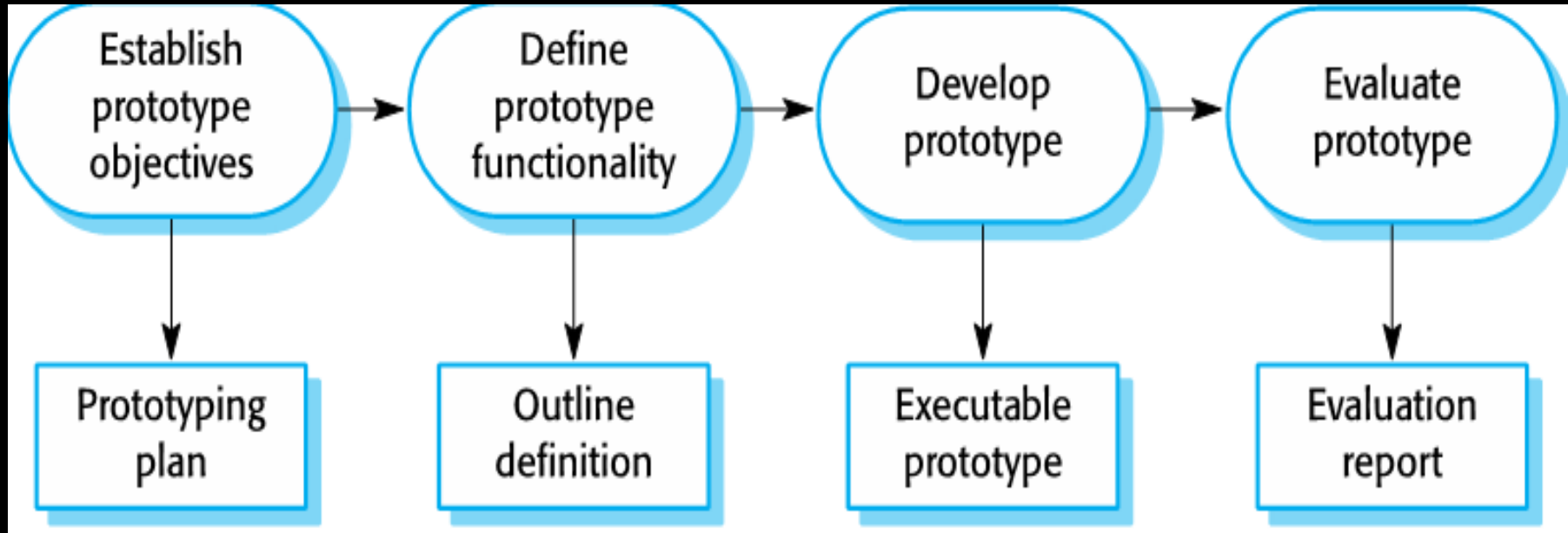
Software Prototyping

- 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.

Benefits of Prototyping

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

The Process of Prototype Development



Prototype Development

- May be based on rapid prototyping languages or tools
- May involve leaving out functionality
 - Prototype should focus on areas of the product that are not well-understood;
 - Error checking and recovery may not be included in the prototype;
 - Focus on functional rather than non-functional requirements such as reliability and security

Throw-away Prototypes

- Prototypes should be discarded after development as they are not a good basis for a production system:
 - It may be impossible to tune the system to meet non-functional requirements;
 - Prototypes are normally undocumented;
 - The prototype structure is usually degraded through rapid change;
 - The prototype probably will not meet normal organisational quality standards.

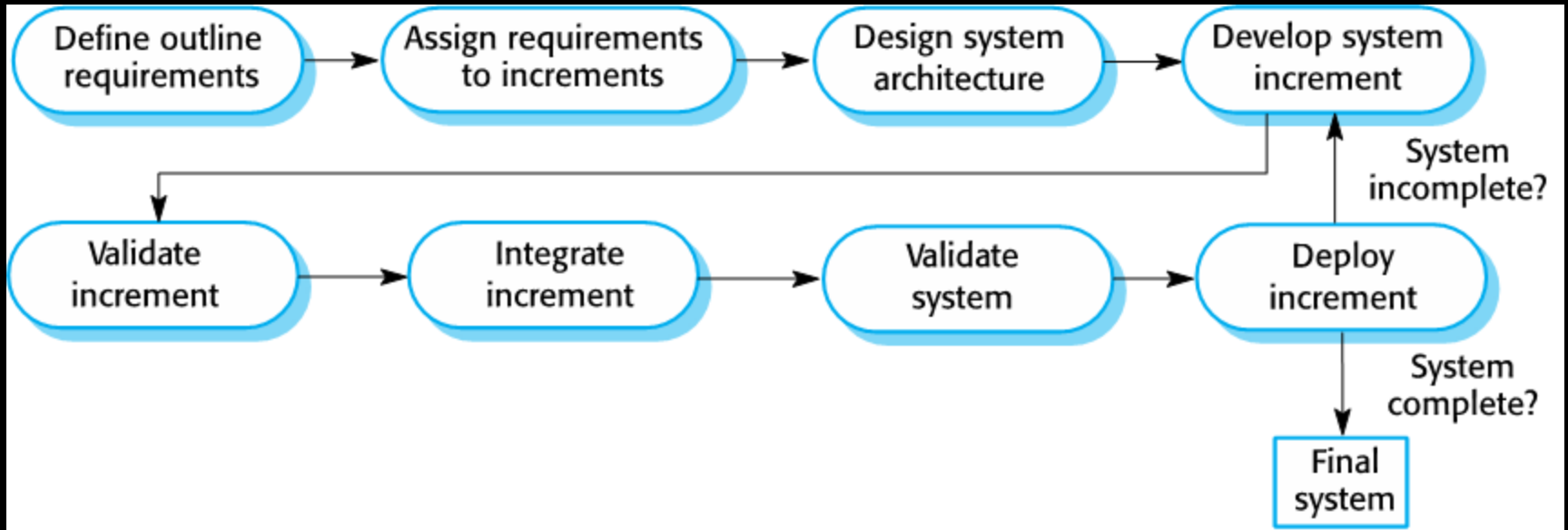
Incremental Delivery

- Rather than deliver the system as a single delivery, the development and delivery is broken down into increments with each increment delivering part of the required functionality.
- User requirements are prioritised and the highest priority requirements are included in early increments.
- Once the development of an increment is started, the requirements are frozen though requirements for later increments can continue to evolve.

Incremental Development and Delivery

- 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.

Incremental Delivery



Incremental Delivery Advantages

- Customer value can be delivered with each increment so system functionality is available earlier.
- Early increments act as a prototype to help elicit requirements for later increments.
- Lower risk of overall project failure.
- The highest priority system services tend to receive the most testing.

Process Activities

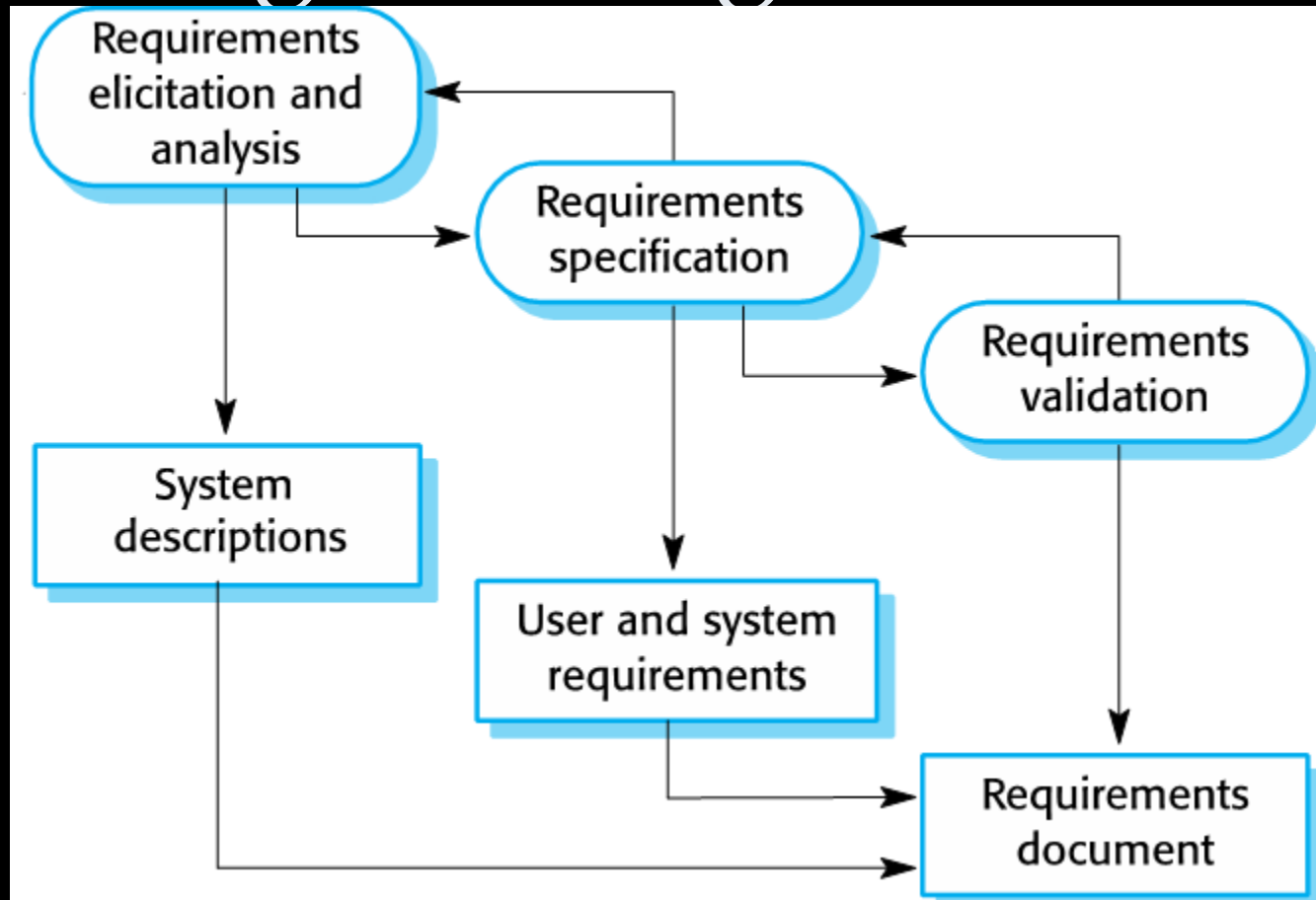
Process Activities

- Real software processes are inter-leaved sequences of technical, collaborative and managerial activities with the overall goal of specifying, designing, implementing and testing a software system.
- The four basic process activities of specification, development, validation and evolution are organized differently in different development processes.
- For example, in the waterfall model, they are organized in sequence, whereas in iterative development they are interleaved.

Software Specification

- The process of establishing what services are required and the constraints on the system's operation and development.
- Requirements engineering process
 - Requirements elicitation and analysis
 - What do the system stakeholders require or expect from the system?
 - Requirements specification
 - Defining the requirements in detail
 - Requirements validation
 - Checking the validity of the requirements

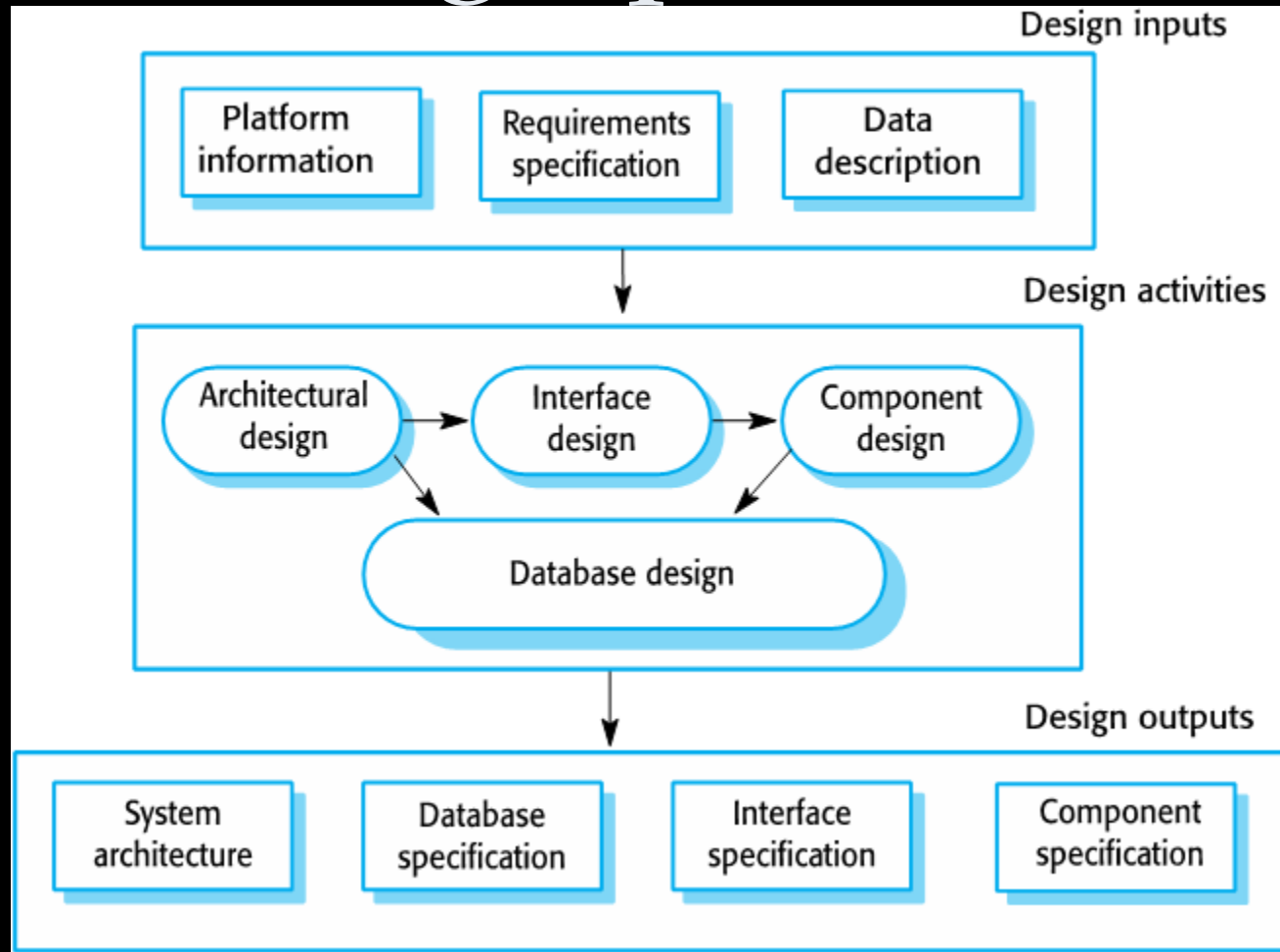
The Requirements Engineering Process



Software Design and Implementation

- The process of converting the system specification into an executable system.
- Software design
 - Design a software structure that realises 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



Design Activities

- *Architectural design*, where you identify the overall structure of the system, the principal components (subsystems or modules), their relationships and how they are distributed.
- *Database design*, where you design the system data structures and how these are to be represented in a database.
- *Interface design*, where you define the interfaces between system components.
- *Component selection and design*, where you search for reusable components. If unavailable, you design how it will operate.

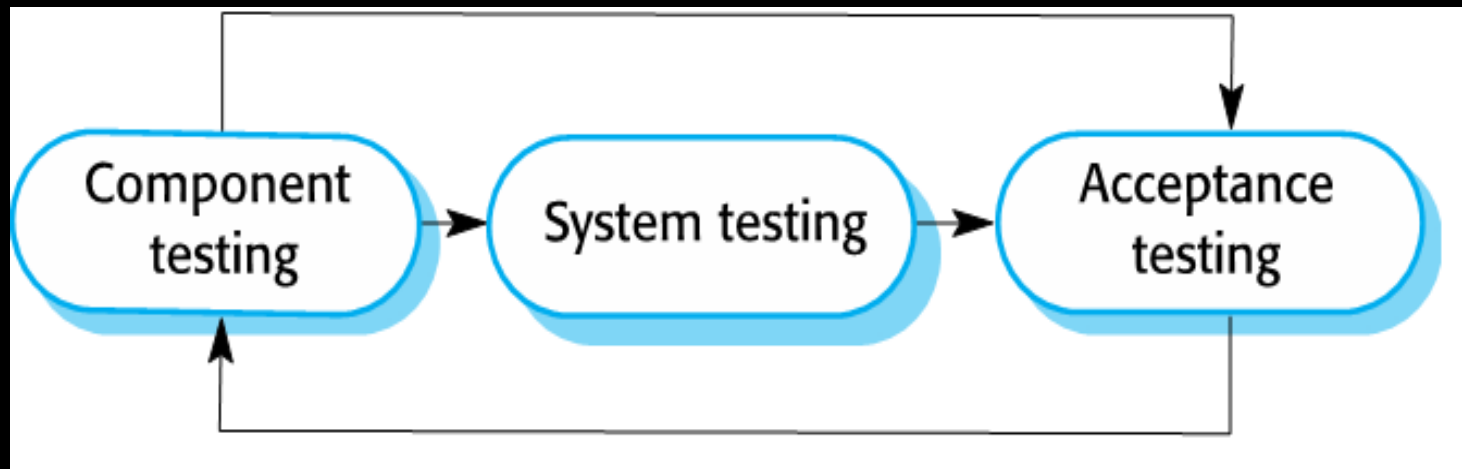
System Implementation

- 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 Validation

- Verification and validation (V & V) is intended 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.
- System testing involves 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.

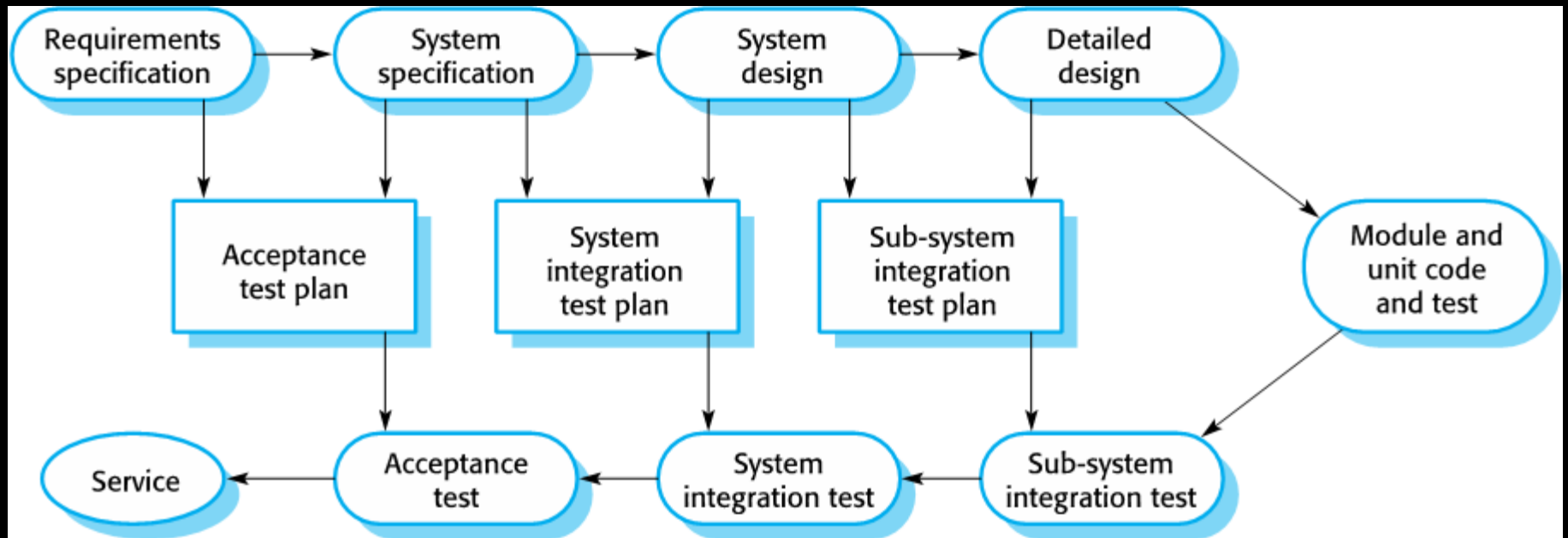
Stages of Testing



Testing Stages

- 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. Testing of emergent properties is particularly important.
- Customer testing
 - Testing with customer data to check that the system meets the customer's needs.

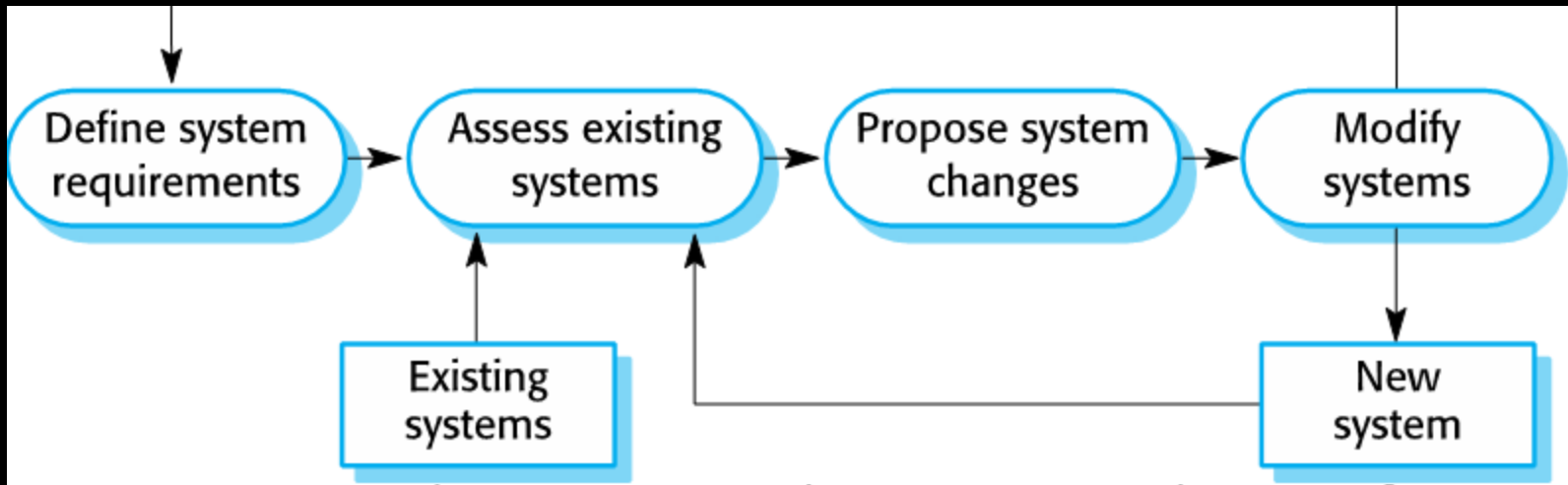
Testing phases in a plan-driven software process (V-model)



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.
- Although there has been a demarcation between development and evolution (maintenance) this is increasingly irrelevant as fewer and fewer systems are completely new.

System Evolution





That is all