



Chapter 2 – Software Processes

Topics covered



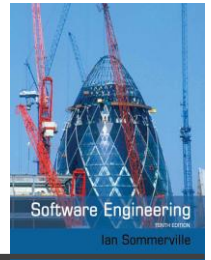
- Software process models
- Process activities
- Coping with change
- Process improvement

The software process



- A structured set of activities required to develop a software system.
- Many different software processes but all involve:
 - Specification – defining what the system should do;
 - Design and implementation – defining the organization of the system and implementing the system;
 - Validation – checking that it does what the customer wants;
 - Evolution – changing the system in response to changing customer needs.
- A software process model is an abstract representation of a process. It presents a description of a process from some particular perspective.

Plan-driven and agile processes



- Plan-driven processes are processes where all of the process activities are planned in advance and progress is measured against this plan.
- In agile processes, planning is incremental and it is 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.

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 incremental development they are interleaved.

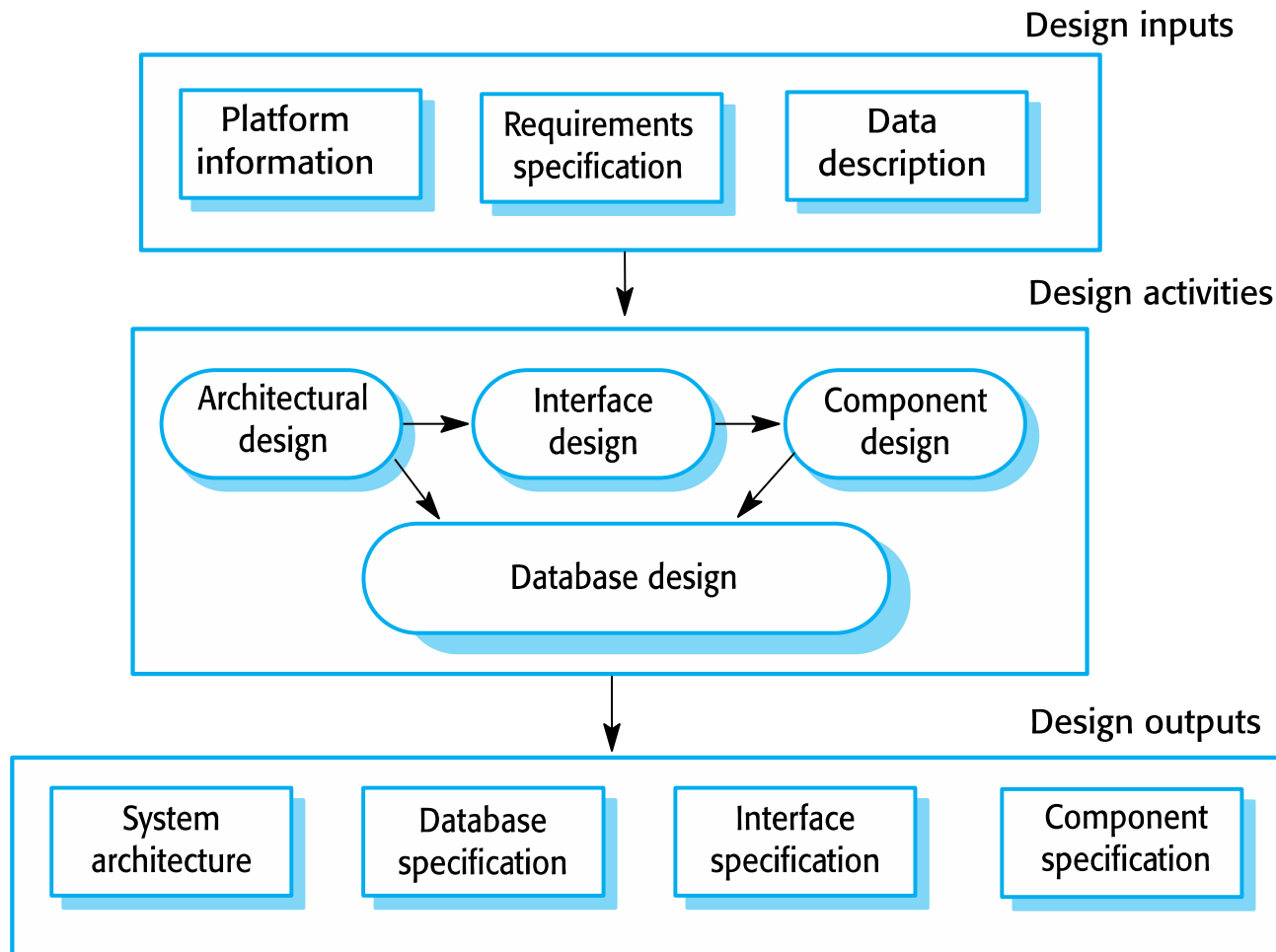
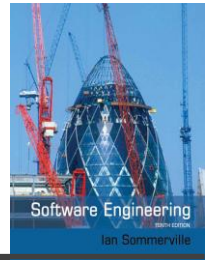


- The process of establishing what services are required and the constraints on the system's operation and development.
- Requirements engineering process
 - Requirements elicitation and analysis
 - What do the system stakeholders require or expect from the system?
 - Requirements specification
 - Defining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

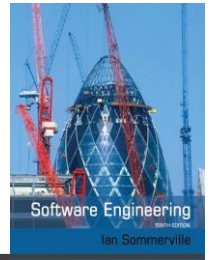


- The process of converting the system specification into an executable system.
- Software design
Design a software structure that realizes the specification;
- Implementation
Translate this structure into an executable program;
- The activities of design and implementation are closely related and may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

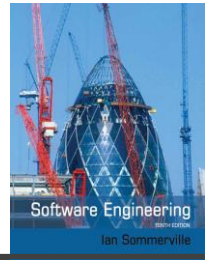
A general model of the design process



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.

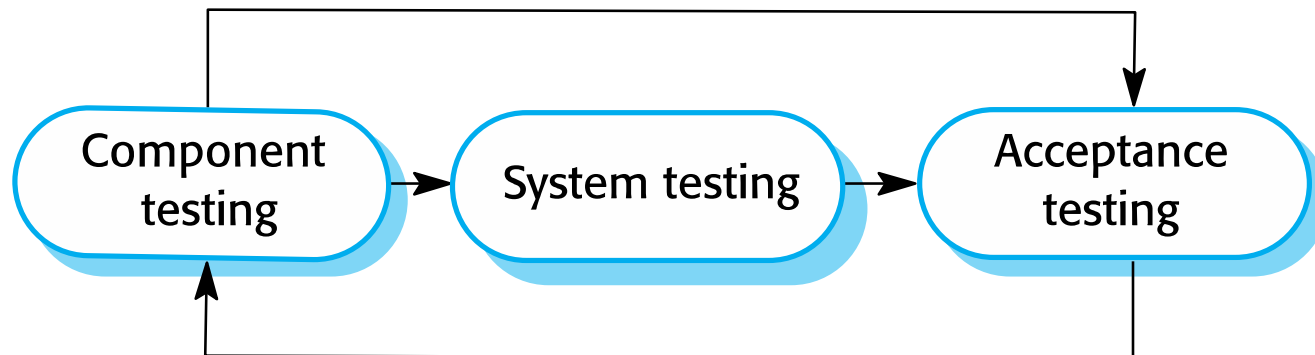


- The software is implemented either by developing a program or programs or by configuring an application system.
- Design and implementation are interleaved activities for most types of software system.
- Programming is an individual activity with no standard process.
- Debugging is the activity of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



- 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



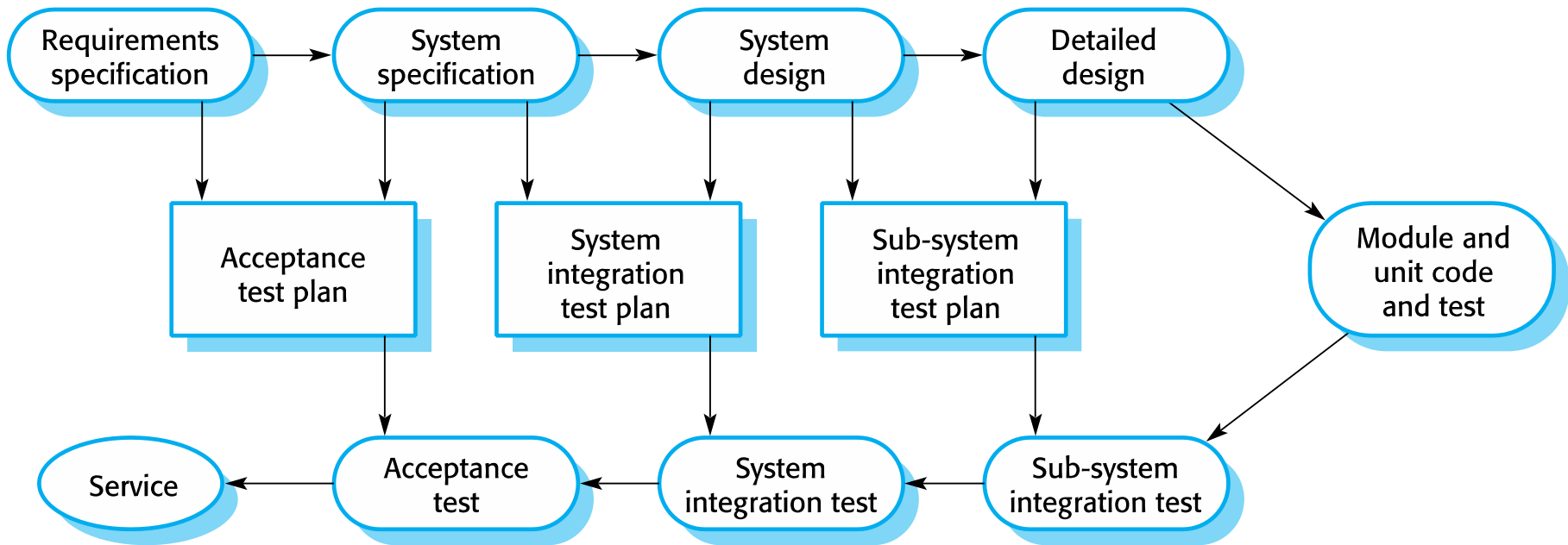
Testing stages

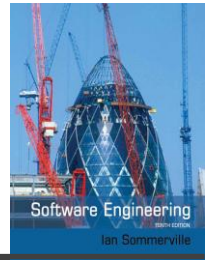


- **Component testing**
Individual components are tested independently;

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.

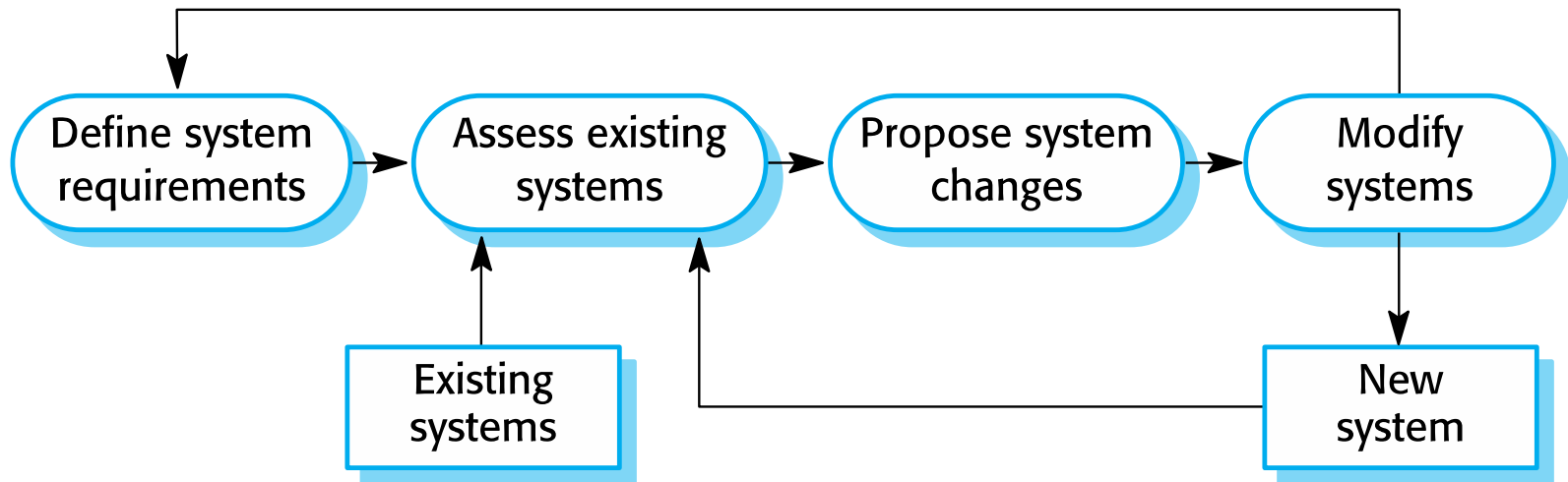
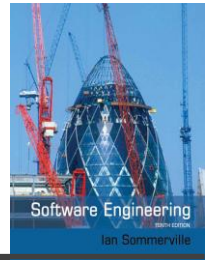
Testing phases in a plan-driven software process (V-model)

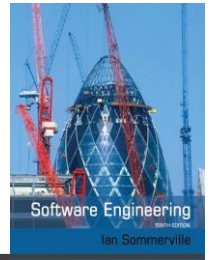




- Software is inherently flexible and can change.
- As requirements change through changing business circumstances, the software that supports the business must also evolve and change.
- Although there has been a demarcation between development and evolution (maintenance) this is increasingly irrelevant as fewer and fewer systems are completely new.

System evolution





- 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 so the costs of change include both rework (e.g. re-analysing requirements) as well as the costs of implementing new functionality