Software Process

- What is a 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

Software Process

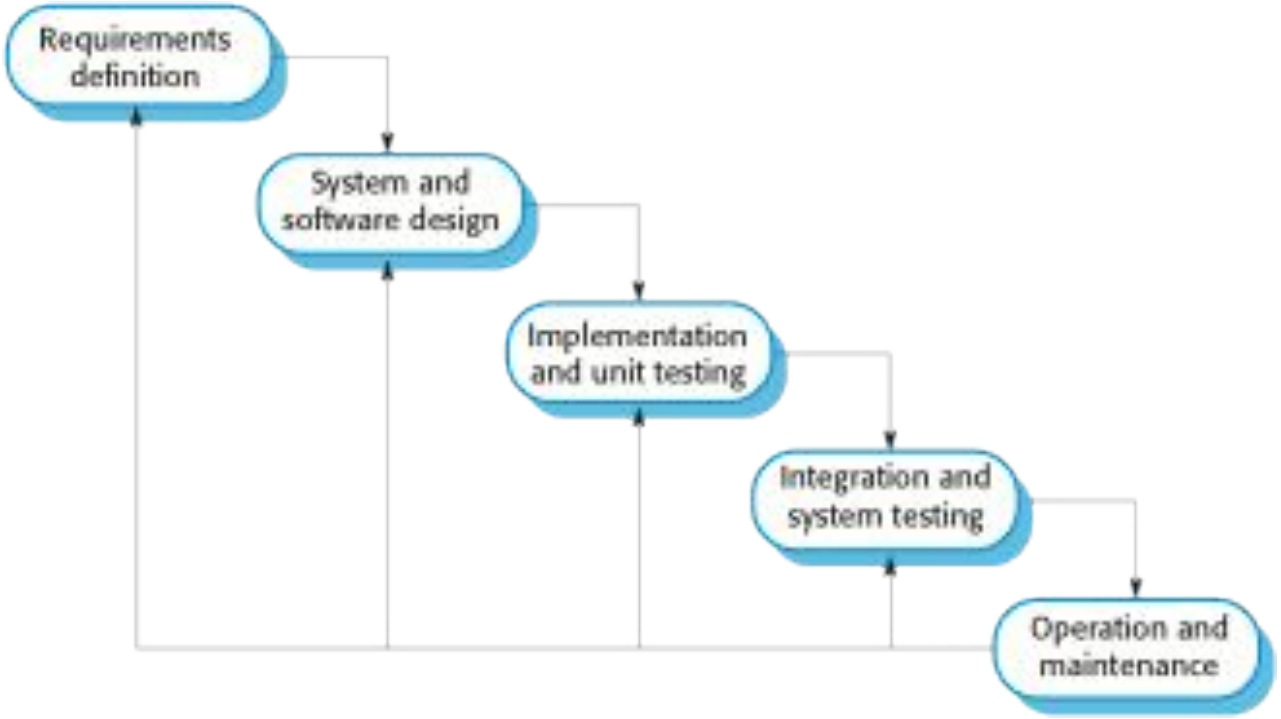
- Why are processes important?
- When we describe and discuss processes,
- we usually talk about the activities in these processes such as specifying a data model, designing a user interface, etc. and the ordering of these activities.
- The artifacts generated by the activities in these processes are an indication of the progress done by the software development team.
- These are also part of the various documents like the SRS and SDS

Software Process

- Attributes of a software process?
- Process descriptions may also include:
  - Products, which are the outcomes of a process activity;
  - Roles, which reflect the responsibilities of the people involved in the process;
  - Pre- and post-conditions, which are statements that are true before and after a process activity has been enacted or a product produced.
  - In a software development team having a well defined process in place is also a part of the project team management.
  - What are the steps involved in the planning of a birthday party?
  - What are the various activities within those steps/phases in the planning of the birthday party?

Software Process Models

- Waterfall Process Model



Waterfall Process Model

- There are separate identified phases in the waterfall model:
  - Requirements analysis and definition
  - System and software design
  - Implementation and unit testing
  - Integration and system testing
  - Operation and maintenance
- The main drawback of the waterfall model is the difficulty of accommodating change after the process is underway. In principle, a phase has to be complete before moving onto the next phase.

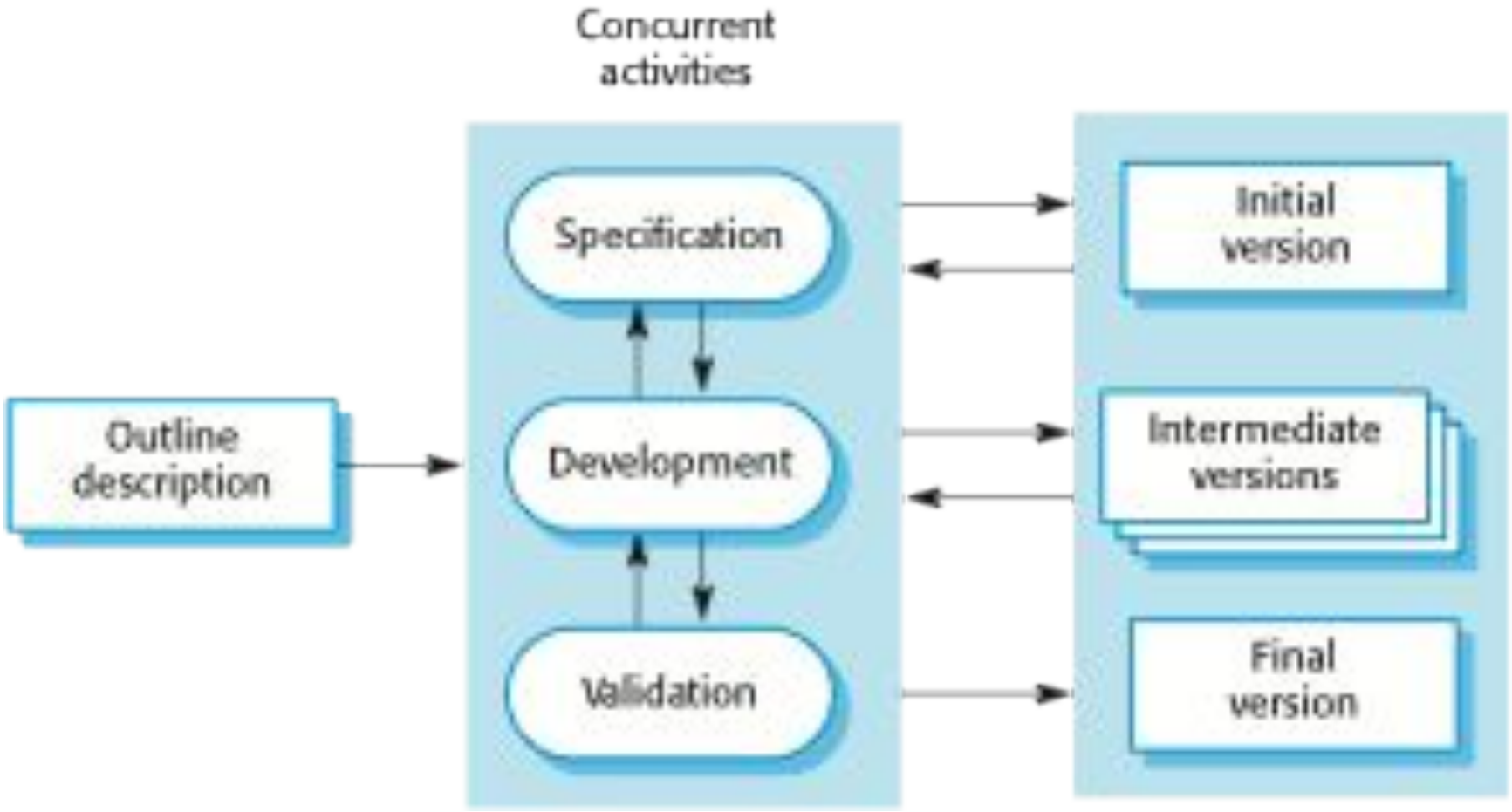
Waterfall Process Model

- Inflexible partitioning of the project into distinct stages makes it difficult to respond to changing customer requirements.
  - Therefore, this model is only appropriate when the requirements are well-understood and changes will be fairly limited during the design process.
  - Few business systems have stable requirements.
- The waterfall model is mostly used for large systems engineering projects where a system is developed at several sites.
  - In those circumstances, the plan-driven nature of the waterfall model helps coordinate the work.

Examples of the systems projects using waterfall model approach

- Energy management systems for electrical utilities and rail traffic control systems for railways. These systems always included delivery of hardware and software for the control center and often include remote telemetry hardware. The rail traffic control systems were often part of a major development with a big infrastructure component. Athens systems always included a lot of database design and development. As these systems were used by power system dispatchers and rail traffic dispatchers there was big user interface component.
- Some of the systems that I participated in the development and delivery (as I remember) were:
- A large Energy Management System (EMS) for Southern California Edison (early 1970s)
- A large EMS system for Florida Power and Light. (1980s)
- A large EMS for Houston Light and Power (late 1980s)
- A SCADA, AGC, and Water Housekeeping system for Statkraft Tokke Control Center (1990s)
- A large EMS system for Hong Kong Electric Company (1990s)
- Tren Urbano, a heavy rail transportation system for San Juan Puerto Rico (1990s)
- A large rail traffic control system for Canadian National Railway (early 2000s)
- Most of these projects took three to five years to complete. They all had stringent safety requirements.
- Reference: Michael Tipton Retired Software Project Manager Siemens (1995 – 2005)

Incremental Process Model



Iterative Approach

- The cost of accommodating changing customer requirements is reduced.
  - The amount of analysis and documentation that has to be redone is much less than is required with the waterfall model.
- It is easier to get customer feedback on the development work that has been done.
  - Customers can comment on demonstrations of the software and see how much has been implemented.
- More rapid delivery and deployment of useful software to the customer is possible.
  - Customers are able to use and gain value from the software earlier than is possible with a waterfall process.

Iterative Approach

- The process is not visible.
  - Managers need regular deliverables to measure progress. If systems are developed quickly, it is not cost-effective to produce documents that reflect every version of the system.
- System structure tends to degrade as new increments are added.
  - Unless time and money is spent on refactoring to improve the software, regular change tends to corrupt its structure. Incorporating further software changes becomes increasingly difficult and costly.

Reuse Oriented Approach

- Based on systematic reuse where systems are integrated from existing components or COTS (Commercial-off-the-shelf) systems.
- Process stages
  - Component analysis;
  - Requirements modification;
  - System design with reuse;
  - Development and integration.
- Reuse is now the standard approach for building many types of business system

Reuse Oriented Approach

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. In the waterfall model, they are organized in sequence, whereas in incremental development they are inter-leaved

Prescriptive Vs Adaptive Process

- 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.
- The success of the project depends on choice of process, skills, time, budget and effort.

Questions

- What is a software process?
- A software process is set of various activities in producing a software system.
- What is a software process model?
- A software process model is an abstract representation of various set of steps that make up the software process.
- What are the general process models?
- The general process models are waterfall model, incremental development and the ruse oriented approach.

Questions

- Give reasons for your answer based on the type of system to be developed, suggest the most generic process model that might be used as a basis for managing the development of the following systems.
- 1. A mobile phone
- 2. An automobile engine
- 3. A embedded software for a driverless car
- 4. An antilock braking system for a car
- 5. A virtual reality system for software maintenance

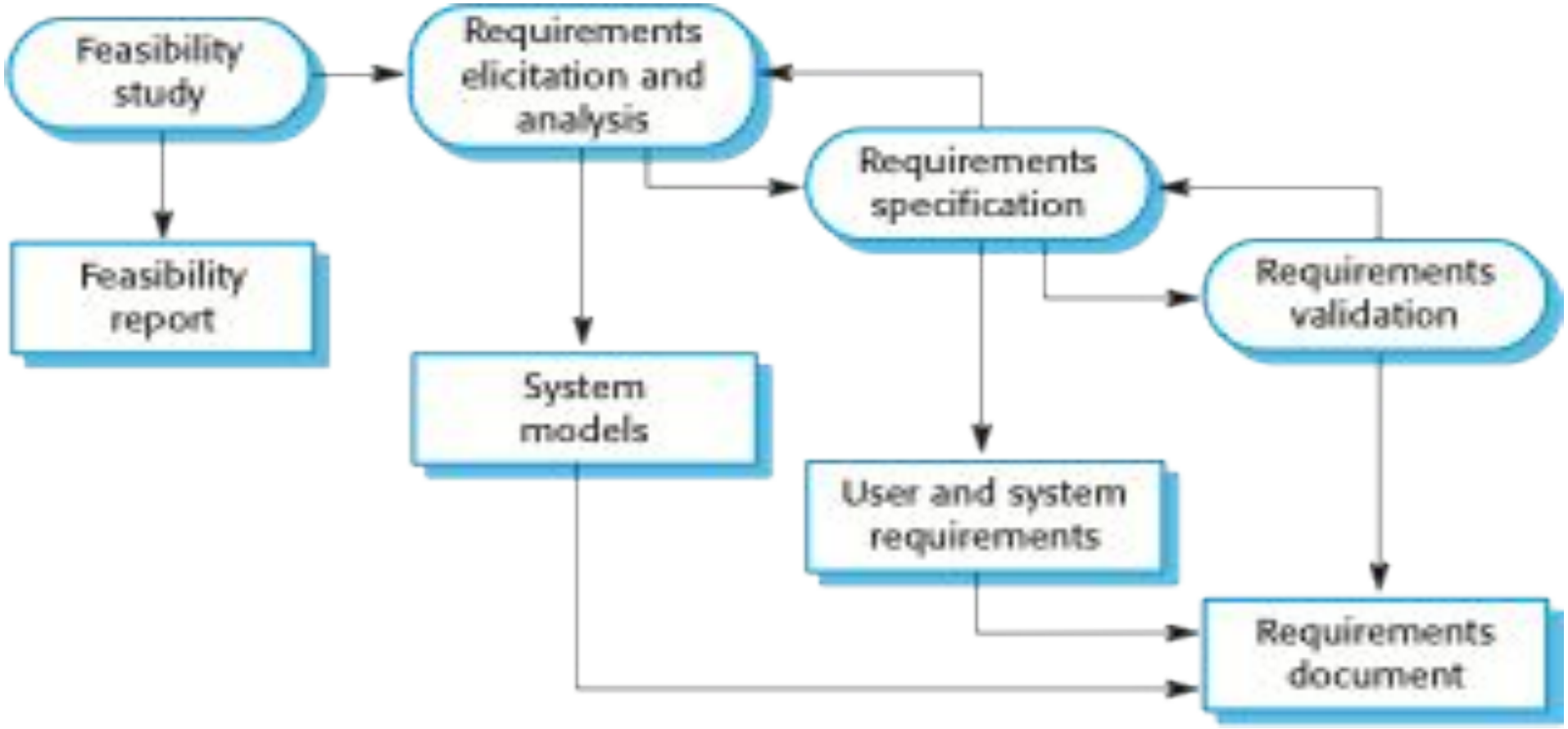
Questions

- A university accounting system
- An interactive travel planner that help user plans the journey.

Software Specification

- The process of establishing what services are required and the constraints on the system's operation and development.
- Requirements engineering process
  - Feasibility study
    - Is it technically and financially feasible to build the system?
  - 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 requirem

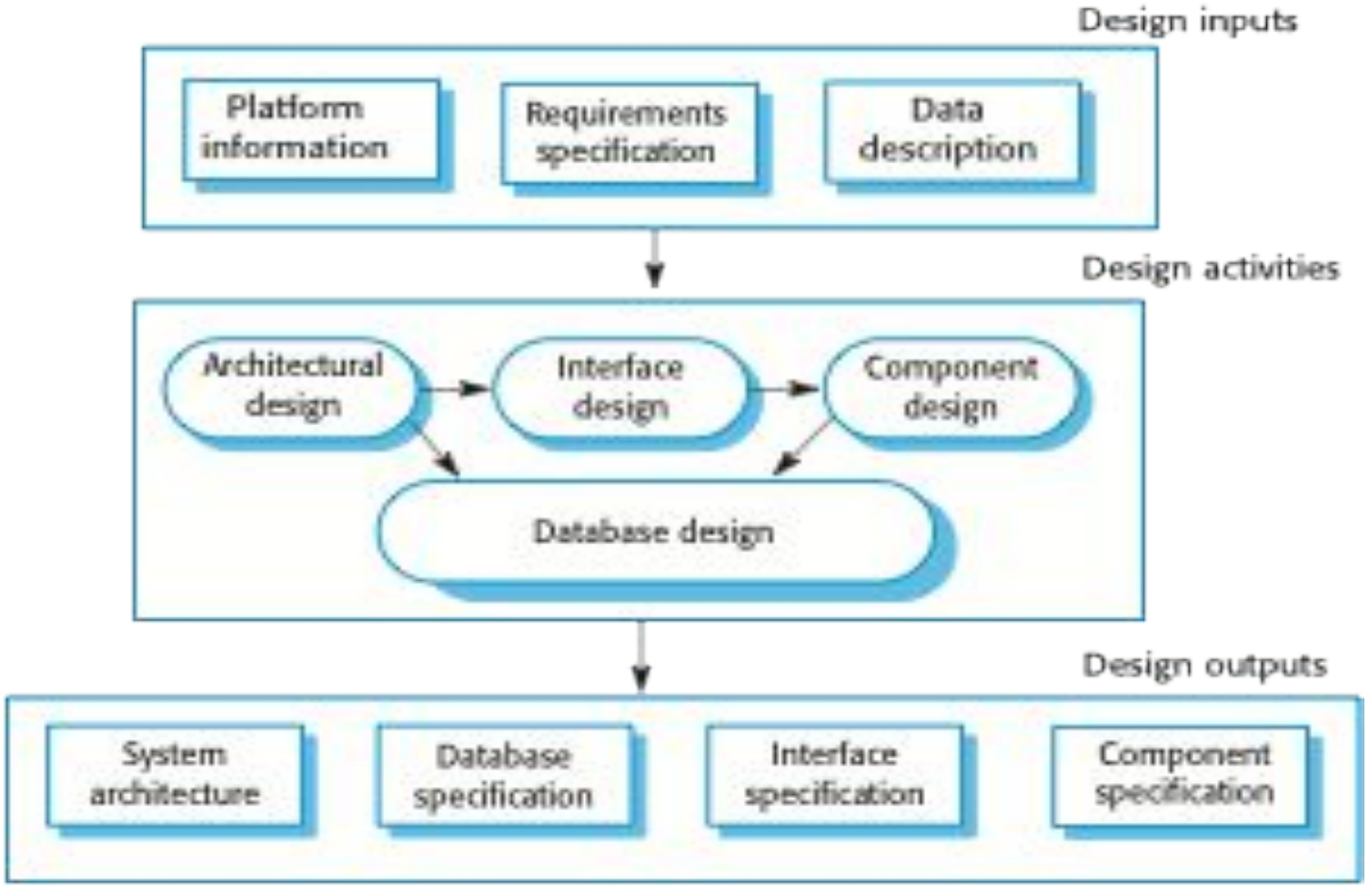
Requirements Engineering Process



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.

General Model of Software Design



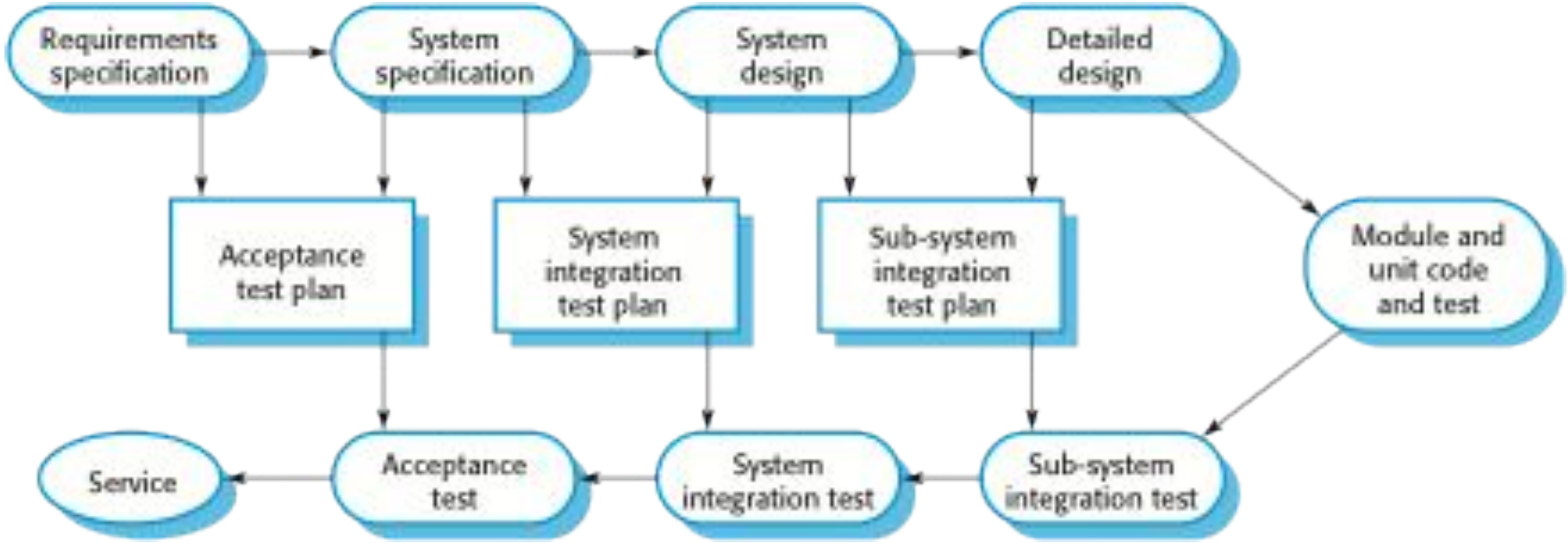
Design Activities

- *Architectural design,* where you identify the overall structure of the system, the principal components (sometimes called sub-systems or modules), their relationships and how they are distributed.
- *Interface design,* where you define the interfaces between system components.
- *Component design,* where you take each system component and design how it will operate.
- *Database design,* where you design the system data structures and how these are to be represented in a database.

Testing

- Development or 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.
- Acceptance testing
  - Testing with customer data to check that the system meets the customer’s needs.

Testing Plan for a Planned Driven Approach



Questions

- What is requirements engineering?
- Requirements engineering is the process of developing a software specification.
- What is design and implementation?
- Design and implementation processes are concerned with transforming a requirements specification into an executable software system.
- What is software validation?
- 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.
- What is software evolution?
- Software evolution takes place when you change existing software systems to meet new requirements. The software must evolve to remain useful.

References

- Ian Sommerville, S*oftware Engineering,* Pearson education, 9th edition
- Wikipedia