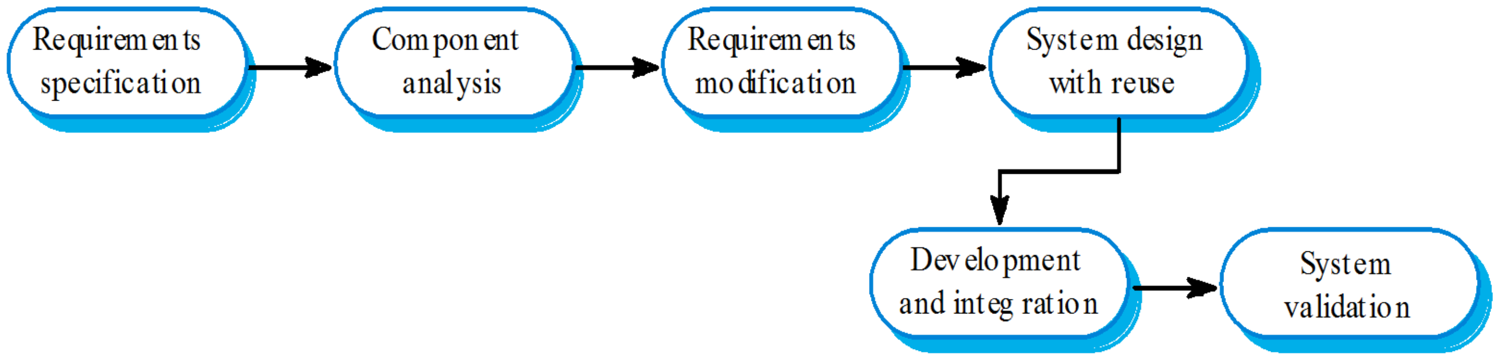
Component-based software engineering

- Based on systematic reuse where systems are integrated from existing components.
- People working on the project → Know of design or code (similar to that required)→ Modify them as needed → incorporate them into their system
- Process stages
  - Component analysis
  - Requirements modification
  - System design with reuse
  - Development and integration
- This approach is becoming increasingly used

Component-based software engineering


Requirements specification
Component analysis
Requirements modification
System design with reuse
Development and integration
System validation

Component-based software Engineering

Component analysis

- Given the requirement specification
- Search is made for components
- Usually there is no exact match.

Requirement Modification

- Requirements are analyzed—using information of discovered components
- If modifications are impossible— component analysis activity may be re-entered to search for alternate solution

System Design with Reuse

- Frame work of system is designed or an existing frame work is reused.

Development and integration Software

- Existing software /modified and newly developed components are integrated.

Component-based software Engineering

Advantages

- Reduce the amount of software to be developed
- Reducing cost and risk
- Faster delivery of software

Disadvantages

- Requirement changes—may lead to a system that does not meet the real needs of users
- Control over the system evolution is lost

Process iteration

- Change is necessary in all large software projects.
- What to do?
- Iteration means earlier stages are reworked in the process for large systems

Two (related) process models

- Incremental delivery
- Spiral development

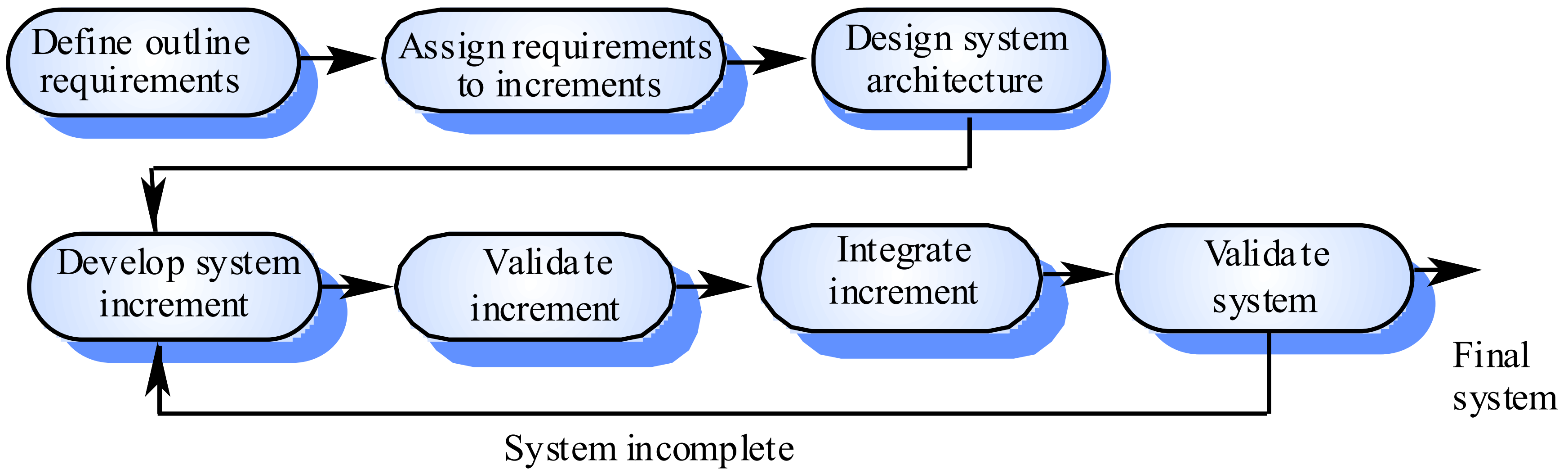
Incremental Approach

- Waterfall model & evolutionary approach?...in case of change
- Incremental delivery—in-between approach

"The software specification, design and implementation are broken down into a series of increments that are each developed in turn"

- System development is decomposed into increments and each delivers a proportion of the system.
- Increments are developed based on their requirement priorities.

Incremental development



Incremental delivery(Steps)

1.Customer identify—services provided by the system

2. Then identify which of the services are most important and which are least important

3. No of delivery increments are then identified with sub-set of the system functionality

4.Highest priority services delivered first

5. Requirements for the first increment are defined in detail

6. First Increment is developed and delivered ..customer can put into service.(Benefit for customer?)

7.During development ….further requirements analysis for later increments can take place

Incremental Model

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.

Disadvantages:

- Increments should be relatively small (20,000 lines of code).
- Can be difficult to map the customer’s requirements onto increments of the right size.
- Hard to identify common functions.

Spiral model

- Rather than represent the software process as a sequence of activities with some backtracking… process is represented as spiral
- Each loop in spiral represents a phase of the software process
- Innermost loop might be concerned with system feasibility
- Next loop with requirements definition…next with system designed and so on.

Spiral model sectors

Each loop in spiral is split into four sectors

- **Objective setting**
  - Specific objectives for that phase of the project are identified
  - Identify Constraints → on process & product.
  - Detail management plan is drawn up
  - Identify Project risks
  - Alternative strategies on these risks ...may be planned

Spiral model sectors

Risk assessment and reduction

- For every identified risk → Detailed analysis is carried out
- Steps are taken to reduce the risk
- E.g.

Risk :

Requirements are inappropriate

Solution / Strategy

A prototype system may be developed

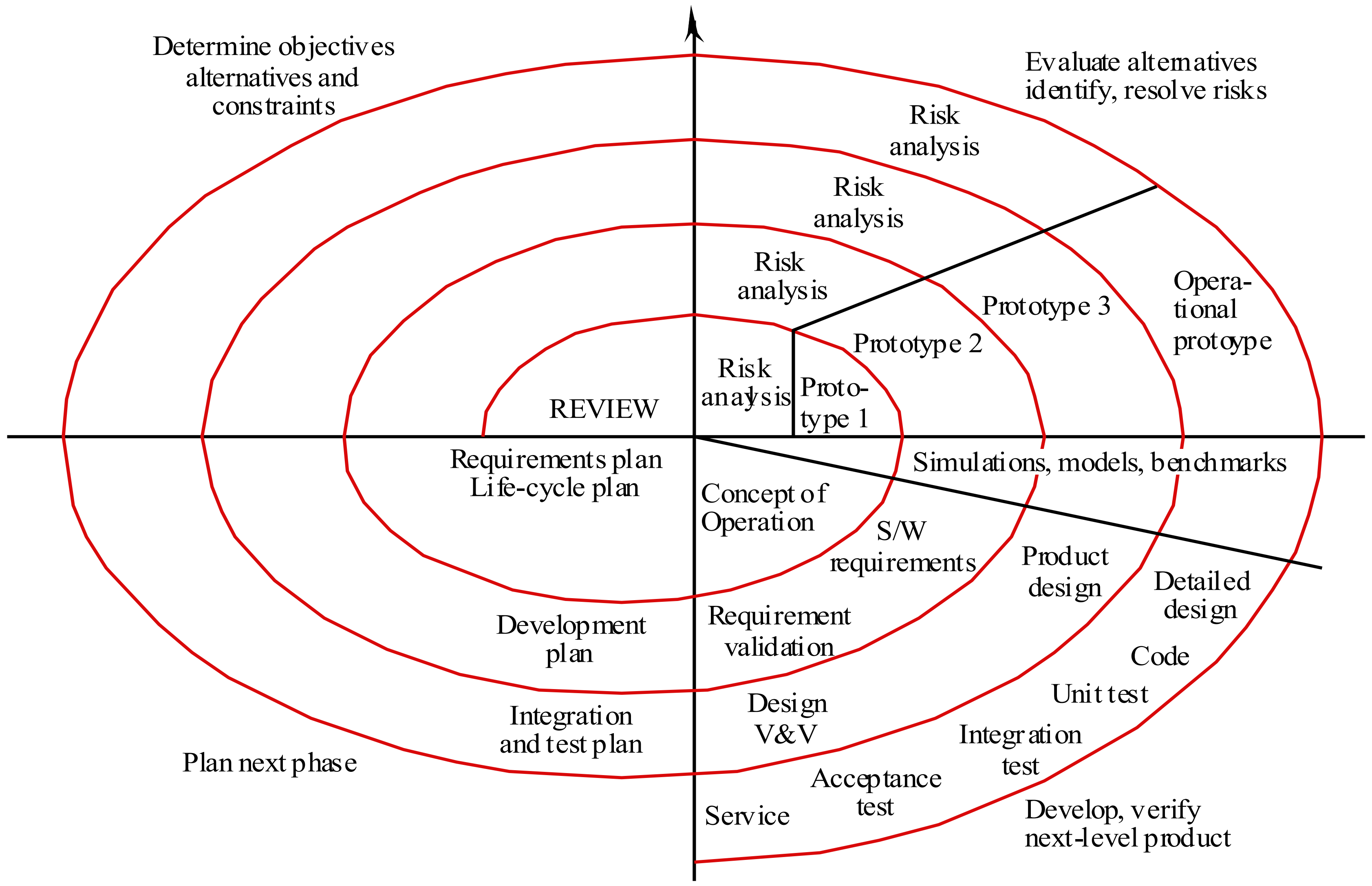
* Risks are

- poorly understood requirements
- poorly understood architecture
- performance problems
- important missing features

Spiral model sectors

- **Development and validation**
  - A development model for the system is chosen which can be any of the generic models
  - For example, if user interface risks are dominant, an appropriate model may be "Evolutionary Prototyping".
  - Selection of model will depend on your risk analysis.
- **Planning**
  - The project is reviewed
  - Decision made whether to continue with a further loop of the spiral
  - If decided to continue , the next phase of the spiral is planned

Spiral model of the software process



Spiral model(Example)

- The evolution of Microsoft Operating System, Compilers and other operating systems.