Software Engineering

Methods and Tools

to

build the right product & build it right

Software Life Cycle

Requirements Phase

What to develop? Involves requirements elicitation activities.

Output of this phase are the Requirements Specs & Test Cases

Design Phase

How to develop? A blueprint to translate specs into code is created.

Output of this phase are the Architecture & Detail Design

Implementation Phase

Product Manifestation takes place.

Output of the phase includes Code, Unit Test Results & Documentation

Testing & QA

Verification (right product) & Validation (build right) takes place

QA & Test Results are captures

Maintenance

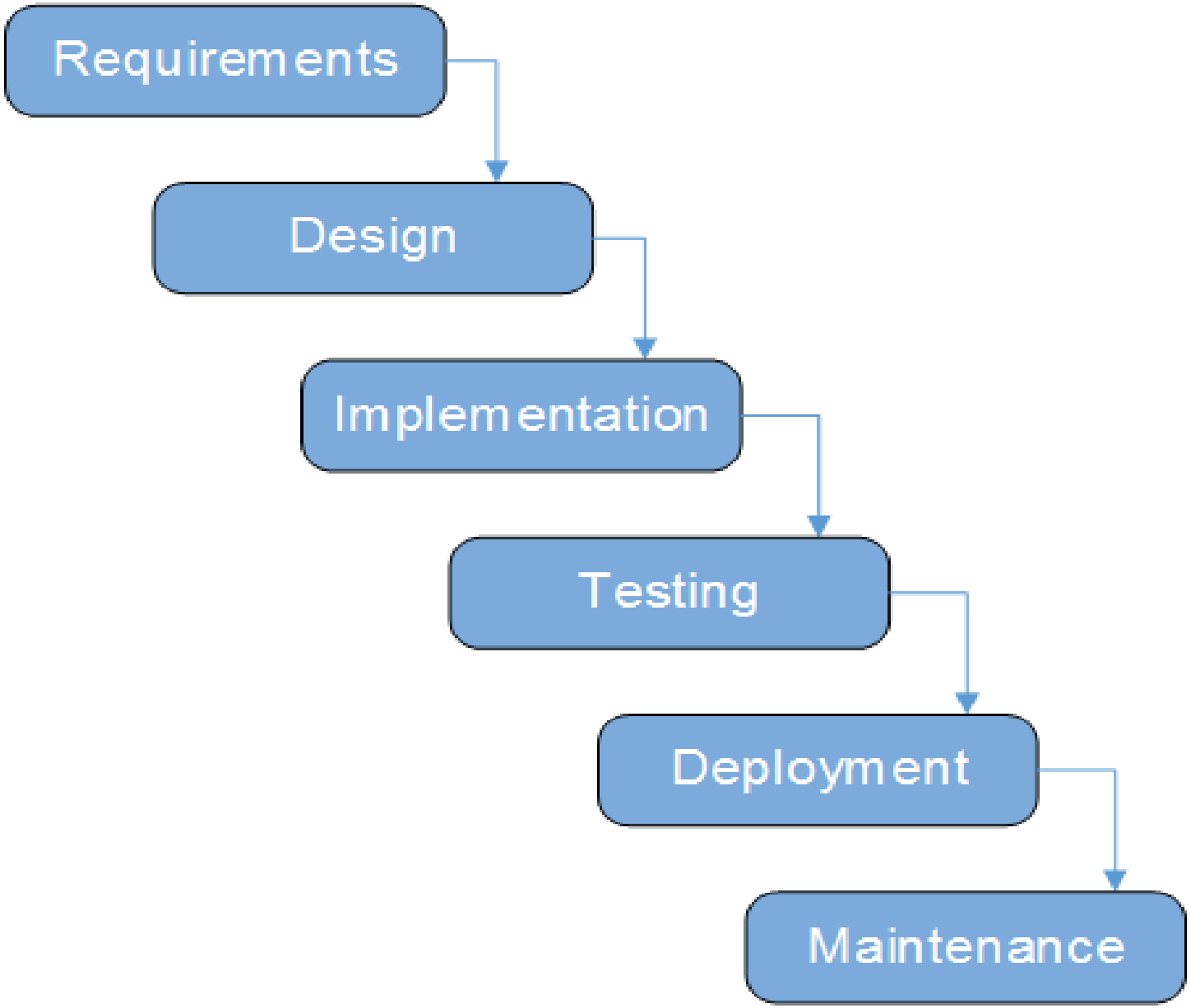
Bug reports are handled and above phases are repeated to add new features

Common patterns of software life cycle are referred to as Software Process Models.

Several such process models have been identified or defined.

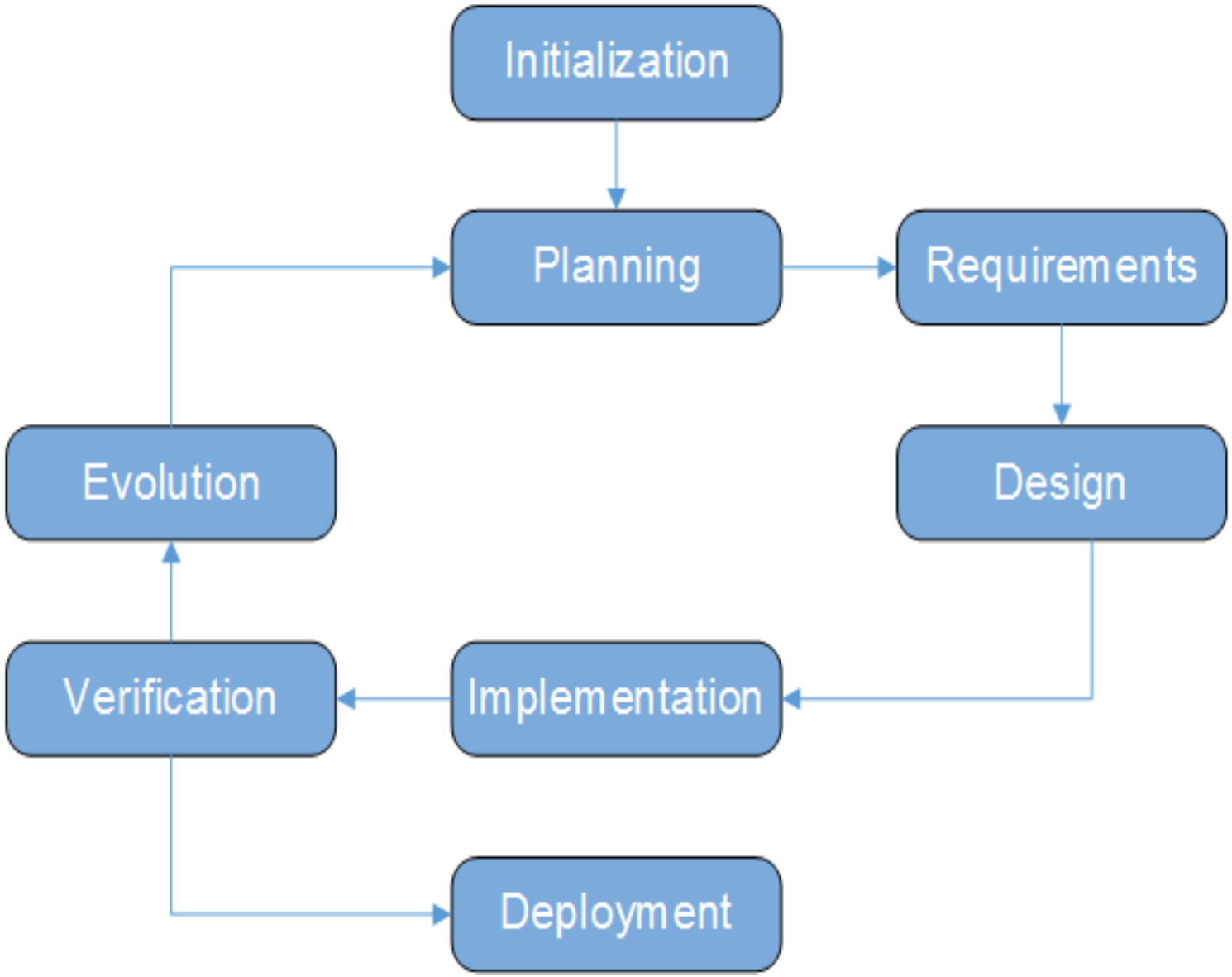
Software Process Models

WaterFall

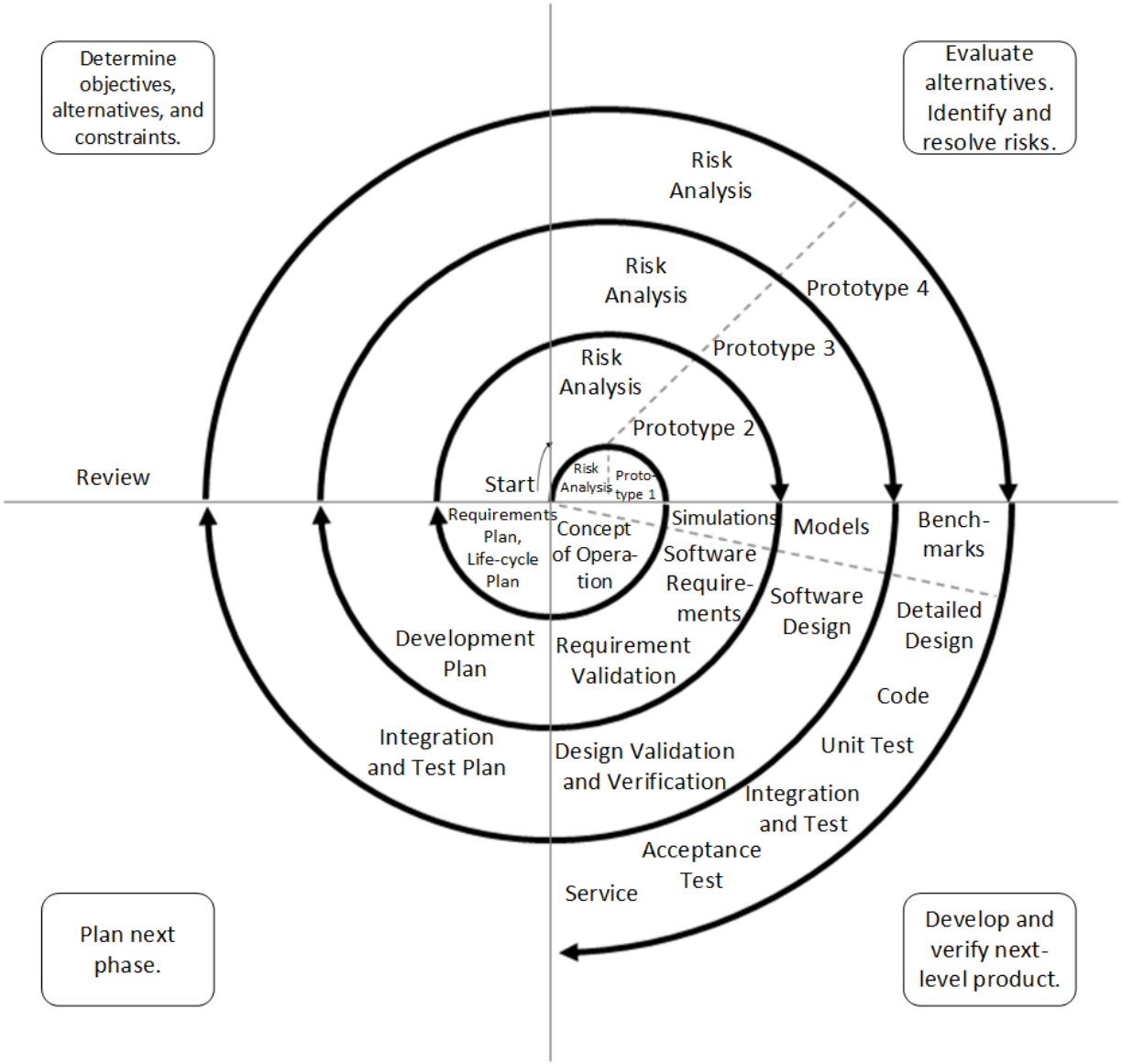


Software Process Models

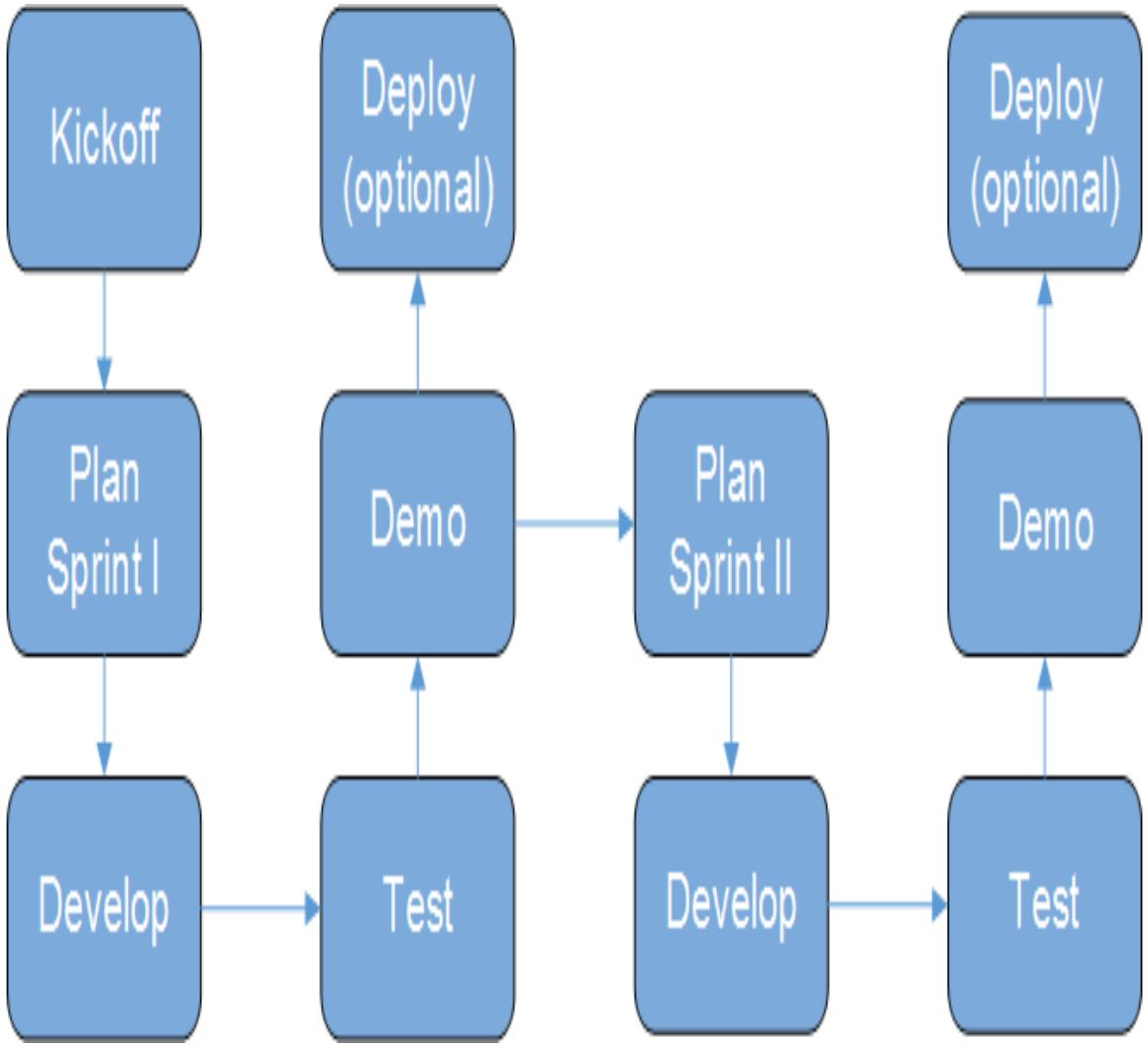
Iterative



Software Process Models



Spiral



Agile

Irrespective of the underlying process model

- a requirements phase to define the product, and
- either a formal design phase upfront or refactoring/redesign after the code is written to address maintainability and other software quality

will take place

Software Definition

- Functional Requirements
  - What functionality the software system provides
    - Expressed as
      - User Stories (Agile Process Model)
      - “shall” statements (traditional IEEE/ISO based process models)
      - UML Use Case Diagram
    - Verified using
      - Acceptance Testing
- Non Functional Requirements
  - How well the functionality is provided
    - Expressed as
      - performance, scalability, reliability, availability, security, maintainability, usability, accessibility measures or metrics
    - Verified using
      - performance, scalability/load, [regression, black box, reliability], stress, security, maintainability, usability, accessibility tests
      - Static code analysis
      - Models

Software Design

- Expressed as
  - Architecture (UML Deployment, Component and/or Package Diagram)
  - Detailed Design (UML Class, and Sequence, State-Machine, Activity and/or Timing Diagrams)

Behaviour (Test) Driven Development
(Android & Java, Espresso, JUnit, Jenkins & Git)

Test Driven development