Software Development Life Cycle (SDLC)

Development Phases

1. Planing
   - Gather requirement
   - Plan project approach
2. Defining Requirements
   - Define and document the requirements
   - Create SRS
3. Design
   - Design the project architecture
   - Create Design Document Specification
5. Build
   - Develop the product as per Design Document Specification
6. Testing
   - Test the product
   - Product defects are
7. Deployment
   - Release formally to the market
   - Do user acceptance testing

SDLC Models

- Waterfall Model
  - The earliest SDLC approach./ A linear and sequential flow/Sequential, no-overlappin phases.
  - Pros: Clearly defined stages./ Well understood milestones./ Process and results are well documented
  - Cons: It is difficult to measure progress within stages./ Cannot accommodate changing requirements.
- V-Model
  - An SDLC model where execution of processes happens in a sequential manner in a V-shape. It is also known as Verification and Validation model.
  - Pros: This is a highly-disciplined model and Phases are completed one at a time./ Works well for smaller projects where requirements are very well understood. / Simple and easy to understand and use.
  - Cons: High risk and uncertainty./ Not a good model for complex and object-oriented projects.
- Big Bang Model
  - The Big Bang model is an SDLC model where we do not follow any specific process.
  - Pros: A very simple model/ Little or no planning required/Easy to manage
  - Cons: Very High risk and uncertainty./ Not a good model for complex and object-oriented projects./ Poor model for long and ongoing projects.
- Agile models

SDLC Models

Traditional

- Models: Waterfall, V-model, Big bang model
- Issues: Cost and schedule overruns, Cancelled projects, Huge effort during the planning phase

Agile

- Principles
  - Our highest priority is to satisfy the customer through early and continuous delivery of valuable software
  - Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
  - The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- Agile Manifesto
  - Individuals and Interaction
  - Working Software
  - Customer Collaboration
  - Responding to change
- Agile is basically a set of values and principles.
- Models: XP, Kanban, Scrum, RAD, Iterative, Incremental, Spiral Model

XP

- XP Team: Common Roles: Product manager / product owner/On-site customersDomain Expert/Interaction desiners/Business analysts/progrmers/Testers/Desingners andTesters/Coces
- XP teams perform nearly every software development activity simultaneously. Analysis, design, coding, testing, and even deployment occur with rapid frequency
- Kanban Practices
  1. Visualize the Workflow (Kanban boards, cards & columns)
  2. Limit Work in Progress
  3. Manage Flow (movement of work items through production process)
  4. Make Process Policies Explicit (processes clearly defined, published & socialized)
  5. Feedback Loops (Daily standups, reviews)
  6. Improve Collaboratively

Kanban

- Kanban is a workflow management method designed to help to visualize work, maximize efficiency and be agile
- Core Principles
  - Start with what you do now.
  - Agree to Pursue Incremental, Evolutionary Change
  - Respect the Current Process, Roles & Responsibilities
  - Encourage Acts of Leadership at All Levels

Scrum

- Scrum Artifacts
  - Product Backlog
  - Sprint Backlog
  - Potentially Releasable Product increment
- FloatingScrum Events
  - Sprint Planning
  - Daily Scrum Meeting
  - Sprint Review
  - Sprint Retrospective
- Process: In scrum, you begin by creating a product backlog—a prioritized list of the features and other capabilities needed to develop a successful product
- Scrum Team: Product owner, ScrumMaster, Development team
- Scrum is an agile approach for developing innovative products and services.

RAD

- RAD Model Pros
  1. Changing requirements can be accommodated.
  2. Progress can be measured.
  3. Productivity with fewer people in a short time.
  4. Reduced development time
- RAD Model Cons
  1. Only for systems that can be modularized can be built using RAD.
  2. Requires highly skilled developers/designers.
  3. High dependency on modeling skills.

Spiral Model

- 4 phases of spiral
  1. Identification
  2. Design
  3. Construct or build
  4. Evaluation & risk analysis
- Spiral Model - Pros
  - Changing requirements can be accommodated.
  - Allows extensive use of prototypes
  - Requirements can be captured more accurately.
  - Users see the system early.
- Spiral Model - Cons
  - Management is more complex
  - End of the project may not be known early: Spiral may go on indefinitely

Prototyping

- Prototyping cons
  - Risk of insufficient requirement analysis owing to too much dependency on the prototype
  - Users may get confused in the prototypes and actual systems.
  - The effort invested in building prototypes may be too much if it is not monitored properly
- Prototyping Pros
  - Increased user involvement in the product even before its implementation
  - Reduces time and cost as the defects can be detected much earlier
  - Quicker user feedback is available leading to better solutions.
- Software Prototyping: A working model of software with some limited functionality
- Models
  - A Horizontal prototype
  - Developing the Initial Prototype
  - A Vertical prototype
  - /Rapid prototyping
  - Evolutionary Prototyping
  - Incremental prototyping
  - Extreme Prototyping