NSS

Software Development Life Cycle (SDLC)

SDLC

Is a structured approach to development of software

Also called a **Software Development Process**

Software Development Activities

Planning

Implementing, Testing & Documentation

Deployment & Maintenance

SDLC Models

Waterfall

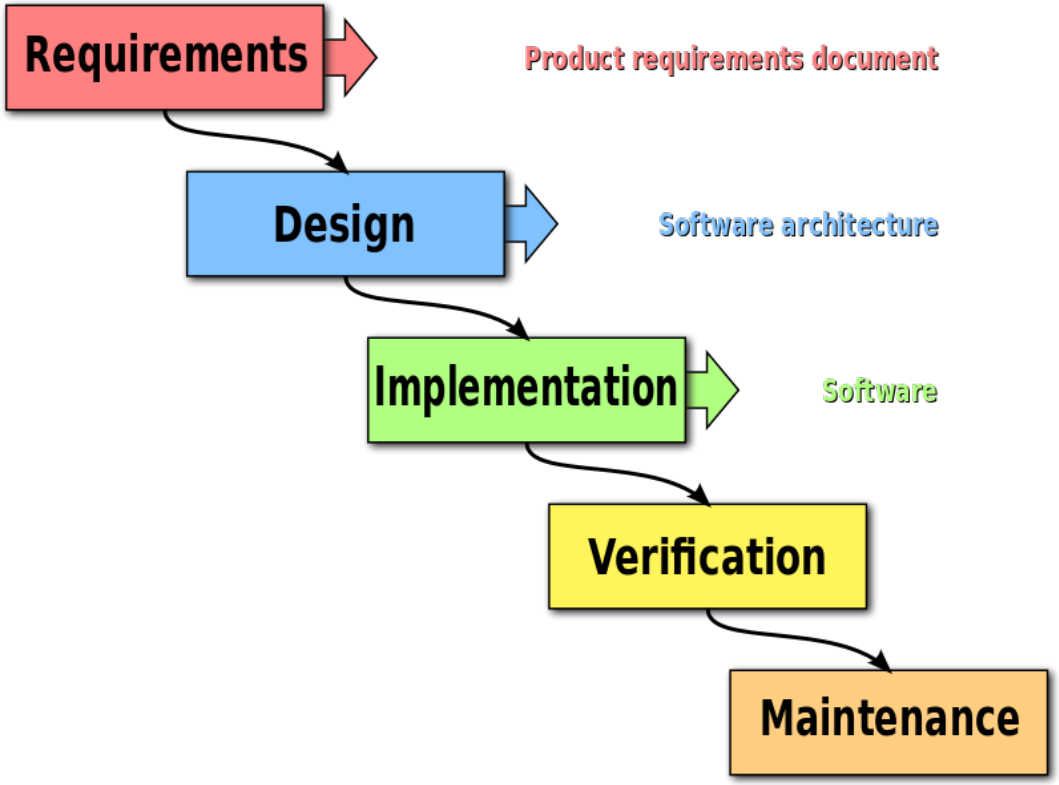
Spiral

Iterative

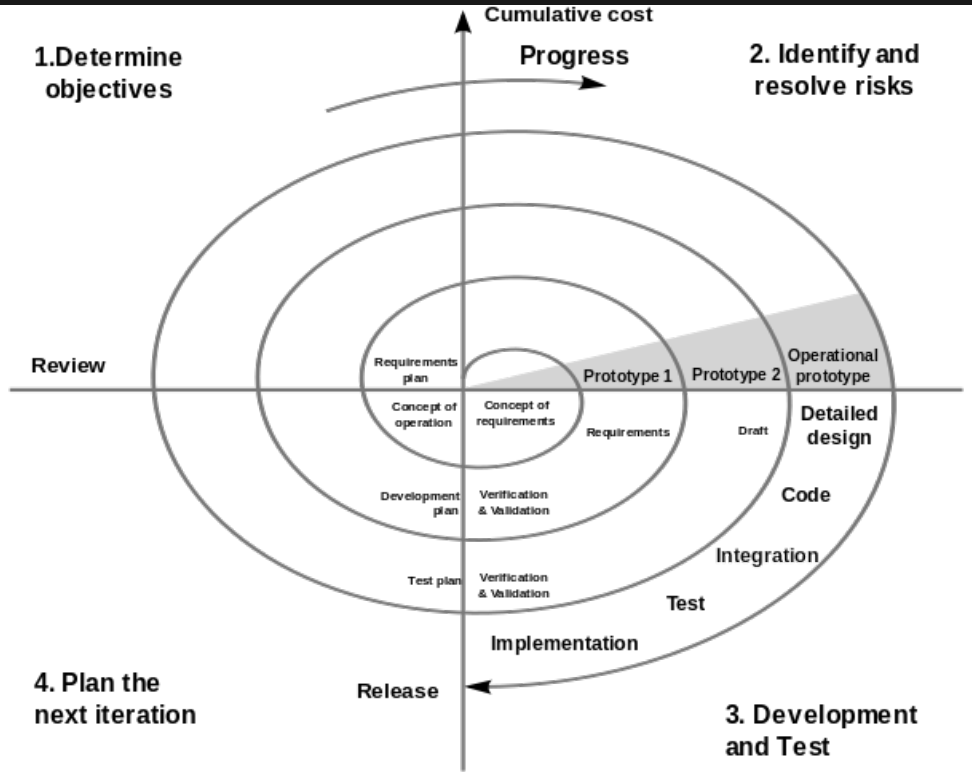
Agile

Rapid Application Development

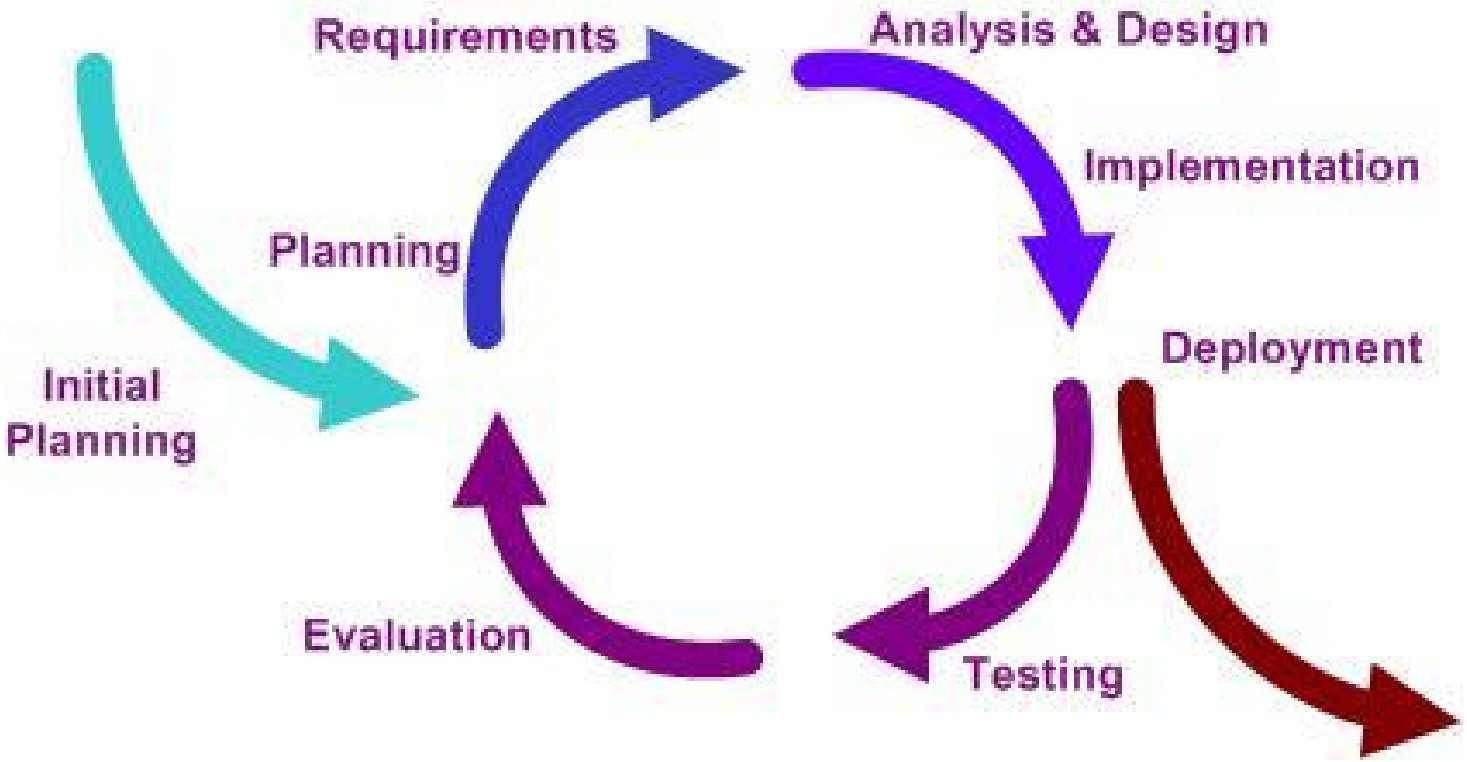
Waterfall


Requirements
Product requirements document
Design
Software architecture
Implementation
Software
Verification
Maintenance

Spiral


Cumulative cost
Progress
1.Determine objectives
2. Identify and resolve risks
Review
Requirements plan
Concept of operation
Concept of requirements
Prototype 1
Prototype 2
Operational prototype
Requirements
Draft
Detailed design
Development plan
Verification & Validation
Code
Integration
Test plan
Verification & Validation
Test
Implementation
Release
4. Plan the next iteration
3. Development and Test

Iterative & Incremental



Agile

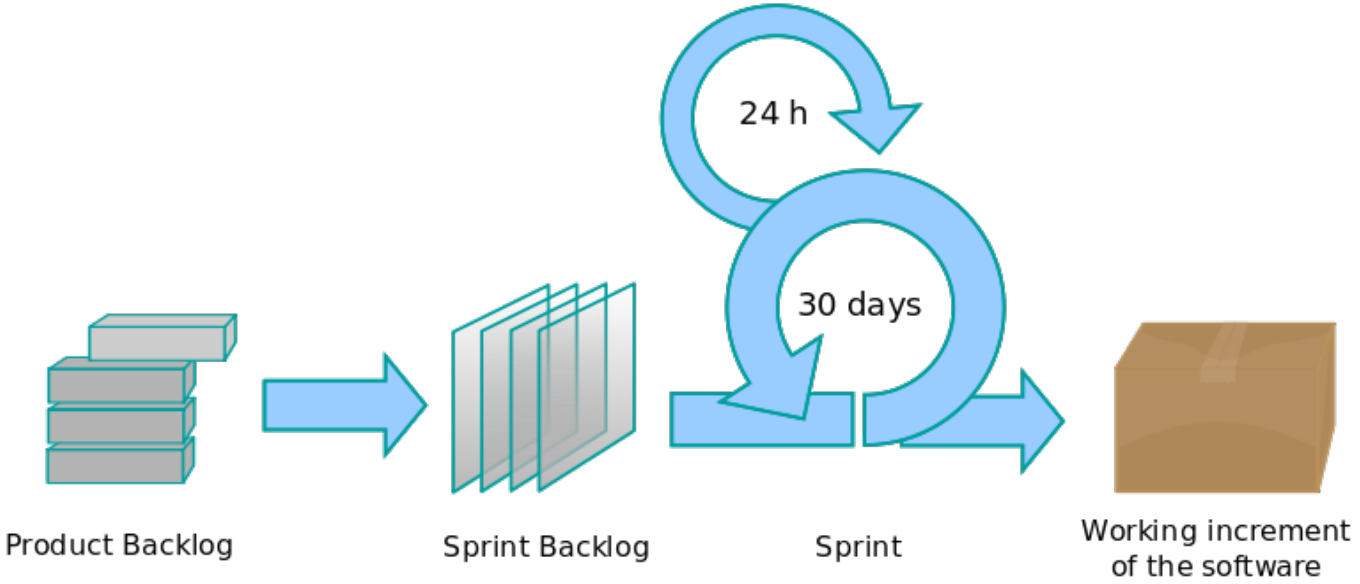
Is iterative process with continuous feedback

Filled with jargon

Has a LOT of subversions

- Scrum
- Kanban
- Lean Software development
- Many more..

Agile (Scrum)



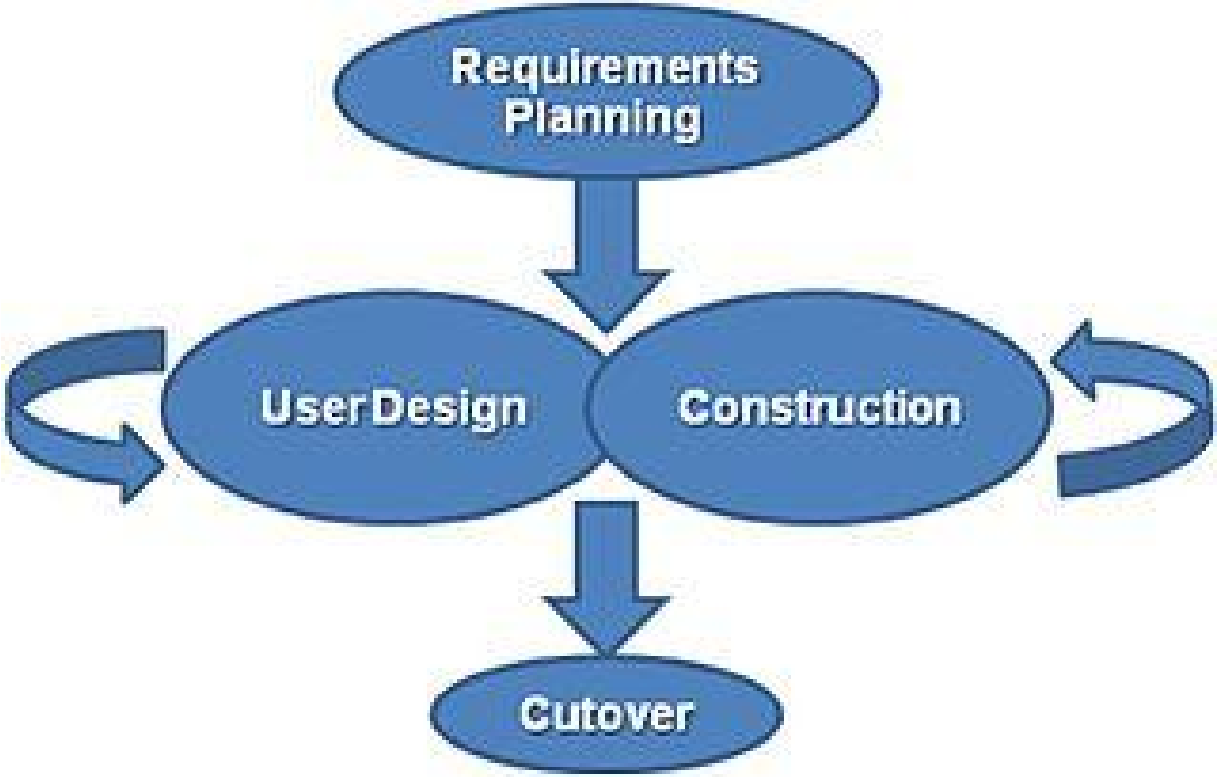
Project Todo list

Items due soon

Time frame

Finished work

Rapid Application Development (RAD)



Name	Pros	Cons
Agile	Minimizes feature creep by developing in short intervals resulting in miniature software projects and releasing the product in mini-increments.	Short iteration may add too little functionality, leading to significant delays in final iterations. Since Agile emphasizes real-time communication (preferably face-to-face), using it is problematic for large multi-team distributed system development. Agile methods produce very little written documentation and require a significant amount of post-project documentation.
Extreme	Lowers the cost of changes through quick spirals of new requirements. Most design activity occurs incrementally and on the fly.	Programmers must work in pairs, which is difficult for some people. No up-front "detailed design" occurs, which can result in more redesign effort in the long term. Thebusiness champion[*clarification needed*] attached to the project full-time can potentially become a single point of failure for the project and a major source of stress for a team.
Joint application	Captures the voice of the customer by involving them in the design and development of the application through a series of collaborative workshops called JAD sessions.	The client may create an unrealistic product vision and request extensive gold-plating, leading a team to over- or underdevelop functionality.

Name	Pros	Cons
Lean	Creates minimalist solutions (i.e., needs determine technology) and delivers less functionality earlier; per the policy that 80% today is better than 100% tomorrow.	Product may lose its competitive edge because of insufficient core functionality and may exhibit poor overall quality.
RAD	Promotes strong collaborative atmosphere and dynamic gathering of requirements. Business owner actively participates in prototyping, writing test cases and performing unit testing.	Dependence on strong cohesive teams and individual commitment to the project. Decision-making relies on thefeature functionality team and a communal decision-making process with lesser degree of centralized project management and engineering authority.
Scrum	Agile framework. Improved productivity in teams previously paralyzed by heavy "process", ability to prioritize work, use of backlog for completing items in a series of short iterations or sprints, daily measured progress and communications.	Reliance on facilitation by a scrum-master who may lack the political skills to remove impediments and deliver thesprint goal. Due to reliance on self-organizing teams and rejection of traditional centralized "process control", internal power struggles can paralyze a team.

Security in the Software Development Life Cycle

- Security must be included in each step of the SDLC
  - Conceptual
  - Requirements and specifications development
  - Application design
  - Threat risk modeling
  - Coding
  - Testing

Security in the conceptual stage

- Presence of sensitive information must be identified
- Information flows
- Access controls (users, administrators, third parties)
- Regulatory requirements
- Application dependencies

Security application requirements and specifications

- Every detail of the software should be specified, down to individual input forms and fields
- Security requirements
  - Roles, access controls, audit logging, configuration management

Security in application design

- Adhere to all requirements and specifications
- Published design documents
- Design reviews
  - Reviewed by all stakeholders including security

Threat risk modeling

- Identify threats and risks prior to development
- Possible changes to specs, req’s, or design

Security in application coding

- Develop safe code
  - Free of common vulnerabilities
- Use safe libraries that include safe functions for input validation
- 1-10-100 rule
  - It costs 10 times as much to secure an application after it has been developed
  - It costs 100 times as much to secure an application after it has been implemented

Security in testing

- Testing should verify correct coding of every requirement and specification
- Use vulnerability scanners

Protect the SDLC itself

- Source code access control
  - Protect source code
- Don't trust it to remain secret, though
  - Record version changes
- Protection of software development and testing tools
  - Protect from unauthorized modifications
- Protection of software development systems
  - Prevent introduction of malware, backdoors, logic bombs

Controls that must be present in a developed application

- Authentication
  - Limiting access to only legitimate, approved users
- Authorization
  - Limiting access only to approved functions and data
- Audit logging
  - Logging of all actions in the application

A Bit on Database Security

Databases and Data Warehouses

- Database
  - Ordered collection of data, such as employee records
- Data Warehouse
  - A database used for decision support and research
  - May contain all customer transactions
  - Business intelligence tools analyze the data to find trends
  - Example: Google's ad-targeting data

Database Architectures

- Hierarchical databases: tree structure like DNS (no longer produced)
- Network databases: complex tree structure (no longer produced)
- Object-oriented databases: OO, methods stored with data

Database Architectures

- Distributed databases: physically distributed, any type
- Relational databases (RDBMS): in widest use today
  - Data is stored in tables, records and fields
  - Tables have relationships
  - Oracle, SQL Server, DB2, MySQL, etc.

Database Transactions

- Records retrieval
- Records update
- Records creation
- Transactional integrity
  - Nested or complex transactions executed as a unit
  - Begin work… <transactions> …end work

Database Security Controls

- Access controls
  - Userids, passwords
  - Table / row / field level access control
  - Read-only or read/write
- Views
  - Virtual tables that are a subset of individual tables, or a “join” between tables
  - Permission given to views just like “real” tables