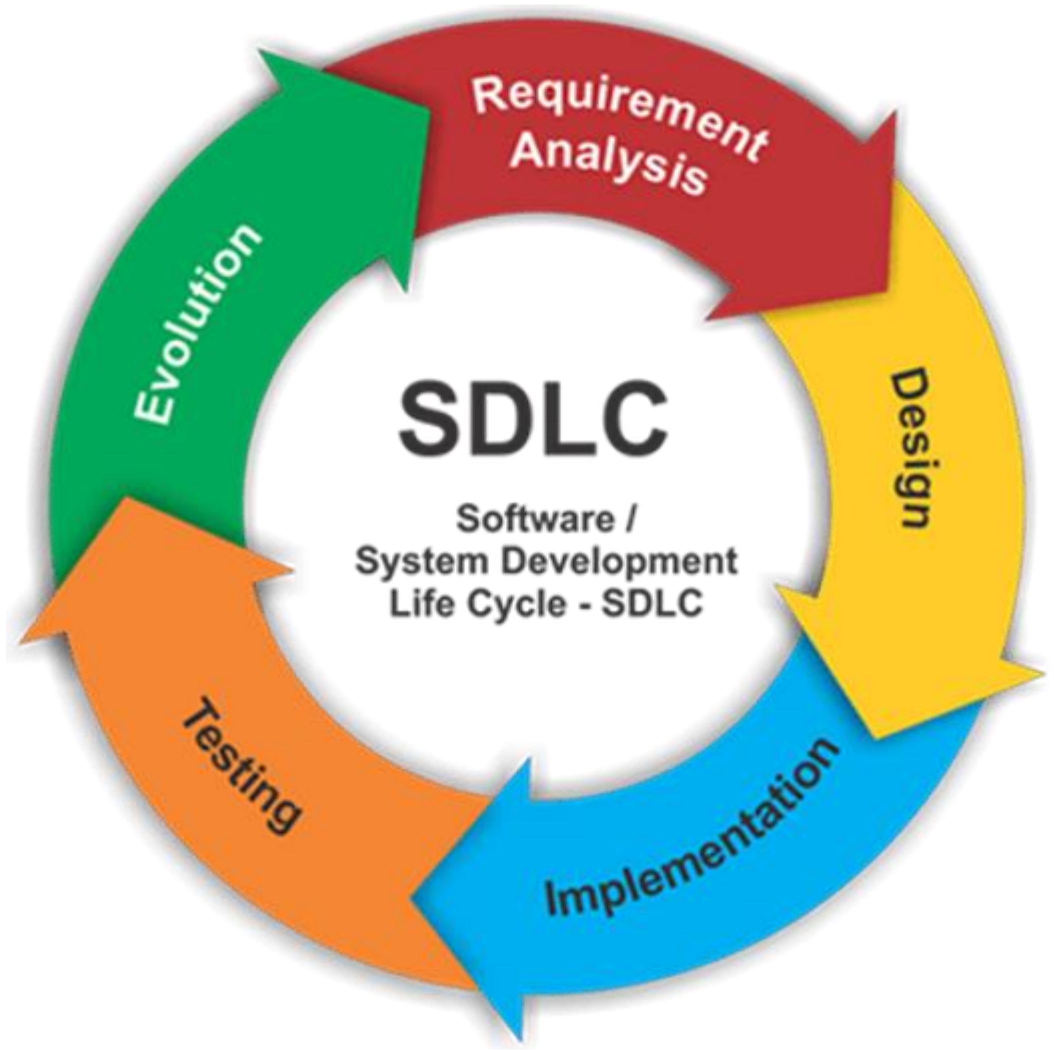
SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC)

Assoc. Prof. Dr. Mehmet Akif Çifçi


Requirement Analysis
Design
Implementation
Testing
Evolution
SDLC
Software / System Development Life Cycle - SDLC

What is SDLC?

- The software development life cycle (SDLC) is a cost-effective and time-efficient process that development teams use to design and build high-quality software. The goal of SDLC is to minimize project risks through forward planning to meet customer expectations during and after production. This methodology outlines a series of steps that break down the software development process into tasks that you can assign, complete and measure.

What are the Advantages of SDLC?

- Increased visibility of the development process for all stakeholders
- Efficient forecasting, planning and programming
- Improved risk management and cost estimation
- Systematic software delivery and better customer satisfaction

The SDLC consists of six phases:

- Planning
- Design
- Application
- Test
- Distribution
- Care

Planning

The planning phase usually includes tasks such as cost-benefit analysis, scheduling, resource estimation and allocation. The development team collects requirements from various stakeholders such as customers, internal and external experts and managers to create a software requirements specification document.

This document sets expectations and defines common goals that help with project planning. The team estimates costs, creates a schedule and has a detailed plan to achieve its goals.

Design

In the design phase, software engineers analyze requirements and determine the best solutions for building the software. For example, they may consider integrating pre-existing modules, make technology choices and identify development tools. They examine how best to integrate the new software into any existing IT infrastructure the organization may have.

Application

In the implementation phase, the development team codes the product. They analyze the requirements to identify smaller coding tasks they can do on a daily basis to achieve the final result.

Test

The development team combines automation with manual testing to check the software for bugs. Quality analysis involves testing the software for bugs and checking whether the software meets customer requirements. Because many teams test the code they write immediately, the testing phase often runs parallel to the development phase.

Distribution

When teams develop software, they code and test it on a separate copy from the one that users access. The software that customers use is called production, while the other copies are called the build environment or test environment.

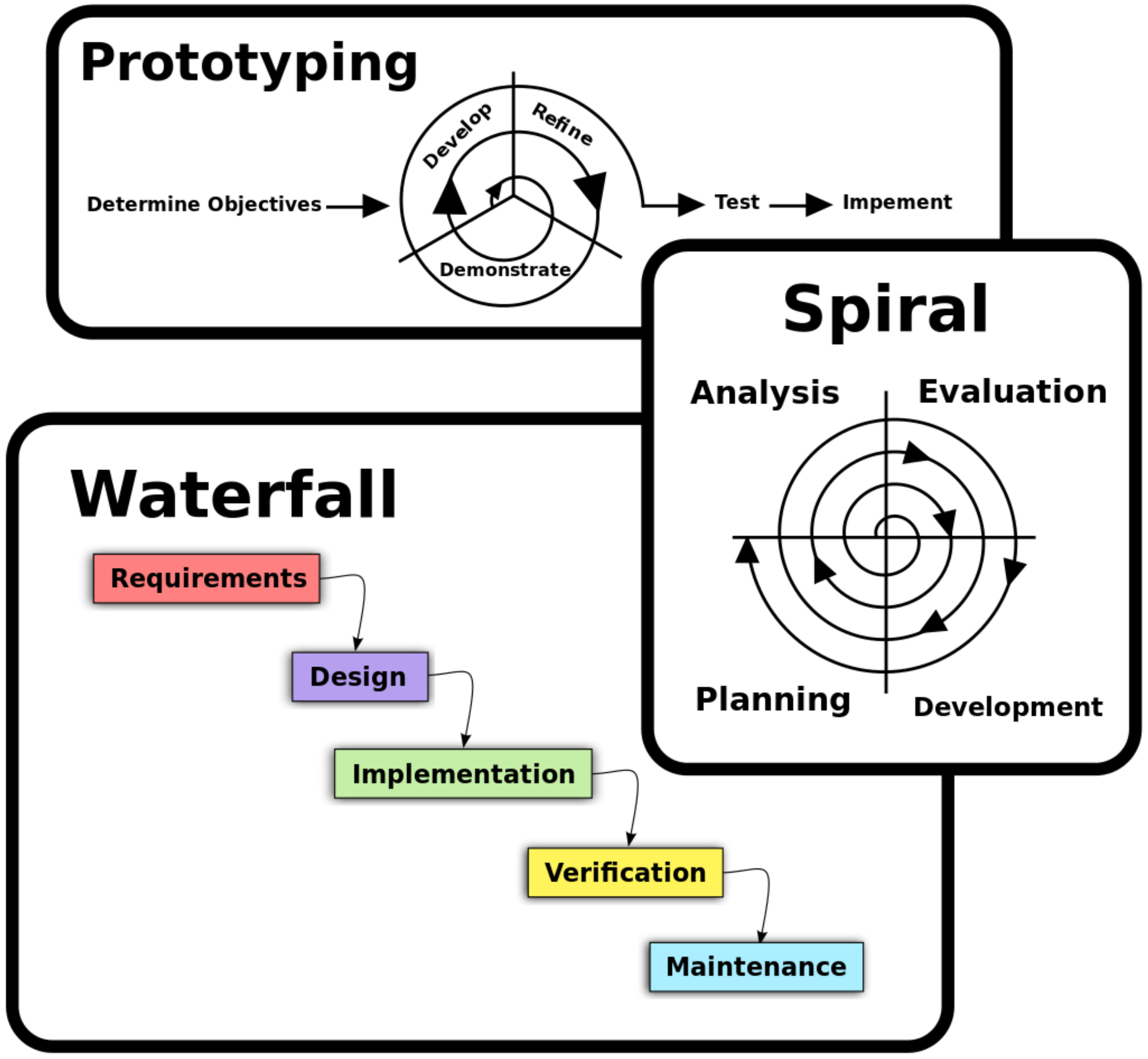
Having separate build and production environments ensures that customers can continue to use the software even when it is being replaced or upgraded. The deployment phase includes various tasks related to moving the latest build copy to the production environment, such as packaging, environment configuration and installation.

Care

During the maintenance phase, the team fixes bugs, resolves customer issues and manages software changes, among other tasks. In addition, the team monitors overall system performance, security and user experience to identify new ways to improve existing software.

What are SDLC Methodologies?

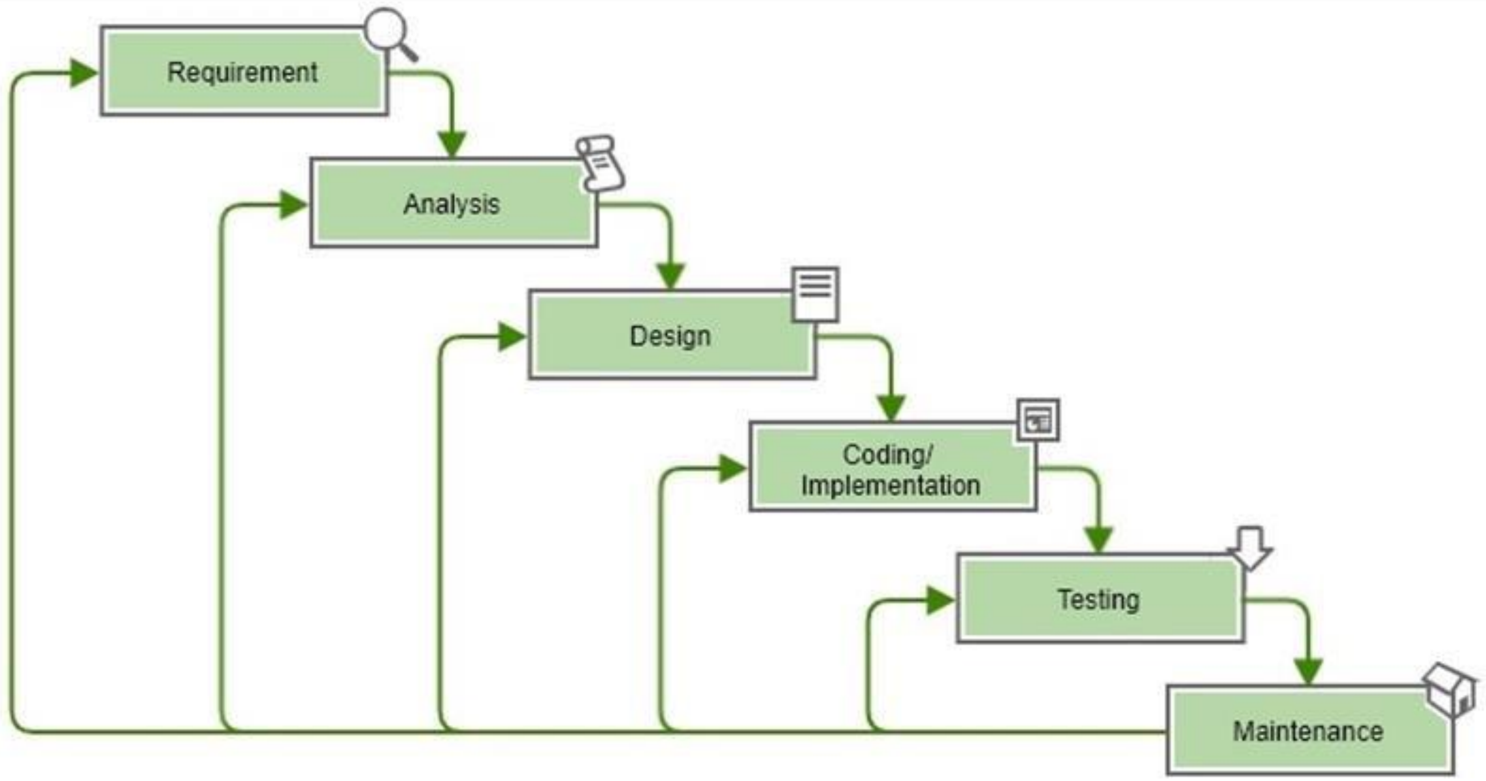
- Waterfall Methodology
- Iterative Methodology
- Spiral Methodology (Spiral Model)
- Agile Methodology
- V Methodology


Prototyping
Determine Objectives
Develop
Refine
Demonstrate
Test
Impement
Spiral
Analysis
Evaluation
Planning
Development
Waterfall
Requirements
Design
Implementation
Verification
Maintenance

Waterfall Methodology

- The waterfall model organizes all stages sequentially in such a way that each new stage depends on the outcome of the previous stage. Conceptually, the design moves from one stage to the next, just like a waterfall.
- The waterfall model provides discipline in project management and gives a tangible deliverable at the end of each phase. However, once a phase is considered complete, too many changes cannot be made, as changes can affect the delivery time, cost and quality of the software. This model is therefore more suitable for small software development projects where tasks are easy to organize and manage and requirements can be accurately specified in advance.

Waterfall Methodology

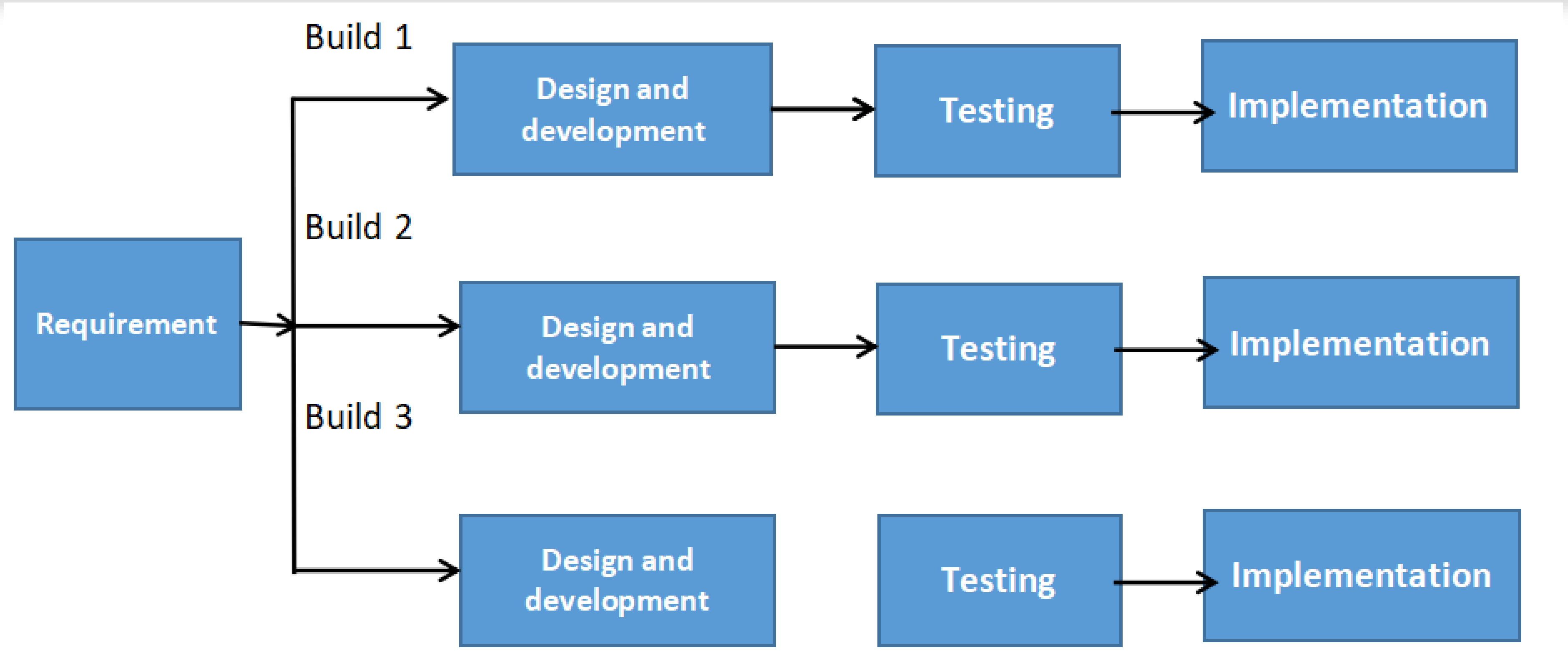


Iterative Methodology

- The iterative process suggests that teams start software development with a small subset of requirements. They then develop versions iteratively over time, until all the software is ready for production. The team creates a new software version at the end of each iteration.

- Risks are easy to identify and manage as requirements may change between iterations. However, repeated cycles can lead to scope creep and underestimation of resources.

Iterative Methodology

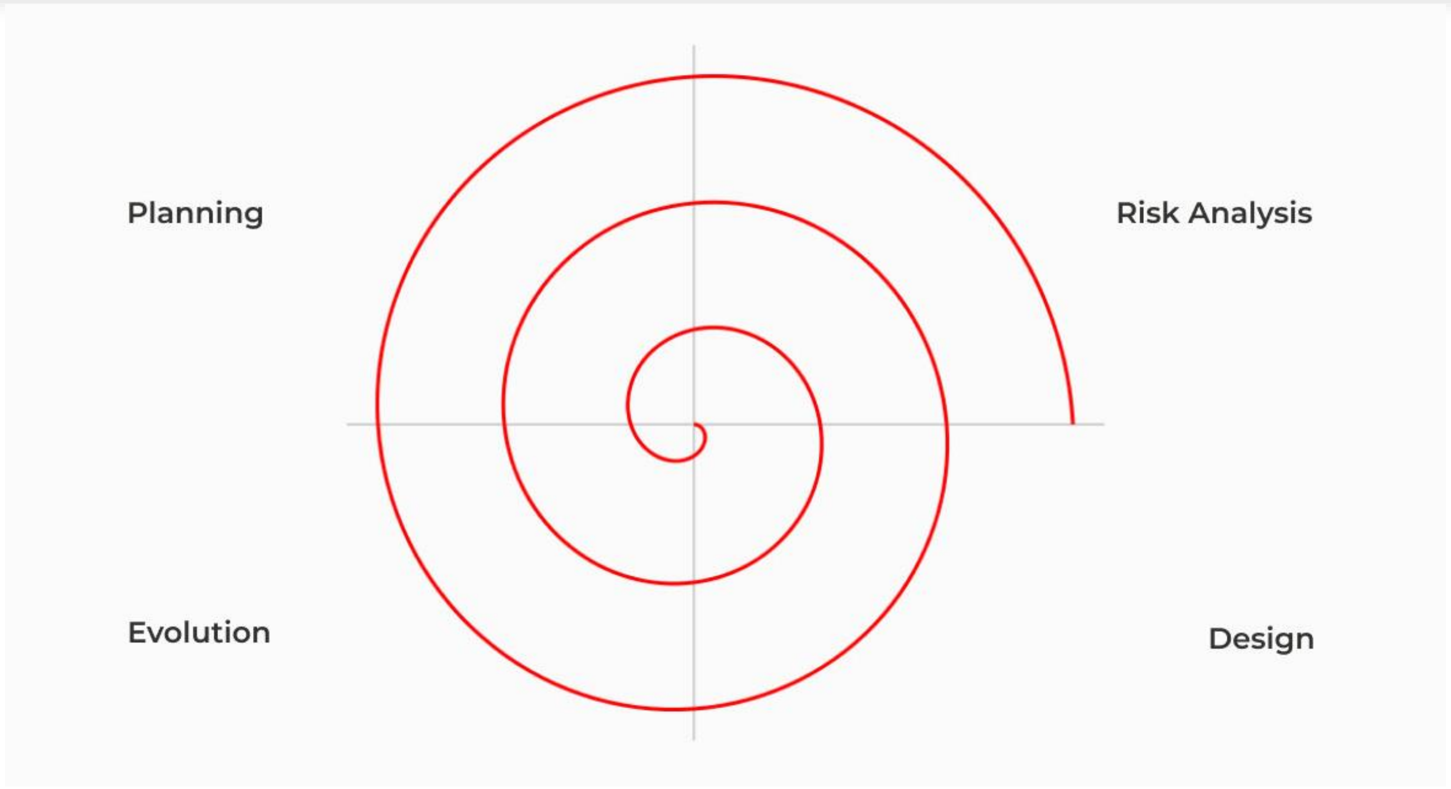


Spiral Methodology (Spirl Model)

- The spiral model prioritizes risk analysis by combining the small repetitive loops of the iterative model with the linear sequential flow of the waterfall model. You can use the spiral model to ensure that software is rolled out in phases and improved by prototyping at each stage.

- The spiral model is suitable for large and complex projects that require frequent changes. But it can be costly for smaller projects with limited scope.

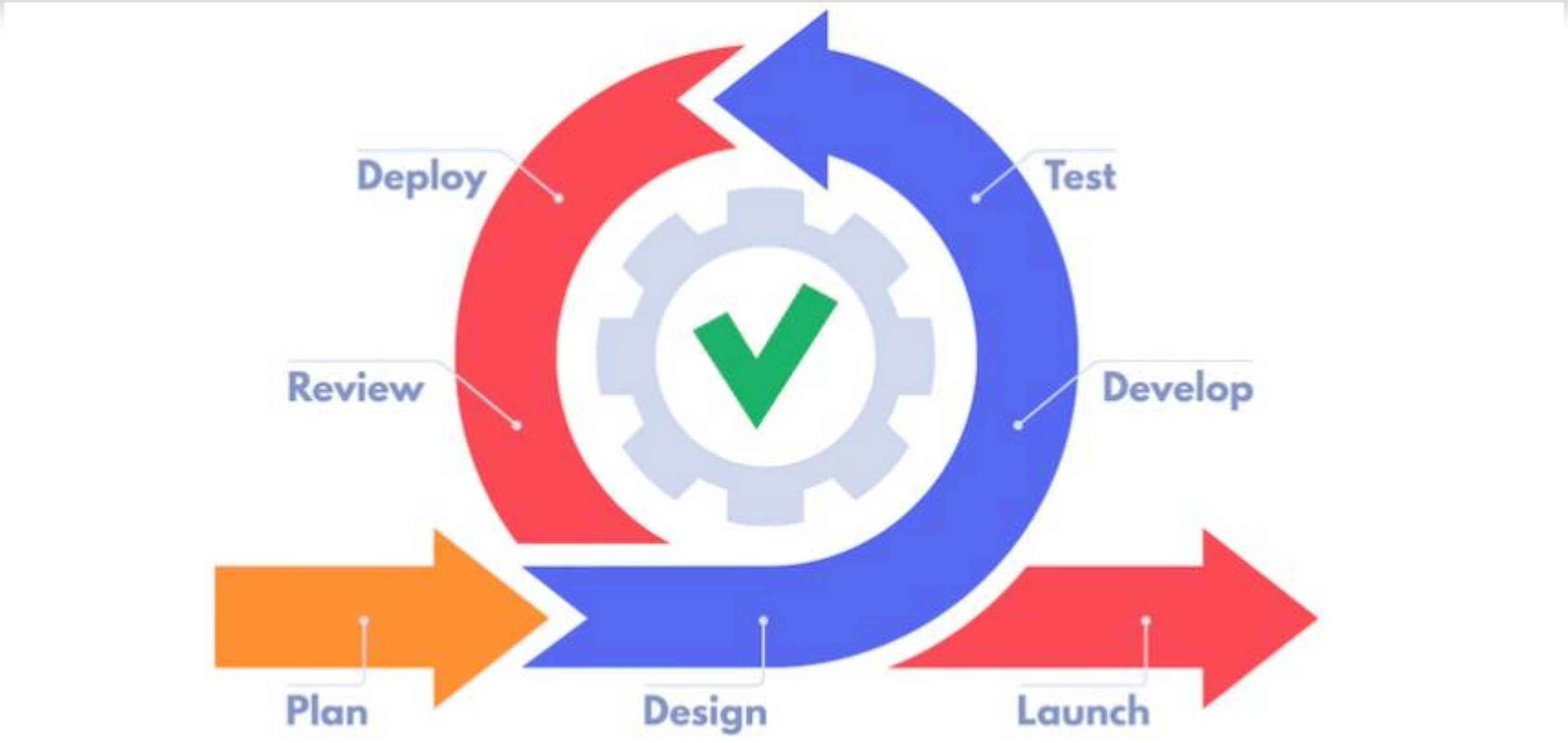
Spiral Methodology (Spiral Model)



Agile Methodology

- The Agile model organizes the SDLC phases into several development cycles. The team iterates quickly through the phases, delivering only small, incremental software changes in each cycle. They continuously evaluate requirements, plans and results to respond quickly to change. The Agile model is both iterative and incremental, making it more efficient than other process models.
- Fast development cycles help teams identify and fix problems in complex projects early and before they become serious issues. They can also engage with customers and stakeholders to get feedback throughout the project lifecycle. However, over-reliance on customer feedback can lead to too many scope changes or a halfway through a project.

Agile Methodology

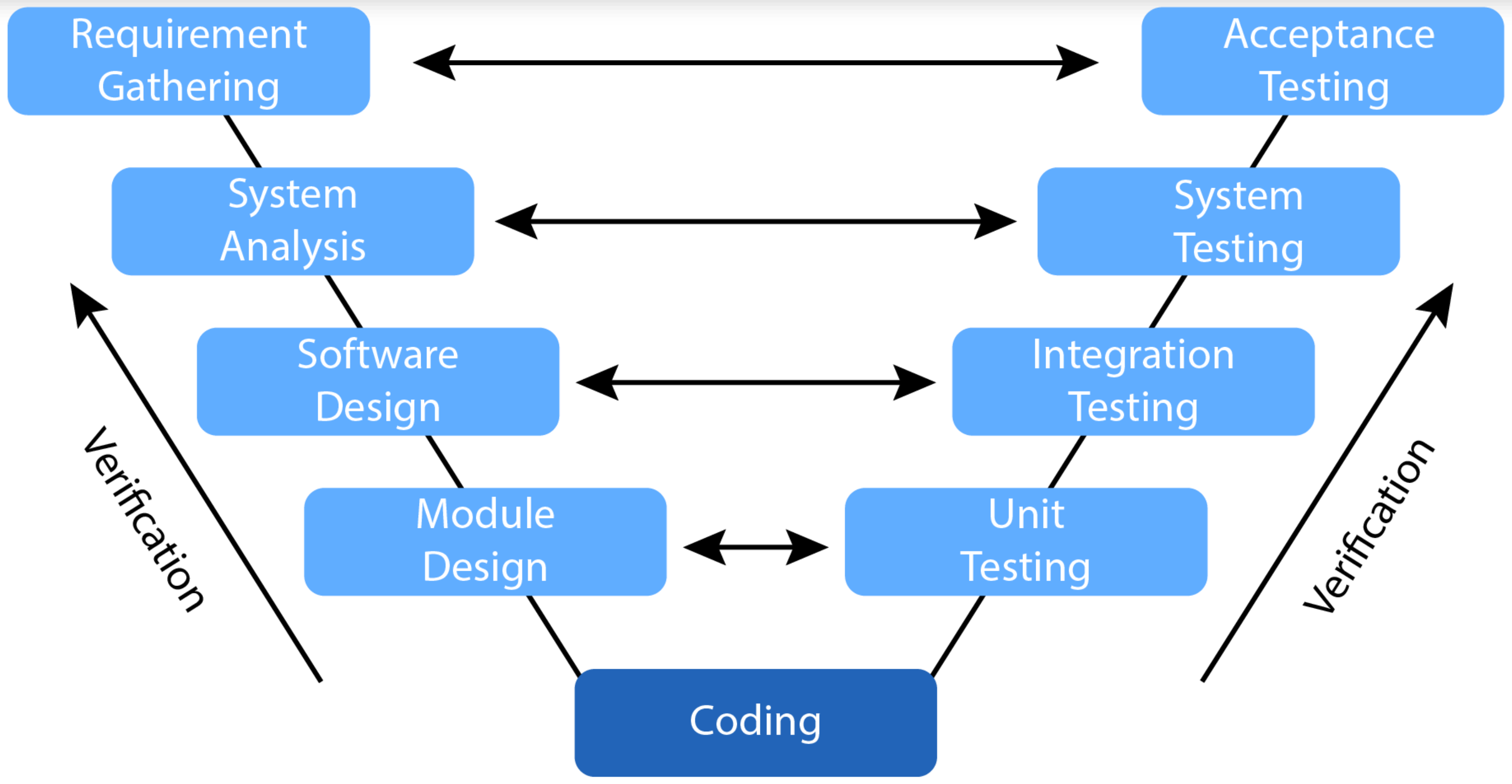


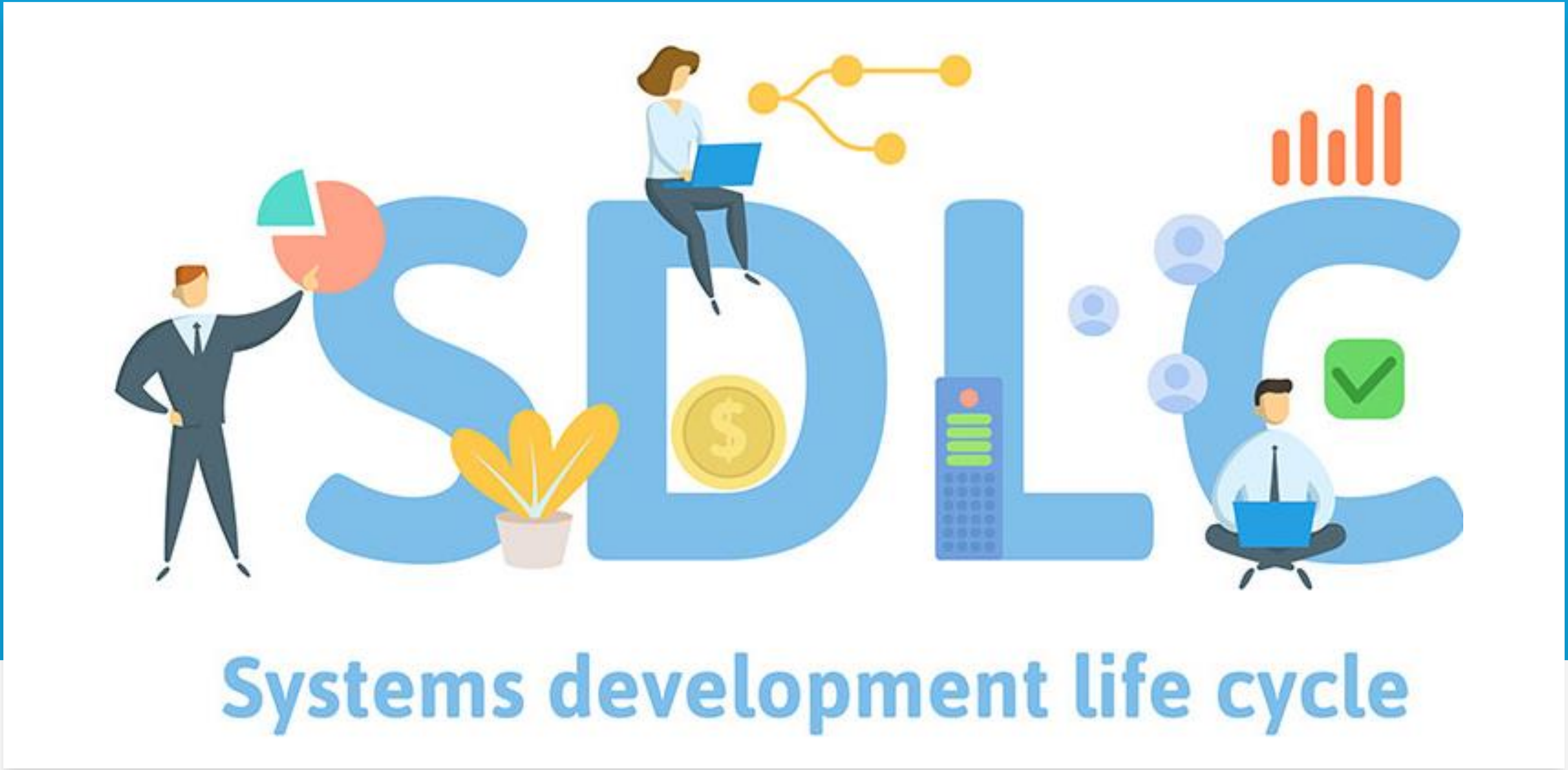
V Methodology

- The V Model is a software development lifecycle model with a test phase for each development phase. This approach, which has many similarities with the waterfall model, takes its name from the "V" shape of the phases.

- This model consists of two sequential processes. The left side of the figure represents the production phases and the right side the testing phases. The coding phase is the step that connects these two sides.

V Methodology




SDLC
Systems development life cycle