**SDLC (SOFTWARE DEVELOPMENT LIFE CYCLE)**

**SDLC:**

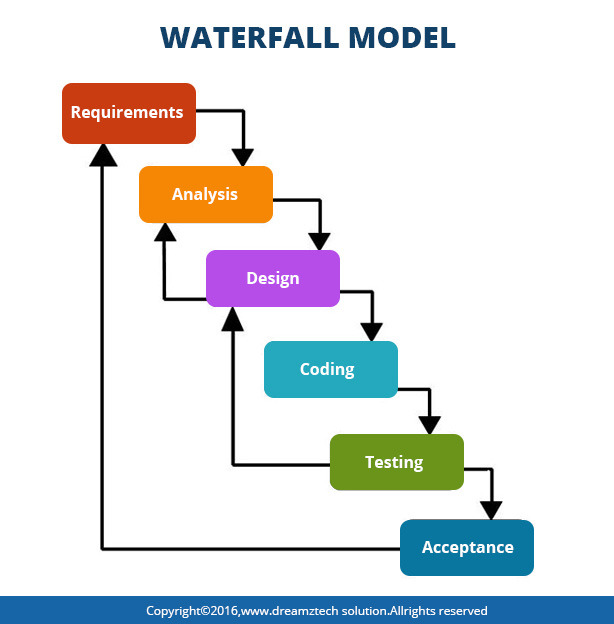
The software development lifecycle (SDLC) is the cost-effective and time-efficient process that development teams use to design and build high-quality software. The goal of SDLC is to minimize project risks through forward planning so that software meets customer expectations during production and beyond.



**Waterfall Model - Design**

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially.

The following illustration is a representation of the different phases of the Waterfall Model.



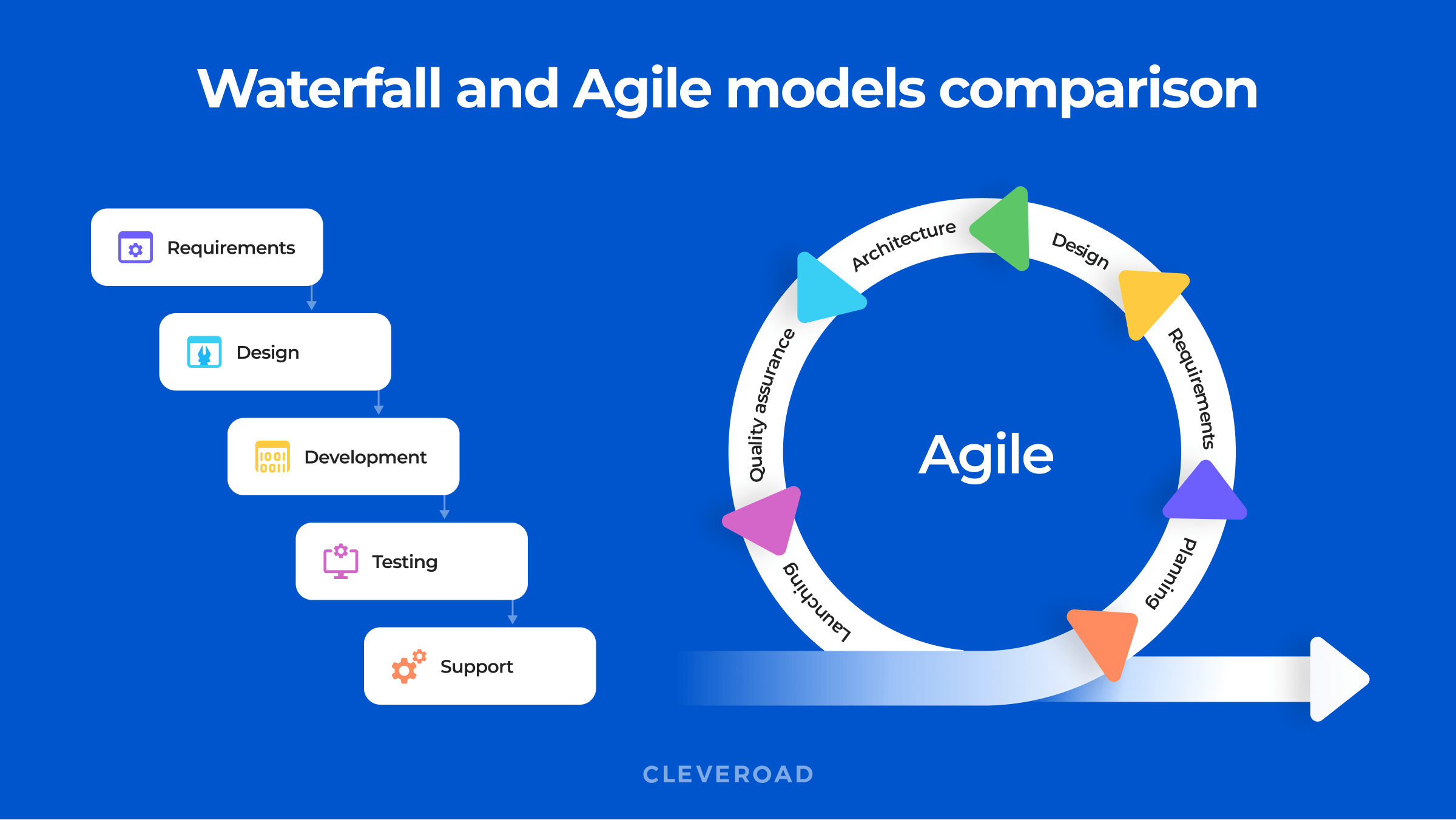
|  |  |
| --- | --- |
| Advantages | Dis-Advantages |
| 1. Before the next phase of development, each phase must be completed | 1. Error can be fixed only during the phase |
| 1. Suited for smaller projects where requirements are well defined | 1. It is not desirable for complex project where requirement changes frequently |
| 1. They should perform quality assurance test (Verification and Validation) before completing each stage | 1. Testing period comes quite late in the developmental process |
| 1. Elaborate documentation is done at every phase of the software’s development cycle | 1. Documentation occupies a lot of time of developers and testers |
| 1. Project is completely dependent on project team with minimum client intervention | 1. Clients valuable feedback cannot be included with ongoing development phase |
| 1. Any changes in software is made during the process of the development | 1. Small changes or errors that arise in the completed software may cause a lot of problems |

## Agile Model:

The Agile Model is an **incremental and iterative process** of software development. It defines each iteration’s number, duration, and scope in advance. Every iteration is considered a short “frame” in the agile process model, which mostly lasts from two to four weeks.



**Waterfall V/S AGILE:**



|  |  |  |  |
| --- | --- | --- | --- |
| # | Case | Agile model | Waterfall model |
| 1. | **Definition** | Agile model follows the incremental approach. | Waterfall model follows a sequential design process. |
| 2. | **Progress** | The measurement of progress is in terms of developed and delivered functionalities. | The measurement of success is in terms of completed and reviewed Artifacts. |
| 3. | **Nature** | Flexible as there is a possibility of changing the requirements even after starting the development process. | This model is rigid as it does not allow to modify the requirements once the development process starts. |
| 4. | **Customer interaction** | There is a high customer interaction. It is because, after every iteration, an incremental version is deployed to the customer. | Customer interaction in waterfall model is very less. It is because, in a waterfall model, the product is delivered to the customer after overall development. |
| 5. | **Team size** | It has a small team size. As smaller is the team, the fewer people work on it so that they can move faster. | The team may consist more members. |
| 6. | **Suitability** | Not a suitable model for small projects. The expenses of developing the small projects using agile is more than compared to other models. | Works well in smaller size projects where requirements are easily understandable. |

**EPIC**: It is a project documentation contains all requirements.

**SPRINT:** It is a mile stone or a plan made for a period of time to implement some work.

**SCRUM:** Group of people involved to develop application

**STORY:** A work that indicates functional requirement of a project.

**TASK:** A developer work indicates part of story.

**BUG:** An issue in functionality/story implemented by developer.

**STORY LIFECYCLE:** End-to-End Process of Story with possible states

**BUG LIFE CYCLE:** End-to-End Process of Bug with possible states

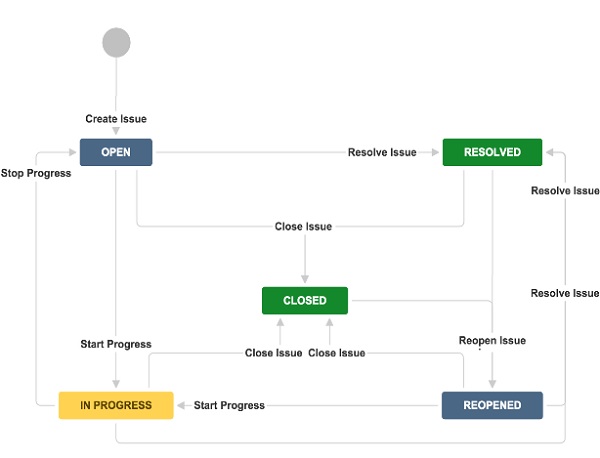
**SPRINT RETERO SPRECTIVE:** A Review Meeting of a Sprint that says what went wrong, good and need to be improved.

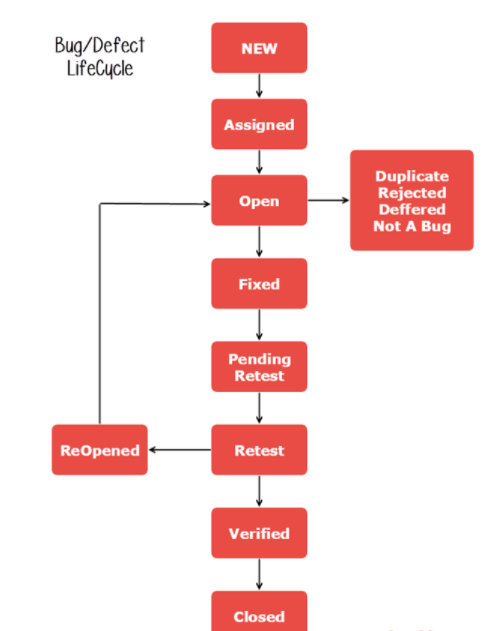
**BURN DOWN CHART:**

Burn down charts and burn-up charts track the amount of output (in terms of hours, story points, or backlog items) a team has completed across an iteration or a project.



**Story Lifecycle:**





**FAQs:**

* Explain SDLC Concept?
* What is Waterfall Model?
* Explain some dis-advantages of Waterfall Model?
* Explain AGILE model and how it is different from waterfall?
* What is the difference between sprint and scrum?
* What is Retrospective meeting?
* Explain Story and Bug Lifecycle?

**AGILE vs DevOps**

**AGILE: Speaks about Development and Testing**

**DevOps: Speaks about Delivery / Deployment.**

## DevOps is the next automation stage of AGILE

## What is DevOps?

DevOps is a software development method which focuses on communication, integration, and collaboration among IT professionals to enables rapid **deployment of products**.

DevOps is a culture that promotes collaboration between Development and Operations Team. This allows deploying code to production faster and in an automated way. It helps to increases an organization’s speed to deliver application and services. It can be defined as an alignment of development and IT operation.

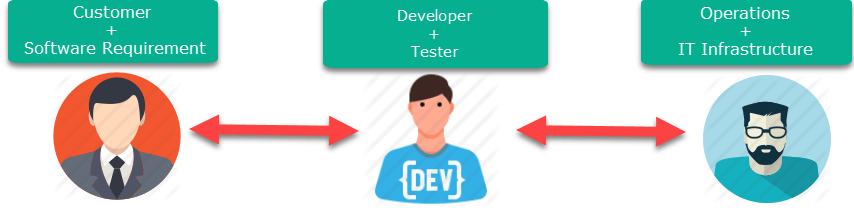
## What is Agile?

Agile Methodology involves continuous iteration of **development and testing** in the SDLC process. This software development method emphasizes on iterative, incremental, and evolutionary development.

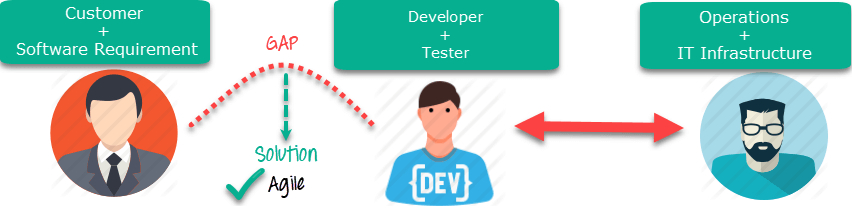
Agile development process breaks the product into smaller pieces and integrates them for final testing. It can be implemented in many ways, including scrum, kanban, scrum, XP, etc.

## Agile Vs. DevOps

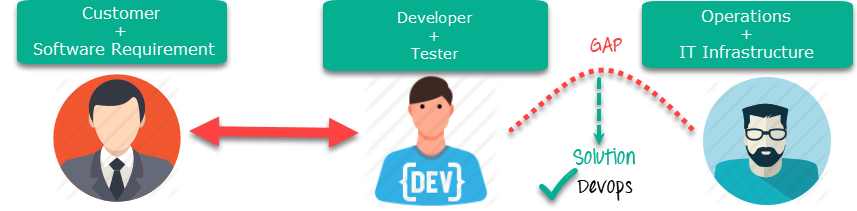
Stakeholders and communication chain in a typical IT process.



Agile addresses gaps in Customer and Developer communications



DevOps addresses gaps in Developer and IT Operations communications



## Difference between Agile and DevOps

|  |  |  |
| --- | --- | --- |
| Parameter | Agile | DevOps |
| What is it? | Agile refers to an iterative approach which focuses on collaboration, customer feedback, and small, rapid releases. | DevOps is considered a practice of bringing development and operations teams together. |
| Purpose | Agile helps to manage complex projects. | DevOps central concept is to manage end-to-end engineering processes. |
| Task | Agile process focusses on constant changes. | DevOps focuses on constant **testing and delivery**. |
| Duration | Agile development is managed in units of **“sprints”.** This time is much less than a month for each sprint. | DevOps strives for deadlines and benchmarks with major releases. The ideal goal is to deliver code to **production DAILY or every few hours**. |
| Automation | Agile doesn’t emphasize on automation. | **Automation** is the primary goal of DevOps. It works on the principle to maximize efficiency when deploying software. |
| Goal | It addresses the gap between **customer need and development & testing teams.** | It addresses the gap between **development + testing and Ops.** |
| Tools used | JIRA, Bugzilla, Kanboard are some popular Agile tools. | Puppet, Ansible, Jenkins, Docker, K8s, AWS are popular DevOps tools. |
| Challenges | The agile method needs teams to be more productive which is difficult to match every time. | DevOps process needs to development, testing and production environments to streamline work. |
| Advantage | Agile offers **shorter development cycle** and improved defect detection. | DevOps supports Agile’s release cycle. |