Software Development Life Cycle

by Devmountain Tutorials

WHAT'S COVERED

This section will explore the software development life cycle by discussing:

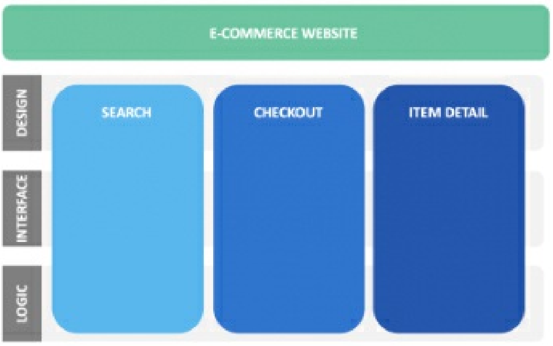
1. HOW DO LARGE COMPANIES ORGANIZE THEIR ENGINEERS?
2. WHAT ARE THE PHASES OF THE SOFTWARE DEVELOPMENT LIFE CYCLE?

Have you ever wondered why large tech companies need tens of thousands of engineers? How does anyone know what to do all day? If a company wants to implement a new business process and needs new hardware or software to support it, how does it happen? While programming is critical to software development, it only represents one phase of the overall process.

1. HOW DO LARGE COMPANIES ORGANIZE THEIR ENGINEERS?

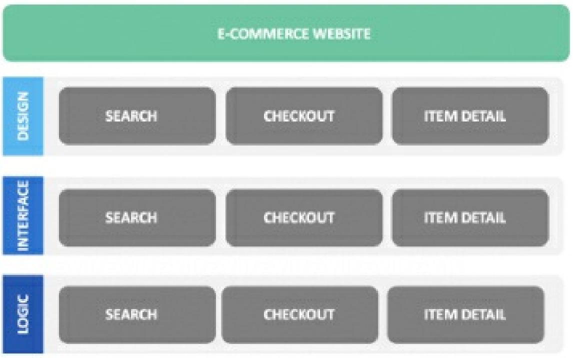
While a single software engineer may be capable of building a product on their own, most companies leverage development teams to reduce the time required to build large products. Development teams are usually organized around a 'product' or a 'functional area.' The number of teams and size of each team will vary by organization, the amount of software being created, and budgets.

Product teams for an e-commerce website could include a team for search, a team for checkout, and a team for item details. The search product team would be responsible for designing and developing all aspects of that experience everywhere it exists. This can be advantageous because the users will see a consistent search experience no matter what part of your software they are using. The complexity of organizing teams by product or feature is that it can require team members to know and manage multiple technologies or skill sets.



Product Team

An e-commerce website organized by functional area could include a team to create designs, a team to implement the website layout, or **interface**, a team to implement the logic. In this structure, the design team would create designs for everything in the entire company. This can be advantageous for teams that are highly specialized to focus on their one activity. The complexity of organizing teams like this is making sure the timing and dependencies between the teams are organized and managed so they can each complete their portion of the work when it is needed. This requires great organizational communication and prioritization.



Functional Team

EXAMPLE Because Poodle Jumper is a small organization, we have one product team that does it all.

In the next section, you will see the different phases that are generally part of the software development cycle and each organization may go through these phases in a unique pattern.

TERM TO KNOW

Methodology
The process or method for how something is done.

Product
What the company builds for its users. Most companies have multiple products. Sometimes you'll hear developers talk about applications or features.

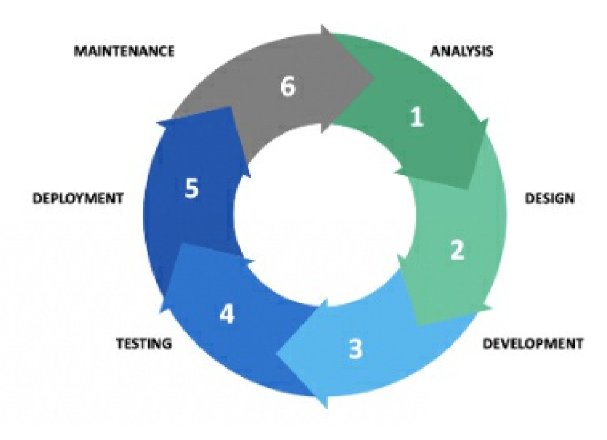
Functional Area
A group of people with the same activity or purpose.

Interface
What the user sees on the screen when they use software. A good portion of software code happens behind the scenes and controls what you actually see on the screen.

2. WHAT ARE THE PHASES OF THE SOFTWARE DEVELOPMENT CYCLE?

The process of developing quality software is not as simple as sitting down and writing code. Most of the time, the creation of software is a resource-intensive process that often involves several different groups of people in an organization and several individuals on a team.

The **software development life cycle** (SLDC) provides the general phases for teams to follow for developing and releasing software. Each phase is characterized by a specific deliverable that should be developed by the end of the phase.



Software Development Life Cycle

ANALYSIS

'Analysis' is the first phase of the process. In this phase, the idea is reviewed. Does this align with our vision and strategy? Is creating a solution possible? What alternatives exist? What is currently being done about it? How do we know it is a problem, and how will we know it is fixed? This step is important in determining if the idea should even be developed. The team members work with customers or stakeholders to determine the specific requirements. The result of this phase is documented requirements that feed into phase 2: design.

DESIGN

The second phase is **design**. This is when the team takes the requirements and research to design how the feature or product should work. This can include when mockups or prototypes are created and shared with customers or stakeholders for feedback. It can also include designing specific technical details like the architecture or identifying risks and dependencies. This results in design artifacts and an agreed-upon approach for the next phase: development.

DEVELOPMENT

The third phase, **development** is when code is written. Using the requirements and designs, a programmer creates the software by breaking down the feature or product into really small chunks. This phase usually requires the most time and effort, but that isn't always the case. It really depends on what you are building, the team building it, and your organization. The result of this phase is an initial working code that meets the requirements identified in the previous phases.

TESTING

In the **testing** phase, the code developed in the previous phase is put through a series of structured tests. Some development teams are continuously testing their code as it is created with unit tests, integration tests, and automation. User Acceptance testing allows the team to verify the development with real users before it is released to everyone. Any bugs, errors, or problems found during testing are addressed and then tested again. The result of this phase is verified code that is ready to be released.

HINT

In software, the terms bug, defect, issue, or problem can be used interchangeably to mean something that is unexpected or not working correctly. The first versions of computers were so large they had real insects get caught in the electronics and cause problems. The term has been attributed to programming pioneer, Grace Hopper, who described an incident where a technician pulled moth from a computer.

DEPLOYMENT

Once the code is developed and tested, it has to be implemented. This is referred to as a 'deployment,' or a release. This phase includes communication, training, and providing documentation or release notes. This is typically the final stage of initial development, in which the system is put into production for the users or organization to use. The frequency of releases depends on the software and the organization's technical maturity. Some software teams push several releases a day and some can go more than a year between major updates.

EXAMPLE At Poodle Jumper, we release updates to our software every few weeks.

MAINTENANCE

Maintenance is the last phase of the SDLC. It includes a structured support process so reported bugs are fixed and requests for new features are evaluated and implemented. It includes measuring performance and verifying that the outcome the software was designed to achieve actually happened. If the outcome isn't met or new ideas or features should be implemented, the process starts all over again with analysis.

The duration, process, and flow between phases will vary between different methodologies. We'll discuss methodologies in the next section.

TERM TO KNOW

Software Development Life Cycle (SDLC)

The general phases for teams to follow for developing and releasing software. Each phase is characterized by a specific deliverable that should be developed by the end of the phase.

Analysis

This step is important in determining if the idea should even be developed. The team members work with customers or stakeholders to determine the specific requirements.

Design
This phase is when the team takes the requirements and researches to design how the feature or product should work.

Development
This phase includes breaking down the design into small chunks to be developed and writing code for each chunk. The result of this phase is an initial working code that meets the requirements identified in the previous phases.

Testing

Plans are made to test the code that has been developed in a structured process during this phase. Any bugs, errors, or problems found during testing are addressed and then tested again. The result of this phase is verified code that is ready to be released.

Deployment
During this phase, the system is put into production for the users or organization to use, often called deployment or release.

Maintenance

This is a structured support process so reported bugs are fixed and requests for new features are evaluated and implemented. It includes measuring performance and verifying that the outcome the software was designed to achieve actually happened.

TERMS TO KNOW

Analysis
This step is important in determining if the idea should even be developed. The team members work with customers or stakeholders to determine the specific requirements.

Deployment
During this phase, the system is put into production for the users or organization to use, often called deployment or release.

Design
This phase is when the team takes the requirements and researches to design how the feature or product should work.

Development
This phase includes breaking down the design into small chunks to be developed and writing code for each chunk. The result of this phase is an initial working code that meets the requirements identified in the previous phases.

Functional Area
A group of people with the same activity or purpose.

Interface
What the user sees on the screen when they use software. A good portion of software code happens

behind the scenes and controls what you actually see on the screen.

Maintenance
This is a structured support process so reported bugs are fixed and requests for new features are evaluated and implemented. It includes measuring performance and verifying that the outcome the software was designed to achieve actually happened.

Methodology
The process or method for how something is done.

Product
What the company builds for their users. Most companies have multiple products. Sometimes you'll hear developers talk about applications or features.

Software Development Life Cycle (SDLC)
The general phases for teams to follow for developing and releasing software. Each phase is characterized by a specific deliverable that should be developed by the end of the phase.

Testing
Plans are made to test the code that has been developed in a structured process during this phase. Any bugs, errors, or problems found during testing are addressed and then tested again. The result of this phase is verified code that is ready to be released.