

# **Module 1: Soft-Dev Principles**

## **CSC3350: Software Development**

**Fall 2023**

Dr. Tushara Sadasivuni, Dr. William Greg Johnson  
Department of Computer Science  
Georgia State University

# Module1: Software Development Principles

- Topic Specifics

- SDLC Software Development Life Cycle or App Life Cycle
- Beyond the S.O.L.I.D. design principles
- Includes organizational structures for efficiency
- Some are simple to understand and operationalize
- Main development methodologies:
  - Agile
  - Waterfall
  - Component based

- Tutorial for Soft Dev from Microsoft

Featured playlist

1



7 videos

[Software Development Fundamentals](#)  
[Akram F. Sulaim](#)

# SDLC or App Life Cycle

- SDLC and App Life Cycle are closely related. The SDLC is a broader term that can be applied to any type of software development, while the App Life Cycle is a subset and considered to be focused on the development in mobile applications.
- The older term 'Application Lifecycle Management' has been more recently linked to the waterfall model of software development.

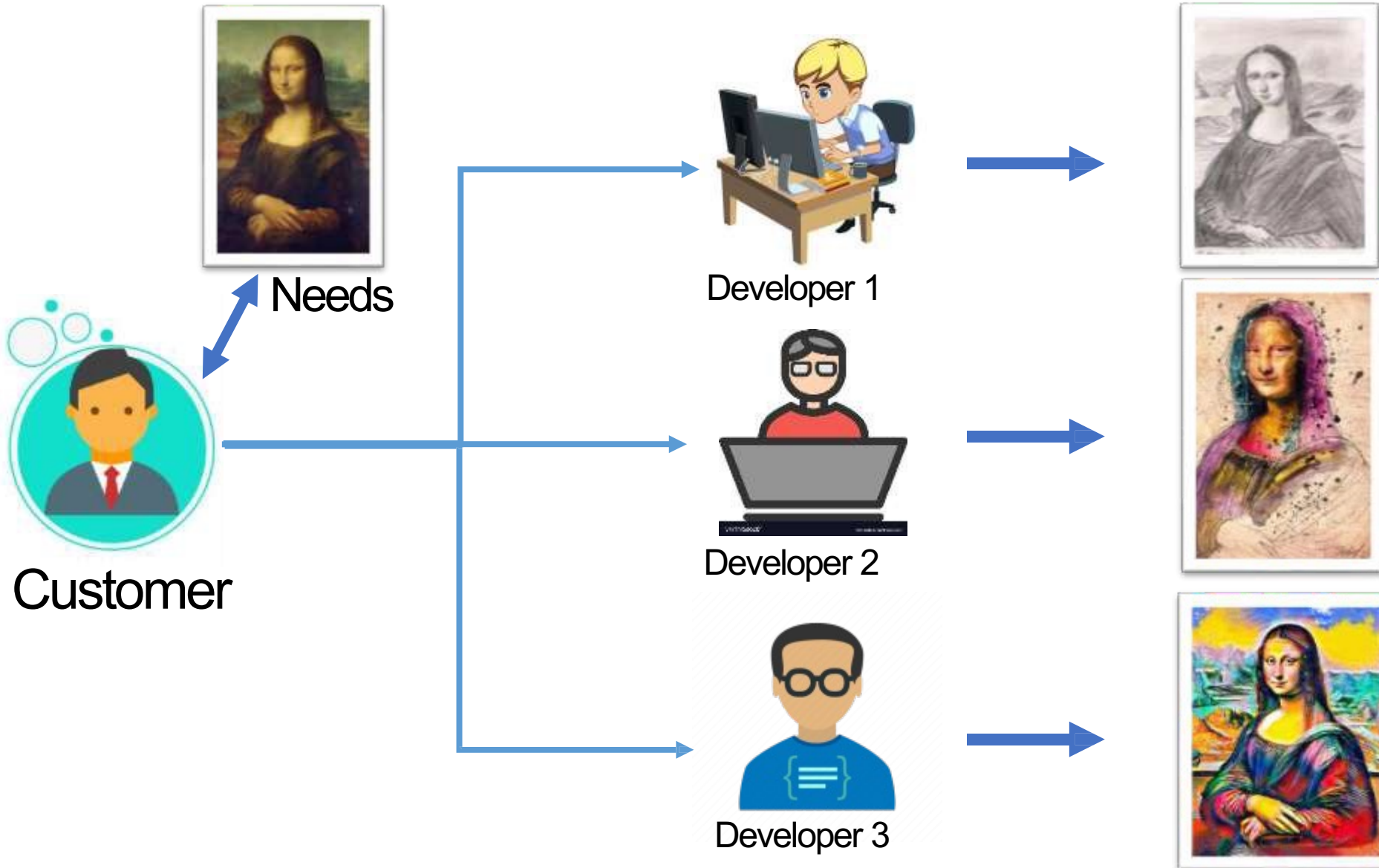
# Phases (stages)

- Phase 1: Requirement collection and analysis (user needs)
- Phase 2: Feasibility study (system and/or business analyst)
- Phase 3: Design (abstraction and algorithms, UX)
- Phase 4: Implementation (coding and debugging)
- Phase 5: Testing (automated and human required)
- Phase 6: Installation/Deployment (Dev Ops and systems engineering)
- Phase 7: Maintenance (updates and upgrades)

# Realistic SDLC phases:

- **Phase 1:** Requirement collection and analysis (user needs) and feasibility study (system and/or business analyst)
- **Phase 2:** Design (abstraction and algorithms, UX)
- **Phase 3:** Implementation (coding and debugging)
- **Phase 4:** Testing (automated and human required)
- **Phase 5:** Evolution is installation or deployment (Dev Ops and systems engineering) and ongoing maintenance (updates and upgrades)

# Is software development (SDLC) an art?



3

# SDLC: a long cycle...



3

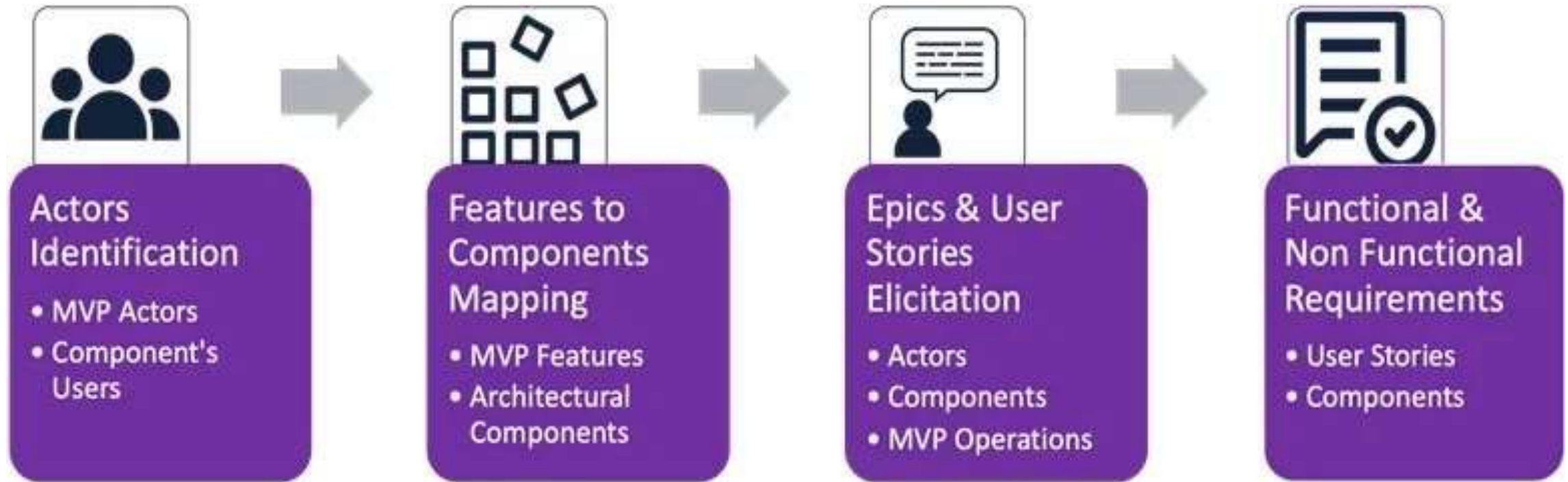
# SDLC Phases: Requirement collection and analysis (user needs)

- Defining requirements is considered part of planning to determine what the application is supposed to do and its requirements. For example, a social media application would require the ability to connect with a friend. An inventory program might require a search feature.
- Requirements also include defining the resources needed to build the project. For example, a team might develop embedded software to control a custom manufacturing machine. The micro-controller and machine are a requirements in the process.
- Often referred to as 'Requirements Engineering'



# SDLC Phases: 'Requirements Engineering'

3



# SDLC Phases: 'Requirements' as user story

3



**As a** <role>  
**I want** <goal>  
**so that** <benefit>

**Acceptance criteria:**  
(Conditions of Satisfaction)

...

...

**As an** Account Manager  
**I want** a sales report of my account  
to be sent to my inbox daily  
**So that** I can monitor the sales  
progress of my customer portfolio

## Acceptance criteria:

1. The report is sent daily to my inbox
2. The report contains the following sales details: ...
3. The report is in csv format.

# SDLC Phases: Design

The Design phase models the way a software application will work.

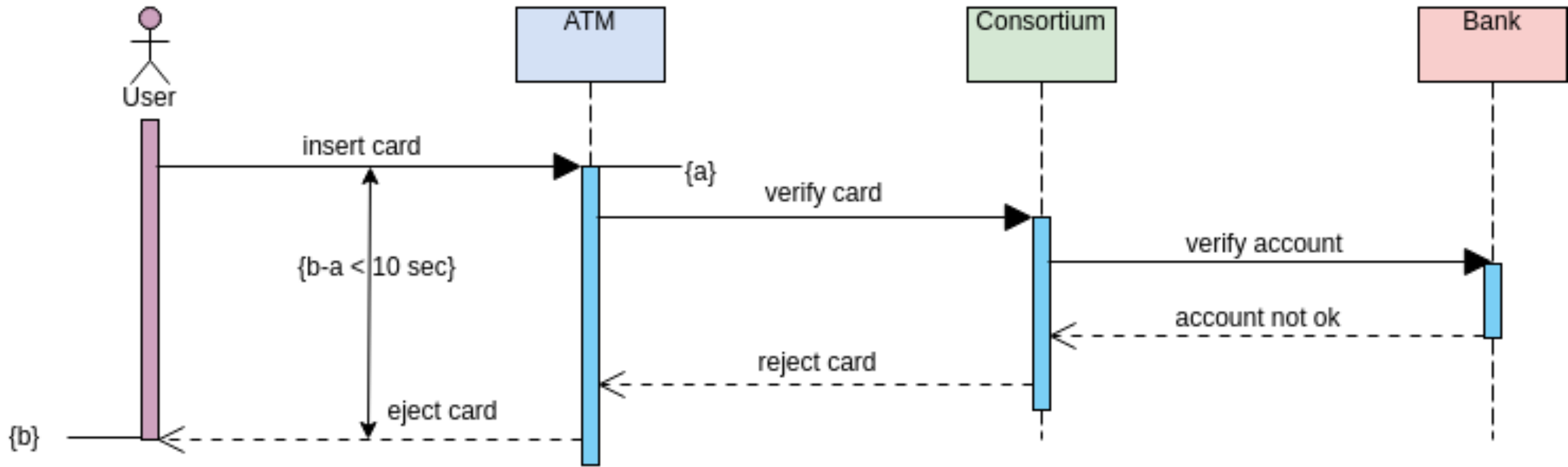
- Architecture – Specifies programming language, industry practices, overall design, and use of any templates or boilerplate
- UX– Defines the ways customers interact with the software, and how the software responds to input
- Platforms – Defines what the software will run upon, such as Apple, Android, Windows version, Linux, or even gaming consoles
- Programming – Not just the programming language, but including methods of solving problems and performing tasks in the application
- Communications – Defines the methods that the application can communicate with other components, such as a central server or other instances of the application
- Security – Defines the measures taken to secure the application, and may include SSL traffic encryption, password protection, and secure storage of user data

# SDLC Phases: Design

- Prototyping can be a part of the Design phase. A prototype is the early versions of software in an iterative software development model. It demonstrates a basic idea of how the application looks and works. This “hands-on” design can be shown to stakeholders. Use feedback to improve the application. It’s less expensive to change the Prototype phase than to rewrite code to make a change in the Development phase.

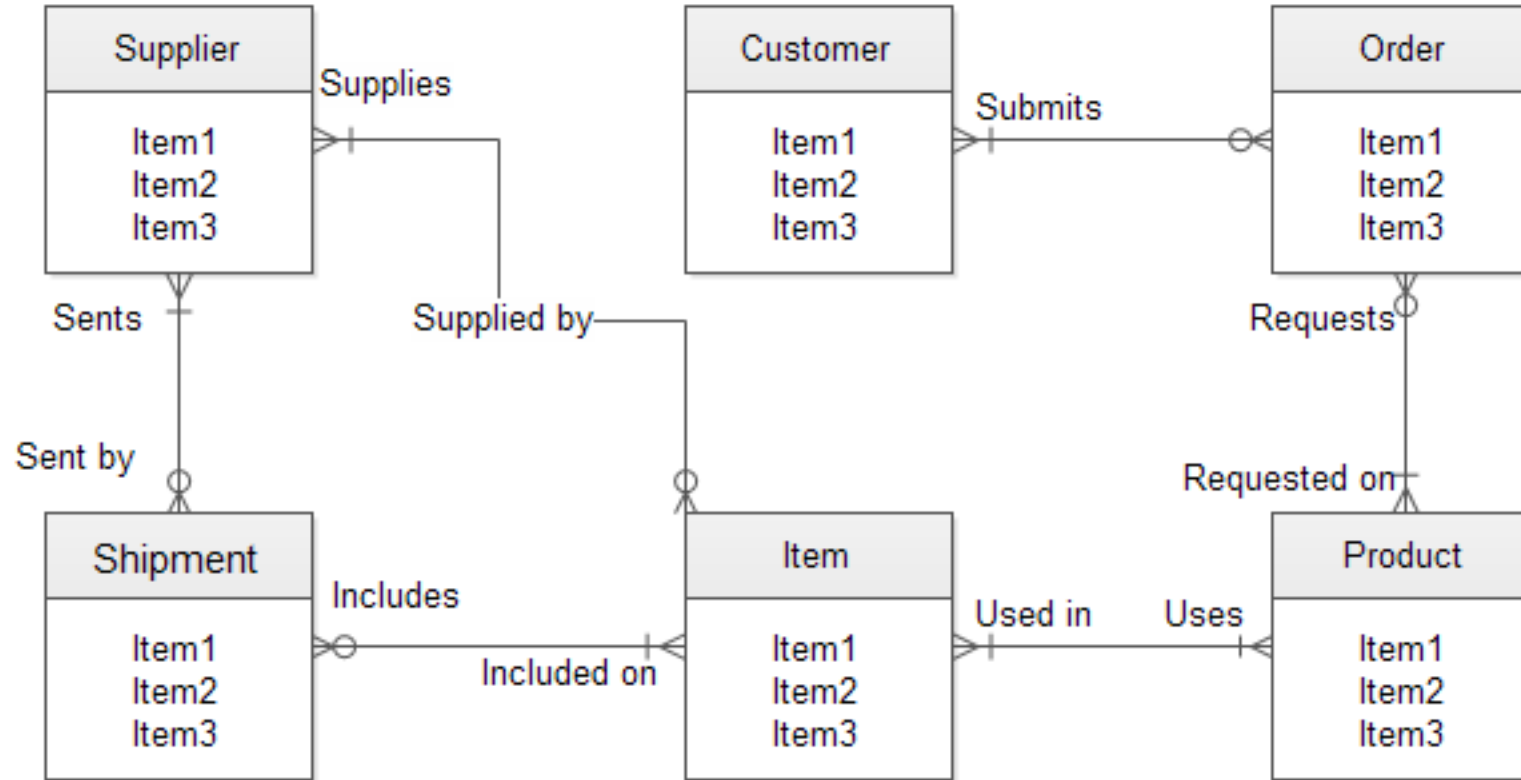
# SDLC Phases: Design

2



UML Sequence Diagram

# SDLC Phases: Design



4

Entity Relation Diagram  
data access object (DAO)

# SDLC Phases: Implementation (coding and debugging)

- This is the actual writing of the program. A small project might be written by a single developer, while a large project might be broken up and worked by several teams. Use a Source Code Versioning application (GIT) in this phase. These systems help developers track changes to the code. They also help ensure compatibility between different team projects and to make sure target goals are being met.
- The coding process includes many other tasks. Many developers need to brush up on skills or work as a team. Tasks often hold up the development process, such as waiting for test results or compiling code so an application can run. SDLC can anticipate these delays so that developers can be tasked with other duties.
- Software developers appreciate instructions and explanations. Documentation can be a formal process, including writing a user guide for the application, usually part of a versioning system. It can also be informal, like comments in the source code that explain why a developer used a certain procedure. Even companies that strive to create software that's easy and intuitive benefit from the documentation.



# SDLC Phases: Testing (automated and human required)

It's critical to test an application before making it available to users.

- Much of the testing can be automated, like security testing. Other testing can only be done in a specific environment – consider creating a simulated production environment for complex deployments.
- Testing should ensure that each function works correctly. Different parts of the application should also be tested to work seamlessly together—performance test, to reduce any hangs or lags in processing.
- The testing phase helps reduce the number of bugs and glitches that users encounter. This leads to a higher user satisfaction and a better usage rate.



# SDLC Phases: Test Case Sample

3

Project Name:

Google Email

Module Name:

Login

Reference Document:

If any

Created by:


Rajkumar

Date of creation:

DD-MMM-YY

Date of review:

DD-MMM-YY


  
[www.SoftwareTestingMaterial.com](http://www.SoftwareTestingMaterial.com)

TEST CASE ID	TEST SCENARIO	TEST CASE	PRE-CONDITION	TEST STEPS	TEST DATA	EXPECTED RESULT	POST CONDITION	ACTUAL RESULT	STATUS (PASS/ FAIL)
TC_LOGIN_001	Verify the login of Gmail	Enter valid User Name and valid Password	1. Need a valid Gmail Account to do login	1. Enter User Name	<Valid User Name>	Successful login	Gmail inbox is shown		
				2. Enter Password	<Valid Password>				
				3. Click "Login" button					
TC_LOGIN_001	Verify the login of Gmail	Enter valid User Name and invalid Password	1. Need a valid Gmail Account to do login	1. Enter User Name	<Valid User Name>	A message "The email and password you entered don't match" is shown			
				2. Enter Password	<Invalid Password>				
				3. Click "Login" button					
TC_LOGIN_001	Verify the login of Gmail	Enter invalid User Name and valid Password	1. Need a valid Gmail Account to do login	1. Enter User Name	<Invalid User Name>	A message "The email and password you entered don't match" is shown			
				2. Enter Password	<Valid Password>				
				3. Click "Login" button					
TC_LOGIN_001	Verify the login of Gmail	Enter invalid User Name and invalid Password	1. Need a valid Gmail Account to do login	1. Enter User Name	<Invalid User Name>	A message "The email and password you entered don't match" is shown			
				2. Enter Password	<Invalid Password>				
				3. Click "Login" button					

# SDLC Phases: Installation/Deployment (Dev Ops and systems engineering)

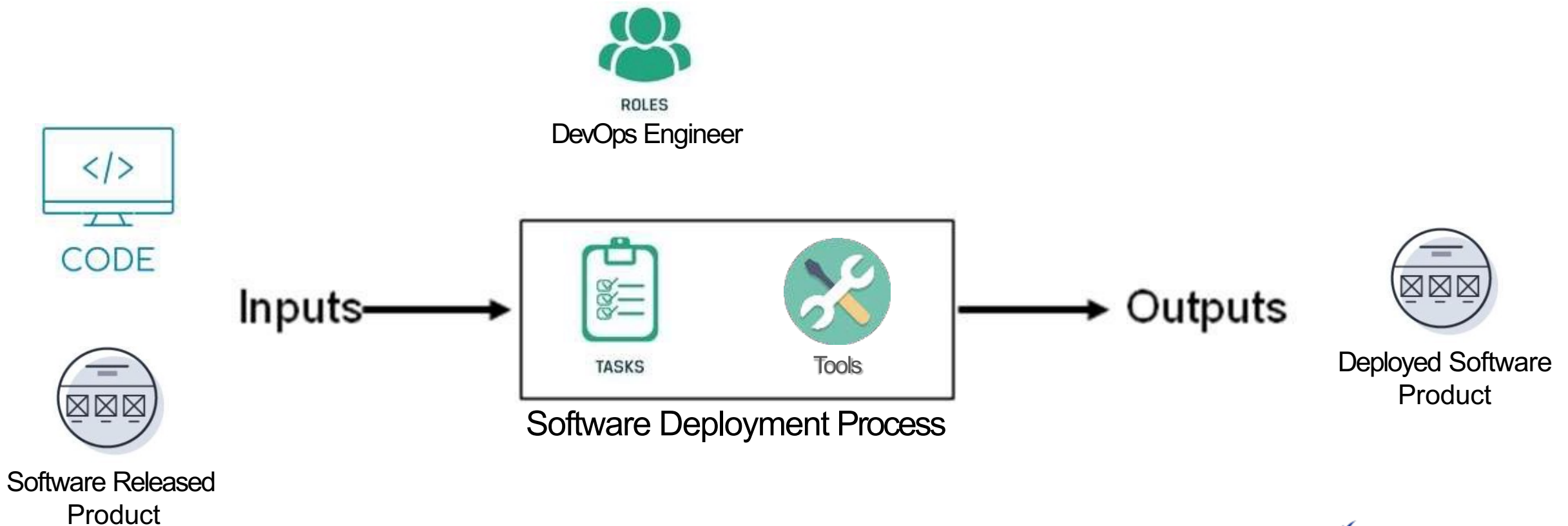
In the deployment phase, the application is made available to users.

3

- Many companies prefer to automate the deployment phase. This can be as simple as a payment portal and download link on the company website. It could also be downloading an application on a smartphone.
- Deployment can also be complex. Upgrading a company-wide database to a newly-developed application is one example. Because there are several other systems used by the database, integrating the upgrade can take more time and effort.

# SDLC Phases: Installation/Deployment (Dev Ops and systems engineering)

3



# SDLC Phases: Maintenance (updates and upgrades)

3

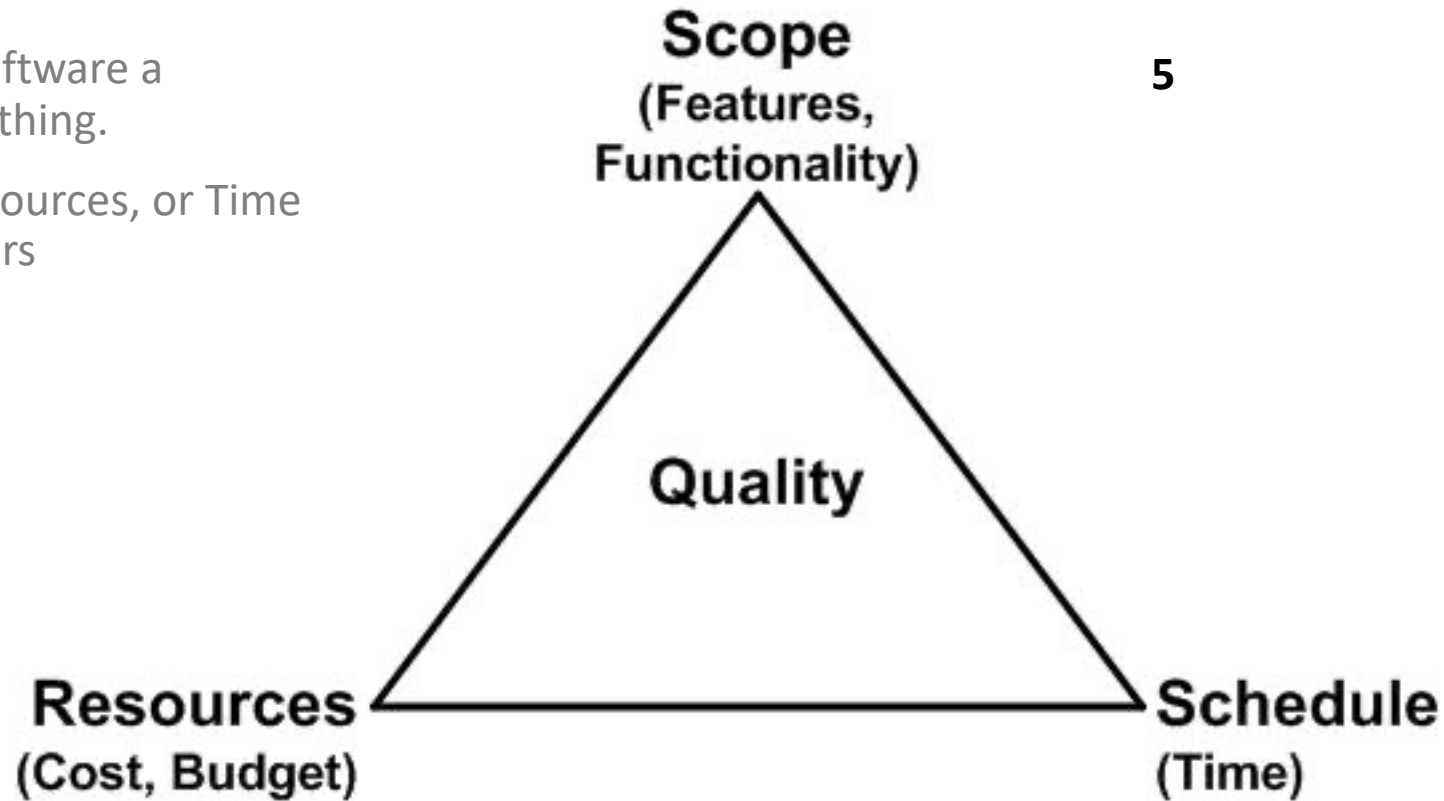
At this point, the development cycle is almost finished.

- The application is done and being used in the field. The Operation and Maintenance phase is still important, though. In this phase, users discover bugs that weren't found during testing. These errors need to be resolved, which can spawn new development cycles.
- In addition to bug fixes, models like Iterative development plan additional features in future releases.
- For each new release, a new Development Cycle can be launched.

# Iron Triangle is real...

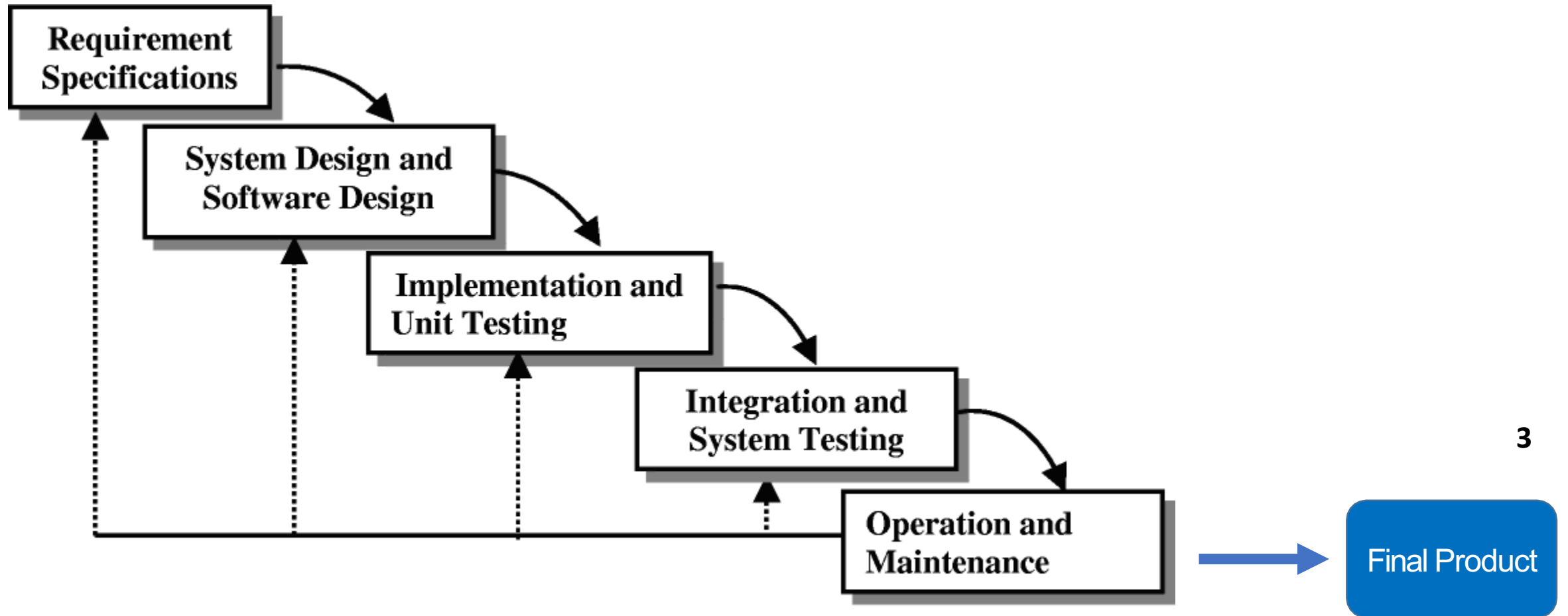
At this point, the impacts to software a development cycle is a simple thing.

- Any change to Features, Resources, or Time changes the other two factors



Copyright 2003-2006 Scott W. Ambler

# Software Development Methodology

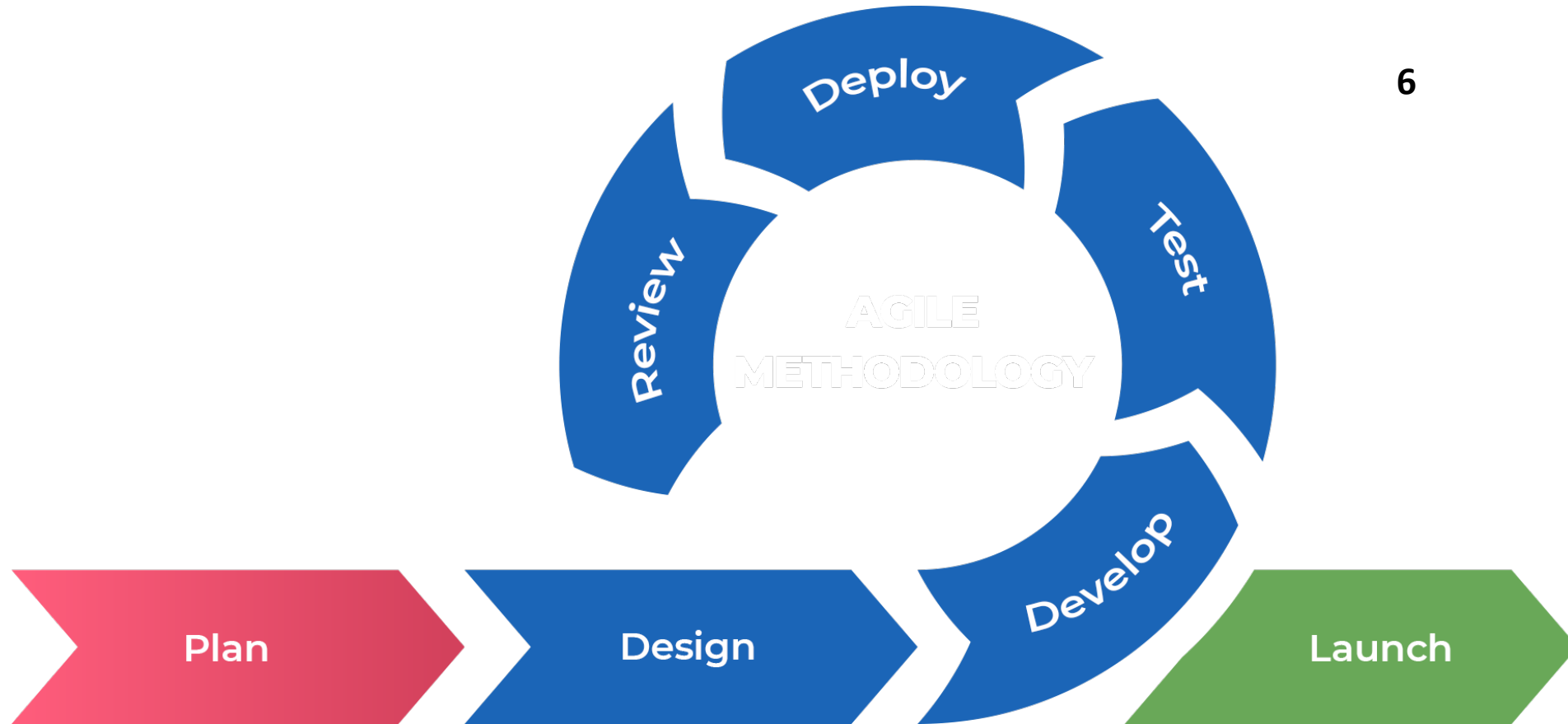


Waterfall

# Disadvantages of “Waterfall” Approach

- In this model, the risk factor is higher, so this model is not suitable for more significant and complex projects.
- This model cannot accept any changes in requirements later, during implementation/development.
- It becomes expensive to go back to the design phase. For example, if the application has now shifted to the coding phase, and there is a change in requirement.
- Since the testing done at a later stage, it does not allow identifying the risks from the earlier phase, so the risk reduction strategy is difficult to prepare.

# Software Development Methodology



6

Agile



# Agile: when, who, why

7

- 1970's, a more flexible way for project management was introduced to handle change during development. Driving industries: aerospace, computer, defense industries
- 2000, group of 17 software developers met to discuss how to speed up the development process for new software.
- 2001, Agile manifesto was introduced to the software development and project management world

# Agile Manifesto: 4 key values

1. Individuals and interaction of process and tool use
2. Working software over comprehensive documentation
3. Customer collaboration over contract negotiation
4. Responding to change over following a plan



3

# When to use the Agile Methodology?

3

- When frequent changes are required.
- When a highly qualified and experienced team is available.
- When need to respond to market dynamics and complete more of their projects successfully.
- When project contains ambiguities in requirements such as R&D projects.
- When the project is complex and takes long time to be developed.

# Agile methodologies

- SCRUM
- Kanban
- DevOps
- Hybrid
- Others: Crystal, Lean, XP, Bimodal
- Agile principles are important because they provide a guide for how software development should be approached. They emphasize collaboration, customer focus, and continuous improvement. These values are very relevant today and help to make sure that software development projects are successful.
- Communication is a critical, foundational part of Agile

# End

Questions?

## **Module 1: Soft-Dev Principles**

### **CSC3350: Software Development**

# Attributions

1. <https://youtube/3zFBV4GwlWQ?si=L2yEY67NBJwbdf>
2. <https://online.visual-paradigm.com/diagrams/templates/sequence-diagram/sequence-diagram-simple-atm-example/>
3. [https://www.slideshare.net/MohamedElshaikh10/software-development-methodologiespptx?from\\_search=28](https://www.slideshare.net/MohamedElshaikh10/software-development-methodologiespptx?from_search=28)
4. <https://www.edrawmax.com/er-diagram/how-to-make-an-er-diagram/>
5. <https://ambysoft.com/essays/brokentriangle.html>
6. <https://www.credencys.com/our-methodology/>
7. <https://study.com/learn/lesson/agile-methodology-history-development.html>