# Week 03: Instructor Notes

## Overview

The students will be researching how QA-Testing is an integral part of the software lifecycle and how Testing is applied to different software methodologies. .

## Objectives

By the end of the week, each student will be able to:

* Summarize how Testing fits into different S/W Development Lifecycles.

## Before the Week Begins (Prepare)

### Partnership/Group work:

* Students will be assigned partnerships
* Each partnership will be given an assigned Methodology and Lifecycle Phase (Group Number). Partners are to create a poster on the assigned Methodology and Software Lifecycle Phase relating to Testing and be prepared to present it to the class.

## Looking Ahead

Next week's announcements:

* Please create and post your Announcement for W04 by Friday of this week.

# Prepare

## Overview

Testing is significant in each phase of any software product and lifecycle. Testing is involved in every aspect of any software lifecycle, from requirements development to design to integrating & delivering.

* This week we will explore each stage of different lifecycles and the roles, responsibilities, and products of Software Testing.
* This week is an overview of Software Lifecycles and not a replacement for CS 272. It focuses on how Testing is part of each of the lifecycles and methodologies.

## Objectives

By the end of the week, each student will be able to:

* Summarize how Testing fits into different S/W Development Lifecycles

## Preparation Material

To be prepared for this module's activities, please read the following and be prepared to start or complete activities for the Teach One Another and Prove assignments.

### Reading

* See [Reading Materials](../Reading/Reading.html)

## Additional Information: Methodology Phases

### Methodologies – Lifecycle Phase:

* Code and Fix - Concept Dev, Code and Fix, Release
* Waterfall – Requirements, Design, Implementation, Testing,
* Spiral – Objectives, Planning, Risk Analysis, Prototyping, Requirement Validation, Design V&V, Simulations/Models/Benchmarks, Detailed Design, Unit test, Integration Test, Acceptance Test, Spirals 1,2,3,4
* Incremental – Req, Analysis, Design, Implementation, Testing, Deployment, Maintenance
* V-Model   - CONOP, Req, Design, Implementation, Testing (Unit, Integration, System, Acceptance), Verification, Validation
* XP – Release Plan, Iteration Plan, Acceptance Test, Unit Test, Pair Programming
* Agile – Brainstorm, Design, Develop, QA, Deployment
* Continuous Integrations and Development – Agile, Integration, Delivery, Deployment, DevOps

The software has a lifecycle, much like a butterfly or moth. Each phase of the development lifecycle can use different methodologies. Each Methodology introduced more detail to each lifecycle phase.

* Concept of Operation
  + Concept
    - The software concept is in Code & Fix, Spiral, and V-Model methodologies. The concept is one of the first stages of the customer's requirement, elicitating what they want and how they think it should work.
  + Software Objectives
    - Software Objectives is the next stage is figuring out the customer's general goals. Even though only spelled out in Spiral, it required for all methodologies.
  + Risk
    - Risk is a common practice throughout all methodologies, but specifically to Spiral. All mature software projects are constantly identifying risks to mitigate issues and problems.
* Requirements
  + System Requirements
    - System Requirements are a high-level list of requirements that usually identify hardware, operating system, communication, and architecture specifications. Specifically mentioned in the Waterfall, System Requirements are needed in all aspects of the Software Development Lifecycle.
  + Software Requirements
    - Software Requirements are specifications specific to the developed software. These requirements interact with the system requirements and identify all the functionality of the developed software. Waterfall and Spiral specifically call these out.
  + Architecture
    - The Architecture phase is the part of the System Requirements and is used to define better what resources the software will run on. Found mostly in V-Model explanations.
  + Requirements Validation
    - Requirements Validation lay out the preliminary research for testing the System and Software Requirements. It is necessary to test each requirement, know as verification. And that the requirements reflect what the customer wants, known as validation. Typically, this is done through a Software Requirements Review (SRR) meeting and is a major milestone in the Software Lifecycle. This phase in the Spiral Methodology is specifically defined.
* Design
  + Analysis
    - The analysis is usually started during the requirements phase and finished during the development phase. It takes the requirements and starts the transformation process breaking the requirements down when programmers can start implementing them in code. This process also addresses product, customer, data, and process workflows.
  + Program Design
    - The Program Design is a process of taking the requirements and coming up with solutions to implement it in code. Program Design, also called Product Design or Detailed Design, is introduced in Waterfall and further defined and explained in Spiral and V-Model methodologies.
  + Prototyping
    - Prototyping is a technique of creating representatives of the software functionality to show to the customer. Spiral Methodology heavily relies on the technique and is used in Demonstrations during Agile. Prototypes do not typically work, and care should be taken to prevent customers from thinking that it is the current working product.
  + Product Validation and Verification
    - As part of the design phase's last step, culminating in a Software Design Review meeting (Waterfall & Spiral), the team makes sure that the customer wants and the design fulfills requirements.
* Implementation
  + Coding/Implementation
    - Coding and Implementation are where the tires meet the road. It is the test if the Requirements and Design are detailed enough to translate the customer's needs down to executable binary code. Found in all methodologies, it is the most important lifecycle phase.
  + Build
    - The Build phase is more complicated than just compiling the code. It involves making sure that it works with other parts of the System, Architecture, and Design. It properly links all the parts together into a simplified process. With the recent development of Continuous Integration and Delivery, this phase has come into its own.
* Testing
  + Unit Testing
    - Unit Tests are typically written (before or) during (or after) the coding and implementation phase. Its main focus is on Black Box testing each of the small coded components to verify that they work correctly.
  + Simulations/Simulators
    - Much like prototyping, simulator and simulations provide a way to verify that the system, architecture, and software work correctly. Specifically spelled out in Spiral, simulators help when the hardware is not available. Simulations model the system to provide performance benchmarks and are a valid testing method known as Testing by Analysis.
  + Integration Testing
    - As part of verifying the build phase, Integration Testing makes sure that interfaces between minor and major parts are working together. That software is properly talking to the architecture, and the system runs as designed.
  + System Testing
    - Found in Waterfall and V-Model methodologies, System Testing is a set of tests to ensure that the identified workflows are working as expected.
  + Acceptance Testing
    - Spiral and XP methodologies focus on Acceptance testing to verifying and validating the originals customer's needs. Typically, just before deployment, a meeting, SAT (Software/System Acceptance Test), is done with the customer. Walkthrough and demonstration tests address each of the customers' needs and concerns.
* Deployment
  + Deployment and Version Release
    - After acceptance with the customer, the customer received the software. The software package includes hardware (if needed), software packages for support, and scripts and programs to configure and set up the software. This phase address concerns of releases and DevOps. The XP methodology specifically requires a Release plan and Agile addresses Deployment procedures.

# What are Tester's Responsibilities?

## Overview

For this assignment, we will be exploring specifically what roles and responsibilities do testers have in the different lifecycles.

## Instruction

Imagine that you are a software tester and asked to attend two of the following meetings:

* Concept Review
  + This meeting usually happens after software Requirements Elicitation has been done. Typically, in the form of Task Analysis, Domain Analysis, and Data Analysis. Some would consider this the project proposal. Considering it a proposal, the customer wants to know, "What parts of the product will Testing verify and what will not? "
* Requirement review
  + The Software Requirements Review (SRR) meeting usually is after the Software Requirements Specification (SRS) is complete. There are two main sections to the SRS. The first is the software requirements, which only specify what and not how it will be implemented. It also provides information about the users and accounts, the elements/components, and the system's workflow/use cases. The second, Software requirement validation, is testing at a high-level approach. The SRS will also cover preliminary architecture, data storage, security, etc.
* Design Review
  + The Software Design Review (SDR) meeting usually is after the Software Design Description (SDD) document has been completed. The SDD will consist of enough details for a programmer to implement the code. Usually, the QA-Tester is working on Acceptance, System, Regression & Integration Test Plans.
* Code Review
  + As the developer finishes implementing each design unit, a mature project required Peer Reviews.  Peer Reviews reduce the number of defects in the delivered product. Software programmers are typically required to provide unit tests as part of their deliverables. These unit tests use developed tests for Acceptance, System, Regression & Integration Tests, so QA-Tester has a vested interest in these tests' detail and quality.
* Integration Review
  + After programming each unit, the next step is integrating the units into components, sub-systems, and systems. Integration tests consist of Interface and Component testing. After addressing integration conflicts for each unit, component, and interface, the  Integration Team, will Review the Integration Plan and Tests and verify they are complete providing evidence that those elements work together as planned.
* System Review
  + The integration of the major components, sub-systems, and systems allows the Workflow and End-to-End processing to be tested and verified. The System review provides a formal deliverable that elements are working as designed.
* Acceptance Review
  + The final product presentation of how it fulfills the customer's business workflow and problems finishes and accepts the work completed.
  + Demos are typically one of the major ways of showcasing the product, along with reviews and validation reports.
* Delivery Review
  + As will all good products, just delivering it to just the development environment is not even close to it working in the real world. DevOps is the process of making sure the product does not fall into the trap of "It worked on my machine." Have the proper configuration for different customers and charging costs is essential to profit and customer satisfaction.

Create a role description. A Role Description is a brief about a particular role a person would play in a review and the responsibilities and items that role is looking for to assure quality (checklist).  
The role is a Tester. Describe the responsibilities that a test would have in the meeting. Provide a five-item checklist that that role will be examining during the meeting to ensure that the product passes to the next phase. Finally, what proof, deliverable, would a Tester need to provide that the tester approved or what conditions would be needs to get approval to move to the next phase of development.

Provide 2 Role Descriptions, one for each of the two meetings you selected.

## Rubric

Use the following rubric to help understand the expectation.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Exceptional 100%** | **Good 90%** | **Acceptable 70%** | **Developing 50%** | **Missing 0%** |
| **1st Role Description  45%** | Either 6+ checklist items, additional information, or identified above and beyond | Well researched and cited | Contains Responbilites, Checklist (5 items), **and** Deliverable information | Contains Responbilites, Checklist (5 items), **or** Deliverable information | No Role Description |
| **2nd Role Description  45%** | Either 6+ checklist items, additional information, or identified above and beyond | Well researched and cited | Contains Responbilites, Checklist (5 items), **and** Deliverable information | Contains Responbilites, Checklist (5 items), **or** Deliverable information | No Role Description |
| **Professionalism 10%** | The paper is easy to read and communicated. | Properly cited, there are no grammar or spelling errors, and the writing style is "professional." | Found an instance of a spelling error, grammar error, incomplete citation, overly verbose wording, poor formatting, or poor writing. | A citation is missing where one is needed (plagiarism alert!). | Gross spelling/grammar errors or other aspects of the writing that make the paper difficult to read. |

The distribution of points starts at 50 percent for the minimal participation expectation, and additionally stated expectations increase the percents.

# What is the relationship between Testing and Software Methodologies?

## Overview

Testing is a major phase of any lifecycle. From requirements development to integrating & delivering valuable products, Testing involves every aspect of any software life cycle.

## Instruction for Posts

Partnership: Each partnership will be assigned a Methodology Phase.

* Partners are to research the assigned Methodology and lifecycle Phase.
* Partners are to create a poster on the assigned Methodology and Software Lifecycle Phase relating to Testing and be prepared to present it.
* Posters should include Testing in the Methodologies and the responsibilities of a Tester in that phase. Answering the following questions:
  + At what point does Testing get involved in the lifecycle?
  + What are the responsibilities, roles, and products affected or produced by Software Testing?

Post a copy of your poster (soft copy) on Testing related to Software Methodologies and Lifecycle.  
Posters should include Testing in the Methodologies and the responsibilities of a Tester in that phase. Answering the following questions:

* At what point does Testing get involved in the lifecycle?
* What are the responsibilities, roles, and products affected or produced by Software Testing?

The poster is a showcase discussion board; please take some time and read over your classmate's posters and use what you learned as part of your pondering.

## Rubric

Use the following rubric to help understand the expectation.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Exceptional 100%** | **Good 90%** | **Acceptable 70%** | **Developing 50%** | **Missing 0%** |
| **The Content relates to Lifecycle 50%** | Partners presented professionally. | Relates Testing to the Lifecycle phase | Describes Lifecycle phase | Mentioned lifecycle phase |  |
| **Methodology Content 10%** | They identified the pre/post lifecycle phase in methodologies | Provided 2+ methodologies that lifecycle appears and relates to Testing | Describes the Methodology that the Lifecycle phase occurs. | Mentioned a methodology | No Questions provided. |
| **Roles and Responsibilities 20%** | Answered two questions from two different posts and added new information. | Answered two questions from two different classmates | Identified multiple responsibilities | Listed an example of what a Tester would do | No Answer provided |
| **Professionalism 10%** | The paper is easy to read and communicated. | Properly cited, there are no grammar or spelling errors, and the writing style is "professional." | Found an instance of a spelling error, grammar error, incomplete citation, overly verbose wording, poor formatting, or poor writing. | A citation is missing where one is needed (plagiarism alert!). | Gross spelling/grammar errors or other aspects of the writing that make the paper difficult to read. |

The distribution of points starts at 50 percent for the minimal participation expectation, and additionally stated expectations increase the percents.

# Prove: QA-Tester Review Checklists

## Overview

Create a checklist for each of the different phases in the software process.

## Backstory

As an individual contributor to a project, you will fulfill one or many roles; each role is a set of responsibilities. One of those responsibilities is to review your co-worker's work for quality. The best practice is to have a checklist, which allows for process improvement. For example, if you miss a defect that is not on your checklist, you make a process improvement and add it to your checklist.

## Instruction

Create a checklist for a tester with items that each of the methodology phases. Each phase should have 4-6 items.

The following phases are required:

* Concept of Operations
  + Concept, Objectives, Risks
* Requirements
  + System, software, Architecture, Validation
* Design
  + Analysis, Program Design, Prototyping, Product Validation
* Implementation
  + Coding, Build
* Testing
  + Unit, Simulations, Integration, System and Acceptance Testing
* Deployment
  + Version-Release

Make sure you include an item from each of the sub-phases.

## Make it your Own

The completion of the core of this assignment is 89%. The assignment needs additional personalized work to achieve 100%.

Software Maintenance

You will be working most of your career in Software Maintenance. Software Maintenance mode can be in any phase of the development cycle since it is considered any code modification for perfection, performance, improvement, and clarity. We have not specifically covered Software Maintenance but from the information you got from the methodology phases, extrapolate what Testing might be like in Software Maintenance.   
Create a checklist for a project that is in the Software Maintenance phase.

## Submission

Make sure that you upload a copy of your document with checks to iLearn.

## Rubric

Use the following rubric to help understand the expectation.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Exceptional 100%** | **Good 90%** | **Acceptable 70%** | **Developing 50%** | **Missing 0%** |
| **Checklist 80%** | Make it your own | Answered all the questions in detail. | It contains all three elements and nine quality characteristics | Has two or more major issues | No answers to questions |
| **Professionalism 10%** | Make it your own | Properly cited, there are no grammar or spelling errors, and the writing style is "professional." | Found an instance of a spelling error, grammar error, incomplete citation, overly verbose wording, poor formatting, or poor writing. | A citation is missing where one is needed (plagiarism alert!). | Gross spelling/grammar errors or other aspects of the writing that make the paper difficult to read. |
| **Citations 10%** | One of the citations is a primary source | Contains 3-4 additional citations. | Contains 1-2 more citations other than the reading | Contains citations from the reading | No Citations |

* The Distribution of points starts at 50 points for minimal expected participation with additional points as difficulty increases.