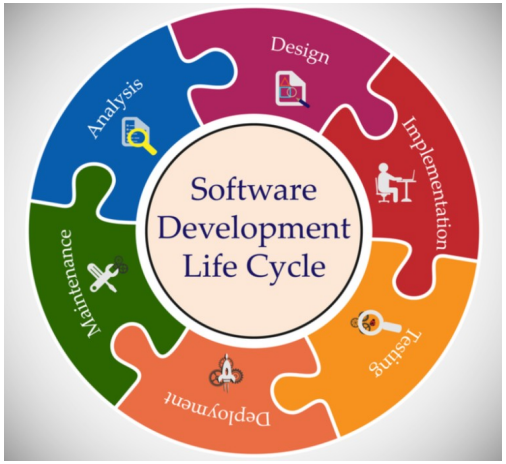
Software Development Life Cycle (SDLC) Waterfall QA Methodology

INFO6255 Software Quality Control & Management

Fall 2020

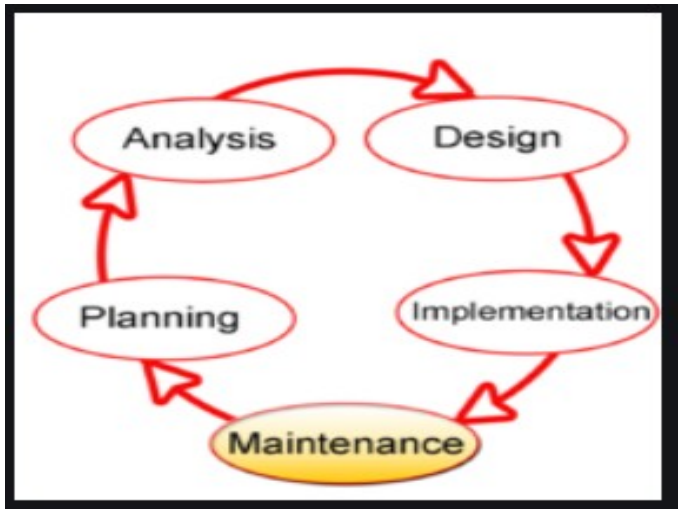
Medi Servattalab

M.Servattalab@northeastern.edu

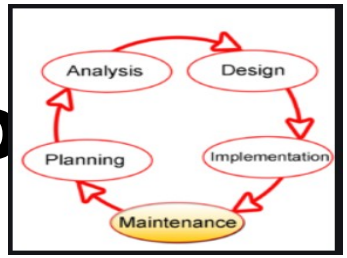


What is SDLC?

- The **systems development life cycle** (**SDLC**), also referred to as the **Software development life-cycle**, is a term used in information systems and software engineering to describe a **process for planning, creating, testing, and deploying an information system.**



6 Basic SDLC Methodolo



1. **Waterfall:** It is widely considered the **oldest** of the structured SDLC methodologies. It's also a very straightforward approach: **finish one phase, then move on to the next**. No going back.

   - Each **stage relies on information from the previous stage** and has its own project plan.

2. **Agile:** The Agile model has been around for **about a decade**. But lately, it has become a major driving force behind software development in many organizations.

   - Some businesses value the Agile methodology so much that they are now applying it to other types of projects, including **non-tech initiatives**

6 Basic SDLC Methodologies (Continued)

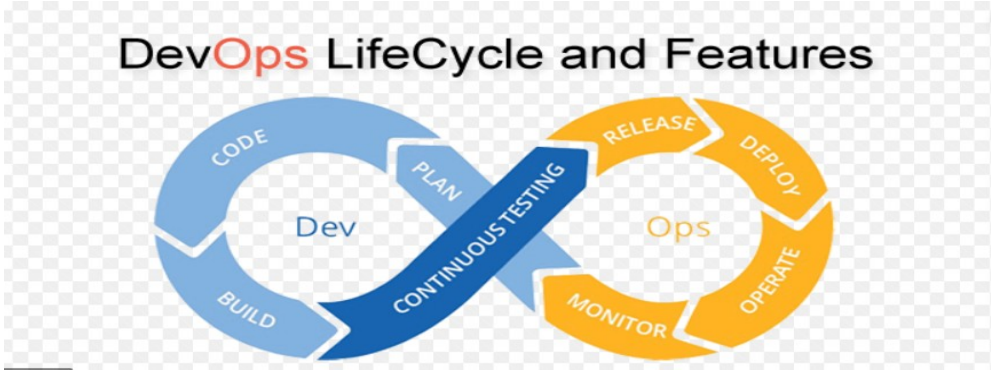
3. **DevOps:** The DevOps methodology is the **newcomer** to the SDLC scene. It emerged from two trends:

   - The **application of Agile and Lean practices to operations work**

   - The **general shift in business toward seeing the value** of collaboration between development and operations staff at all stages of the SDLC process

More on the DevOps Model…

- The high pace of code builds per day delivering stability and reliability.
- **The DevOps model is to:**
    1. Development **writes** the code
    2. Development **Automatically deploys** the code into the Automated Test environment
    3. Test **team execute Automated** Tests
    4. **Deploys into the production** environment



Top 6 Basic SDLC Methodologies (Continued)

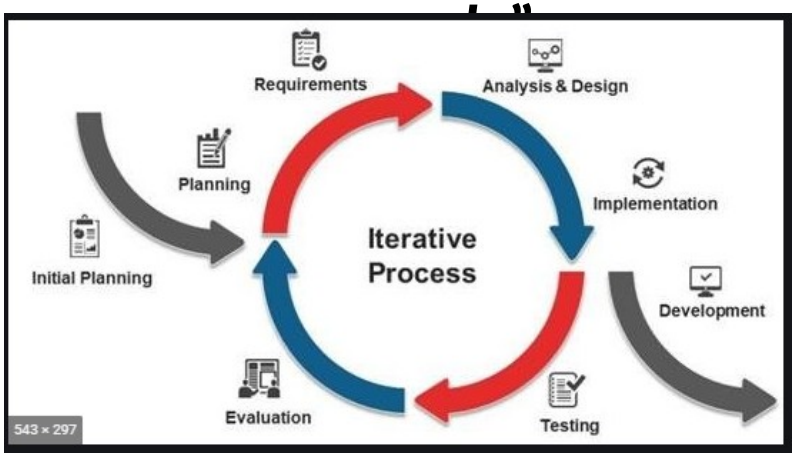
4. **Lean:** The Lean model for software development is inspired by **lean manufacturing practices and principles**.

- The seven Lean principles are:
  - **Eliminate waste**
  - **Amplify learning**
  - **Decide as late possible**
  - **Deliver as fast as possible**
  - **Empower the team**
  - **Build integrity in**
  - **and see the whole**

Top 6 Basic SDLC Methodologies (Continued)

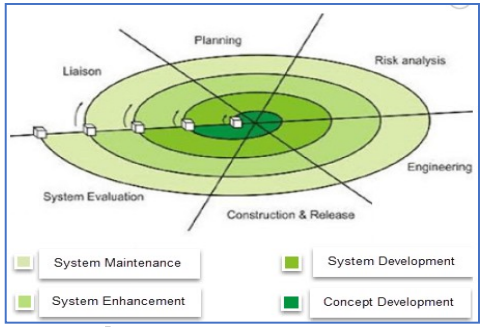
5. **Iterative:** The Iterative model is repetition incarnate. Instead of starting with fully known requirements, project teams implement a set of software requirements, **then test, evaluate and pinpoint further requirements**.

"***A new version of the software is produced with each phase, or iteration. Rinse and repeat until the complete system is***



Top 6 Basic SDLC Methodologies (Continued)

6. **Spiral:** One of the most flexible SDLC methodologies, **the Spiral model takes a cue from the Iterative model and its repetition;** the project passes through four phases:
   - Planning
   - Risk Analysis
   - Engineering
   - Evaluation



over and over in a "spiral" until completed, allowing for multiple rounds of refinement.

- **Spiral Model is a combination of a waterfall model and iterative model.**

https://www.guru99.com/what-is-spiral-model-when-to-use-advantages-disadvantages.html

The Definition of the Business Requirements

- **Business requirements** describe the **characteristics of the proposed system** from the viewpoint of end user like a Concept of Operations.

- **Business Requirements** define what the system should do from the Business (end user) perspective.

- For Example:

  As a Product Owner, I would like to let our customers know about our promotions once they purchase any of our B2B Services.

From Wikipedia…

The Definition of the Functional Requirements

- A **functional requirement**, in software and systems engineering, **is a declaration of the intended function of a system and its components**.

- Based on **functional requirements, an engineer determines the behavior** (output) that a device or software is expected to exhibit in the case of a certain **input**.

Techopedia

More on Functional Requirements…

- Functional Requirements **describe how the features** of a product must behave.

- A **Developer can implement the functional requirements** to enable the user to accomplish his/her tasks.

- Example:

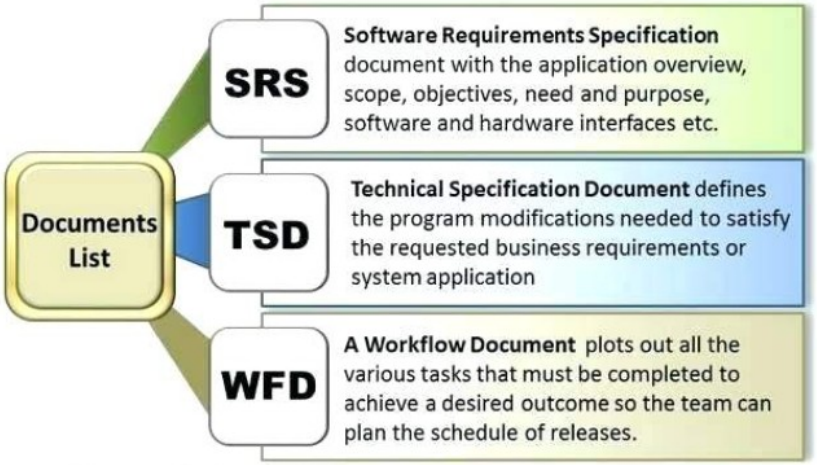
After the user confirms the purchase of the eService, prompt the user with a list of top 10 of our products in an alphabetical order.

From Wikipedia

Technical Specification Document (TSD)

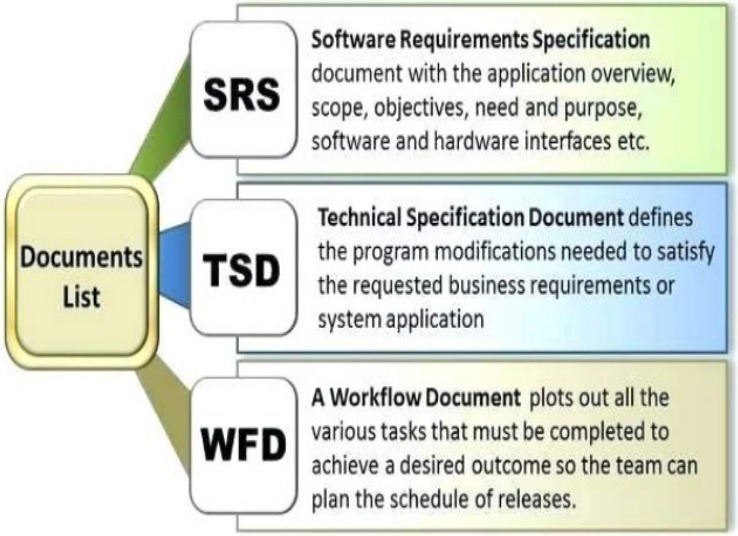
- A **technical specification document** defines the requirements for a project, product, or system.

- A **specification** is the information on **technical** design, development, and procedures related to the requirements it outlines.

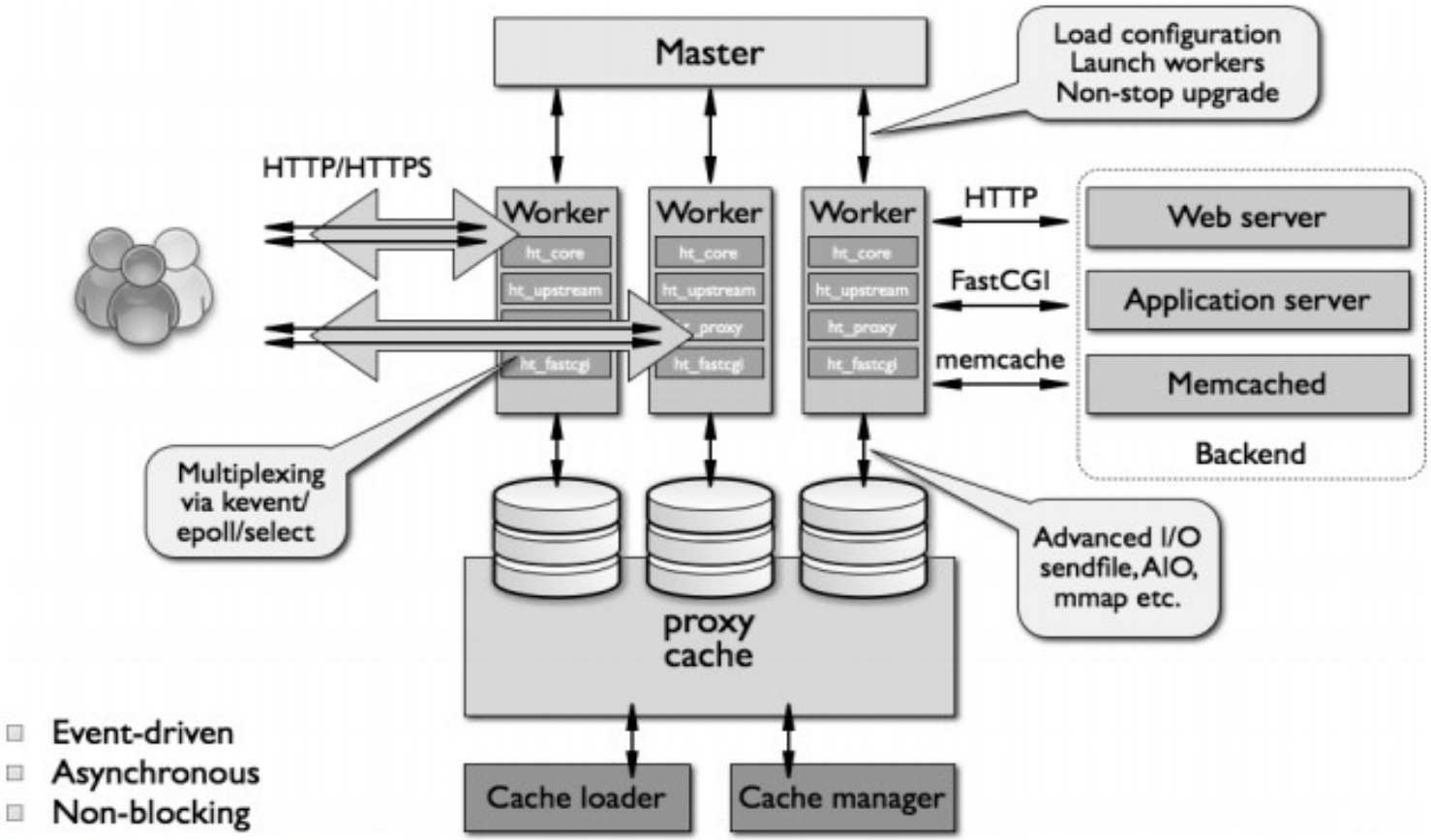


Technical Specification Document (TSD)

- It describes the
  - System Architecture
  - System Design Considerations Inputs/Outputs
  - Database Design
  - APIs
  - Communications Protocols
  - Security Architecture
  - Data Formatting
  - Choice of the Technology
  - and etc.



Example of System Architecture Diagram

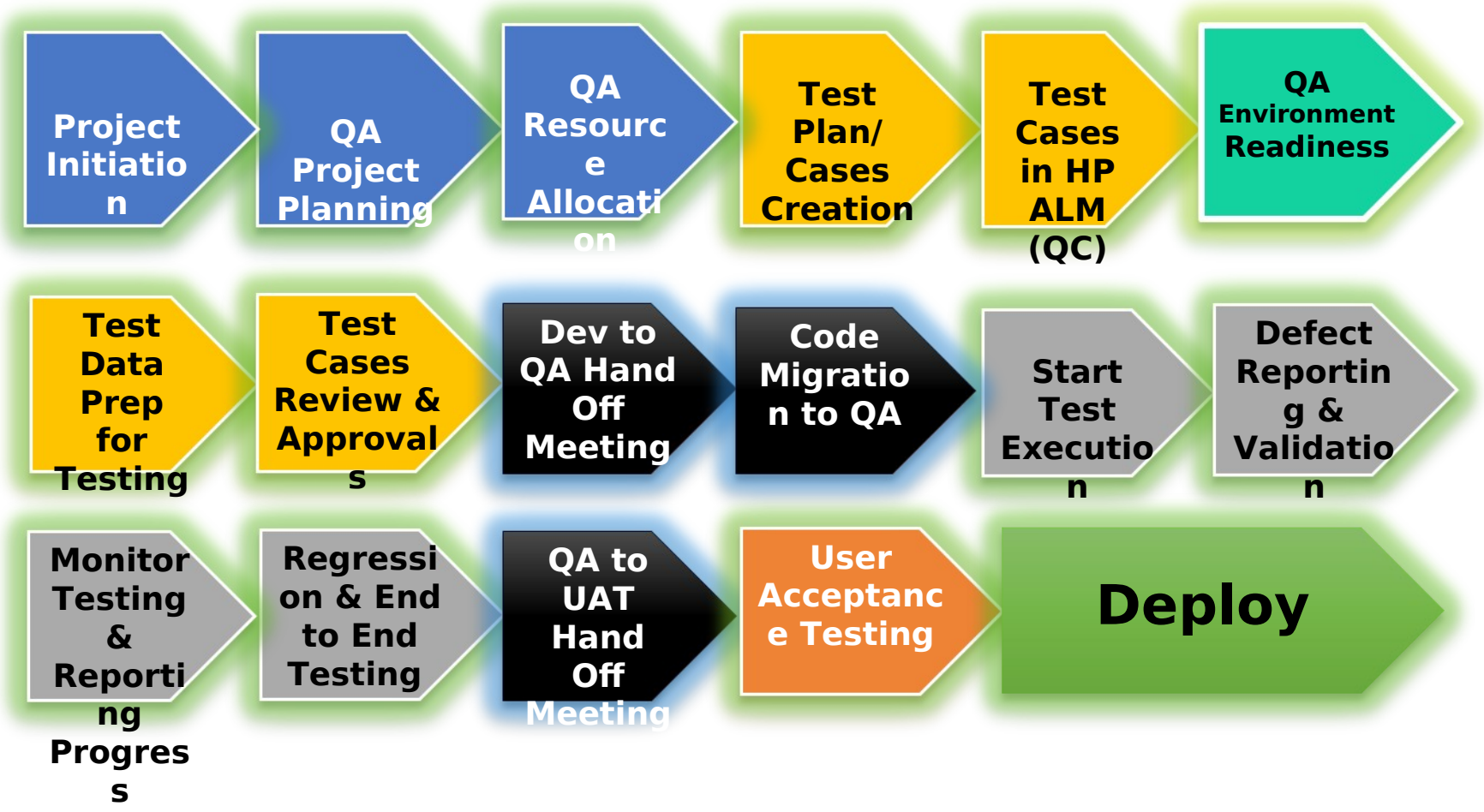


- Event-driven
- Asynchronous
- Non-blocking

Figure 4: Hardware architecture

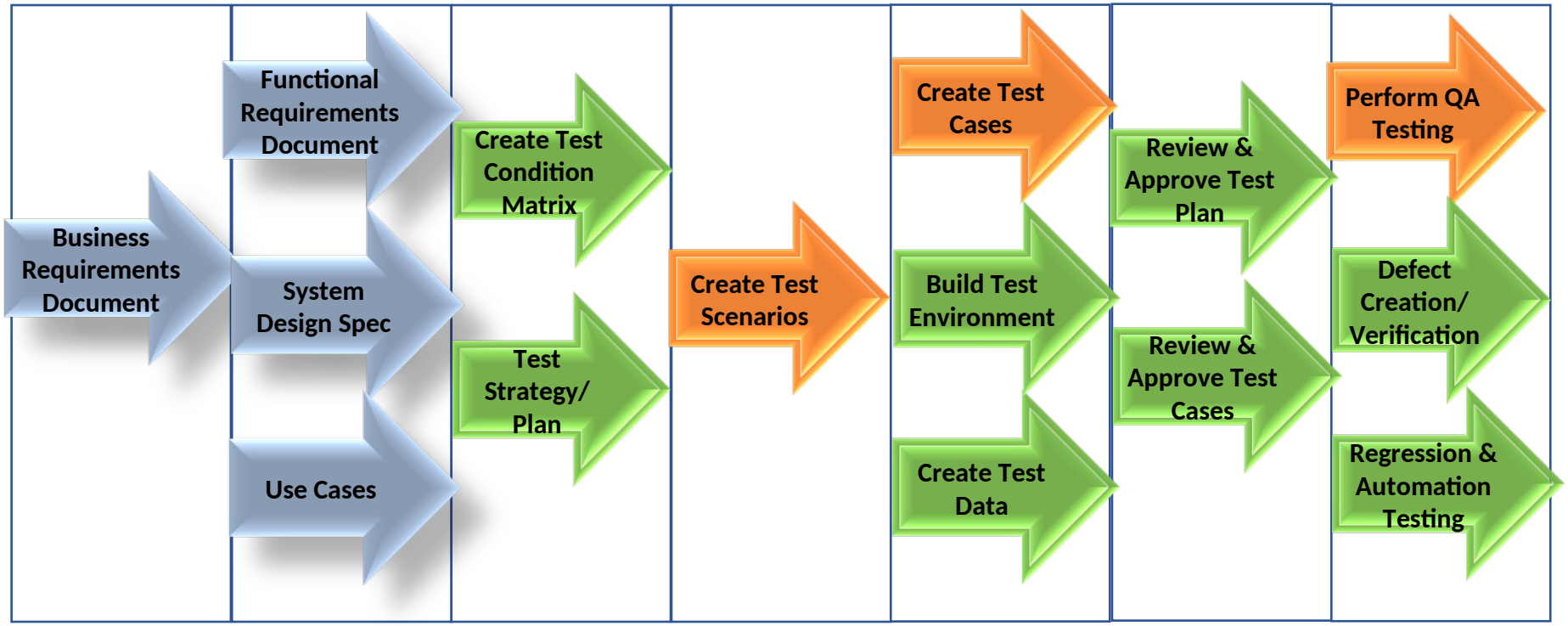
Table 4: Systems descriptions

A Simplistic QA Workflow (Waterfall)



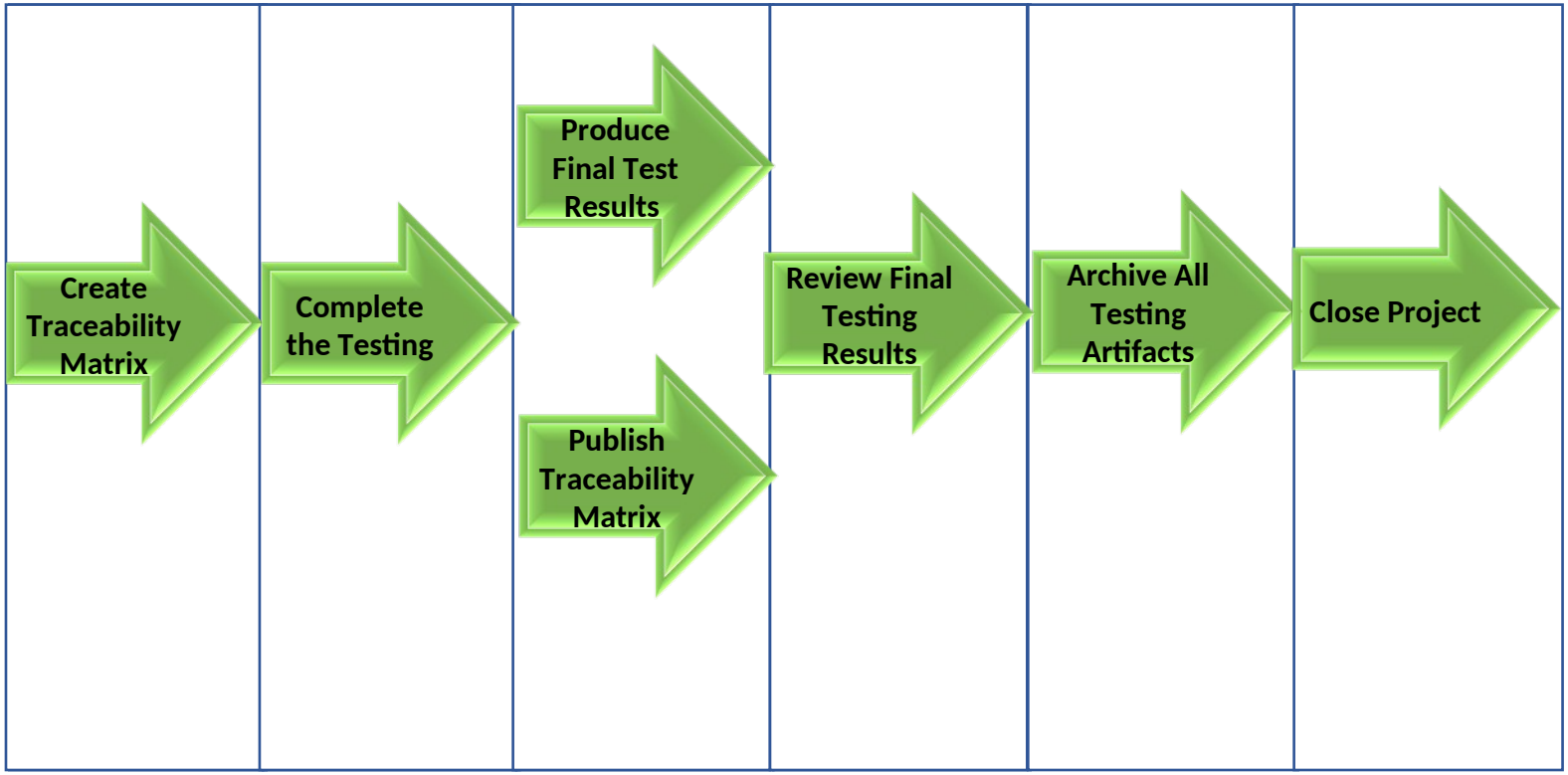
QA Workflow

QA Testing during different phases



Test Creation/Project Closure

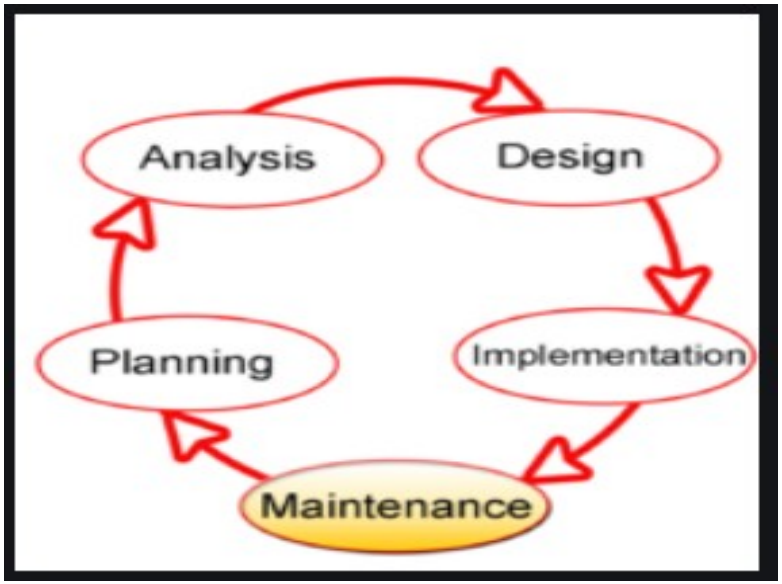
QA Testing during different phases…



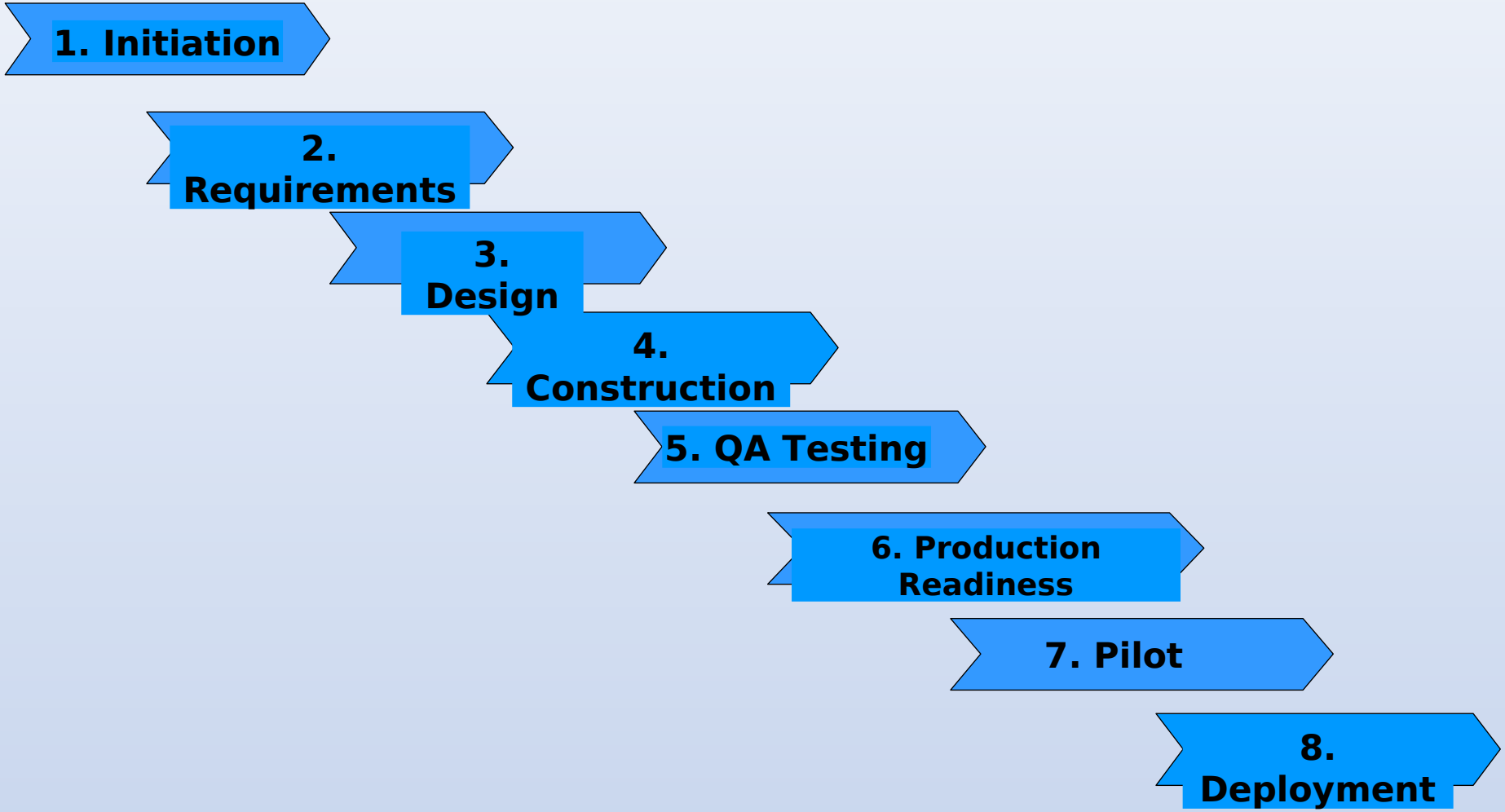
Waterfall Methodology Stages

Sample Phases:

1. Initiation
2. Requirements gathering
3. Design
4. Implementation (Construction)
5. QA Testing
6. Production Readiness
7. Pilot
8. Deployment
9. Maintenance



Software Development Life Cycle at a Glance



SDLC– “Waterfall” Methodology

1. Initiation	2. Requirements	3. Design	4. Construction
Project / Application Track Activities			
1. Project Charter & Scope (D) 2. Identify Project Sponsor (A) 3. Set Project Goals (D) 4. Define Scope (D) 5. Fund Project (A)	1. Business Requirements Document (BRD) (D) 2. Identify Project Team (A) 3. Decide on High Level Architecture (A) 4. Estimate Project (A) 5. Deliver Project Plan (D)	1. Functional Spec Document (FSD) (D) 2. Deliver Initial Architecture Document (D) 3. Prototype (A) 4. Refine Estimates (A) 5. Environment Specification (A)	1. Technical Spec Document (TSD) (D) 2. Develop & Unit Test Components (A) 3. Plan & Execute Code Reviews (A) 4. Refine Estimates (A) 5. SIT Testing (A) 6. Release Documents (D)
QA Testing Track Activities			
1. No QA Activities 2. QA Information Gathering	1. Draft QA Entry/Exit Criteria (D) 2. Draft Test Plan/ Test Strategy (D)	1. Draft QA Entry/Exit Criteria (D) 2. Draft Test Plan (D)	1. Set up Defect Tracking (HP QC) (A) 2. Review QA Entry Criteria (A) 3. Develop QA Test Cases (D) 4. QA Rqmts/Code Freeze (A) 5. Set up QA Environment (A) 6. Obtain Signoffs for the QA Test Cases (A)

A = Activity, D=Deliverable

SDLC– “Waterfall” Methodology

5. QA Testing	6. Production Readiness	7. Pilot	8. Deployment
Project/Application Track Activities			
1. Perform Testing (A) 2. Triage, Fix, Regress Bugs (A) 3. Develop Cutover Plan (A) 4. Develop Pilot Plan (A) 5. Review QA Exit Criteria (A)	1. Develop Cutover Checklist (A) 2. Create Install Plan (A) 3. Design Deployment Strategy 4. Determine Pilot facilities (A)	1. Evaluate Application (A) 2. Evaluate Training Materials (A) 3. Evaluate End-User Training (A) 4. Evaluate Process Changes (A)	1. Deploy Application (A) 2. Deploy Training Materials (A) 3. End-User Training (A) 4. Establish Feedback Loop (A)
QA Testing Track Activities			
1. Execute - Functional Testing (A) - QA Interface Testing (A) - End-End Testing (A) - Performance Testing (A) - UAT Testing (A) 1. Document QA Test Results (D, A) • Log/Verify Defects in the Test Management Tool	1. Develop a Production Sanity Check	1. Provide QA Support 2. Perform a Production Sanity Check	1. Provide QA Support 2. Perform a Production Sanity Check

A = Activity, D=Deliverable

What is a Test Plan or a Test Strategy Document?

- **Test Plan is a dynamic document**. The success of a testing project depends upon a well-written Test Plan document that is current at all times.
- Test Plan is more or less like **a blueprint of how the testing activity is going** to take place in a project.
- The Test Plan is shared with the Business Analysts, Project Managers, Dev team and the other teams. This helps to enhance the level of transparency of the QA team's work to the external teams.
- It is documented by the **QA lead** based on the inputs from the QA team members.
- The **more detailed and comprehensive** the plan is, the **more successful** will be the testing activity.

https://www.softwaretestinghelp.com/how-to-write-test-plan-document-software-testing-training-day3/

The Test Plan Sections

- Test Plan/ Test Strategy is a document that outlines the
  - Scope of testing
  - Testing Schedules
  - Roles and Responsibilities
  - Testing Phases
  - Features to be Tested (in scope
  - Features not to be tested (out
  - QA Entrance Criteria
  - QA Exist Criteria
  - Dependencies
  - Risks
  - Defect Severities/Processes

Developed by the Lead QA, reviewed by all members on the project team. Approved by the managers.

Testing Phases

- **Unit Testing**
  - Testing performed at the module level in the Development environment by the developers
- **Systems Integration Testing (SIT)**
  - Interface testing performed in the Development environment by the Business Systems Analysts and the developers
- **QA Functional Testing**
  - QA testing of the application functionality
- **QA Interface Testing**
  - Testing of the interfaces using the applications as the end points
- **QA End to End Testing**
  - Complete functional and Interface testing using business processes
- **User Acceptance Testing (UAT)**
  - User Acceptance Testing using business processes (performed by the UAT team)

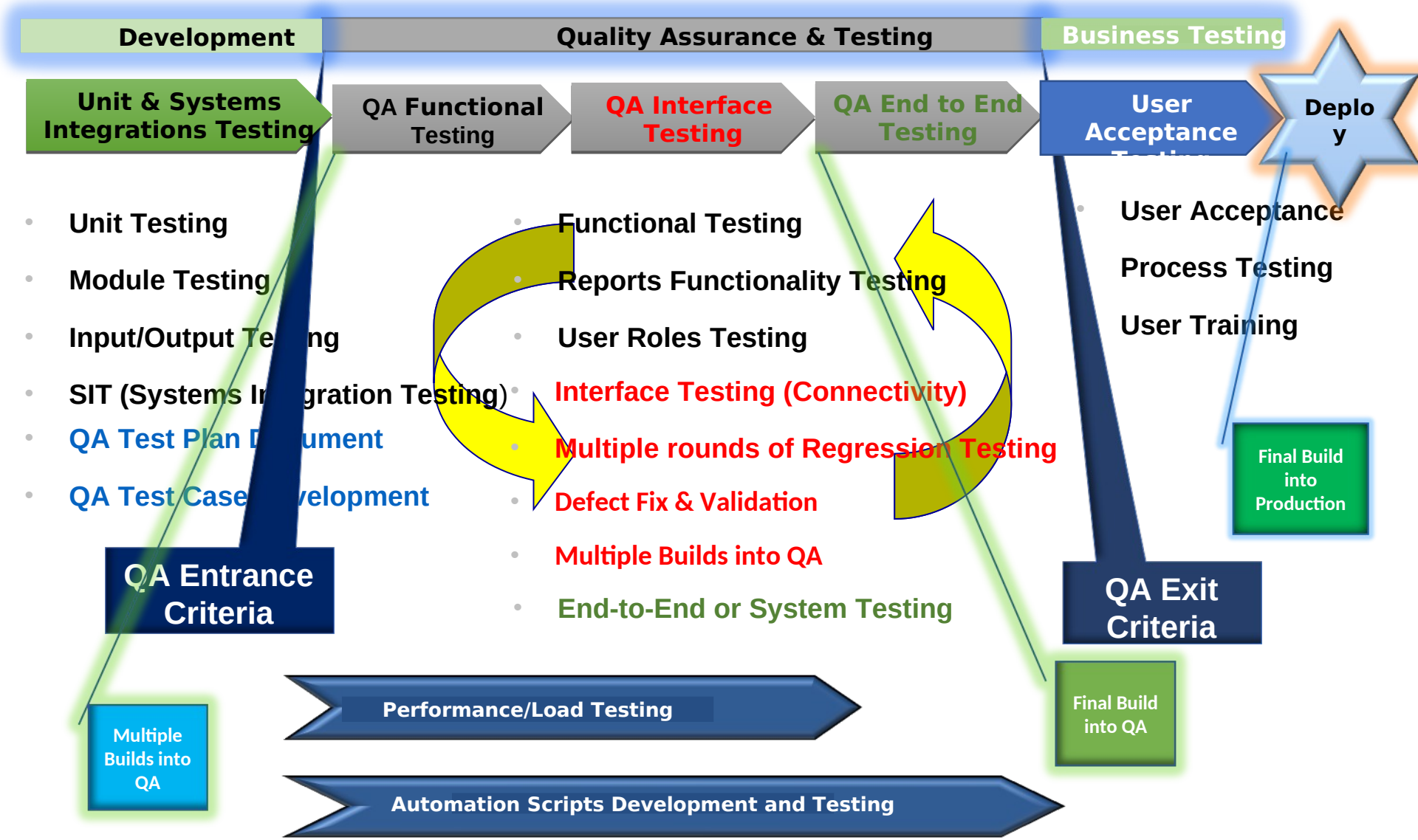
More Testing Related Definitions…

- **Smoke Testing** is a type of software **testing** that comprises of a non-exhaustive set of **tests** that aim at ensuring that the most important functions work. A.K.A "**Build Verification Testing**".
  - *The term '**smoke testing**' is from a similar type of hardware **testing**, in which the device passes the **test** if it **does** not catch fire (or smoke) the first time it is turned on.*

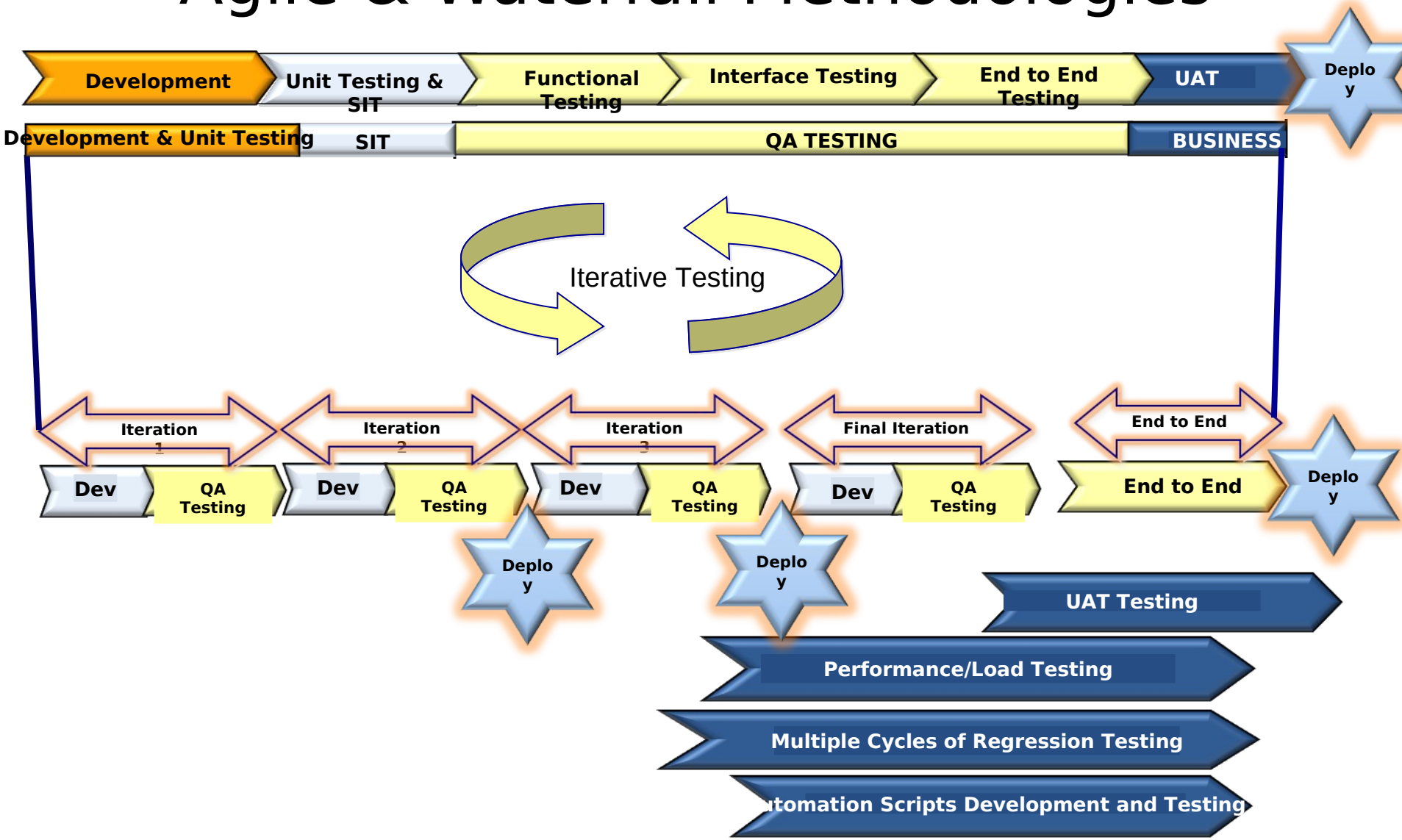
- **REGRESSION TESING** is to ensures any changes made to a **Build** did not negatively impact any of the functionality of the site or an application.
  - It is a much **deeper level testing,** and it is usually primed for <u>test automation</u>.

- **A Build** is a versioned release of a software that has been built and is being delivered to QA for testing. All Builds have specific release numbers for identification purposes **(e.g., Release 5.1)**

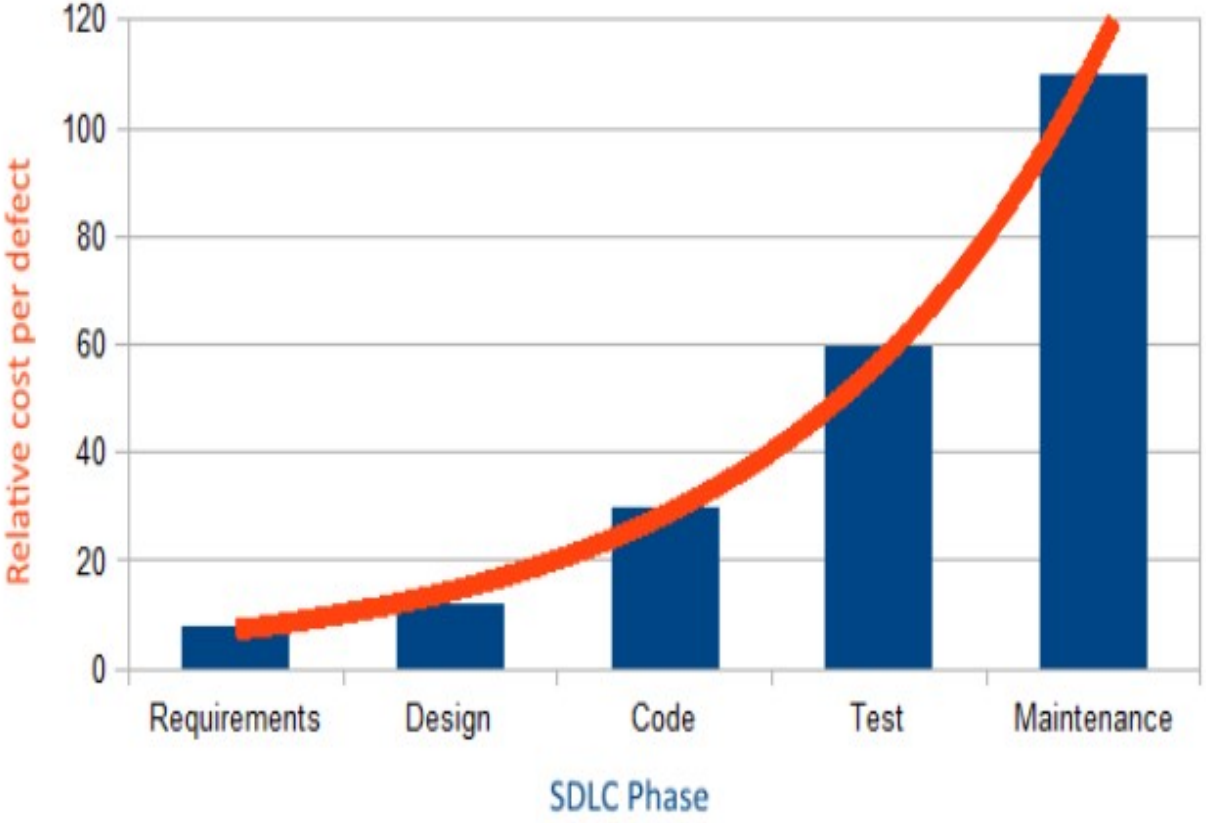
Waterfall Testing Methodology



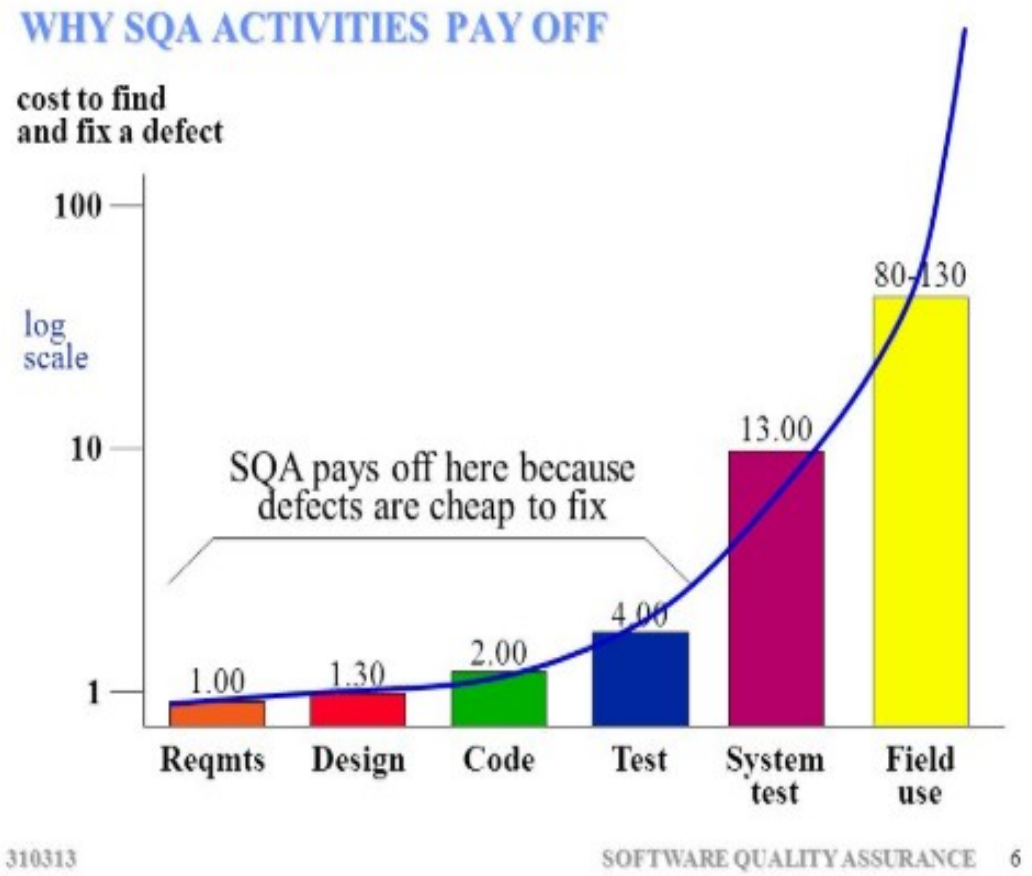
Agile & Waterfall Methodologies



Cost of Defects



Cost of Defects – Another View



Traceability Matrix

- **Purpose:**
  - The purpose of the Requirements **Traceability Matrix** is to ensure that all requirements defined for a system are tested in the test protocols.

- **Why is it Important?**
  1. To ensure that All of the requirements have been mapped to the test cases for a complete testing coverage.
  2. When requirements change midway through a project, a **traceability matrix** allows you to identify all of the impacted workflows, test cases, training materials, software code, etc.

- **What does it Map?:**

Business Requirements -> Functional Requirements -> Test Cases -> Defects

Traceability Matrix

REQUIREMENTS TRACEABILITY MATRIX					
Project Name: Online Flight Booking Application					
Business Requ			nts Document		Test se Document
Business Requirement ID	Business Use case	Functional Requirement ID#	Functional Requirement / Use Case	Priority	Test Case ID#
BR_1	Reservation Module	FR_1	One Way Ticket booking	High	TC#001 TC#002
		FR_2	Round Way Ticket		TC#003 TC#004
		FR_3	Multicity Ticket booking	Hi	TC#005 TC#006
BR_2	Payment Modu	FR_4	By Credit Card		TC#007 TC#008
		FR_5	By De		TC#009
		FR_6	By Re		TC#010 TC#011

Business Requirement ID #

Functional Requirement ID #

Test Case ID #

One Business Requirement to Many Functional Requirements Relationship

One Functional Requirement to Many Test Cases Relationship

RACI chart

- **RACI chart** shows the roles of the resources on a project. It is a Project Management task:
  - **R = Responsible -** Those **who do the work to complete** the task. There is at least one role with a participation type of responsible, although others can be delegated to assist in the work required.
  - **A = Accountable -** The one ultimately **accountable for the correct and thorough completion** of the deliverable or task. An accountable must sign off (approve) work that responsible provides.
  - **C = Consulted -** Those whose opinions are **sought, typically subject matter experts (SME)**; and with whom there is two-way communication.
  - **I = Informed -** Those who **are kept up-to-date on progress**, often only on completion of the task or deliverable; and with whom there is just one-way communication.

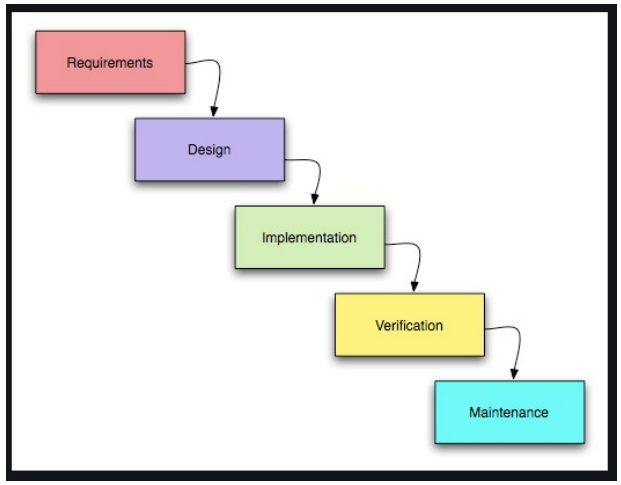
Responsibility Assignment Matrix - RACI Chart

	Jeff	Michael	Reto	YOU	Alex	Anna	Bill	Cindy	Felix	Fred	Hans	John	Livio	Luc	Marco	Paul	Peter	Sue	Ted	Tim
Planning / Schedule	R	A	I	C					C											Q
Risk Management		I	I	Q						A								R		
Quality Management			R	C						R										A
Procurement				R		Q				R								R		A
1. Specifications Listing								A		R								R		R
2. Site Requirements		C	A	R	Q						R									
3. Call for Tenders				Q	A	R	C				R							R		
4. Budget Approval				A	Q					R							R			R
5. Contract Negotiations			A		Q	R	R											R		

* R – Responsible (works on), A – Accountable, C – Consulted, I – Informed, Q – Quality Reviewer

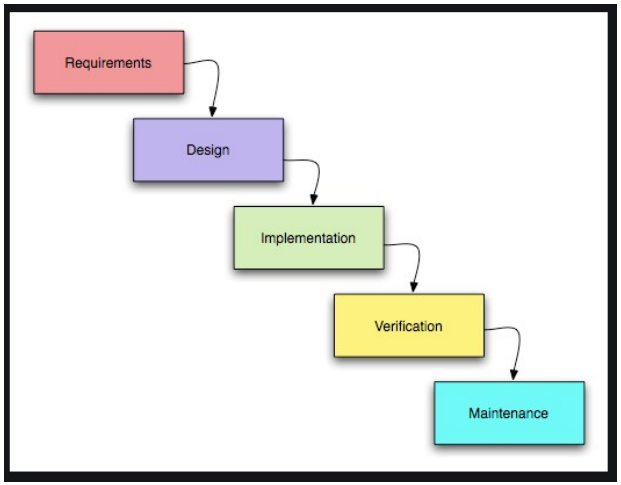
Pros and Cons of Waterfall Methodology

- **Pros:**
  - Everyone **understands the objectives** through the technical documentation.
  - **Timelines are more achievable** due to the "phased" development.
  - **Costs can be estimated** more accurately once the requirements have been defined.
  - The **testing is easier** due to the creation of the various documentation.
  - The **outcome** is very **clear**.
  - The documentation **drives the testing**, and the planning can be done more accurately.



Pros and Cons of Waterfall Methodology

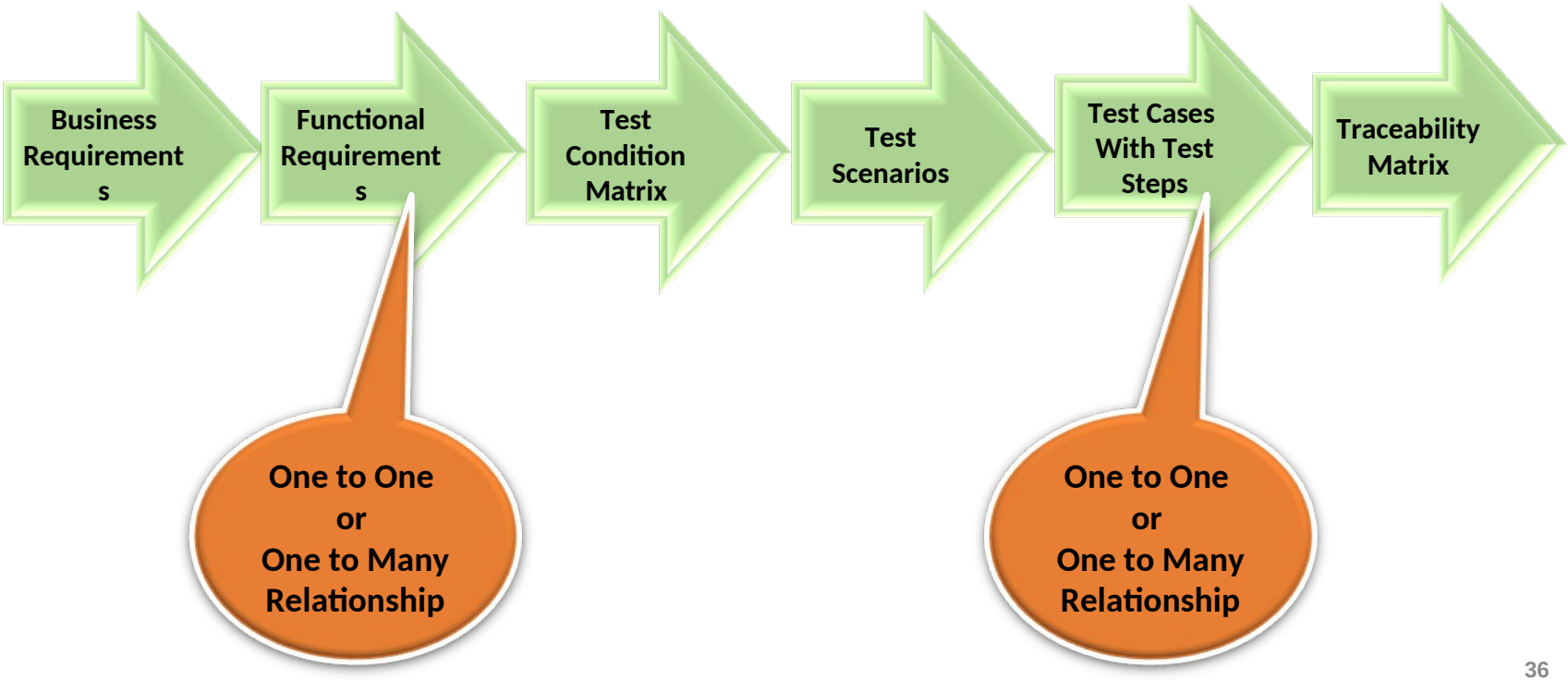
- **Cons:**
  - **No flexibility**. Everything is locked down.
  - Difficult to make changes to the requirements.
  - **Longer delivery** time.
  - **Dependency** of each phase on another phase.
  - The user's **involvement is very minimal** in the process.
  - The **users' needs** may have changed since the beginning of the project.



https://www.dcslsoftware.com/pros-cons-waterfall-software-development/

A Summary of the Testing Artifacts

To create Test Cases for a project a QA tester needs the Functional Requirements document



What is a Test Management Tool?

- **Test Management Tools** are used to **store information on how testing is to be done**, plan testing activities and **report the status of quality assurance activities**.
- The tools have different approaches to testing and thus have different sets of features.
- A Test Management tool for testing is **Mandatory**.

Two Examples:

- **Quality Center – ALM**
- **Jira**
- **TestRail**

Test Management T

- **Why Needed?**
  - To create and store the **Business and Functional Requirements.**
  - To create and store the **test cases** based on the requirements.
  - Allows mapping of the **Requirements to Test Cases** (Traceability Matrix).
  - Allows the **execution** of the manual or automated test cases.
  - Allows the **storage** of the test results/test snapshots.
  - **Create reports** on the testing progress.
  - **Management** of the Defect Workflow.
  - **To store** the Testing Artifacts (Actual Results, Screen Shots, Defects and etc.)

Quality Tools

HP ALM (Formerly QC)

- Ability to create re-usable test cases and execute manual or automated tests
- Ability to manage defects and trace defects to tasks, tests and requirements
- Sprinter to run the test cases more efficiently

Micro Focus Application Life Cycle Management

Converting to TestRail

HP Performance Center (Load Runner)

- Allows integration with ALM (QC)
- Allows flexibility with 2 controllers running with 1800 Vusers
- Allows centralization of the load testing
- Currently using Performance Center for a number of projects in the IT
- 1800 Web Virtual User Licenses
- 500 Flash Virtual User Licenses

Micro Focus Unified Functional Testing

Micro Focus Performance Center

(HP PC 11.5)

HP UFT (Formerly QTP)

- Automate testing of multi-layer test scenarios, including GUI and API testing
- Easy conversion of manual tests to automated tests
- Framework definition for better test management with tight integration to HP Business Process Testing and HP Application Lifecycle Management

What is a defect?

- A **defect** is an **error in coding or logic** that causes a program to malfunction or to produce incorrect/unexpected results.
- Defects are found during different **cycles of testing**, especially in the QA cycle.
- A Defect can be found in many different situations:
  - When the Actual Results deviate from the Expected Results of a test case.
  - When an error message in a program or a system is displayed on the screen.
  - When the performance of a function is not within the SLA (Service Level Agreement)
  - A **"crash"** during the execution of a function.
  - And more….

http://tryqa.com/what-is-defect-or-bugs-or-faults-in-software-testing/

Defect Life Cycle and the Defect Fields

- **Defect ID** – The unique identification number
- **Reported Date** – The Date when the bug is reported
- **Reported By** – The details of the tester who reported the bug like Name and ID
- **Status** – The Status of the defect like **New, Assigned, Open, Retest, Verification, Closed, Failed, Deferred, etc.**
- **Version Found In**– The product version of the application in which the defect is found.

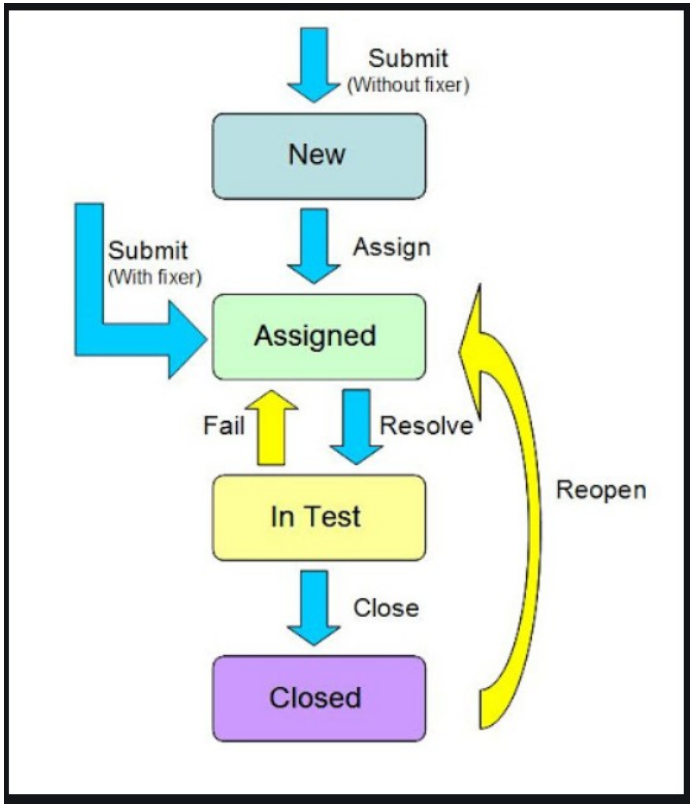
Defect Fields…

- **Defect Description** – The abstract of the issue. This includes the detailed steps of the issue with the screenshots attached so that developers can recreate it.
- **Fixed by** – The details of the developer who fixed it like Name and ID
- **Date Closed** – The Date when the bug is closed
- **Severity –** Shows the impact of the defect or bug in the software application. Critical, Major or Minor.
- **Priority –** The order of fixing the defect can be made. High, Medium and Low.

http://tryqa.com/what-is-defect-or-bugs-or-faults-in-software-testing/

Defect Severity vs. Defect Priority

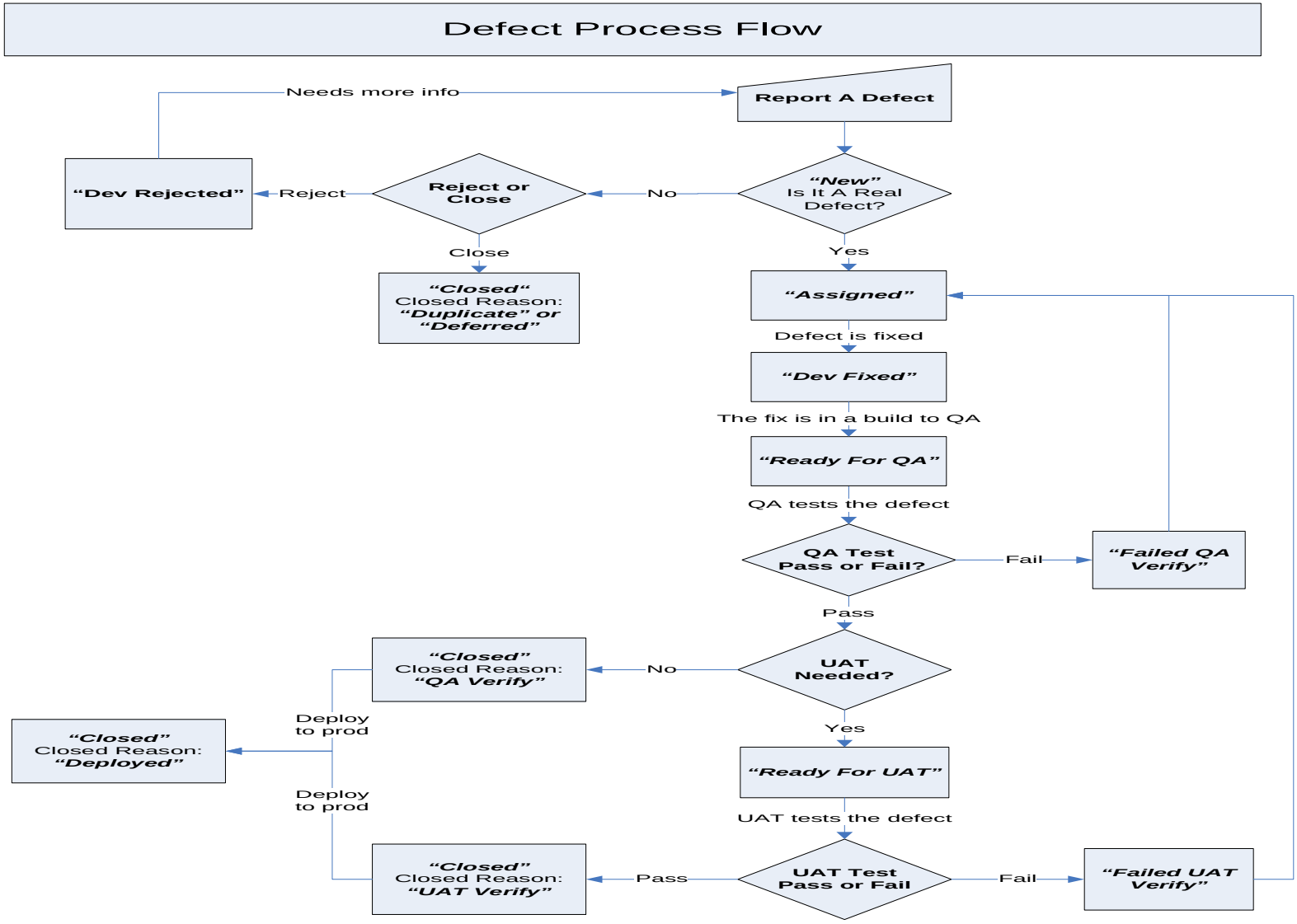
- **Defect Severity** is the degree of impact that a **defect** has on the system; whereas, **Bug Priority** is the order of **severity** which has impacted the system.
- The **QA Tester** may set up the Defect Impact/Severity:
  - Critical
  - Major
  - Minor
  - Low
- The **Product Manager** may set up the Defect Priority:
  - Severe
  - High
  - Medium
  - Low



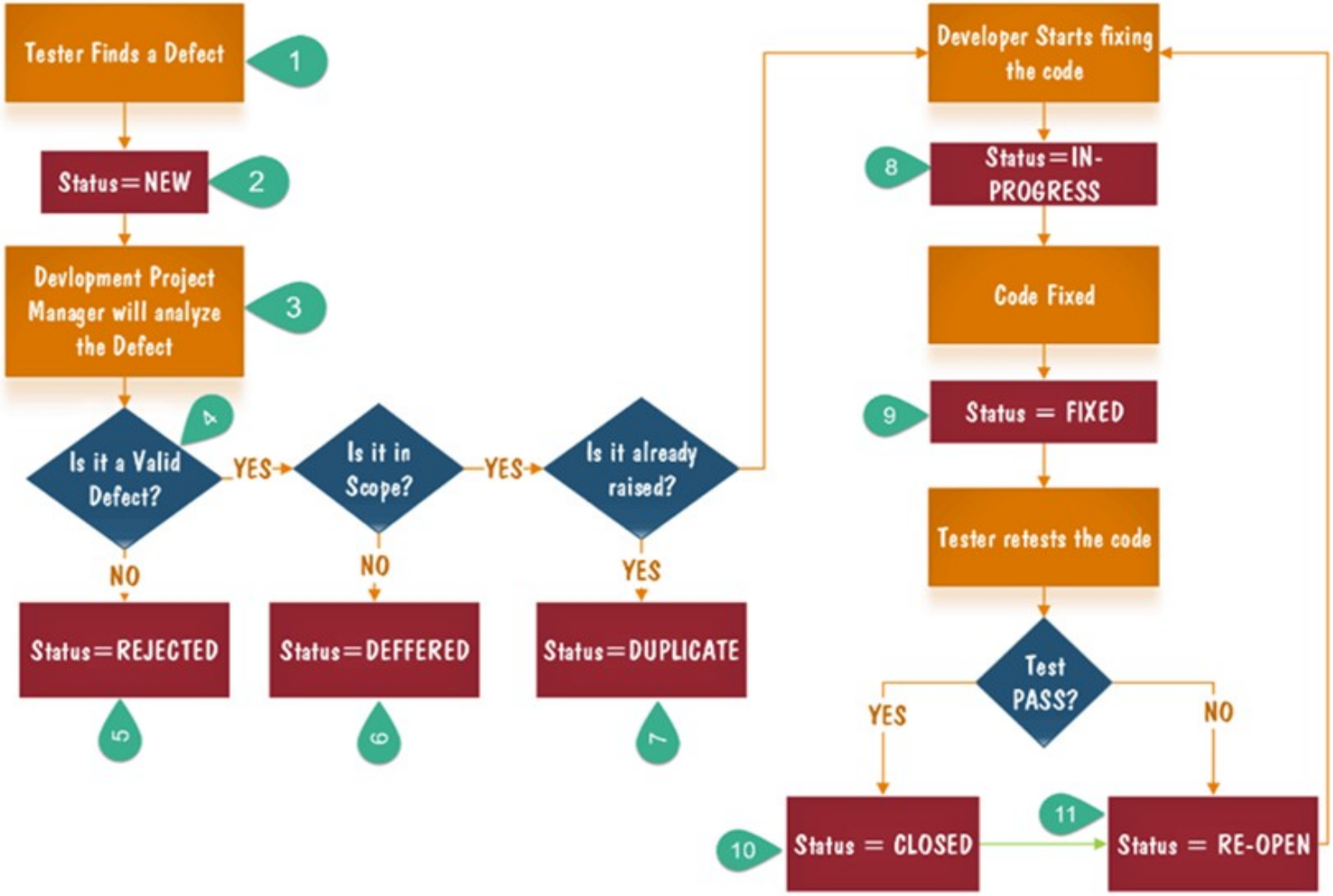
Defect Process & Statuses

Status	Description
New	This defect has been discovered and reported
Assigned	This defect has been **Assigned** to a development resource for investigation
Dev Fixed	A development resource has solved the issue but the defect is not yet associated with a build
Dev Rejected	The Developer has rejected the defect because either he/she did not understand the steps and requires more information, or was not able to reproduce the problem.
Ready For QA	The fix for this defect has been introduced into a build and is pending verification by QA tester.
Closed	This defect has been **Closed** by QA to remove it from process consideration (The Closed Reason must be filled out)
Closed Reason: **Deferred** **Documentation** **Duplicate** **Not a Bug** **QA Verified**	The reason for the defect closure. Mandatory field
Failed QA Verify	The fix for this defect has been tested but has **Failed Verification**--it is still a defect.

Defect Process

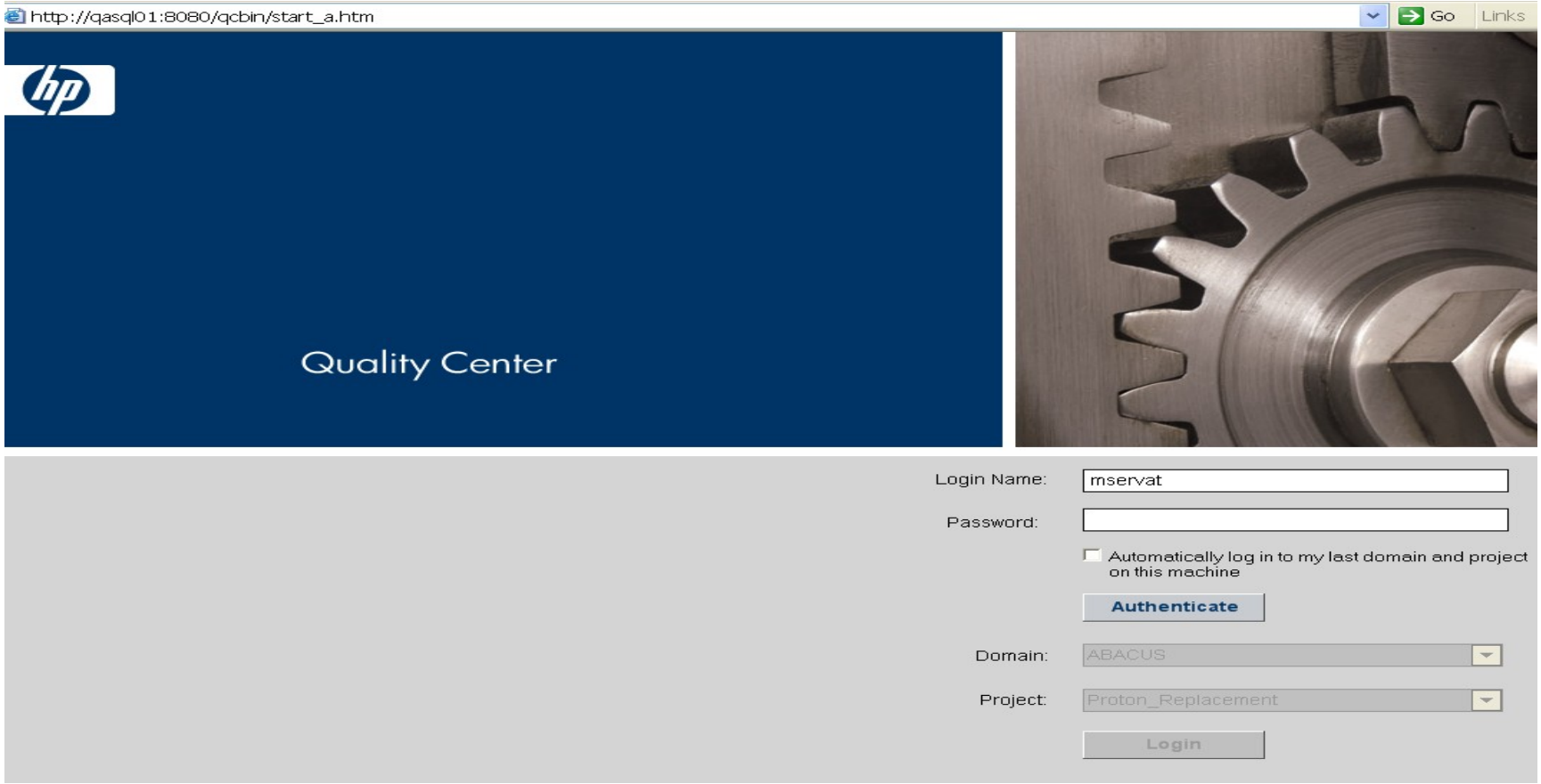


Another Example of a Defect Process



https://www.guru99.com/defect-life-cycle.html

Quality Center(10.0) - Defects



Quality Center – Defects

HP Software | < BACK | FORWARD > | TOOLS | HELP | LOGOUT

Quality Center

Domain: ABACUS, Project: Proton_Replacement, User: mservat

Releases
Requirements
Test Plan
Test Lab
Defects

Defects | Edit | View | Favorites | Analysis

New Defect... | Favorite : <None>

Bug	Detected on Dat	Target I	Bug Severity	Assign	Assigned Dev	Bug P	Bug Statu	Sub_Proje	Functionality	Summary
1	11/27/2007			pgubba	eminnucci	4. Low	Closed	Chairside :	Shift List - Patient	Shift list - Cancel of Shift list screen displays Machine Setup screer
2	11/27/2007			pgubba	bparsons	4. Low	Closed	Chairside :	Pre-Dialysis V/E	Update Pre-Dialysis Access - Select is displaying the resultant scre
3	11/28/2007			hsaini	lmarchetti	4. Low	Closed	Chairside :	Other	No Confirmation message on Log Out
4	11/28/2007			hsaini	jjohnson	4. Low	Closed	Chairside :	Machine Setup	Dialyzer Text don't wrapped properly
5	11/28/2007	6/18/20(		hsaini	rmeidal	4. Low	Closed	Chairside :	Machine Setup	SELECT SEARCH - Dialyzer, Machine Type, Arterial and Venous S
6	11/28/2007			pgubba	lmarchetti	4. Low	Closed	Chairside :	Other	Logout - As per design spec , logout button should close the applic
7	11/28/2007			hsaini	jjohnson	3. Med	Closed	Chairside :	Other	Search Add Non Scheduled Screen: ChairSide crashed
8	11/28/2007			pgubba	jjohnson	4. Low	Closed	Chairside :	Machine Setup	Edit machine settings - Both Performed by / Reviewed by after sigr
9	11/28/2007			pgubba	ali	3. Med	Closed	Chairside :	Pre-Dialysis V/E	Update Pre-Dialysis Evaluation - NO-button is prefilled or selected b
10	11/28/2007			pgubba	ali	2. High	Closed	Chairside :	Pre-Dialysis V/E	Update Pre-Dialysis Evaluation - On first selection Check of 'Hospita
11	11/28/2007			pgubba	jjohnson	2. High	Closed	Chairside :	Pre-Dialysis V/E	Update Pre-Dialysis Evaluation - Check of 'Hospitalizationt' is gettii
12	11/29/2007			hsaini	ali	3. Med	Closed	Chairside :	Other	Switch Patient: No Show button changes the functional flow of applii
13	11/29/2007			pgubba	jjohnson	4. Low	Closed	Chairside :	Lab Results	Lab Results - Mismatch is observed between Screen Design Spec 2
14	11/29/2007			pgubba	lmarchetti	4. Low	Closed	Chairside :	Other	PATIENT TREATMENT MENU after screen shot has to be correctec
15	11/29/2007			hsaini	ali	3. Med	Closed	Chairside :	Other	Select Patient: Selecting left and Right Patients distrub the Tabs--n
16	11/29/2007			hsaini	rmeidal	4. Low	Closed	Chairside :	Other	UI: Selected Tab gets unhighlighted.
17	11/29/2007			hsaini	jjohnson	4. Low	Closed	Chairside :	Pre-Dialysis V/E	Update Pre-Dialysis vital Screen: Button has wrong label.
18	11/29/2007			hsaini	jjohnson	2. High	Closed	Chairside :	Other	UI:Start Treatment button appears without signing all four Screens.
19	11/29/2007			pgubba	lmarchetti	4. Low	Closed	Chairside :	Machine Setup	Machine Setup - The space between OLC-V, Dialyzer, Machine #, :
20	11/29/2007			hsaini	jjohnson	4. Low	Closed	Chairside :	Pre-Dialysis V/E	Update Pre-Dialysis Evaluation: Field has wrong label.
21	11/29/2007			pgubba	jjohnson	3. Med	Closed	Chairside :	Machine Setup	Edit Machine Settings - Field label OLC-V is missing
22	11/29/2007			hsaini	jjohnson	4. Low	Closed	Chairside :	Pre-Dialysis V/E	Update Pre-dialysis Vitals: Wrong field Label.
23	11/29/2007			pgubba	lmarchetti	4. Low	Closed	Chairside :	Machine Setup	Update Dialyzer--- > Mismatch field type is is observed for field "Rea
24	11/29/2007			hsaini	rmeidal	4. Low	Closed	Chairside :	Other	KEYBOARD - Wrong Keyname on Numeric Touch Screen keyboarc

Bug Description | Attachments * | History

* Summary: Shift list - Cancel of Shift list screen displays Machine Setup screen with no Header and tab section

* Bug Description:

mbaigSteps

1. Follow all the steps and select the paitent . Click switch patient and hit cancel .Noticed Machine Setup screen is displayed with no headers and tabs. Attached is the snap shot

Comments: | Add Comment

Bill Parsons <bparsons>, 11/29/2007: Cancel not working in 2.0.0.2

Jeffrey Johnson <jjohnson>, 3/18/2008: old.

Pathi Gubbala <pgubbala>, 3/20/2008: Verified

Quality Center - Test Lab

Execution | Test Sets | Tests | Search | Hosts | Analysis

Requirements
Test Plan
Test Lab
Defects
Dashboard

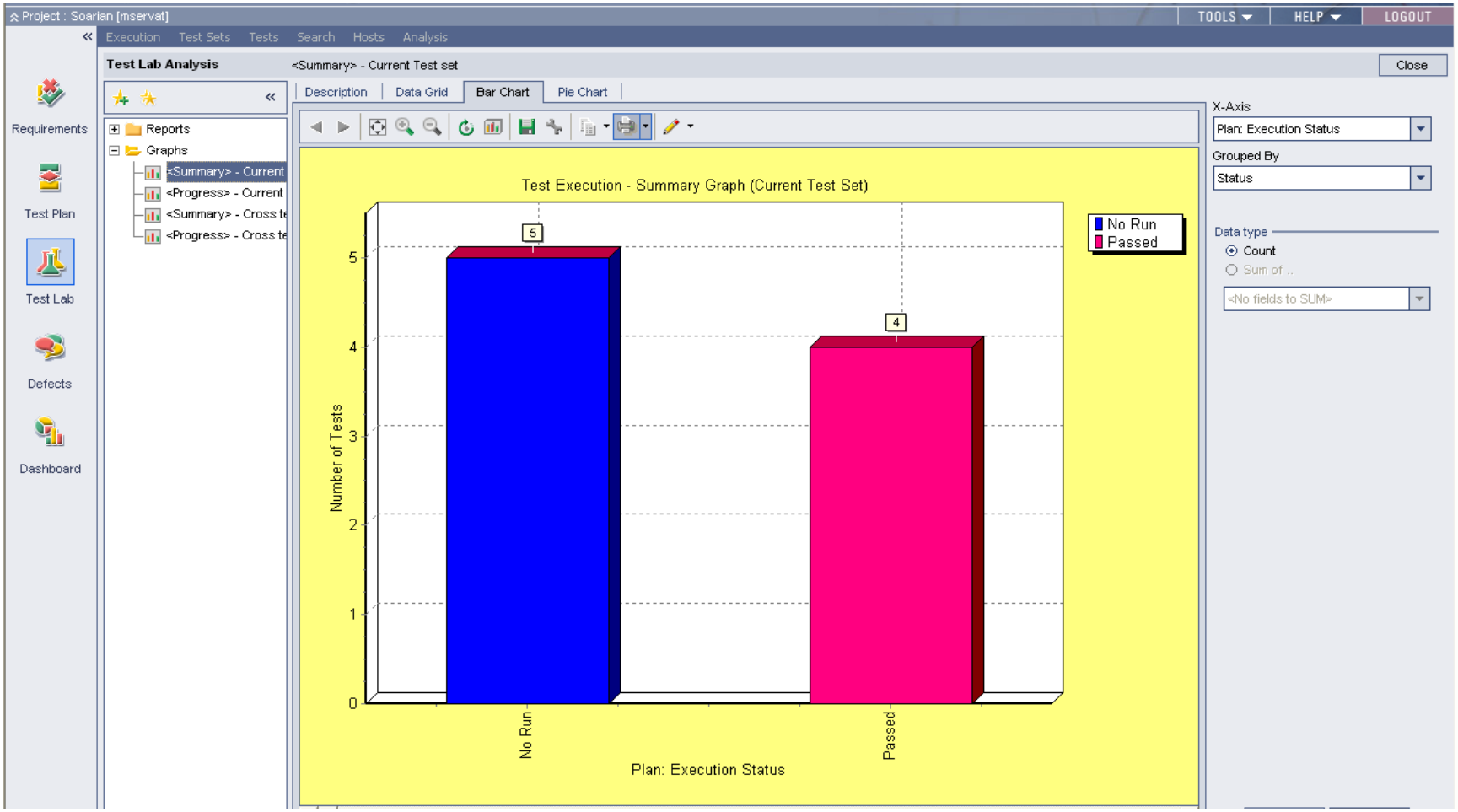
Root
- Unattached
- ADT - Pilot 1
- DI
- eCPL - Docs Recieved Re
- Lexmark Functionality Tes
  - Functionality
- Nightly Process Testing
- test

Execution Grid | Execution Flow | Test Set Properties

Select Tests | Run | Run Test Set

Plan: Test Name	Plan: Type	Status	Planned Host	Responsible	Exec Date	Time	Planned Exec	Planned Exec
[1]Test Case 1_Invalid Last Name	MANUAL	Passed			10/6/2006	3:11:30 PM		
[1]Test Case 2_Invalid Clinic #	MANUAL	Passed			10/6/2006	3:11:39 PM		
[1]Test Case 3_Patient related billing documents	MANUAL	No Run						
[1]Test Case 4_Scan PT - LOA doc	MANUAL	Passed			10/6/2006	3:11:49 PM		
[1]Test Case 5_Patient related clinical documents	MANUAL	No Run						
[1]Test Case 6_Non Patient documents Acutecon	MANUAL	Passed			10/6/2006	3:12:32 PM		
[1]Test Case 7_Non Patient documents Org Bankruptcies	MANUAL	No Run						
[1]Test Case 8_Non Patient documents Bulletins from Payers	MANUAL	No Run						
[1]Test Case 9_Non Patient documents Daily Cash Remits	MANUAL	No Run						

Quality Center – Reporting & Graphs



ALM Test Plan Execution Status Report

TestPlan - Report Manager - Windows Internet Explorer

http://localhost/Reports/Pages/Report.

TestPlan - Report Manager

Home > HP QC reports - www.rbreporting.com > TestPlan

Home | My Subscriptions | Help

Test Type: All

Execution Status: All

View Report

Show Levels: 1

Test Status: Ready

Search

Show Percentage: True False

1 of 1 100% Find | Next

<YOUR COMPANY NAME>

HP Quality Center Report

Generated with parameters: Test Status: Ready

Test Plan Execution Status

Test Area	Test Exec. Status			Total Tests
	Pass	Fail	Other	
Flight Application (BPT Flow Demo)	62%	12%	25%	8
Completed BPT Tutorial	100%			1
Flight Reservation	48%	17%	34%	29
Itinerary	75%		25%	4
Mercury Tours Site	25%		75%	12
Compiled Modules			100%	1
Profiling	62%	8%	29%	24
Total	52%	10%	38%	79

http://www.rbreporting.com/hp_quality_center_reports.htm

References

- https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm
- https://www.youtube.com/watch?v=Y_A0E1ToC_I
- https://www.smartsheet.com/content-center/best-practices/project-management/project-management-guide/waterfall-methodology
- https://www.softwaretestinghelp.com/defect-management-process/