INFO6255 - Software Quality Control and Management

QA Testing Fundamentals

Medi Servattalab

M.Servattalab@northeastern.edu

Fall 2020

About me!!

- **My Name** is Matthew Medi Servattalab (Medi Servat professionally)
- NU Alumni
- 30+ years of experience in IT (Development, Project Management and QA)
- Worked in a lot of **small and large** companies in Mass
- Currently live in Hopkinton, MA **(the start of the Boston Marathon!!!)**
- Work at **Fresenius Medical Care**, Lexington MA for 15 years.
    - **VP of Production Support Services**
- Interests: Tennis, Golf, Soccer and Running (Jogging!).

House Keeping Rules!!!!

- **Participate, Participate, Participate. Participate in the class….**
  - If you show up physically in this class, be here mentally as well.
  - Show me what you got!
  - Your chance to improve your written & verbal communications
- **Missing classes:** I rather not see any, however if you must take an absence let me know in advance. University Rules: 2X
- **Late shows:** Please be on time or you will miss the quiz.
- **Laptops:** I'd rather you close your laptops during the class, however you can use it if you must.
  - **DO NOT HIDE** behind your laptop.
  - **DO NOT DO** other class work during this class.
  - **DO NOT GET DISTRACTED: Now that we are remote.**
- **Phones/Watches:** Silenced
- **Quiz:** Most weeks there will be a quiz.
  - Don't miss any.
- **Groups: You all will be placed in teams.**

INFO6255 - Teaching Assistants:

- Two TAs can help you in this class:
  - **Aishwarya Ajay Mane**
    - eMail: mane.ai@northeastern.edu
  - **Rishi Jatania**
    - Email: jatania.r@northeastern.edu
- Slack channel for communication with the TAs:

https://join.slack.com/t/info6255softw-zsh5187/shared_invite/zt-h5r06rd1-QPnv5E96Cez~01iwvQ_uZw

What are you going to learn?

Week 1	**Introductions and the House Keeping rules**
Week 2	SDLC and the QA Life Cycle
Week 3	Anatomy of a Test Plan/Strategy document
Week 4	Analysis of Business & Functional Requirements/Test Condition Matrix/Traceability Matrix
Week 5	Agile Methodology; DevOps Model;
Week 6	Change and Release Management
Week 7	**Test Automation Project: MicroFocus UFT (Group Presentations)**
Week 8	Optimize Quality for Business Outcome Part1 and 2 – **Assignment**
Week 9	Anatomies of Test Cases
Week 10	Intro to ALM; Security Testing & Cyber Security
Week 11	Cross Browser and Mobile testing; Usability & User Experience Testing
Week 12	**Test Automation Project: Selenium (Group Presentations)**
Week 13	Performance & Load Testing using Performance Center
Week 14	Career Management; IT Leadership, Group Thinking; Motivation & Decision Making Skills
Week 15	QA Job Positions & Job Interviewing Questions/Answers
Week 16	**Final Project Presentation (Group Presentations)**

Course Assessment

- **Class Participation (10%) –10% makes a difference!!!**
- **Tests/Quizzes (30 %)** – Short quizzes almost every week.
  - Timely Submission
  - Completeness & Correctness
- **Projects/Assignments (25%)** – UFT/Selenium/Agile assignments for each group.
- **Final Exam – Project (35%)** – Final project. Your team will do the following:
  - Test Condition Matrix/Test Scenarios/Test Cases/Traceability Matrix
  - Presentation
  - Test Plan

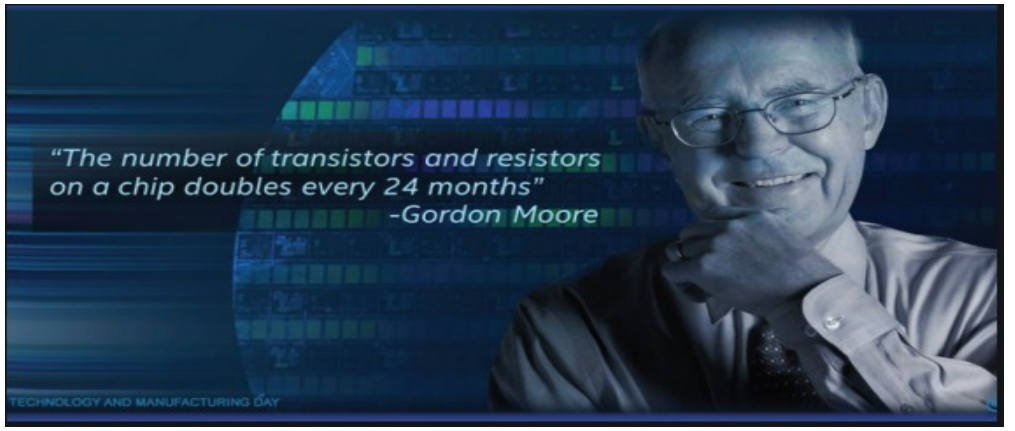
More on the Quizzes…

- There will be a quiz almost every week on the topics that were discussed the previous week.
- The quizzes are Open Book, Open Notes, Open Canvas.
- **Important Notes:**
  - The quizzes will be based on what was covered in the previous classes.
  - **Quizzes are Fixed lengths**. If you go over the time, you will lose points.
  - **No copying and pasting of the answers**. **All answers must be typed in**.
  - It is not a good time for you to do research on the web during the quizzes.
  - Quizzes start around 6:10, 6:15 PM. If you are late or you are a no show, you will miss the quiz.

About You!

- Who are you?
- What is your major?
- What do you want to get out of this course?
- When are you graduating?
- Coop Experience?
- Interests?
- Career plans?

Moore’s Law



This ‘**law**’ developed from the thinking of Gordon Moore, one of **Intel’s founders**, underpins much of the thinking and predictions about when our technology will become intelligent than us.

The simplified version of this **law** states that processor speeds, or overall processing power for computers will double every two years.

https://humanswlord.wordpress.com/2014/01/14/brainfood-moores-law-explained/

1 The accelerating pace of change ...

Agricultural Revolution ← 8,000 years → Industrial Revolution ← 120 years → Light-bulb ← 90 years → Moon landing ← 22 years → World Wide Web ← 9 years → Human genome sequenced

2 ... and exponential growth in computing power ...

Computer technology, shown here climbing dramatically by powers of 10, is now progressing more each hour than it did in its entire first 90 years

3 ... will lead to the Singularity

2045
Surpasses brainpower equivalent to that of all human brains combined

Surpasses brainpower of human in 2023

Surpasses brainpower of mouse in 2015

Colossus
The electronic computer, with 1,500 vacuum tubes, helped the British crack German codes during WW II

UNIVAC I
The first commercially marketed computer, used to tabulate the U.S. Census, occupied 943 cu. ft.

Apple II
At a price of $1,298, the compact machine was one of the first massively popular personal computers

Power Mac G4
The first personal computer to deliver more than 1 billion floating-point operations per second

COMPUTER RANKINGS
By calculations per second per $1,000

Analytical engine
Never fully built, Charles Babbage's invention was designed to solve computational and logical problems

Hollerith Tabulator
IBM Tabulator
National Ellis 3000
Zuse 2
Zuse 3
ENIAC
IBM SSEC
BINAC
Whirlwind
EDVAC
DEC PDP-4
IBM 1130
Datamatic 1000
IBM 1620
Data General Nova
DEC PDP-10
Intellec-8
IBM PC
Compaq Deskpro 386
Pentium PC
Pentium II PC
Dell Dimension 8400
Mac Pro
Nvidia Tesla GPU & PC

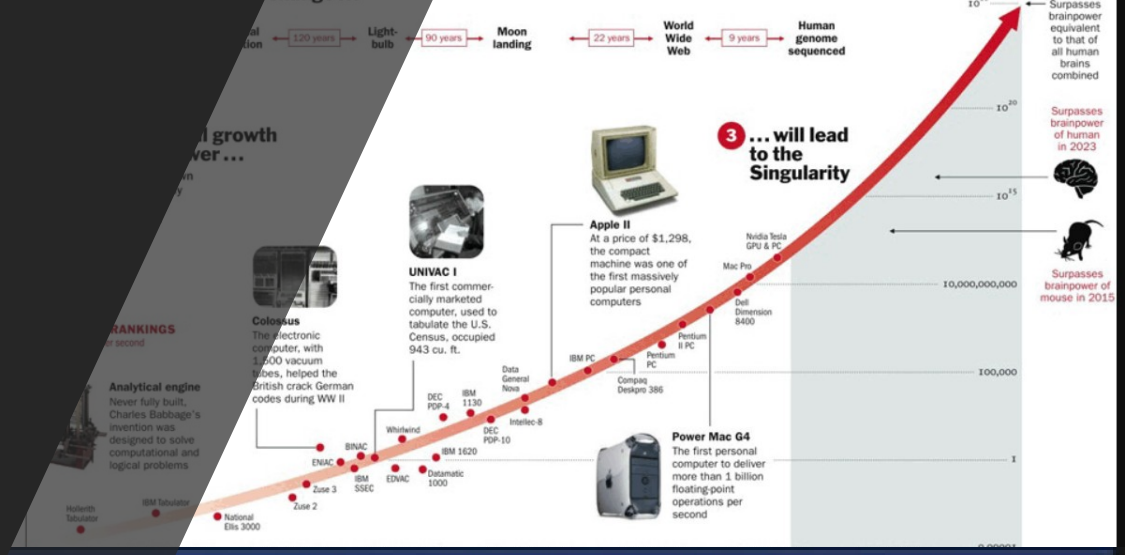
10^{26}
10^{20}
10^{15}
10,000,000,000
100,000
1
0.00001

ELECTROMECHANICAL — RELAYS — VACUUM TUBES — TRANSISTORS — INTEGRATED CIRCUITS

1900 1920 1940 1960 1980 2000 2011 2020 2045

What does all of this really mean?

- The Internet is getting **bigger and more complicated** every day!!
- **Software Applications** will become more complicated!!
- The **testing** will get more complicated!
- There will be greater need for **Software Developers and Software testers!!!**



Quality Control Management (QCM)

- **QCM** is the act of overseeing all activities and tasks needed to maintain a desired level of excellence. This process includes implementing a strategy for:
  - Quality Planning
  - Quality Assurance
  - Quality Improvement
  - Quality Control

https://isoready.com/quality-control-management/

Quality Management System's Basics

- QMS' Basic Rules:
  1. Document your QMS by stating the processes.
  2. Follow the documented processes.
  3. Prove that you are following the processes.
     - By the form of Audits.
  4. Continually review the processes and improve them.

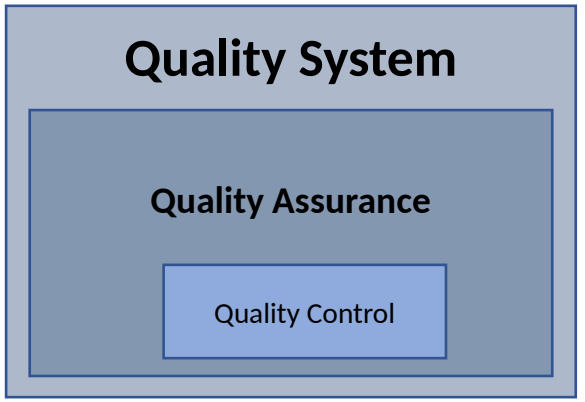
ISO9001 is an example of a Quality Management System

What is a Quality System?

A **quality system** is a specific implementation of quality philosophies/concepts, standards, methodologies and tools, for the purpose of achieving quality-related goals

Quality System = Quality Assurance + Quality Control

- **Quality Assurance** aims to prevent defects with a focus on the process used to make the product.
  - It is a proactive quality process.
- **Quality Control** is a set of activities for ensuring quality in the final products.
  - Monitor operations to ensure that they meet production standards.
  - It is a subset of QA.



https://asq.org/quality-resources/quality-assurance-vs-control

Quality Control

- **Quality Control** is the process that allows firms to ensure the conformity of their products or services. It is used to make sure it meets the:
  - Correct Specifications
  - Quality Benchmarks

- **QC** Tools used:
  - Checklists
  - Fishbone diagrams
  - Control Charts
  - Histograms
  - And more…

https://bizfluent.com/how-7640484-develop-quality-control-plan.html

IT Compliance

- **IT Compliance** is taking appropriate control of and protecting the information. It includes:
  - How the information is **obtained and stored**
  - How it is **secured**
  - How it is distributed **internally and externally**
  - How the **data is protected**

https://www.smartsheet.com/understanding-it-compliance

IT Compliance…

- **A few standards:**
  - **The Sarbanes -Oxley Act (SOX) of 2002** is a sweeping statute to regulate financial transparency and reporting.
  - **HIPAA, or Health Insurance Portability and Accountability Act's Title II** section articulates policies and guidelines for regulating information, especially Protected Health Information (PHI) by **insurers, medical providers, and employers** who provide health care insurance.
  - **Basel III** applies to the **banking industry** and helps determine the amount of capital they need to reserve in order to recover in the case of a loss.
  - **The Payment Card Industry Data Security Standard of 2001 (PCI DSS)** is an industry deployed recommendation instituted by **MasterCard, Visa, and other credit card companies** to provide identity protections for members and service providers.

https://www.smartsheet.com/understanding-it-compliance

What is the focus of this course?

- Software Quality Assurance (SQA) or Software Testing
- Where SQA fits within the SDLC
- Learning the QA Methodologies
- Learning about the QA Phases
- Learning two of the mostly used QA Tools
  - **(UFT and Selenium)**
- Learn more about the QA Jobs

QA - Theory vs Practice

This course is not about just a **'theoretical'** Software testing.

This course is about different **attributes** of the software testing:

- From **my perspective** & based on my experience
- What **skills** I believe a QA resource should have
- What **kind of attitude** a QA resource should have
- A problem **solver** or a problem **creator**?
- How well he/she works in a **team environment**?
- Hard **worker or a slacker**?

Reviewing the Course Topics

- What is Software testing?
  1. ***Software testing*** *is a process, to evaluate the functionality of a software application with an intent to find whether the developed software met the specified requirements or not. To identify the defects to ensure that the product is defect free in order to produce a quality product.*

  2. ***Software testing*** *is an activity to check whether the actual results match the expected results and to ensure that the software system is Defect free.*

  3. ***Software Testing*** *is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not.*

  4. ***Software testing*** *is the process of evaluating a system with the intend of finding bugs. It is performed to check if the system satisfies its specified requirements.*

https://artoftesting.com/manualTesting/what-is-software-testing.html

https://www.softwaretestingmaterial.com/software-testing/

Software Testing…

What is Software testing?

- ***Software testing*** *is a* ***process****, to evaluate the functionality of a software application with an intent to find whether the developed software* ***met the specified requirements or not****. To* ***identify the defects*** *to ensure that the product is defect free in order to produce a quality product.*

The 'Key points':

- *It is a* ***Process***
- ***Met the Specified Requirements*** *or not*
- ***To Identify the Defects***
- *The product is* ***Defect Free? -***
  - ***BTW - <u>No software is defect free!</u>***

What makes a great QA Tester?

- Attributes of a great **QA Analyst**:
  - Great **Analytical Skills**
  - Great **Communication Skills**
- What does "**Analytical Skills**" really mean?
  - **'Analytical Skill'** is defined as the ability to visualize, articulate, conceptualize or solve both complex and uncomplicated problems by making decisions that are sensible given the available information.

- Why "**Communications Skills**" are needed?
  - **In Writing:** to write test plans and test cases, defects and reports.
  - **Verbal:** to explain the test cases, test reasoning and the issues/problems.

Reviewing the course topics

- **SDLC – QA Methodologies**
  - **Waterfall**
    - Business Requirement vs. Functional Requirements?
    - Test Plan/Test Strategy
    - Test Condition Matrix
    - Traceability Matrix
    - Equivalence Class Partitioning
    - Test Coverage
  - **Agile:**
    - Epics, Sprints, Scrums, User Stories, etc.
  - Automation Testing
  - Performance/Load Testing
- When do you know you are done with testing?

More on the QA Methodologies

- **Waterfall vs. Agile**
  - **Business Requirements** vs **Functional Requirements**
  - How to link the **Requirements and the Test Cases** and why?
  - How to create a **traceability matrix to ensure accurate coverage**
  - How to **write test cases**
  - How to conduct testing in **Agile**
  - What to do as part of the **Sprints**
  - More about Scrums, Sprints, QA role?

“Equivalence Class Partitioning”

- It greatly **reduces the number of test cases without compromising the test coverage or quality of testing.**
  - It helps in reducing the overall test execution time due to reduced set of test data.
  - It is highly used in cases where exhaustive testing is not possible but at the same time test good coverage needs to be maintained.
- **Disadvantages of equivalence class partitioning**
  - The identification of equivalence classes relies **heavily on the expertise** of the tester.
  - Having **incorrectly identified equivalence classes** leads to higher risk of defect leakage and less test coverage.
  - The equivalence classes needs to be **partitioned just to the right amount** and groups as having too large partitions leads to risk of missing defects.
  - Whereas, partitioning into more **groups of smaller sizes leads** to redundant tests.

Testing Phases & Types

- Unit Testing
- Integration Testing
- Functional Testing
- System Testing
- Stress Testing
- Performance Testing
- Usability Testing
- Acceptance Testing
- Regression Testing
- Beta Testing
- Longevity testing
- User Roles Testing
- Manual Testing
- Automation Testing
- Load Testing
- White Box Testing
- Black Box Testing
- Grey Box Testing
- Security Testing
- Vulnerability testing
- Smoke Testing
- Scalability Testing
- Interoperability Testing
- International/Localization Testing
- Concurrency Testing

https://www.inflectra.com/ideas/topic/testing-methodologies.aspx

More on QA...

- **Happy Path Testing:** very simple flow testing (1 path).
- **Smoke Testing:** A high level validation of the functionality.
- **Equivalence Partitioning:** A technique to determine the minimum set of test cases that must be executed.
- **RAD Model:** Rapid Application Development an iterative process.
- **Test Stub:** A “dummy” routine to enable a function to be tested.
- **Test Harness:** a pool of test stubs and test drivers put together.
- **Test Summary Report:** A document showing the test results in a compacted format.

Reviewing the semester topics

- How to write test plans and construct Test Cases
- How the Waterfall and Agile processes work, and the phases associated with it.
- Create a Traceability Matrix
- Change vs. Release Management
- QA Tools: ALM, Jira, **UFT, Selenium**, Performance Center **(Two hands on assignments)**
- HP Optimize Quality for Business Outcomes…
- Cross Browser Testing/Mobile Testing
- Verbal and Written Communications
- Security Testing & Cyber Security
- Decision Making and Group Thinking
- Job Interviewing Techniques
- The Final Project (eCare application)

Types of QA Jobs

- **Different types of QA jobs:**
  - **Manual Tester** – All tests are through the GUI or the application interface.
  - **Automation Tester** – Writes scripts for Automation testing using one of many tools in the market, e.g. Selenium, HP UFT, or other scripting languages.
    - Or – uses different languages to write scripts to test the interconnectivity of an application, such as (VB Scripting, Python, and etc.)
  - **Data/Data Base Tester** – All testing is done through writing and executing SQL statements or running various scripts for testing.
  - **Performance & Load Tester** – Performance, Load & Stress testing using one of many tools that are in the market. E.g. HP Performance Center.
  - **Security/Cybersecurity Tester –** Performs application security testing, or Cybersecurity testing.

QA Organizations Reporting Structure

QA is under the same umbrella as the Development

- Reporting to the same VP or an Organizational head
- Pros and Cons!!
  - **Cons:** May get swayed to move faster through QA

QA is under a different organization than the Development

- Reporting to a different VP or an Organizational head
- Pros and Cons!!!
  - **Pros:** Independent and more free to conduct the testing

QA Terminology

- **Test Dashboard/Test Report** – A matrix showing the test execution summary with pass/failure of the tests.
- **SIT** – Systems Integration Testing done before the actual testing begins.
- **Unit Testing** – Testing done by the development to ensure that their code functions as designed.
- **Integration Testing:** Testing multiple components interfacing with each other.
- **Regression Testing** – Repeating the running of the older test cases.
- **Manual Testing** – Running the test cases by humans.
- **Automated Testing** – Running the test cases using automated tools or scripting.

QA Terminology…

- **Performance Testing –** Running test to ensure that the product can function under a specified load.
- **Load and Stress Testing** – Same as above. Stress testing goes above the Performance testing to find the level of failure.
- **Test Evidence/Test Results** – Showing proof of testing by taking screen shots of the actual results.
- **Defects** – Reporting the failure of the test cases to the development.
- **Defect Severity & Priority** – The place in the queue for fixing and deploying into QA/Production.
- **Defect Process** – Agreed upon process on how to handle the defects Reporting, prioritization and fixing.
- **Builds** – Versioned code ready to be deployed into an environment.
- **Builds management** – Includes tools to deliver the code to QA in a consistent and controlled manner, including versioning.

QA Terminology…

- **Source Control –** Using various tools to keep track of what goes into a build so a well measured and a solid build is delivered to QA.
- **Dev/QA/Prod Environments** – QA environment is where the testing can be performed. It needs to be controlled from the code perspective and all builds into it are installed via Build Management software.
- **Test Data** – Ensuring the test data exists in the QA database for testing.
- **Change Request** – Change tickets that introduce changes to the frozen functionality.
- **QA Entrance/Exit Criteria** – Conditions that allows the start and finish the testing cycle.
- **QA Testing tools** – Running the testing using an automation tool as a regression test to speed up the process.
- **QA Signoff** – QA end of testing.
- **Sprint** – a small bundle of wish list items and the decisions on how to develop them.
- **Scrum** – is the agile software development methods.

Top 3 QA Methodologies

1. **Waterfall:** the Waterfall model 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.
3. **DevOps:** The DevOps methodology is the newcomer to the SDLC scene. It emerged from two trends:
   - The application of Agile
   - Lean practices to operations work

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

https://www.roberthalf.com/blog/salaries-and-skills/6-basic-sdlc-methodologies-which-one-is-best

Other QA Methodologies

- **Lean:** The Lean model for software development is inspired by **lean manufacturing** practices and principles. The seven Lean principles (in this order) are:
  - Eliminate waste
  - Amplify learning
  - Decide as late possible
  - Deliver as fast as possible
  - Empower the team
  - Build integrity in
  - See the whole

Other QA Methodologies

- **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 ready.

- **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 and evaluation**) over and over in a “spiral” until completed, allowing for multiple rounds of refinement.

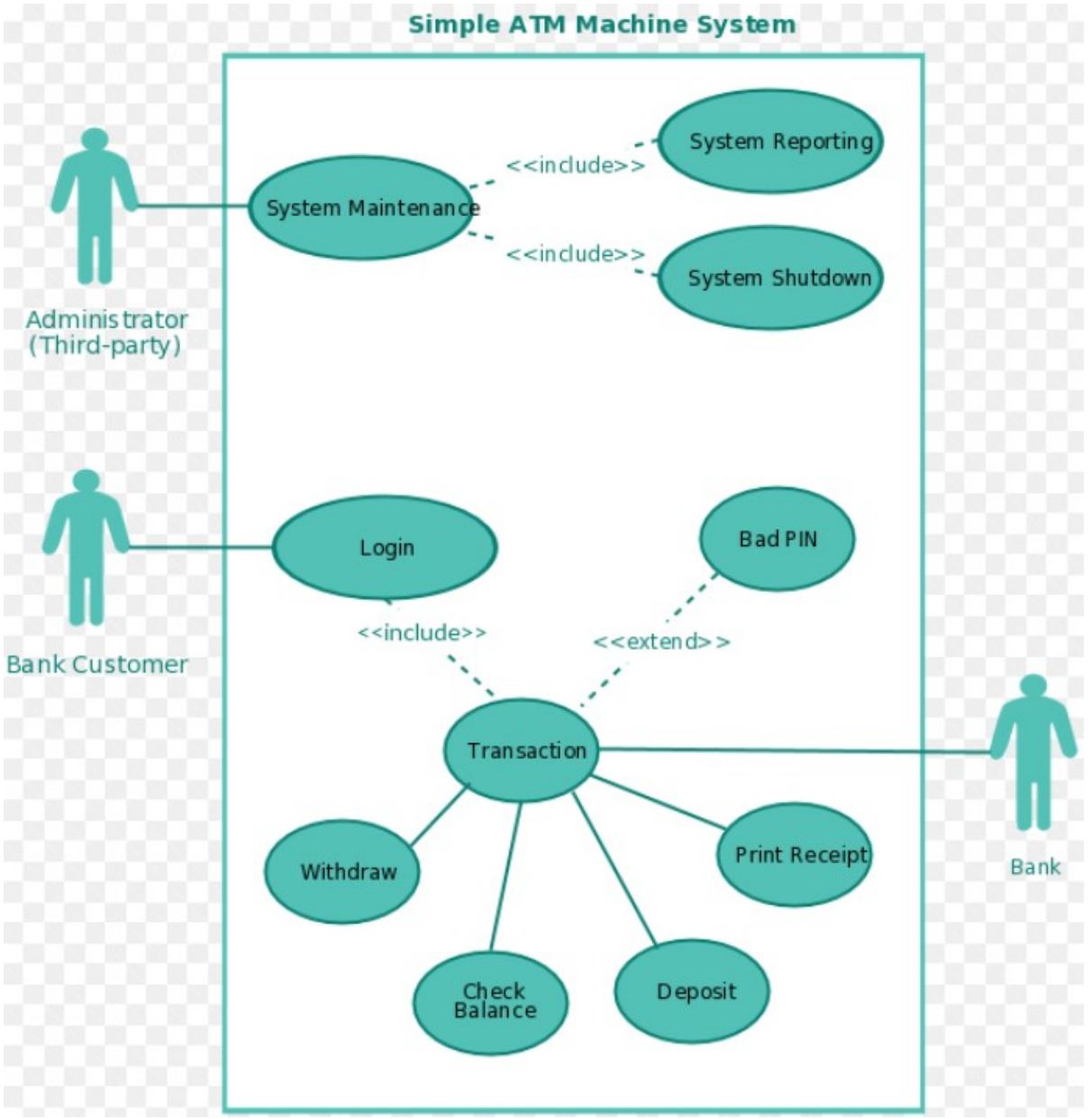
What is a Use Case?

- **Use Case** is another way of defining functional requirements.
- It introduces an **Actor** who will perform various functions in the application.
- It shows the **interaction** of the **Actor** with the system or an application.
- The advantage is that it allows for better **understanding** of the process flow when an **Actor** is involved.
- There could be **multiple Actors** describing the Use Cases from different user perspectives.
- **Use Cases are not User Stories**. User Stories are short descriptions of what the customers do within the application.

USE CASE TEMPLATE

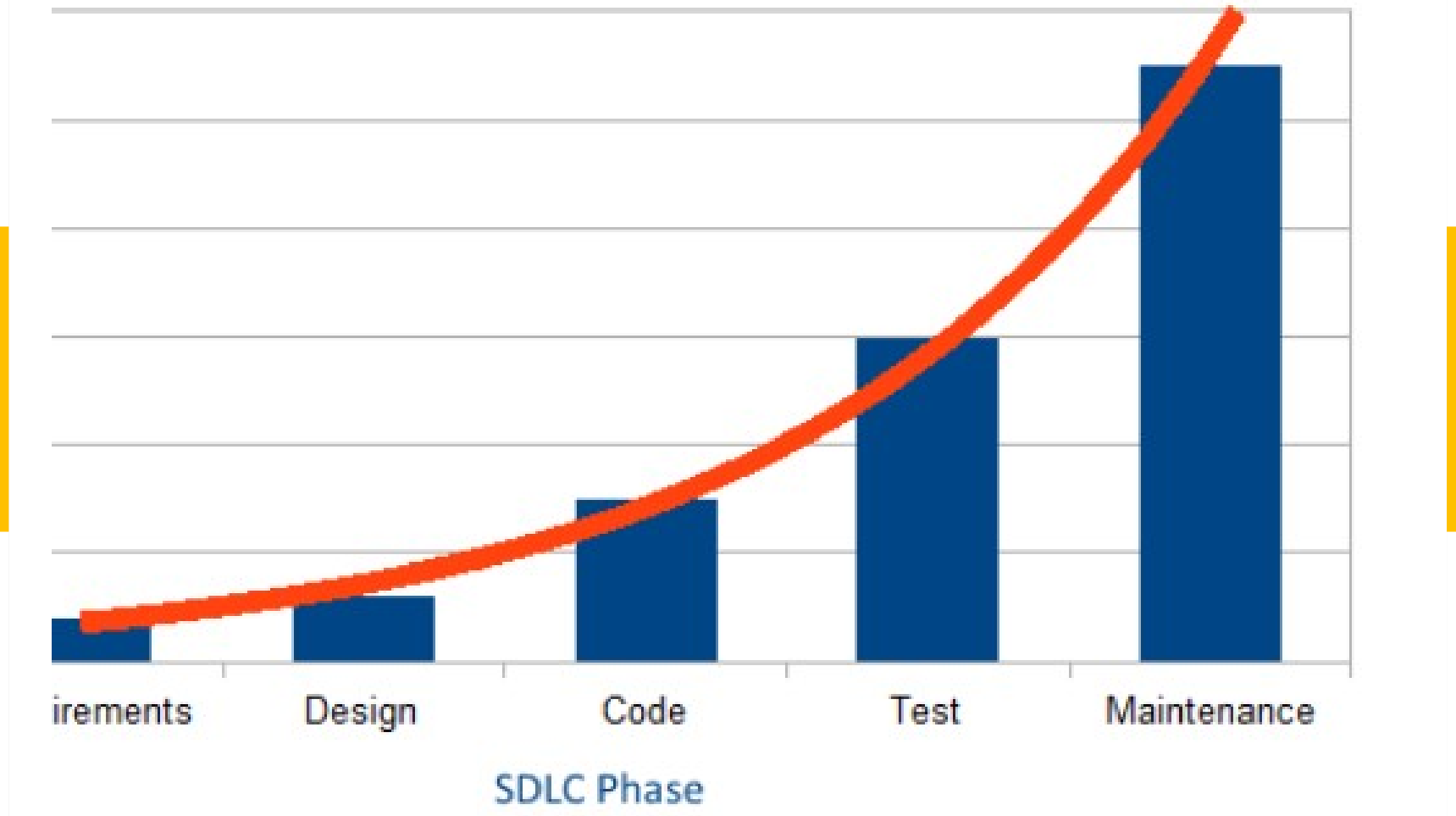
- **Name** – A clear verb/noun or actor/verb/noun descriptor that communicates the scope of the use case.
- **Brief Description** – A brief paragraph of text describing the scope of the use case.
- **Actors** – A list of the types of users who can engage in the activities described in the use case. Actor names should not correspond to job titles.
- **Preconditions** – Anything the solution can assume to be true when the use case begins.
- **Basic Flow** – The set of steps the actors take to accomplish the goal of the use case. A clear description of what the system does in response to each user action.
- **Alternate Flows** – Capture the less common user/system interactions, such as being on a new computer and answering a security question.
- **Exception Flows** – The things that can happen that prevent the user from achieving their goal, such as providing an incorrect username and password.
- **Post Conditions** – Anything that must be true when the use case is complete.

Use Case Diagram example



More on QA...

- **POC:** Proof of Concept. A prototype.
- **Test Log:** Keeping a log of the test execution during testing.
- **User Story:** A user/customer function that results into a system function (equivalent to a Functional Requirement).
- **Walkthrough:** A review session to walk through a test plan, test cases or performing a Code Walkthrough to go through the code.



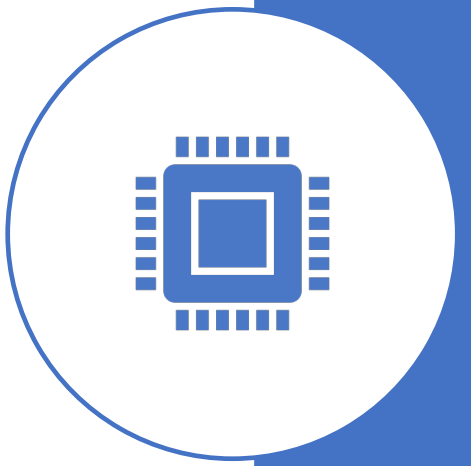
Cost of Defects

Web Based Testing

- Web Based Application Testing
  - Typical topology: **Client -> Web Server -> App Server -> DB**
  - Thin-Client vs. Thick-Client testing
  - **Thin-Client** – most processing is happening on the Server side
  - **Thick-Client** – Most processing is happening on the Client side (Java Applets and ActiveX controls on the Client side)
  - Advantages and disadvantages
    - **Thick Client:** Performance testing focus
    - **Thin Client:** Less of a performance issues risks

Web Based Application

- **Testing Considerations:**
  - **Browser Testing:** Test as many as browsers as specified in the support document.
  - **Functionality Testing:** The functions are working?
  - **Integration Testing:** Are all interfaces working?
  - **Usability and User Interface testing**
  - **Security Testing:** Role based testing
  - **Load and performance testing:** For ensuring performance meets the SLAs
  - **Printing**
  - **Caching**
  - **File Upload/Download**



Verification vs. Validation

- **Verification** comes early on in the project to verify the
  - Functional Requirements
  - Code Inspections & Walkthroughs
  - Application Design

- **Validation** comes later in the project during the actual testing. It includes all phases of the testing:
  - Functional Testing
  - Integrations Testing
  - End to End or System Testing
  - UAT

Testing Related Tools

- **Test Management Tools:** QC, ALM, Jira, TestRail
- **Test Automation Tools:** UFT, Selenium
- **Load Testing Tools:** Performance Center, LoadRunner, Jmeter, Webload
- **Cross Browser Testing Tools:** BrowserShots, Saucelabs, Selenium Grid
- **Security Testing Tools:** Wfuzz, Grabber, Zed Attack Proxy, SQLMap
- **Cyber Security Testing Tools:** Metasploit, Nmap, Wireshark

What do you know about your profession?

What do you want to do professionally?

Do you know how to Interview for a job?

Do you know what the work entails?

Can you work in a group environment?

Do you know your personality?

IT jobs Relative to QA

- Programmer, or Developer
- Program Director
- Project Manager (PM)
- Business Systems Analyst (BSA)
- Systems Analyst
- Data Analyst
- Systems Architect
- Database Base Analyst (DBA)

Summary

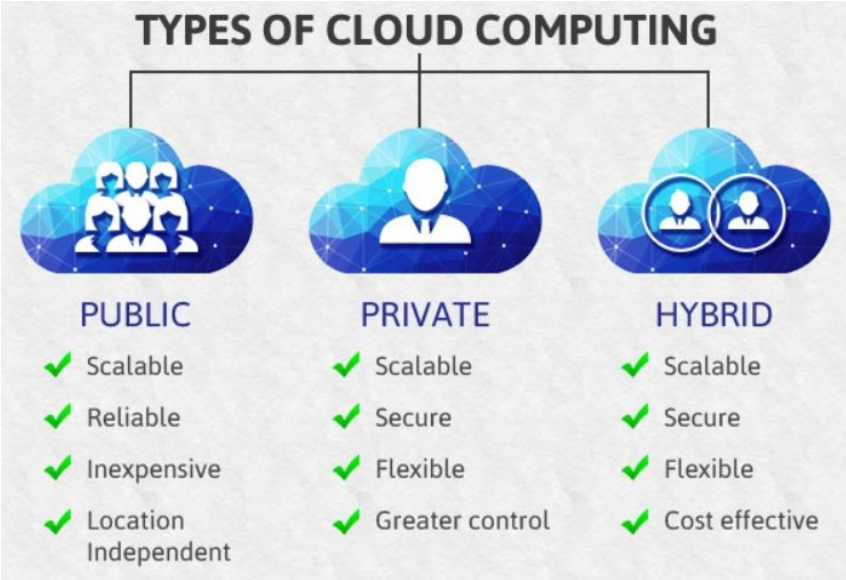
- You will learn
  - What it takes to **develop and deploy** applications from start to finish.
  - How to **maintain and update** the applications running in Production.
  - What **performance testing** is and why it is so important.
  - What **automation testing** is and how it automates the regression testing.
  - **Hands on automation** testing in Selenium and UFT.
  - **Protecting** the applications and the network from **Hackers**.
  - Learn about **Cybersecurity**.
  - **Career Management;** What does it take to start and advance your career. How to interview.
  - **Decide** on what you want to do in **IT**.

Cloud Computing

- What is the Cloud?

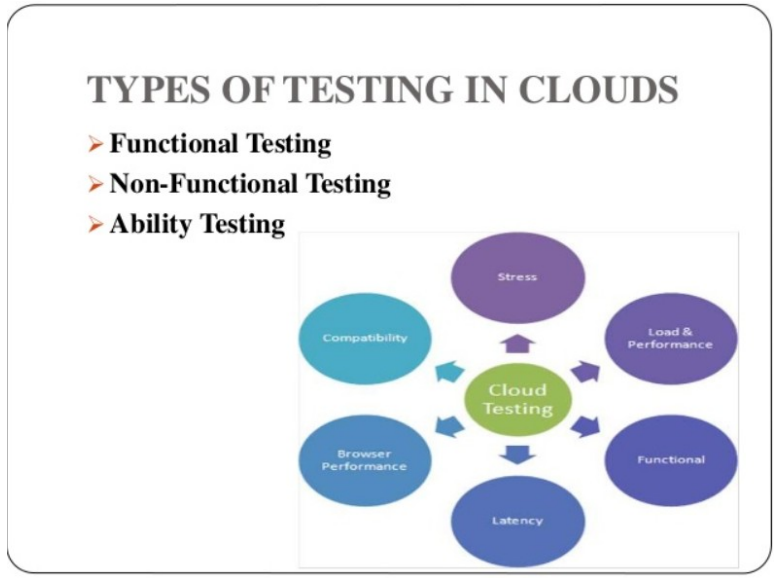
“the practice of using a network of remote servers hosted on the Internet to store, manage, and process data, rather than a local server or a personal computer”

The Cloud Types



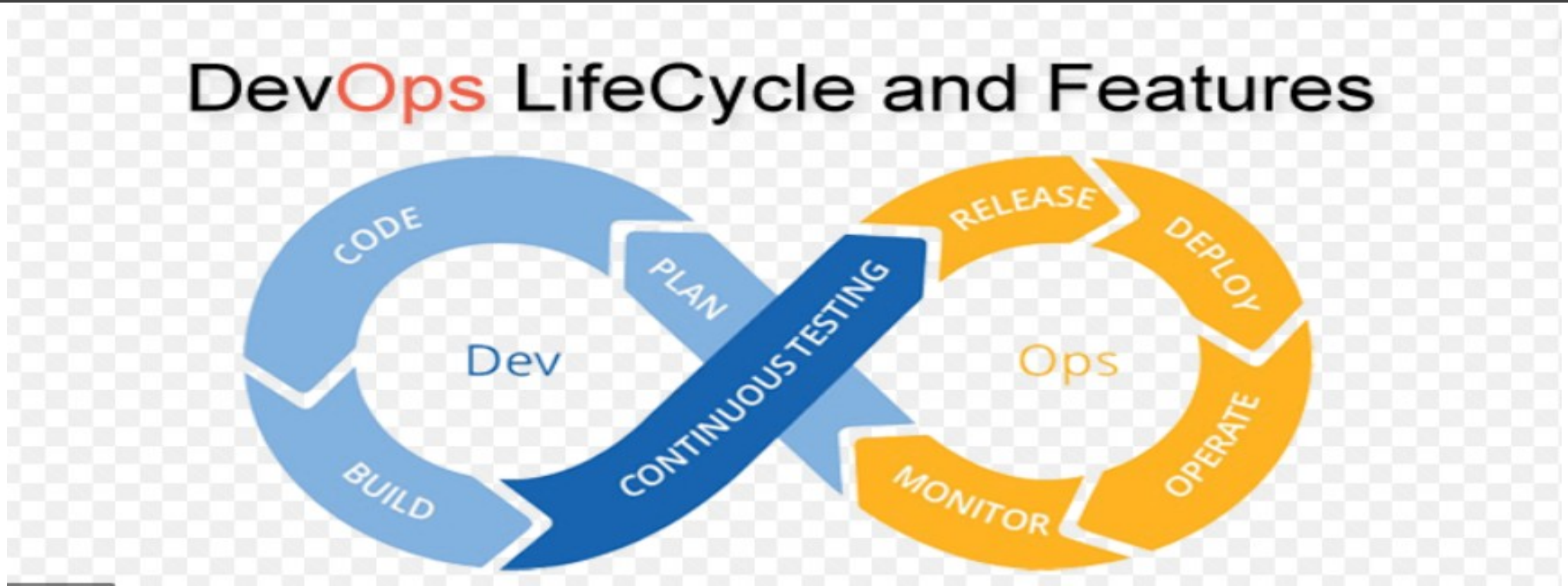
- **Private Cloud** – Operated for the organization only.
- **Community Cloud** – Shared by specific community
- **Public Cloud** – Available to the public
- **Hybrid Cloud** – Combining two or more clouds

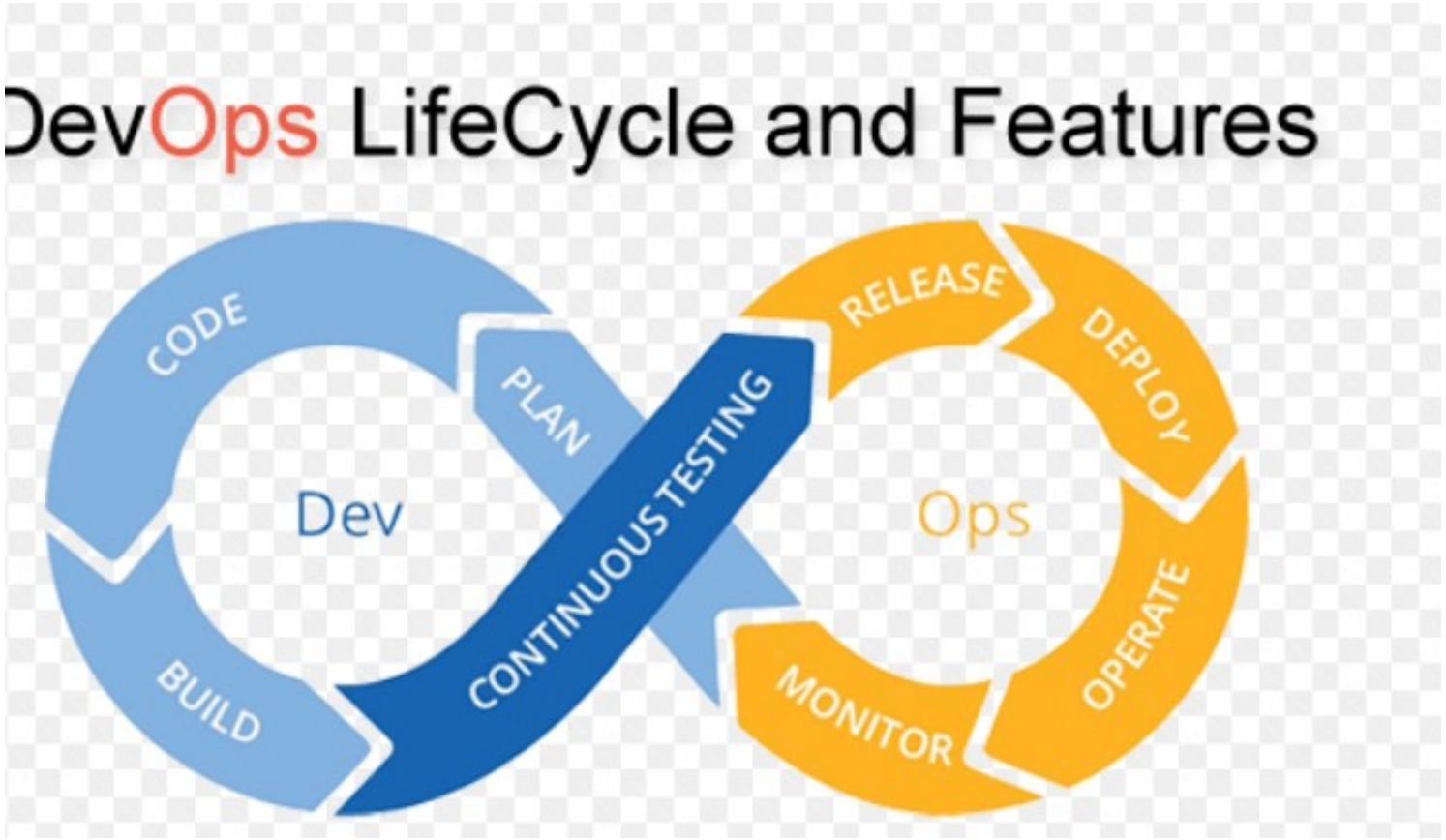
Cloud Testing Types



- **Testing in the Cloud**
  - Performance Testing
  - Security Testing
  - Availability Testing
  - External API testing

DevOps Methodology





DevOps

- The word 'DevOps' is a combination of two words 'development' and 'operations.'
- Why is it important?
  - Frequent new builds per day rather than per week
  - Stability, reliability and Security for all code builds
  - All major companies (google, Facebook, etc.) deliver world class builds into production
  - Seamless integrated model:

    Code Development -> Automated Test Environment -> Test Team executes the Automated tests -> Deploy into Production
  - The DevOps key to success is ‘Automation’

Back End Testing

- We will discuss this more
  - DB Testing
  - API Testing
  - Interface Testing

What’s on Your Mind?

Questions?