Software Crisis
* Software cost/schedules are grossly inaccurate. Cost overruns of several times, schedule slippage's by months, or even years are common.
* Productivity of people has not kept pace with demand Added to it is the shortage of skilled people.
* Productivity of people has not kept pace with demand Added to it is the shortage of skilled people.

Software Myths
Management Myths
* Software Management is different.
* Why change or approach to development?
* We have provided the state-of-the-art hardware.
* Problems are technical
* If project is late, add more engineers.
* We need better people.

Developers Myths
* We must start with firm requirements
* Why bother about Software Engineering techniques, I will go to terminal and code it.
* Once coding is complete, my job is done.
* How can you measure the quality..it is so intangible.

Customer's Myth
* A general statement of objective is good enough to produce software.
* Anyway software is "Flexware", it can accommodate my changing needs.

What do we do ?
* Use Software Engineering techniques/processes.
* Institutionalize them and make them as part of your development culture.
* Adopt Quality Assurance Frameworks : ISO, CMM
* Choose the one that meets your requirements and adopt where necessary.

Software Quality Assurance:
* The purpose of Software Quality Assurance is to provide management with appropriate visibility into the process being used by the software project and of the products being built.
* Software Quality Assurance involves reviewing and auditing the software products and activities to verify that they comply with the applicable procedures and standards and providing the software project and other appropriate managers with the results of these reviews and audits.

Verification:
* Verification typically involves reviews and meetings to evaluate documents, plans, code, requirements, and specifications.
* The determination of consistency, correctness & completeness of a program at each stage.

Validation:
* Validation typically involves actual testing and takes place after verifications are completed
* The determination of correctness of a final program with respect to its requirements.

Software Life Cycle Models :
* Prototyping Model
* Waterfall Model – Sequential
* Spiral Model
* V Model - Sequential

What makes a good Software QA engineer?
* The same qualities a good tester has are useful for a QA engineer. Additionally, they must be able to understand the entire software development process and how it can fit into the business approach and goals of the organization. Communication skills and the ability to understand various sides of issues are important. In organizations in the early stages of implementing QA processes, patience and diplomacy are especially needed. An ability to find problems as well as to see 'what's missing' is important for inspections and reviews.

Testing:
* An examination of the behavior of a program by executing on sample data sets.
* Testing comprises of set of activities to detect defects in a produced material.
* To unearth & correct defects.

- To detect defects early & to reduce cost of defect fixing.
  * To avoid user detecting problems.
  * To ensure that product works as users expected it to.

Why Testing?
* To unearth and correct defects.
* To detect defects early and to reduce cost of defect fixing.
* To ensure that product works as user expected it to.
* To avoid user detecting problems.

Test Life Cycle
* Identify Test Candidates
* Test Plan
* Design Test Cases
* Execute Tests
* Evaluate Results
* Document Test Results
* Casual Analysis/ Preparation of Validation Reports
* Regression Testing / Follow up on reported bugs.

Testing Techniques
* Black Box Testing
* White Box Testing
* Regression Testing
* these principles & techniques can be applied to any type of testing.

Black Box Testing
* Testing of a function without knowing internal structure of the program. Synonyms for black-box include: behavioral, functional, opaque-box, and closed-box.

White Box Testing
* Testing of a function with knowing internal structure of the program. Also known as glass box, structural, clear box and open box testing

Regression Testing
* To ensure that the code changes have not had an adverse affect to the other modules or on existing functions.

Functional Testing
* Study SRS
* Identify Unit Functions
* For each unit function
* - Take each input function
* - Identify Equivalence class
* - Form Test cases
* - Form Test cases for boundary values
* - From Test cases for Error Guessing
* Form Unit function v/s Test cases, Cross Reference Matrix
* Find the coverage

Unit Testing:
* Unit - smallest testable piece of software.
* A unit test is a method of testing the correctness of a particular module of source code. Typically done by the programmer and not by testers .

Benefits
The goal of unit testing is to isolate each part of the program and show that the individual parts are correct.

Limitations

It is important to realize that unit-testing will not catch every error in the program. it will not catch integration errors, performance problems and any other system-wide issues

Integration Testing:
is the phase of software testing in which individual software modules are combined and tested as a group.
Integration Testing takes as its input modules that have been checked out by unit testing, groups them in larger aggregates
There are two ways integration performed. It is called Pre-test and Pro-test.
* Pre-test: the testing performed in Module development area is called Pre-test. The Pre-test is required only if the development is

done in module development area.

Alpha testing:
* Testing of an application when development is nearing completion minor design changes may still be made as a result of such testing. Typically done by end-users or others, not by programmers or testers.

Beta testing:
* Testing when development and testing are essentially completed and final bugs and problems need to be found before final release. Typically done by end-users or others, not by programmers.

System Testing:
* A system is the big component.
* System testing is aimed at revealing bugs that cannot be attributed to a component as such, to inconsistencies between components or planned interactions between components.
* Concern: issues, behaviors that can only be exposed by testing the entire integrated system (e.g., performance, security, recovery).

Volume Testing:
* The purpose of Volume Testing is to find weaknesses in the system with respect to its handling of large amounts of data during short time periods. For example, this kind of testing ensures that the system will process data across physical and logical boundaries such as across servers and across disk partitions on one server.

Stress testing:
* This refers to testing system functionality while the system is under unusually heavy or peak load; it's similar to the validation testing mentioned previously but is carried out in a "high-stress" environment. This requires that you make some predictions about expected load levels of your Web site. Stress testing a subset of load testing.

Usability testing:
* Usability means that systems are easy and fast to learn, efficient to use, easy to remember, cause no operating errors and offer a high degree of satisfaction for the user. Usability means bringing the usage perspective into focus, the side towards the user.

Security testing:
* If your site requires firewalls, encryption, user authentication, financial transactions, or access to databases with sensitive data, you may need to test these and also test your site's overall protection against unauthorized internal or external access.

Test Plan:
* A Test Plan is a detailed project plan for testing, covering the scope of testing, the methodology to be used, the tasks to be performed, resources, schedules, risks, and dependencies. A Test Plan is developed prior to the implementation of a project to provide a well defined and understood project roadmap.

Test Specification:
* A Test Specification defines exactly what tests will be performed and what their scope and objectives will be. A Test Specification is produced as the first step in implementing a Test Plan, prior to the onset of manual testing and/or automated test suite development. It provides a repeatable, comprehensive definition of a testing campaign.
Performance Testing:
Performance testing is testing that is performed to determine the performance under a particular workload. It tells whether system meets the performance criteria or not.

Installation Testing:
This type of testing is performed to ensure that all Install features and options function properly. It is also performed to verify that all necessary components of the application are, indeed, installed.

Stability Testing

In software testing, stability testing is an attempt to determine if an application will crash.

Acceptance Testing:
User acceptance testing (UAT) is one of the final stages of a software project and will often occur before the customer accepts a new system.