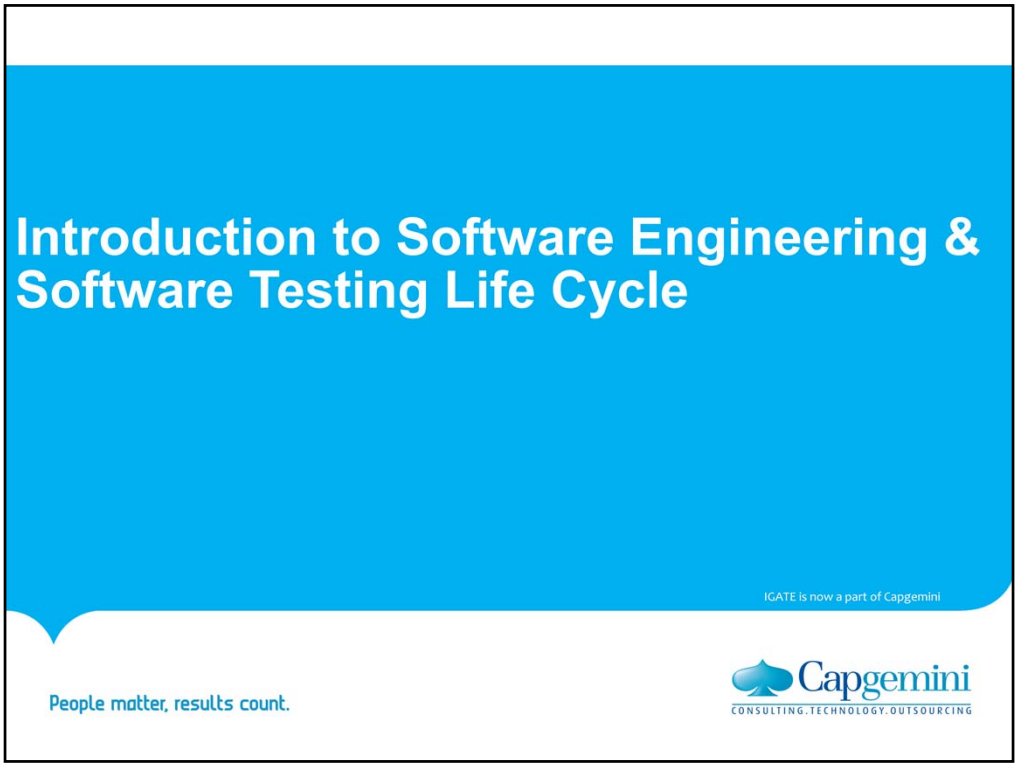


Document History

Date	Course Version No.	Software Version No.	Developer / SME	Reviewer (s)	Approver	Change Record Remarks
21-May-2013	0.1D		Latha S			
2-Sep-2013	1.0		Latha S			Incorporated Review comments
June-2016	1.1		Amruta Rakhonde & Shilpa Bhosle	Shilpa Bhosle	Mahima Sharma	Post-Integration revamp
July-2016	1.1		Anjulata			Refinements

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

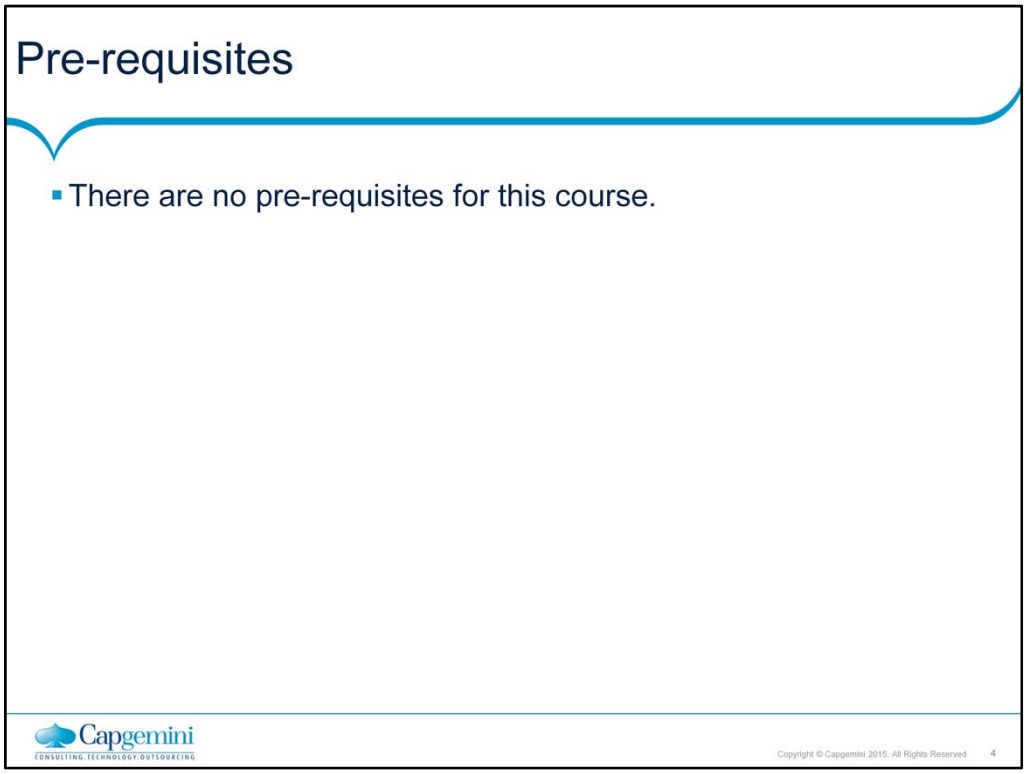
 2

Course Goals and Non Goals

- Course Goals
  - To provide an overview of software engineering
  - To provide an overview of software testing life cycle
- Course Non Goals
  - This course is only an introductory course to Software engineering and software testing life cycle.
  - This course does not intend to delve into details on activities involved in phases like requirements, high level design etc.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 3

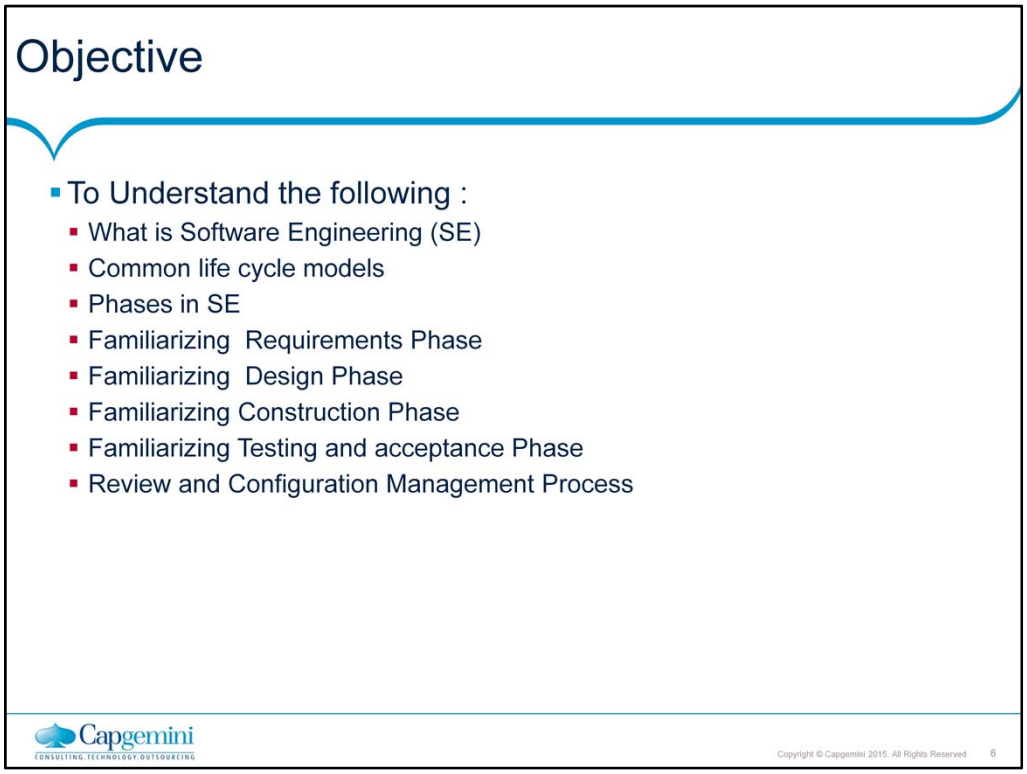

Pre-requisites
There are no pre-requisites for this course.
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
4

Intended Audience

- New entrants to the organization (Fresher's batches)

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 5

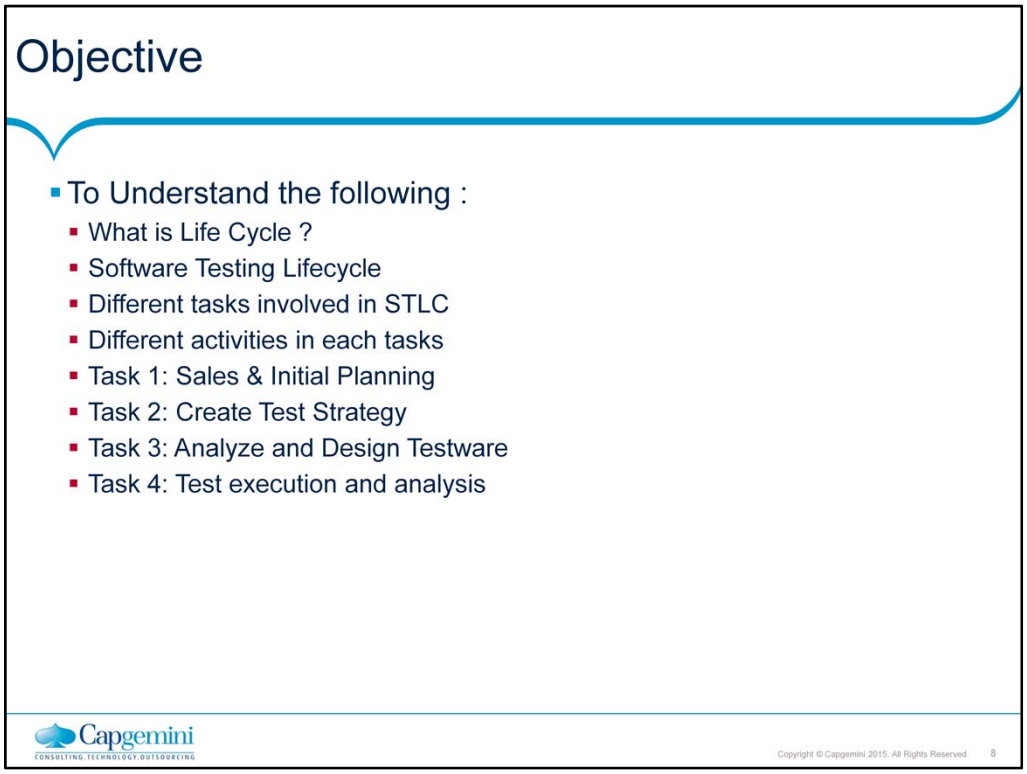

Objective
To Understand the following :
What is Software Engineering (SE)
Common life cycle models
Phases in SE
Familiarizing Requirements Phase
Familiarizing Design Phase
Familiarizing Construction Phase
Familiarizing Testing and acceptance Phase
Review and Configuration Management Process
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
6

Objective

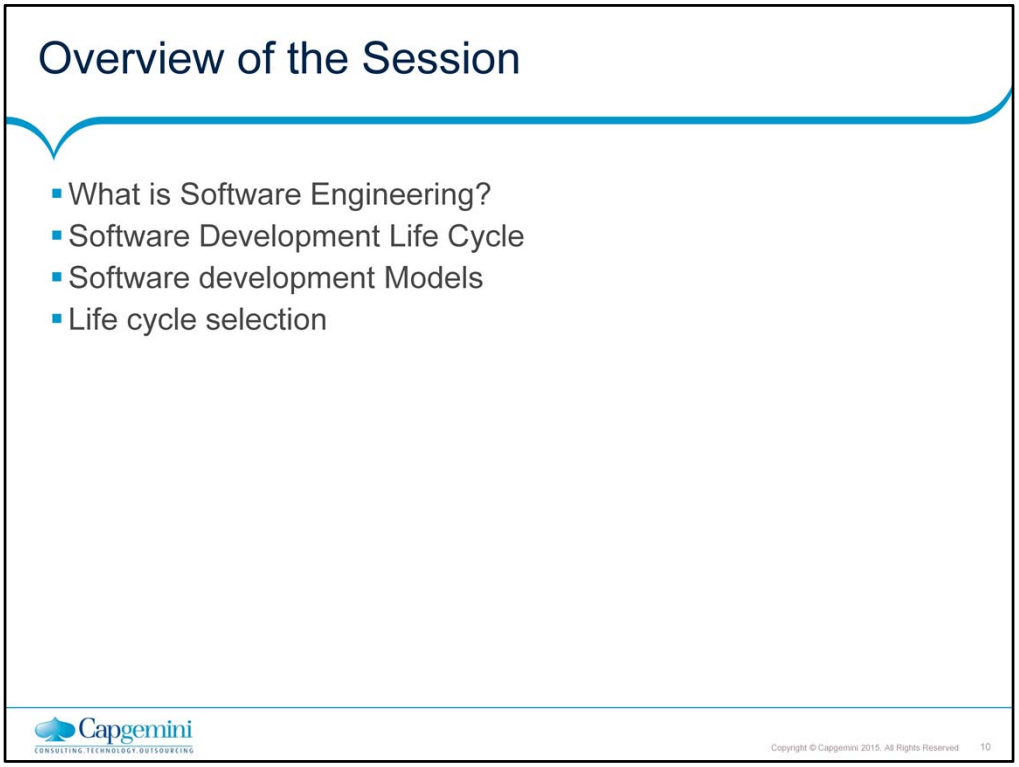
- To Understand the following :
  - What is Software Quality?
  - Other definitions
  - Quality Assurance & Quality Control
  - Importance of QA/QC
  - What is V&V?
  - Static and Dynamic V&V
  - Static and Dynamic Techniques
  - Why V&V?
  - Types of V&V

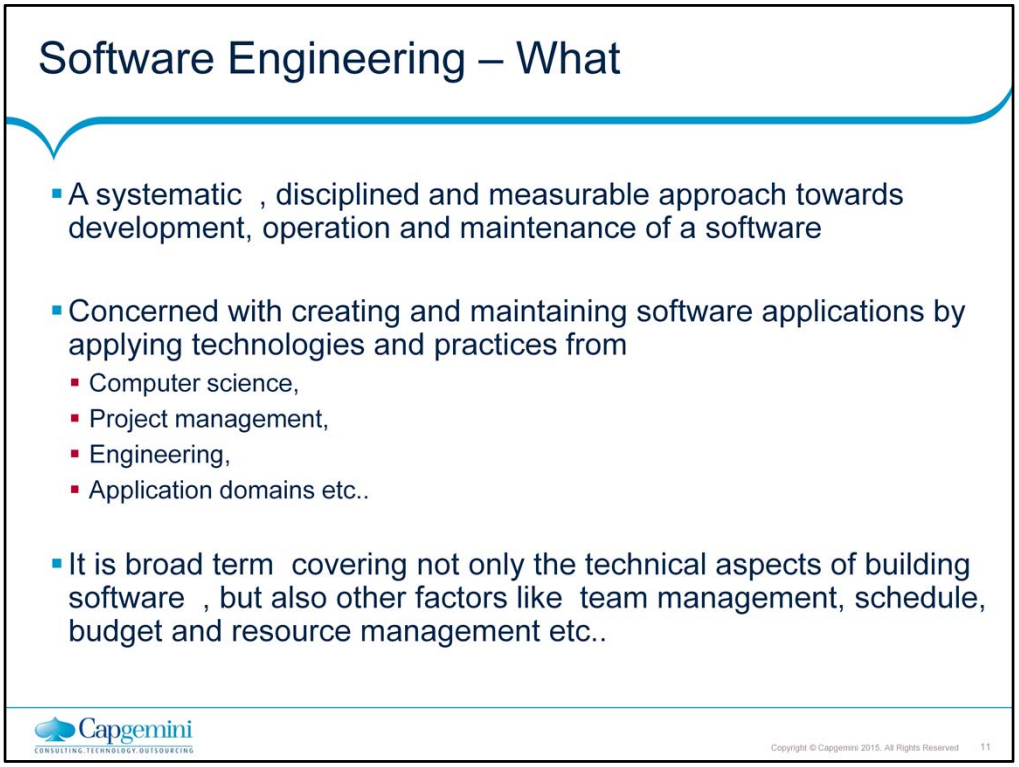
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 7


Objective
To Understand the following :
What is Life Cycle ?
Software Testing Lifecycle
Different tasks involved in STLC
Different activities in each tasks
Task 1: Sales & Initial Planning
Task 2: Create Test Strategy
Task 3: Analyze and Design Testware
Task 4: Test execution and analysis
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
8

Introduction to Software Engineering


Overview of the Session
What is Software Engineering?
Software Development Life Cycle
Software development Models
Life cycle selection
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
10

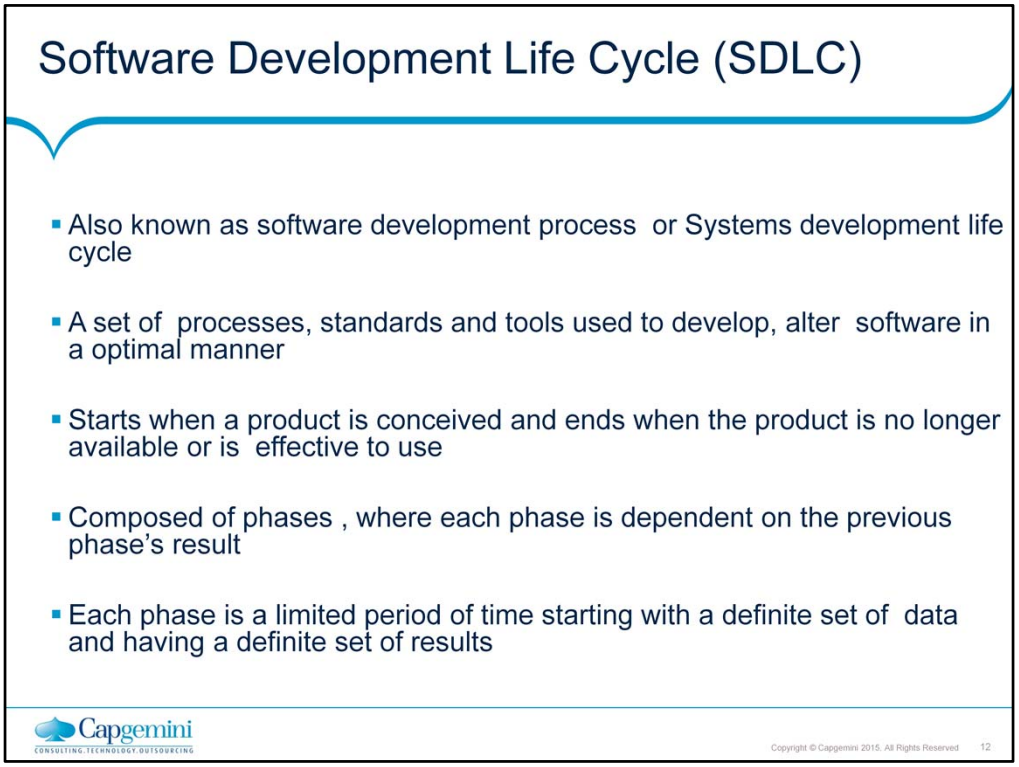


Typical other formal definitions of software engineering are

"an engineering discipline that is concerned with all aspects of software production" "the establishment and use of sound engineering principles in order to economically obtain software that is reliable and works efficiently on real machines“

Traditional engineers use science to construct “real” artifacts and software engineers use mathematics, science to construct “abstract” artifacts

In layman terms it is application of engineering towards development of a software

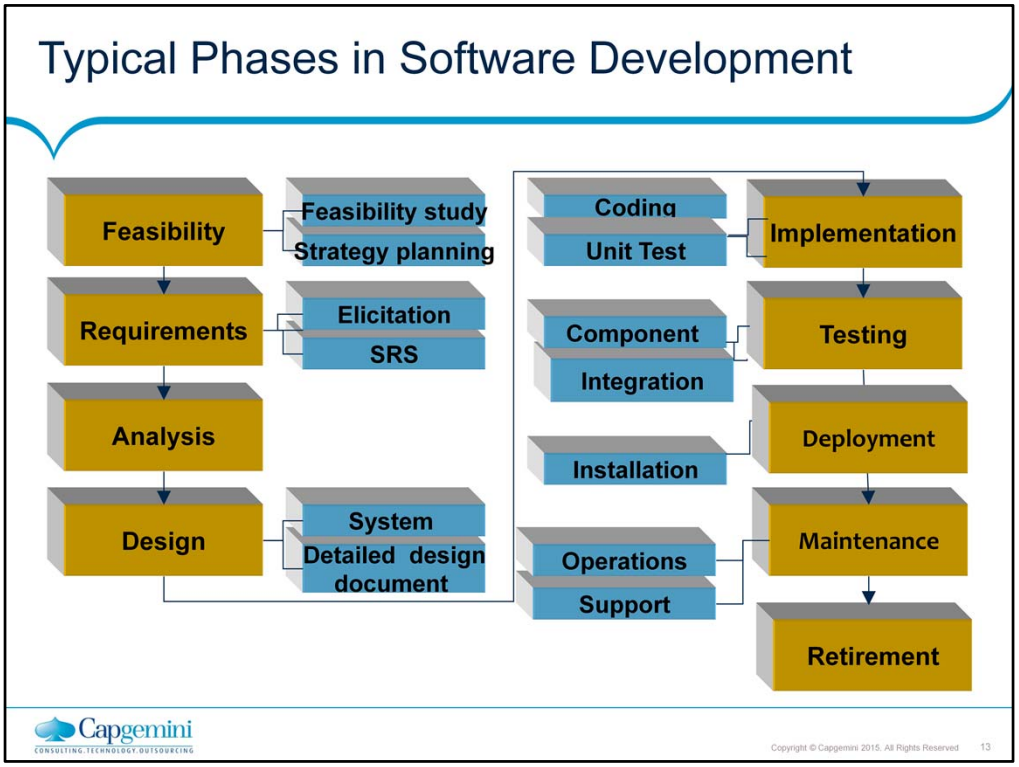


Also known as **systems development life cycle (SDLC)**, or **software development process**, or **Software Development Life Cycle**

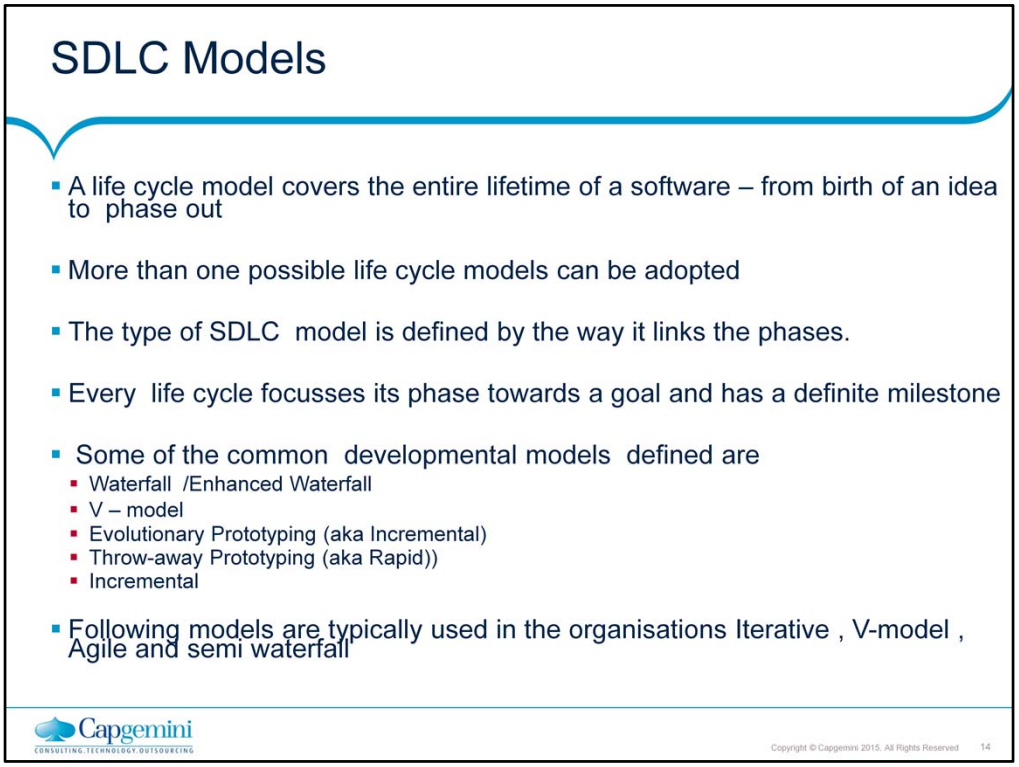
It is a process of creating or altering information systems using various models and methodologies The SDLC aims to produce a high quality system that meets or exceeds customer expectations, reaches completion within times and cost estimates, works effectively and efficiently

The SDLC framework provides a sequence of activities for system design and development . It consists of a set of steps or phases in which each phase of the SDLC uses the results of the previous one.

.



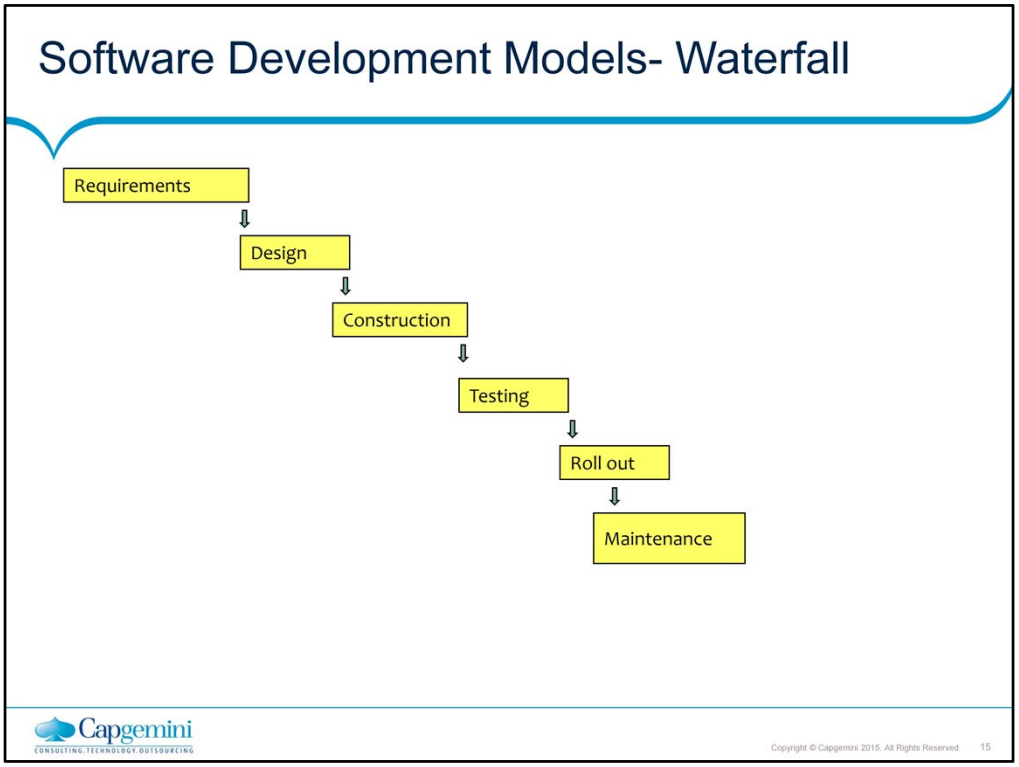
Activities during phases
Requirements: establish the customer's needs
System Design: develop the system's structure
Detailed Design: develop module structures
Implementation: code or otherwise
Testing: check what's been developed
Installation: bring the system into production
Maintenance: correct, adapt, improve



A (software/system) *lifecycle model* is a description of the sequence of activities carried out in an SE project, and the relative order of these activities.

Provides a generic framework for various activities to be done for the lifetime of the s/w - design, develop and maintain

More than 1 model can be chosen or models can be changed between releases of the s/w



Waterfall processes

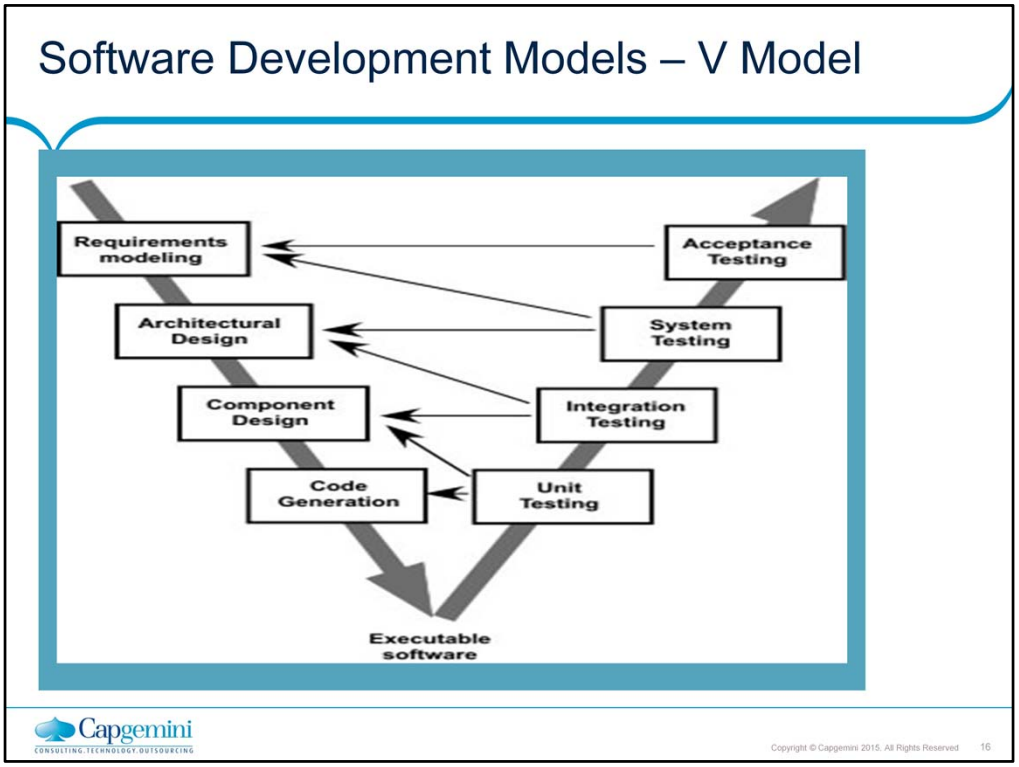
The waterfall model is a sequential software development model in which development is seen as flowing steadily downwards (like a waterfall) through the phases of requirements analysis, design, implementation, testing (validation), integration, and maintenance.

To follow the waterfall model, one proceeds from one phase to the next in a purely sequential manner. For example, one first completes "requirements specification" — they set in stone the requirements of the software.When and only when the requirements are fully completed, one proceeds to design. The software in question is designed and a "blueprint" is drawn for implementers (coders) to follow — this design should be a plan for implementing the requirements given. When and only when the design is fully completed, an implementation of that design is made by coders. Towards the later stages of this implementation phase, disparate software components produced by different teams are integrated.

After the implementation and integration phases are complete, the software product is tested and debugged; any faults introduced in earlier phases are removed here. Then the software product is installed, and later maintained to introduce new functionality and remove bugs.

Thus the waterfall model maintains that one should move to a phase only when its preceding phase is completed and perfected. Phases of development in the waterfall model are thus discrete, and there is no jumping back and forth or overlap between them.

Typically Waterfall model is chosen when requirements , development environment and technology are well known and is more or less permanent



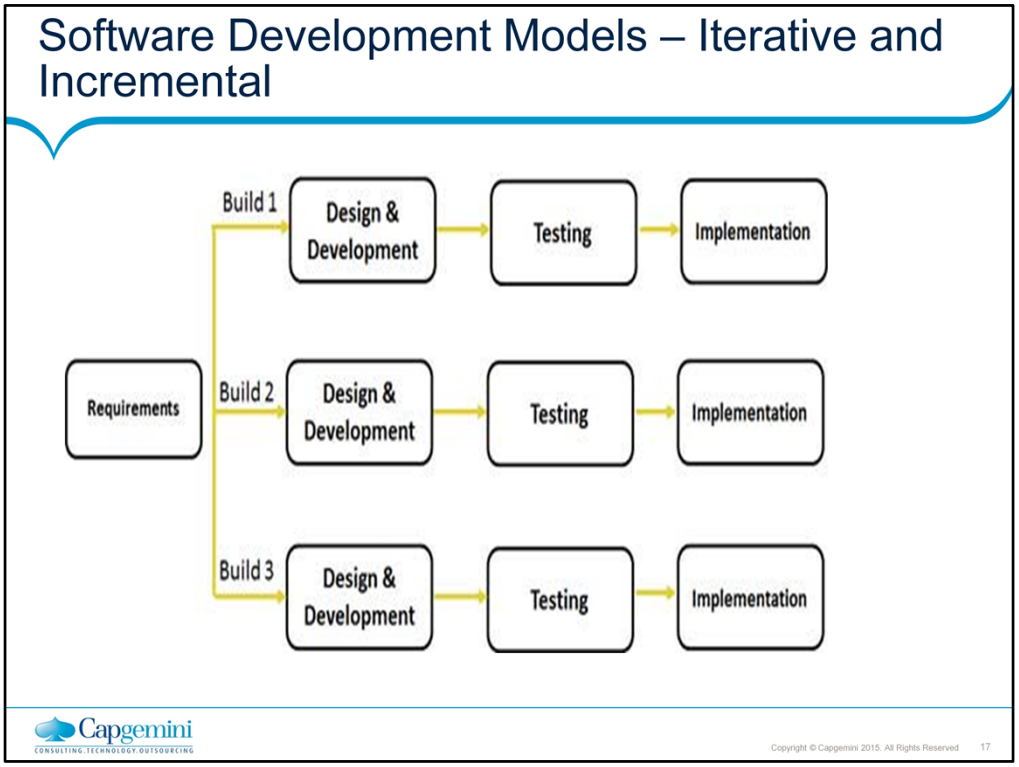
It is also known as Verification and Validation model .

This is a SDLC model which emphasizes the verification and validation of the product. The quality assurance activities are performed in the each phase of Software Testing Life Cycle phase. Based on the requirement document both development team and testing team start their activities in parallel . The developer team started working on the design and after completion on design start actual implementation . The testing team in parallel starts working on test planning, test case writing, test scripting..

Main focus in V mode l is proactive defect tracking and avoiding the downward flow of defects . However though this model helps in planning for Verification and Validation in early stages of product development , it cannot handle the following

- concurrent events
- dynamic changes in requirements
- risk analysis and feedback to the previous iteration

V-Model is ideal for projects where requirements are well known upfront , needs very high level of reliability and is of small size



The iterative and incremental model (IID) method of s/w development is to build the s/w in a incremental way . Every increment will involve analyzing requirements , design , code , test , deploy iteratively adding a little more to the product until the whole product is finished.

The basic idea behind iterative enhancement is to develop a system incrementally, allowing the development team to take advantage of what was being learned during the development of earlier, incremental, deliverable versions of the system. Learning comes from both the development and use of the system, where possible.
Key steps in the process were to start with a simple implementation of a subset of the software requirements and iteratively enhance the evolving sequence of versions until the full system is implemented. At each iteration, design modifications are made and new functional capabilities are added.

The goal for the design and implementation of any iteration is to be simple, straightforward, and modular, supporting redesign at that stage

Note : Fixing defects identified is NOT an iteration. Every iteration has a concrete deliverable and undergoes all the phases . Requirement prioritization helps in deciding the deliverables in each iteration . For example we may deliver web interface in first iteration and mobile interface in second iteration

This model is suitable for projects having evolving requirements , need to get to market early . Lengthy schedule and new technologies .

Agile Modeling

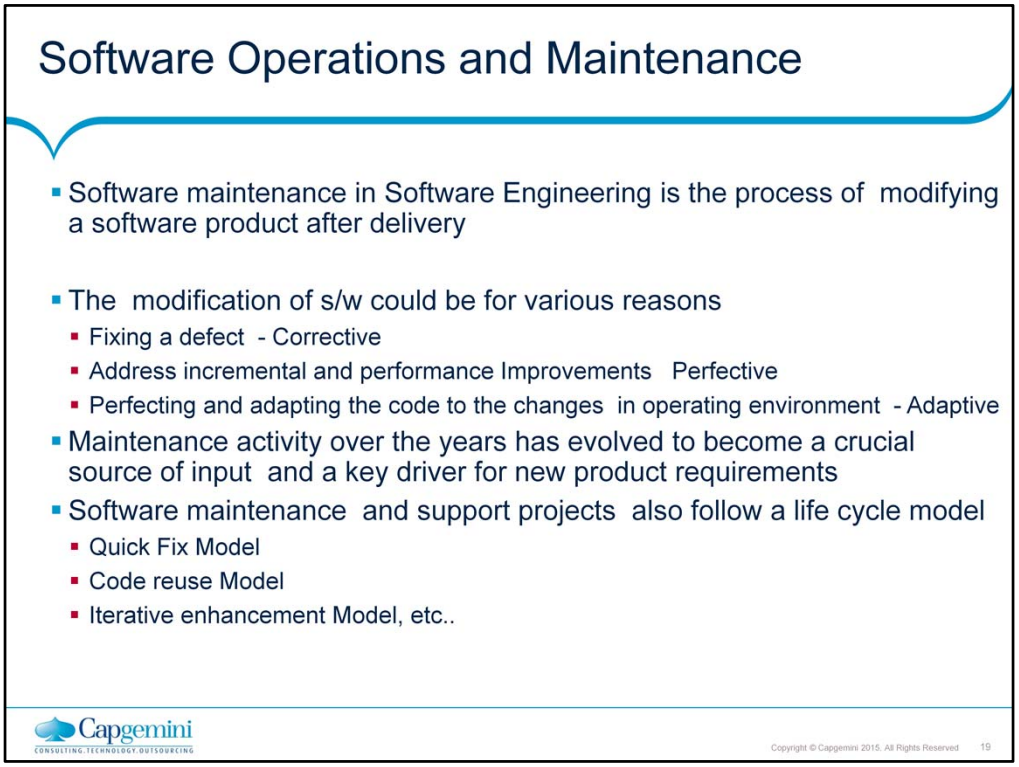
- It is a variant of the Incremental model
- It enables developing customized software with a process that helps in meeting current requirement as well as future through suitable adjustment
- The following principles that enable this methodology to be effective and light weight
  - **Communication**
    - Open communication between stakeholder and development team at every stage
  - **Simplicity**
    - The model emphasize the need to keep concepts and ideas in simple manner like simple tools , simple design , content etc..
  - **Feedback**
    - Model allows quick feedback from shareholders to ensure that things are on track
- Typically used when requirements are volatile and applications are time critical and the team is aware of the agile practices

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 18

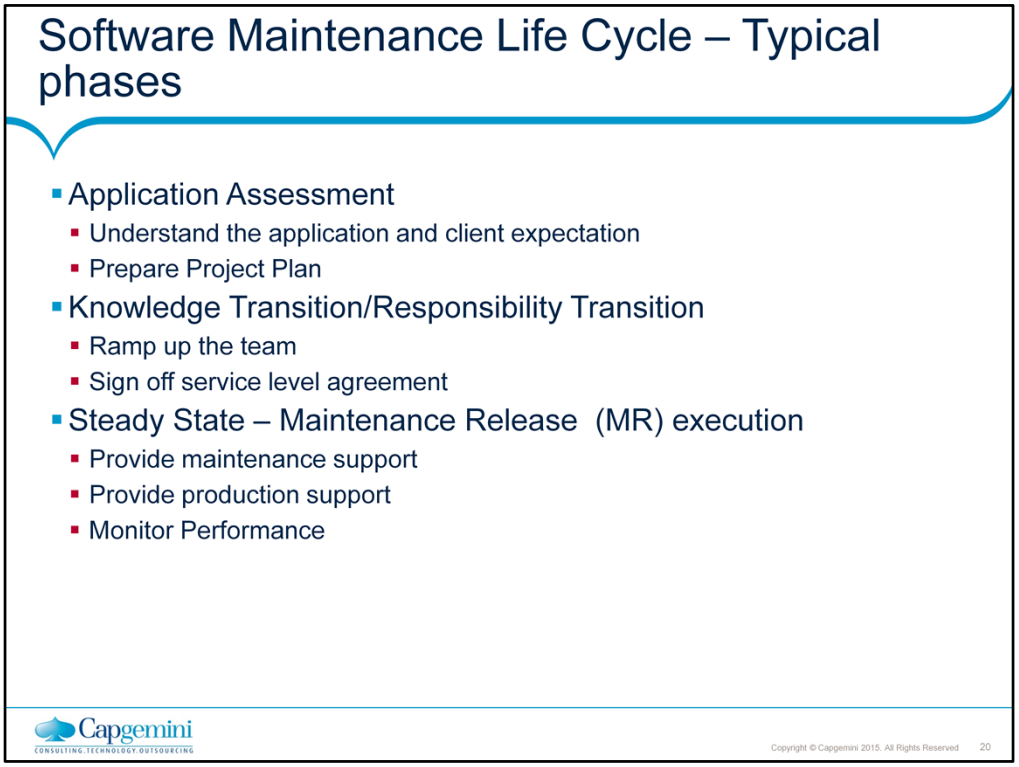
Agile Modeling enables developers to develop a customized software development process that actually meets their current development needs and is flexible enough to adjust in the future.

Agile Model Characteristics:
Active stakeholder involvement
Collective ownership
Keep it simple (content , design , tools)
Model in small increments
Apply modelling standards



Software Maintenance and support has over the years evolved into a crucial phase and also supports life cycle model . Some of the models are discussed in brief

- **Quick-Fix Model :** This model is adhoc and has a “firefighting approach” . That is fix the bug quickly when it occurs
- **Bohem’s Model :** Here the list of “approved changes” are identified by management using various cost effective strategies , which are then executed based on the budgets and resources .
- **Iterative Enhancement Model :** Originally proposed for developmental model later extended to maintenance as well . Follows the same iterative methods as that of development. Implementation of the changes to the system are done in an iterative way throughout the life of the system
- **Reuse Oriented Model :** This model is based on the principle that maintenance is an activity based on reuse of existing components Has the following activity
    - Identify parts of the old system which are candidates for reuse
    - Understand the items
    - Modify the items based on new requirements
    - Integrate the modified parts into the new system



Application Assessment : Here the application is thoroughly studied in terms of functionality, quality, volatility , workflows, user satisfaction (As-is) , along with customer future goals and expectation . Based on the assessment a high level project plan is prepared .

Knowledge Transition : Here a detailed plan is drawn to transit all the necessary information from earlier vendor of the customer including Documents , lessons learnt , service status and expectation etc. Transition from earlier vendor is desirable , if not possible both new vendor and the customer work together to ensure that the new vendor is effective enough to take over the projects/services etc . A detailed plan is drawn to ramp up the resources .

Execution : Here application maintenance team and production support team perform tasks to ensure the application is up and running without any defects . Requirements are consolidated , analyzed , estimated , coded , tested and deployed as per the *SLA* , against MR (Maintenance Request) .

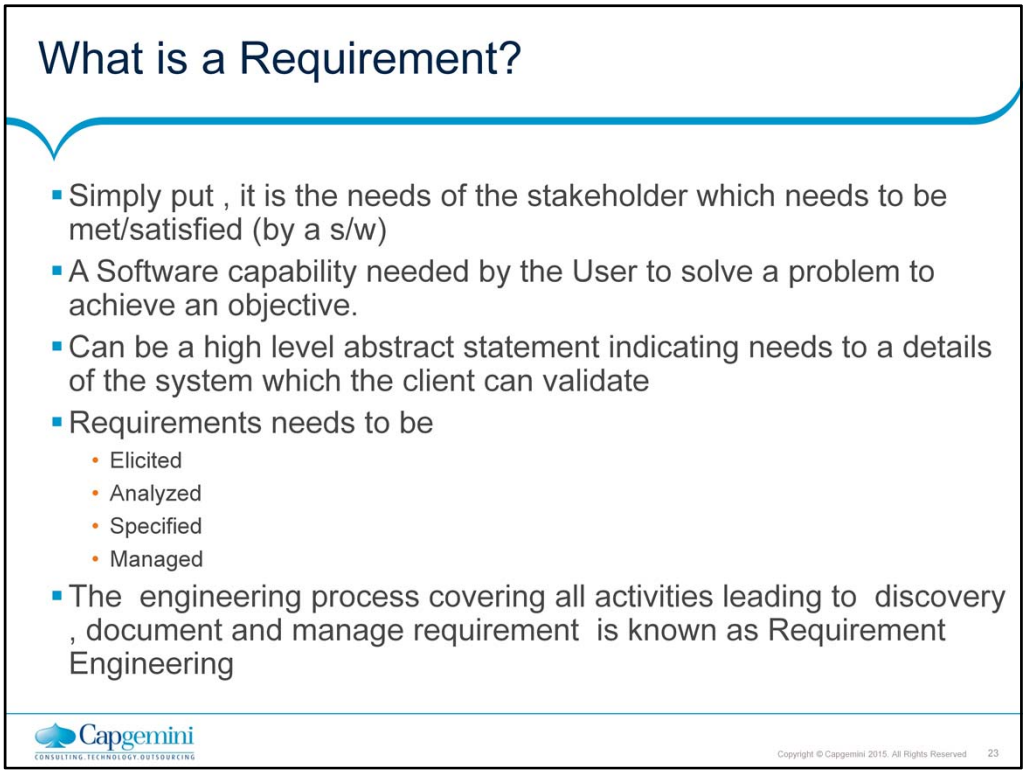
Selection of life cycle and support

- Based on nature of project
  - Clarity of requirements
  - Priority of implementation
  - Need to address change management
  - Need for prototypes
- Organization Support
  - QMS Procedure /guidelines for performing various activities
  - Templates/ forms/checklist/metrics for tracking , measuring and analysing various activities and taking corrective action
  - Tools for automating and improvement
    - SVN /TFS (for CM) (Recommended)

Note : In some cases we may use tools/templates suggested by clients

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

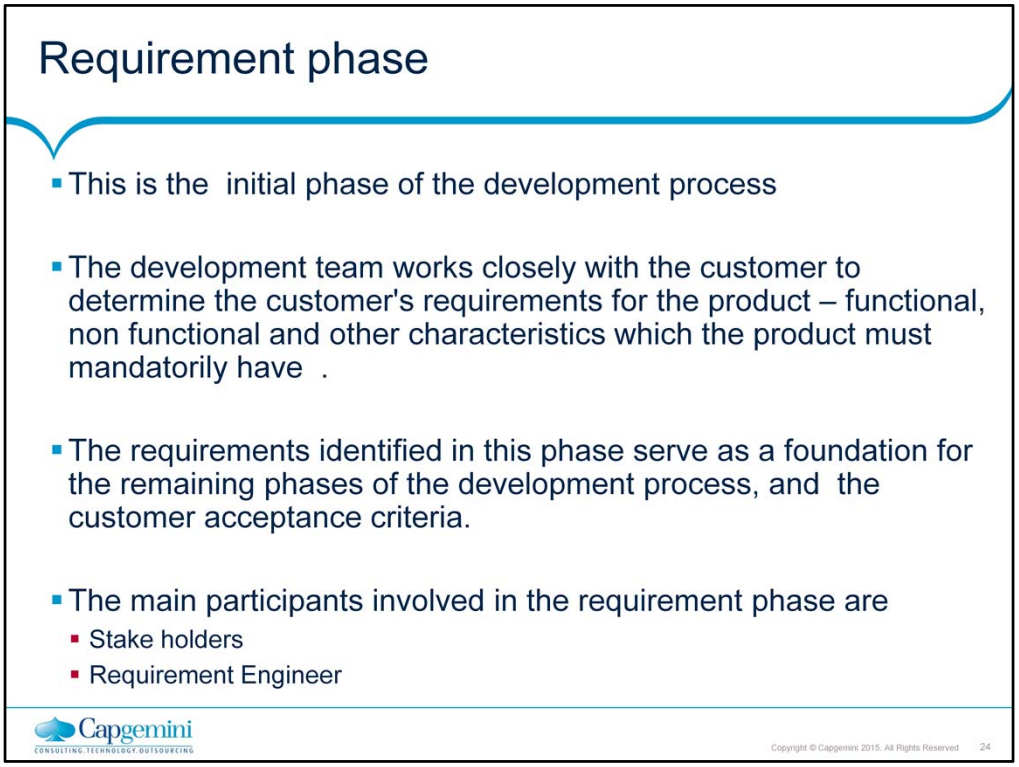
Introduction to Requirements Phase



A requirement is a capability or condition to which the system must conform. Software requirements provide a “black box” definition of the system. They define only those externally observable “What’s” of the system, not the “How’s.”

Requirements are very important for any project, or sub-section of a project, because they define what will be built, hence requires a rigorous engineering process, , hence the term Requirement engineering .

Requirement engineering is a continuous activity throughout the lifetime of a software as requirements are subject to change . New requirements needs to be elucidated existing requirements revamped etc.



Stakeholders are individuals who affect or are affected by the software product
They have some influence over the software , in terms of requirements
Stakeholders can be categorized as

Acquirers of the software (both management and users)
Suppliers of the software (individuals and team , management)
Others (Sales , Legal teams , other internal teams)

RE who are also known as requirements engineer, business analyst, system analyst, product manager, or simply analyst
RA's primary responsibility is to gather, analyze, document and validate the needs of the project stakeholders.They help to determine the difference between what customers say they want and what they really need

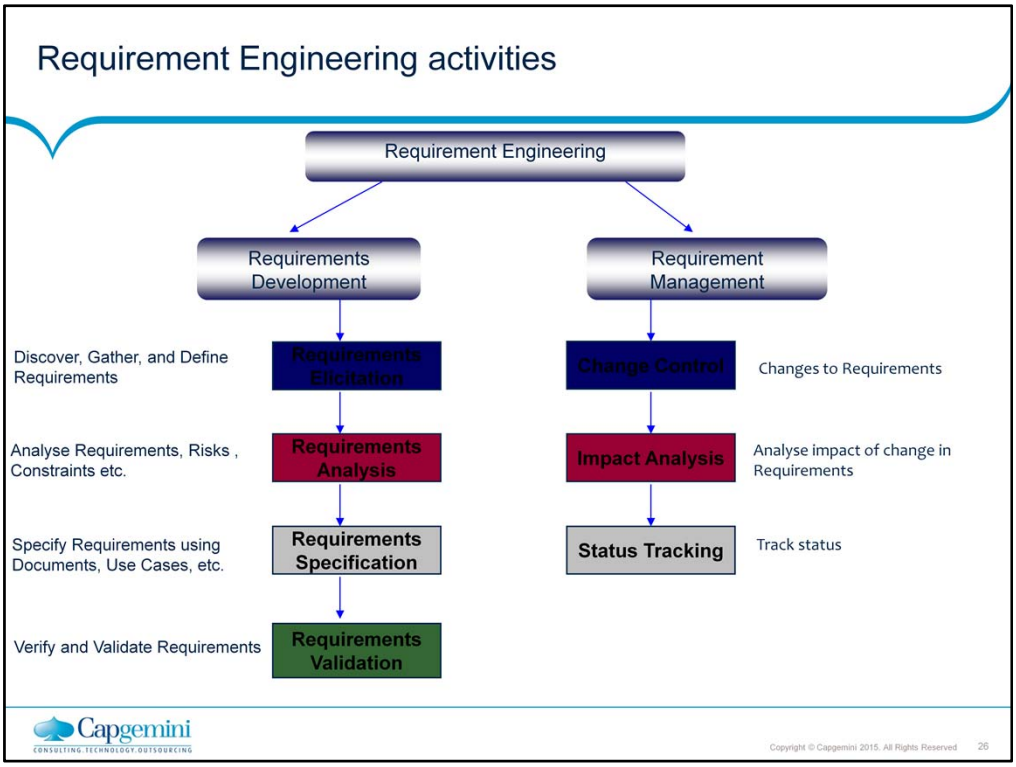
Identifying and considering the needs of all of the different stakeholders can help prevent requirements from being overlooked.
Requirement s can be classified under two categories :
Functional : Requirements what the system should do or provide for users .They can include all the business processes /funcionality, reports and queries and details of data to be stored and managed .
Non Functional : Non-functional requirements are constraints, targets or control mechanisms for the new system. They describe how, how well the system should e provide services like response time , ease of use-usability , security, recoverability etc.

Need for good requirements

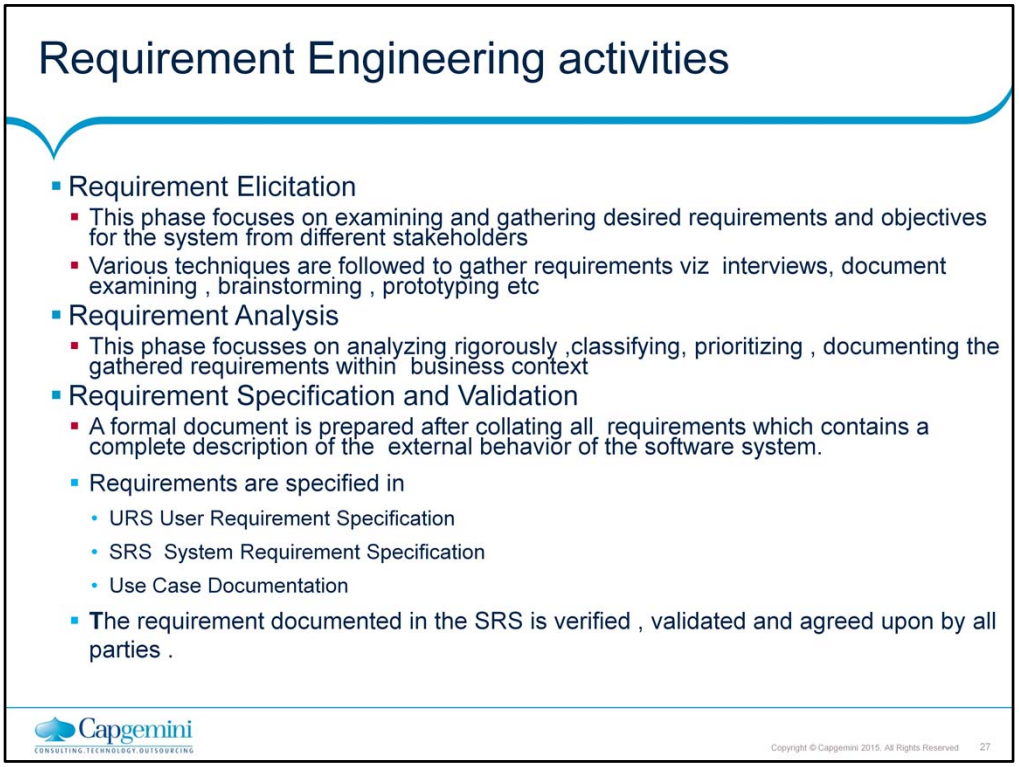
- Requirement Problems are the single No.1 reason for projects failing over
  - Schedule
  - Budget
  - Scope
  - Quality
  - And even getting Cancelled!!
- Reworking requirements cost 40-50% of project effort
- Many problems found during design, testing, or operation of a system are the result of incorrect, incomplete, or missing requirements

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 25



Requirements Engineering = Requirements Development + Requirements Management



Requirements Specifications include

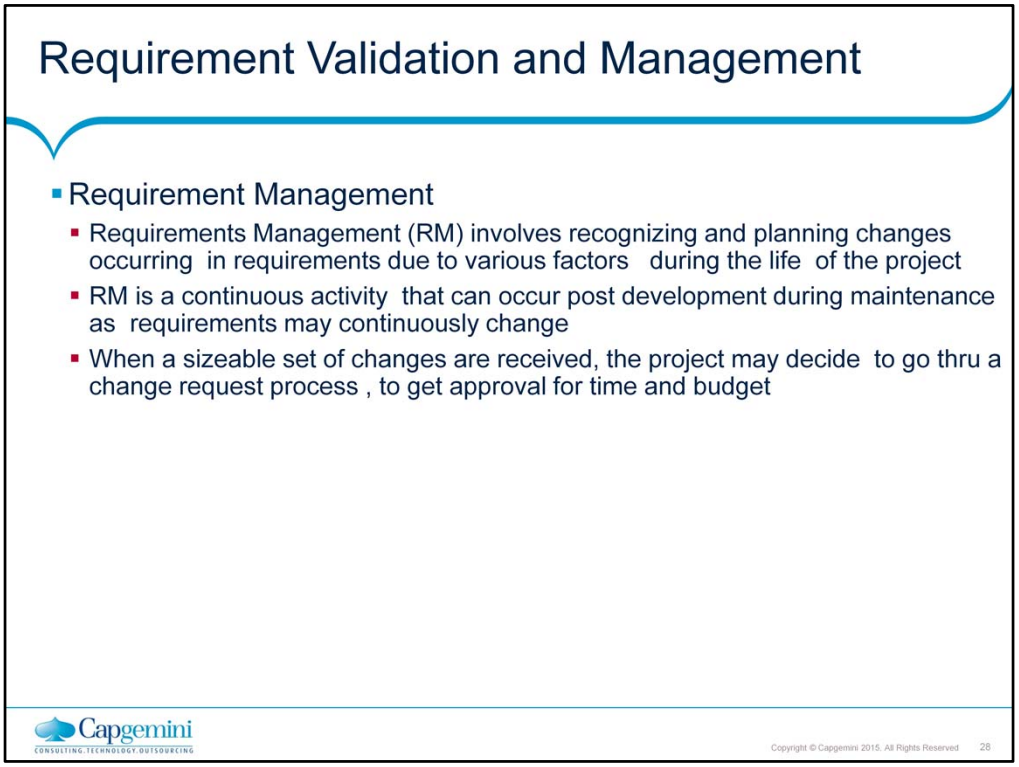
- What is in scope and out of scope
- Related or referenced documents (Customer supplied artifacts and materials)
- Requirement providers and stakeholders of the project
- Deliverables & delivery dates
- Risks and assumptions
- Current and proposed business system
- Acceptance criteria and Customer CTQs
- Functional and non functional requirements
- Limitations and constraints

URS : User Requirement Specification
Typically written prior to the SRS, based on the user's experience and expectations, with inputs from stakeholders

SRS : System Requirement Specification
This information includes detailed descriptions of the operations performed by each screen, the data that can be entered into the system , work-flows performed by the system and system reports or other outputs,
. An SRS also specifies who can enter data into the system as well as how the system meets regulatory requirements that are applicable to the specific system.

Use Case Documents : The document and diagrams together forms the UCD .
Typically done when the approach is Use case modelling

QMS provides templates for creating specification document

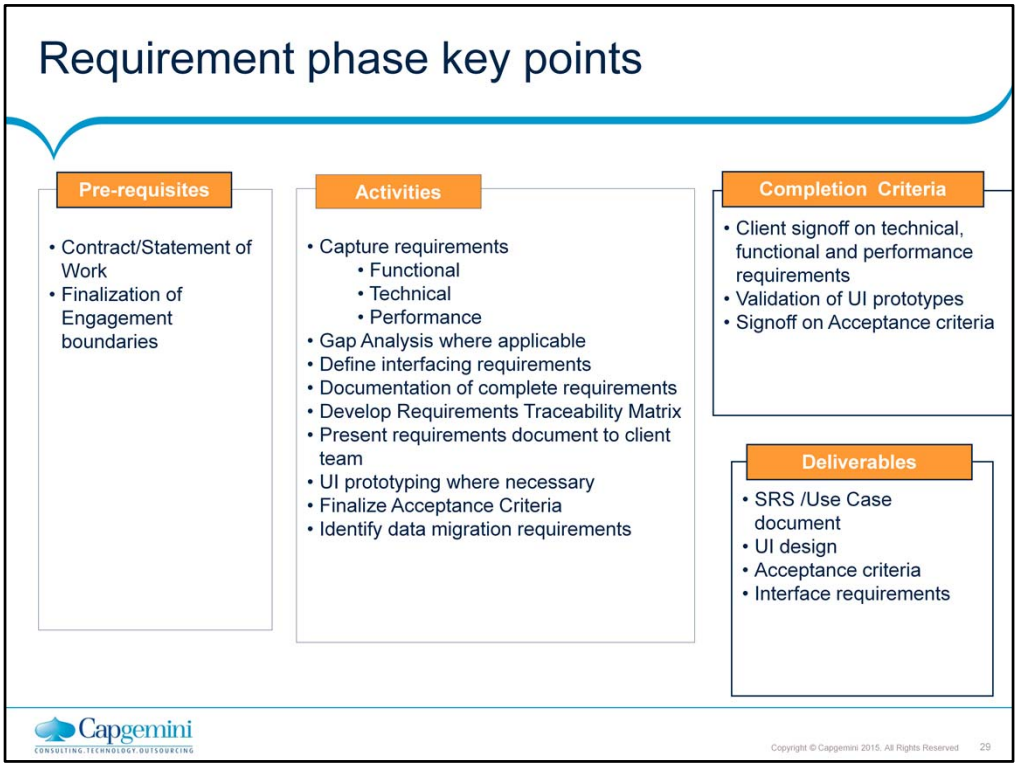


RM phase controls and tracks the changes of agreed requirements, relationships between requirements, and dependencies between the various produced during software engineering process

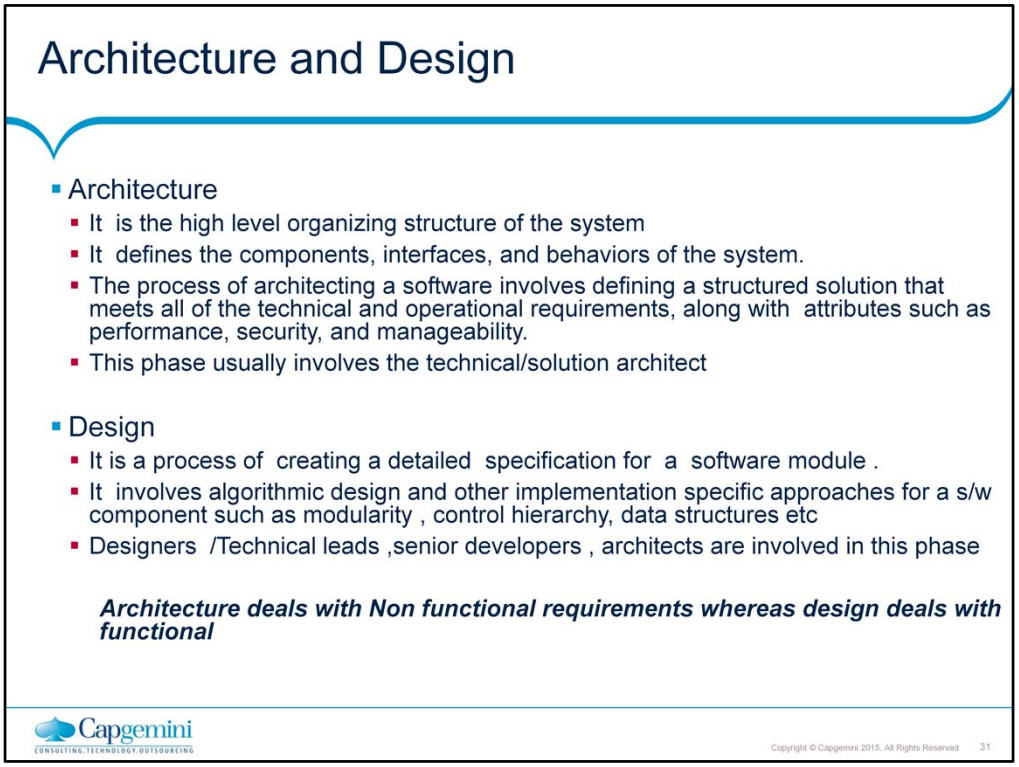
Requirement may change due to various reasons

- A bug
- Technology change
- Change in business

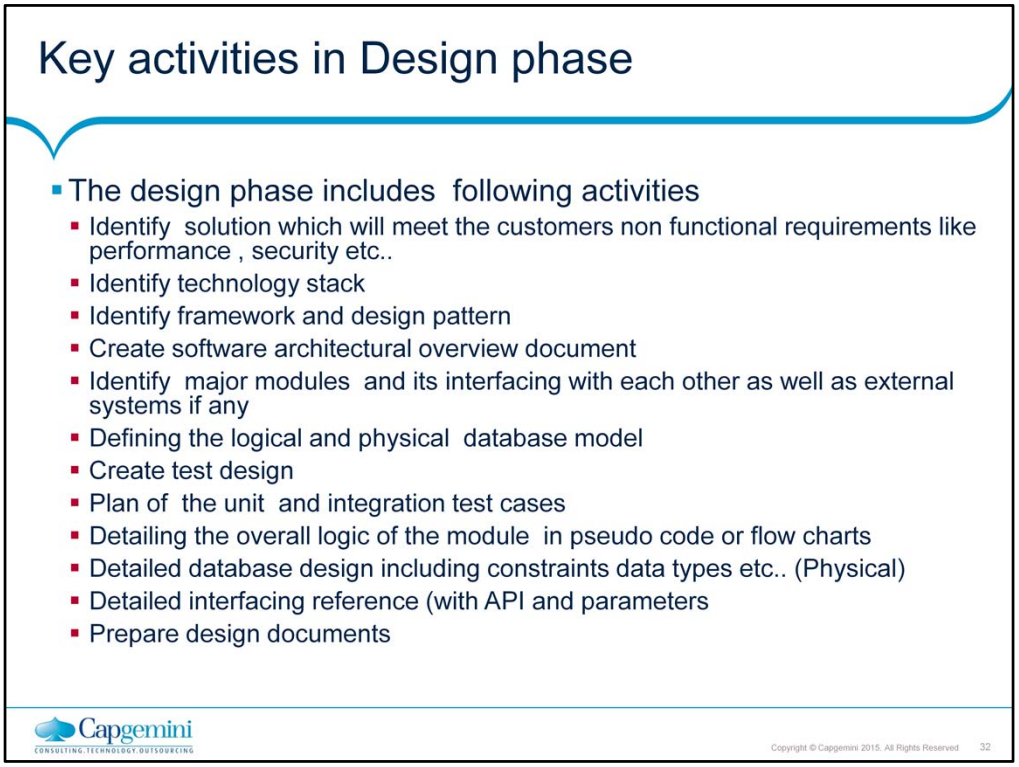
When sizable requirement changes are received the changes are incorporated via a change management process .


Requirement phase key points
Pre-requisites
• Contract/Statement of Work
• Finalization of Engagement boundaries
Activities
• Capture requirements
• Functional
• Technical
• Performance
• Gap Analysis where applicable
• Define interfacing requirements
• Documentation of complete requirements
• Develop Requirements Traceability Matrix
• Present requirements document to client team
• UI prototyping where necessary
• Finalize Acceptance Criteria
• Identify data migration requirements
Completion Criteria
• Client signoff on technical, functional and performance requirements
• Validation of UI prototypes
• Signoff on Acceptance criteria
Deliverables
• SRS /Use Case document
• UI design
• Acceptance criteria
• Interface requirements
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
29

Introduction to Design Phase



The architecture of a system is its 'skeleton'. It's the highest level of abstraction of a system. What kind of data storage is present, how do modules interact with each other, what recovery systems are in place.
Software design is about designing the individual modules / components. What are the responsibilities, functions, of module x? Of class Y? What can it do, and what not? What design patterns can be used?
So in short, Software architecture is more about the design of the entire system, while software design emphasizes on module / component / class level

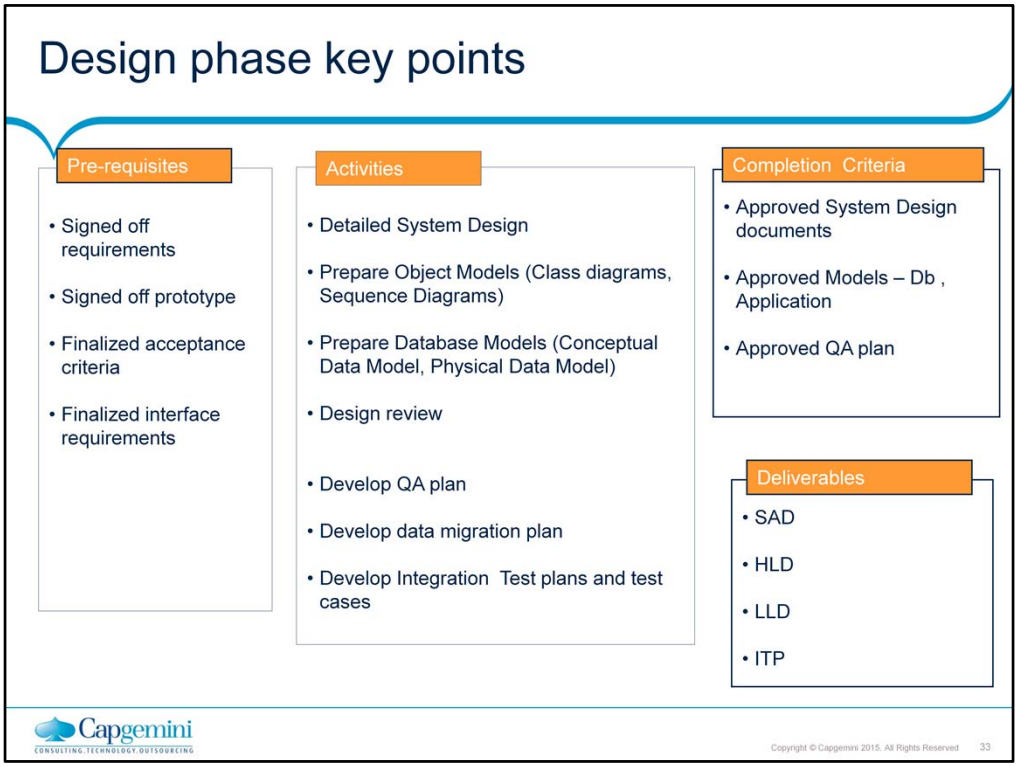


Architecture constitutes of the following key activities:

- Solution space for “non-functional requirements”
- Decision on Technology Stack
- Framework requirements definition and solution
- Critical decisions for some risky "functional" requirements

Architecture activities are delivered by the Technical Architect and supported by the Design lead **Design** is mainly focused on modeling the functional aspects of an application.
Solution space for “functional requirements” based on defined architecture
Design Pattern choice
Application design
Logical ER Data Model (entities, attributes, relationships)
UML Models - Class, Sequence , Activity etc
Analysis Model (domain entities, control and boundary classes, their functional attributes and associations)
Additional UML diagrams (as needed)
Data types of attributes
Additional classes, attributes for technical implementation (ex. primary key)

Design activities are delivered by the Design Lead and the Designer .Design Lead is a key role and which acts as a communicator between the architect and designers


Design phase key points
Pre-requisites
• Signed off requirements
• Signed off prototype
• Finalized acceptance criteria
• Finalized interface requirements
Activities
• Detailed System Design
• Prepare Object Models (Class diagrams, Sequence Diagrams)
• Prepare Database Models (Conceptual Data Model, Physical Data Model)
• Design review
• Develop QA plan
• Develop data migration plan
• Develop Integration Test plans and test cases
Completion Criteria
• Approved System Design documents
• Approved Models – Db , Application
• Approved QA plan
Deliverables
• SAD
• HLD
• LLD
• ITP
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
33

Introduction to Construction Phase

Construction phase

- Also known as implementation phase
- Main objective of this phase is to translate the software design into code , each component identified in design is implemented as a program module following coding guidelines
- Each module in this phase is reviewed and unit tested to determine correct working (White Box testing)
- Unit tested code are then integrated in a planned and a phased manner .
- In each integration step the partially integrated system is tested

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

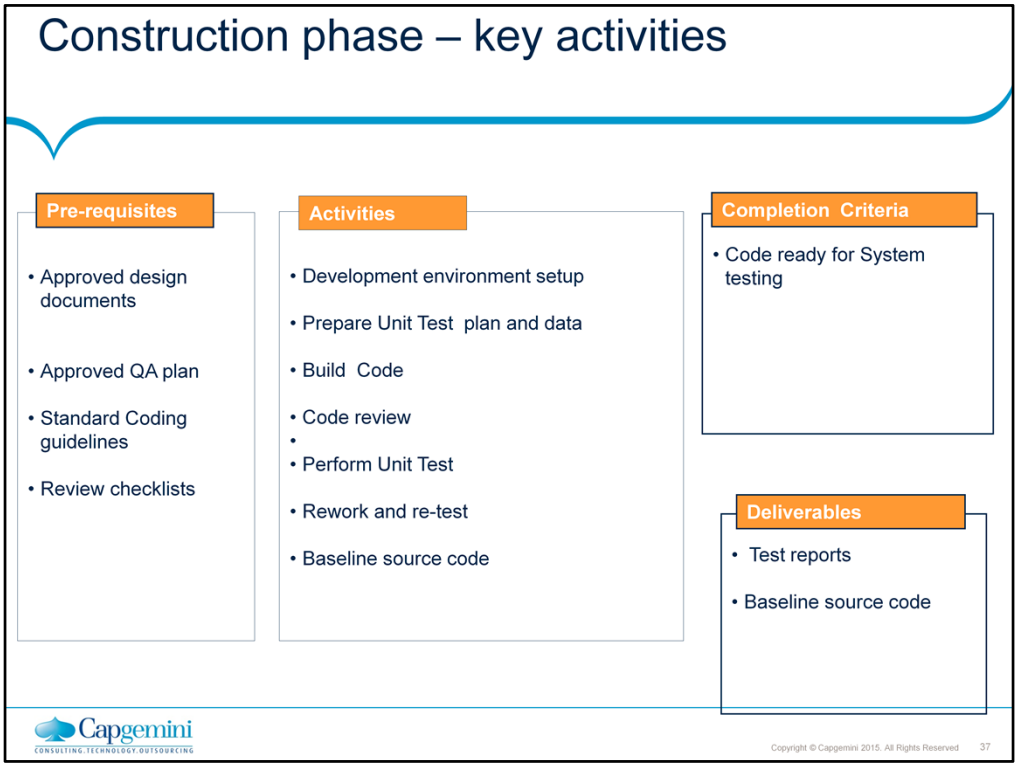
 35

Construction phase

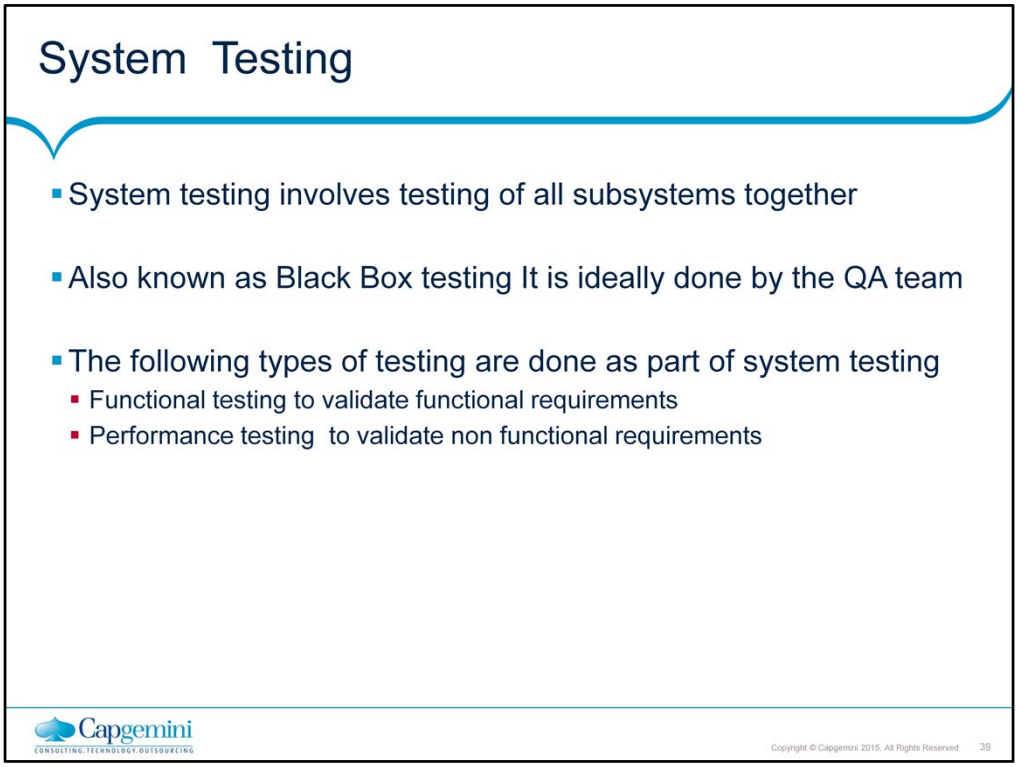
- In addition to the major activities the following activities are also carried out as well
  - Prepare unit test plan and test case
  - Prepare unit test data
  - Setup coding guidelines
  - Setup the environment for Configuration Management as per CM guidelines
  - Provide suitable environment for base lining code and continuous integration
  - Defect reporting and fixing
- The main role players in this phase are
  - Developers
  - Team Leads

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 36


Construction phase – key activities
Pre-requisites
• Approved design documents
• Approved QA plan
• Standard Coding guidelines
• Review checklists
Activities
• Development environment setup
• Prepare Unit Test plan and data
• Build Code
• Code review
•
• Perform Unit Test
• Rework and re-test
• Baseline source code
Completion Criteria
• Code ready for System testing
Deliverables
• Test reports
• Baseline source code
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
37

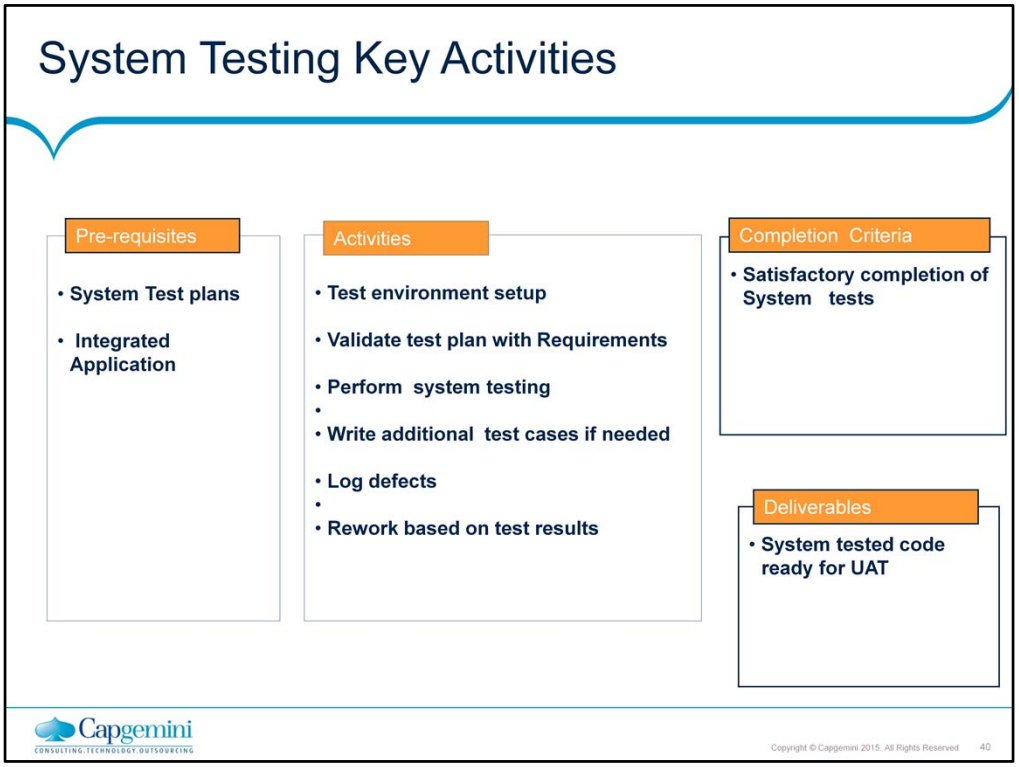
Introduction to Testing Phase



Goal of functional testing is to test the functionality of the system . The test cases are written from the requirement documents by the QA team as a parallel activity once the requirements are frozen . The system is treated as a black box (implementation independent) .

Goal of the performance testing is to validate the non functional requirement of the system (captured during requirements) , In this kind of testing the system is pushed to its limits to see how it behaves . Some of the performance tests

- **Stress testing** to test stress limits of system (maximum # of users, peak demands etc)
- **Volume testing** to test large volume of data
- **Security Testing** to test if the system behavior on security violation
- **Recovery Testing** to test system's response to loss of data and presence of errors
- **Usability testing** to test the ease of Use of the system

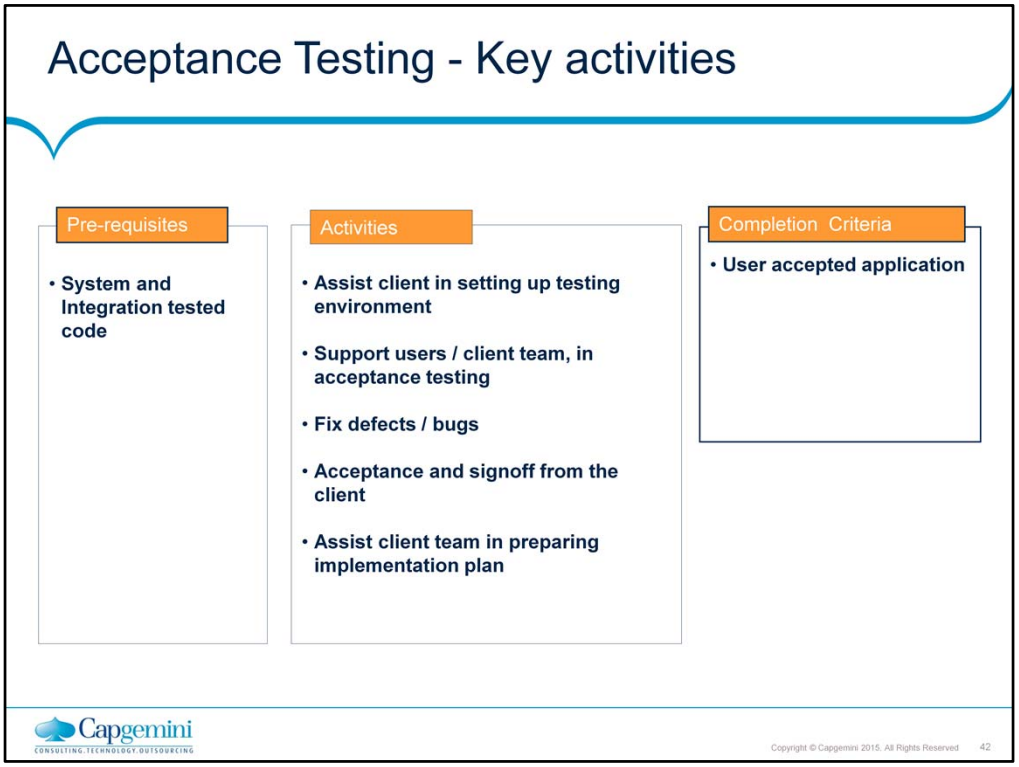

System Testing Key Activities
Pre-requisites
• System Test plans
• Integrated Application
Activities
• Test environment setup
• Validate test plan with Requirements
• Perform system testing
•
• Write additional test cases if needed
• Log defects
•
• Rework based on test results
Completion Criteria
• Satisfactory completion of System tests
Deliverables
• System tested code ready for UAT
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
40

Acceptance Testing

- Usually done at the client location by the client , after the findings of System testing is fixed
- Focus of Acceptance test is to evaluate the system's compliance with the business requirements and assess readiness for delivery.
- Acceptance Testing is done in two ways
  - Alpha Testing or Internal Acceptance Testing
    - **done by s/w vendors**
  - Beta Testing or User Acceptance testing
    - **Done by end users of customers or customer's customer**
- Outcome of the acceptance testing will enable the user, customers or other authorized entity to determine whether or not to accept the system.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 41


Acceptance Testing - Key activities
Pre-requisites
• System and Integration tested code
Activities
• Assist client in setting up testing environment
• Support users / client team, in acceptance testing
• Fix defects / bugs
• Acceptance and signoff from the client
• Assist client team in preparing implementation plan
Completion Criteria
• User accepted application
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
42

Post Acceptance phase

- After successful acceptance testing plans are made to move the application to the “live environment”
- Activities like knowledge transfer , end user training , project signoff are also done .
- Once when the customers starts using the developed system the maintenance team supports and monitors the system to resolve errors and performance .

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Software Reviews

Reviews and CM process are processes which will be used in all the phases of Software development . These are umbrella process .

Reviews – What

- An assessment of a work product created during the software engineering process
- Ensure completeness and consistency of the work product
- Identify needed improvements
- It is a Quality Assurance mechanism to identify discrepancy /deviation from the accepted standards
- Goal of Review
  - To detect and eliminate defects early, effectively and before delivering the product to the customer

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 45

Reviews – Why

- Intermediate software products which are not testable as standalone units
- Can find errors not possible through testing
  - E.g., Maintainability: Comments, Consistency, Standards
- Are proactive measure to find out 60-80 % of defects
- **Reduce Rework Effort and Improve Schedule adherence**
- Enables Quantitative Quality Assessment of any work product

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

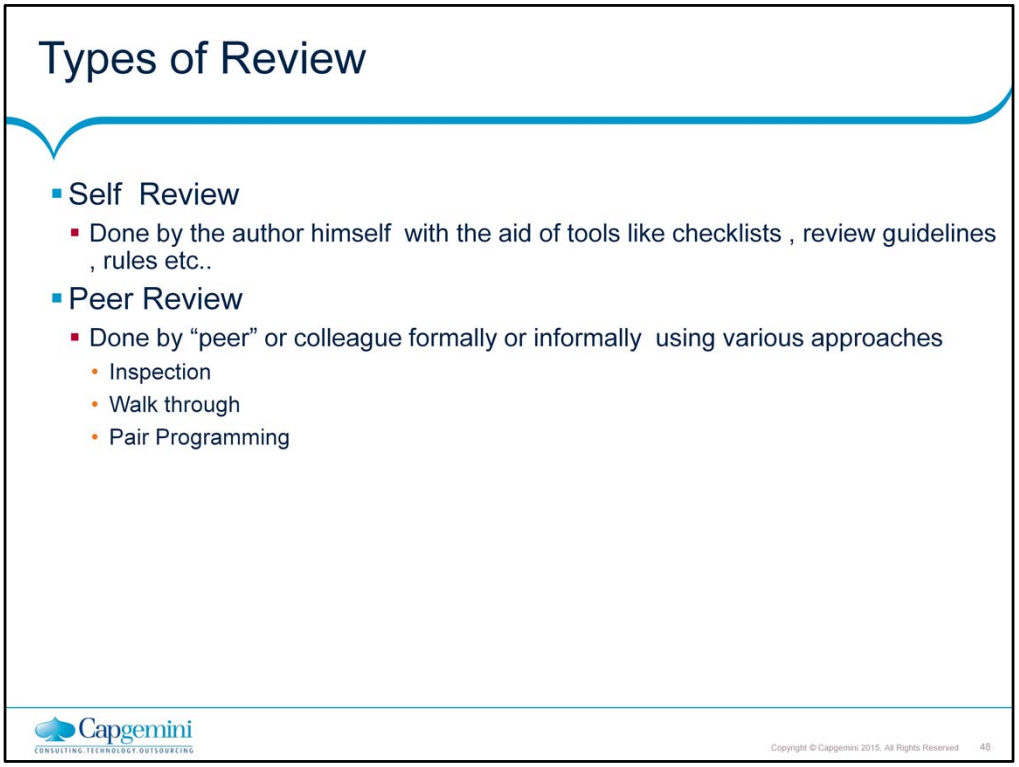
 46

Software Reviews – When , where

- Can happen in all phases of SDLC
- All artifacts can go for reviews
  - Proposals, contracts, statement of work
  - All project work products - Plans, Configuration Mgmt, Test Plans
  - Deliverable and non deliverable work products
  - Software(e.g.: source code) and non software work products (e.g.: documents, test data, etc.)
- Process descriptions
- Policies, brochures, reports, guidelines, standards, training material where required

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 47



Inspection – It is a more systematic and rigorous type of peer review. Inspections are more effective at finding defects than are informal reviews. In inspection reviewer drives the review process .

Walkthrough – It is an informal review because the work product's author describes it to some colleagues and asks for suggestions. Walkthroughs are informal because they typically do not follow a defined procedure, do not specify exit criteria, require no management reporting, and generate no metrics.

Pair Programming – In Pair Programming, two developers work together on the same program at a single workstation and continuously reviewing their work.

Review Process

- Input
  - Work Product , Specifications, Checklists, Guidelines, Historical Data
- Process
  - Prepare for Review
  - Conduct Reviews
  - Analyze Deviations
  - Correct Defects
- Output
  - Review Form, reviewed work product,

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 49

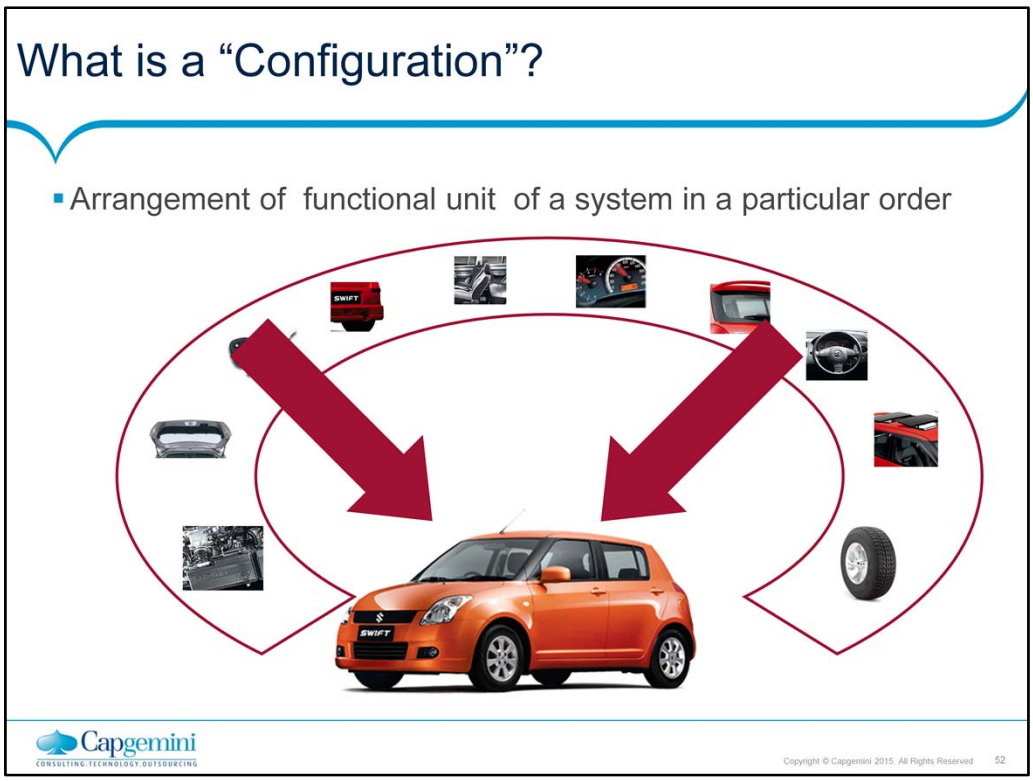
Introduction to Configuration Management Process

Agenda

- What is SCM?
- Why SCM?
- Elements of SCM
- Change Management
- Information on SCM tools

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 51



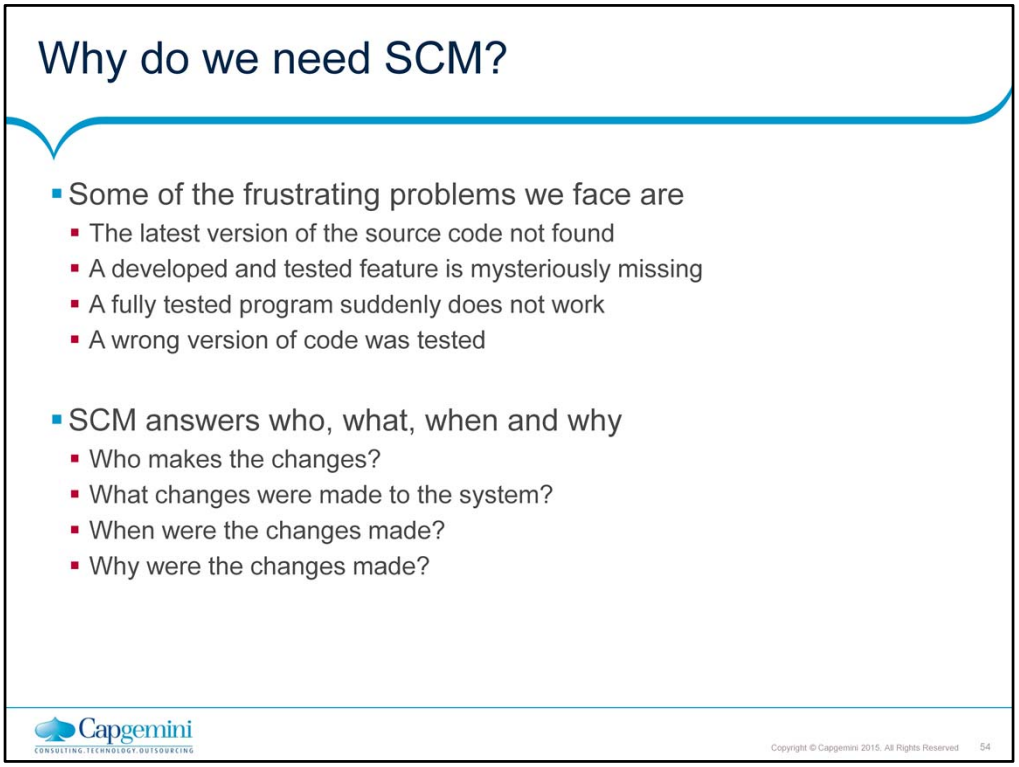
A configuration is an arrangement of functional units according to their nature, number, and chief characteristics. Often, configuration pertains to the choice of hardware, software, firmware, and documentation. The configuration affects system function and performance.

What is Software Configuration Management?

- SCM is the overall management of a software project as it evolves into a software system.
- This includes managing , tracking, organizing, communicating, controlling modifications made in project including release plan
- Also includes the ability to control and manage change in a software project
- Configuration Managers Are responsible for planning the CM activities of their project
- The configuration details of the project are documented in the CMP (Configuration management Plan)

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 53



SCM is the process that defines how to control and manage change.
The need for an SCM process is acutely felt when there are many developers and many versions of the software. Suffice to say that in a complex scenario where bug fixing should happen on multiple production systems and enhancements must be continued on the main code base, SCM acts as the backbone which can make this happen.

Without configuration Management the following can happen

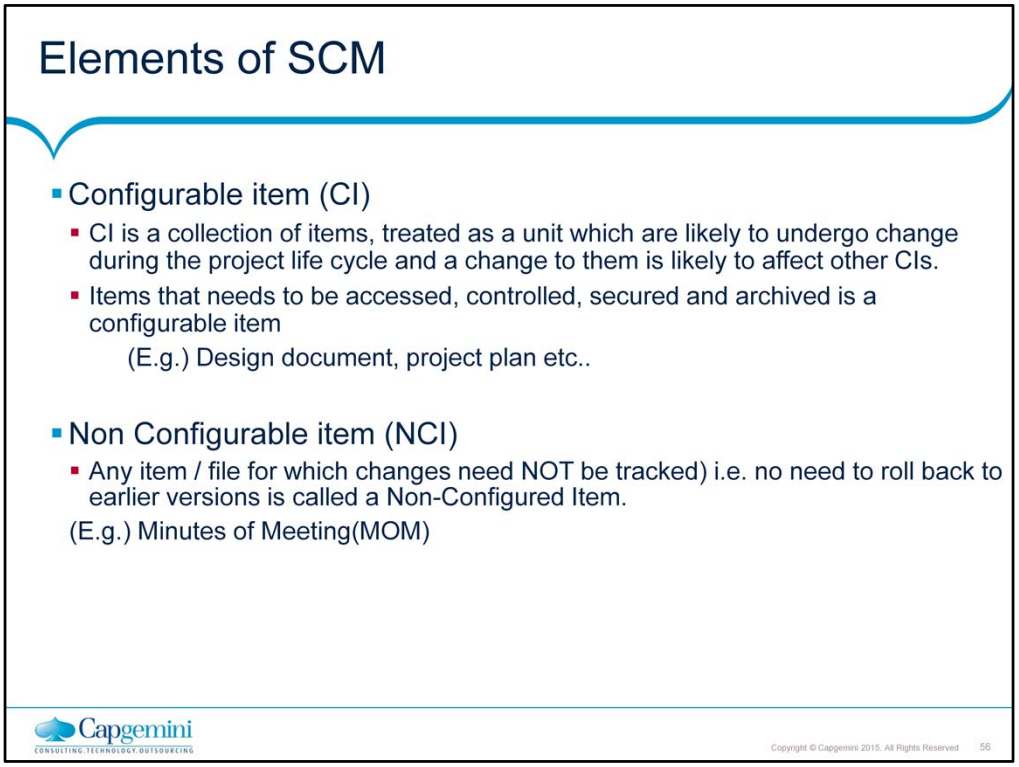
- Unorganized project items
- Confused naming conventions
- Review / Delivery of wrong version of code
- Development based on old version of specifications
- No proper access / privilege control; Unauthorized access to secure information
- Redundant file creation
- Change Management becomes ineffective

Elements of SCM

- Configuration identification
  - CI
  - NCI
- Configuration control (Elements)
  - Library Control
  - Access Control
  - Version Control
  - Establish Naming conventions
  - Establish Baselines
  - Branching, Merging and Labeling
- Change Management
- Auditing (Verification)

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 55



Version:

The term 'version' is used to define a stage in the evolution of a CI, for example versions of source code, etc.

Library Structure

- Controlled collection of software and related documentation designed to aid in
  - software development
  - use
  - Maintenance

Organized structure

- The folder structure is indicated in the CMP

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 57

Example of Libraries Structure

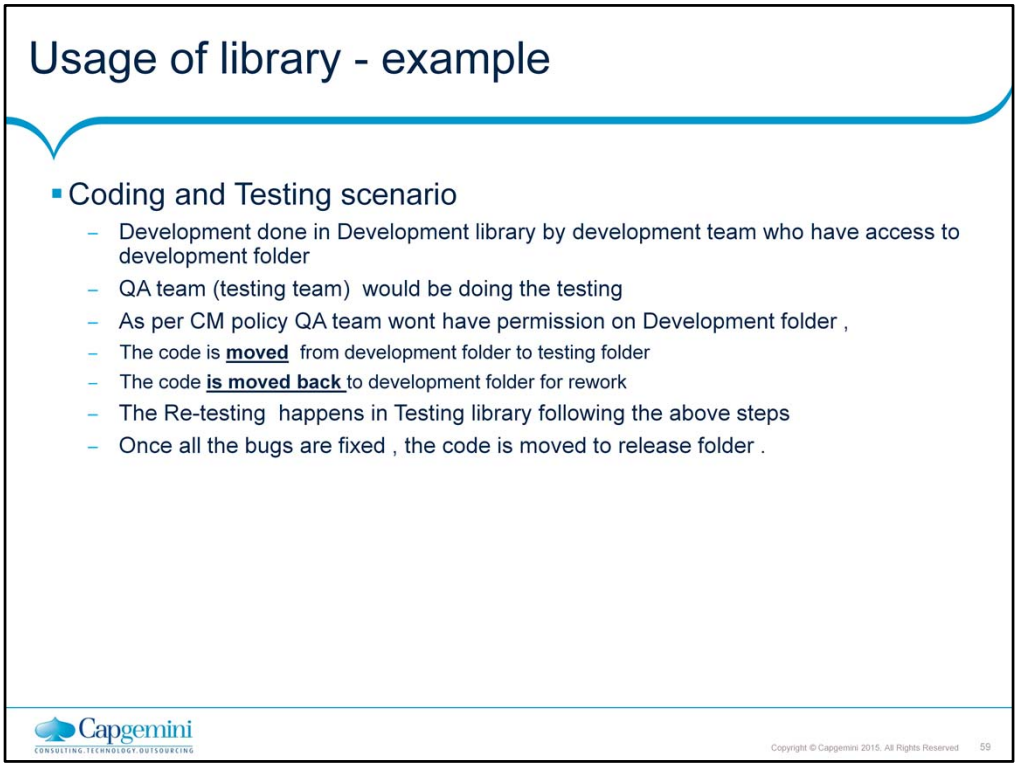
- Input Library
- Development Library
- Testing / Review Library
- Release / Delivery Library
- Template Library
- Project Management Library

Tip: The library (folder) can be created on need basis for the project. No thumb rule to create the same.

SDLC  **Folders**

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 58


Usage of library - example
Coding and Testing scenario
Development done in Development library by development team who have access to development folder
QA team (testing team) would be doing the testing
As per CM policy QA team wont have permission on Development folder ,
The code is moved from development folder to testing folder
The code is moved back to development folder for rework
The Re-testing happens in Testing library following the above steps
Once all the bugs are fixed , the code is moved to release folder .
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
59

Version Numbering

- A version number is a unique number or set of numbers assigned to a specific release of a software/hardware/firmware
- As updates and new editions of product are released, the version number will increase
- Version numbers are usually divided into sets of numbers, separated by decimal points
- Draft version has X . Y

X	Y
• Represents the Version number • Changed when there is a - CI is completely overhauled - Substantial Change	• Represents the revision number • Changed when there is a - No change in the overall structure and flow - Existing content is minimally

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 60

Naming Conventions

- Helps in easy identification
- The name may include
  - Project name/ Application name/ Request ID
  - Document/ Work product name
  - Version number/ Date (If manual configuration)
  - Status – Draft, review, approval etc.,

Sample Document	Sample Naming conventions
Management documents	Est_<Project name>_dd_mm_yy_<ver_ x.y>.doc, PP_<Project name>_<ver_x.y>.doc, SCH_<Project Name>_<ver_x.y>.xls/mpp
Source code	<Component name>_<ver_ x.y>.java, <Module name>_<component name>_<ver_ x.y>.java
Unit Test Plan	UTP_<Component name>_<ver_ x.y>.doc
Quality Records	C-rev_<component name>_<ver_ x.y>**.xls, C-rev_<Module name>_<component name>_<ver_ x.y>**.xls

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

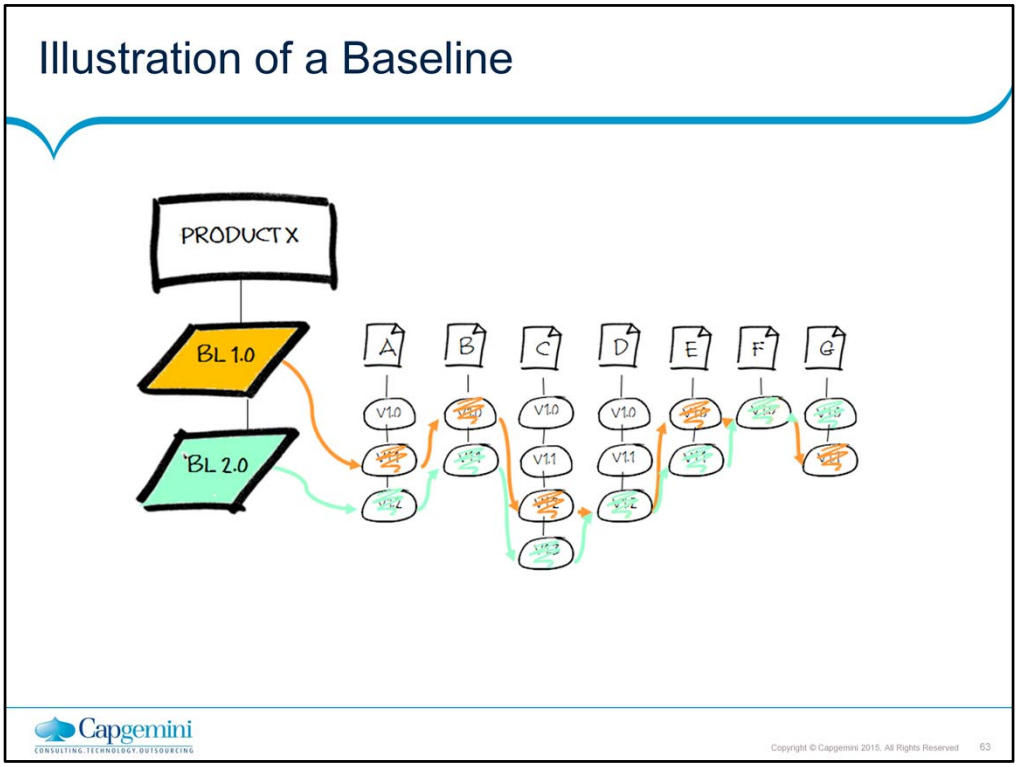
 61

Baselines

- Baseline
  - A 'baseline' is a CI that has been reviewed and agreed upon and is a basis for further development.
  - After base-lining all changes to CI are controlled through a formal change process (Such as Change Management and Configuration Control).
  - For example a reviewed and approved Project plan is used as a basis for execution of the project.
  - One baseline may have several work product , each having different version number
  - Baselining can be of many types – Input baseline , design baseline , code baseline etc

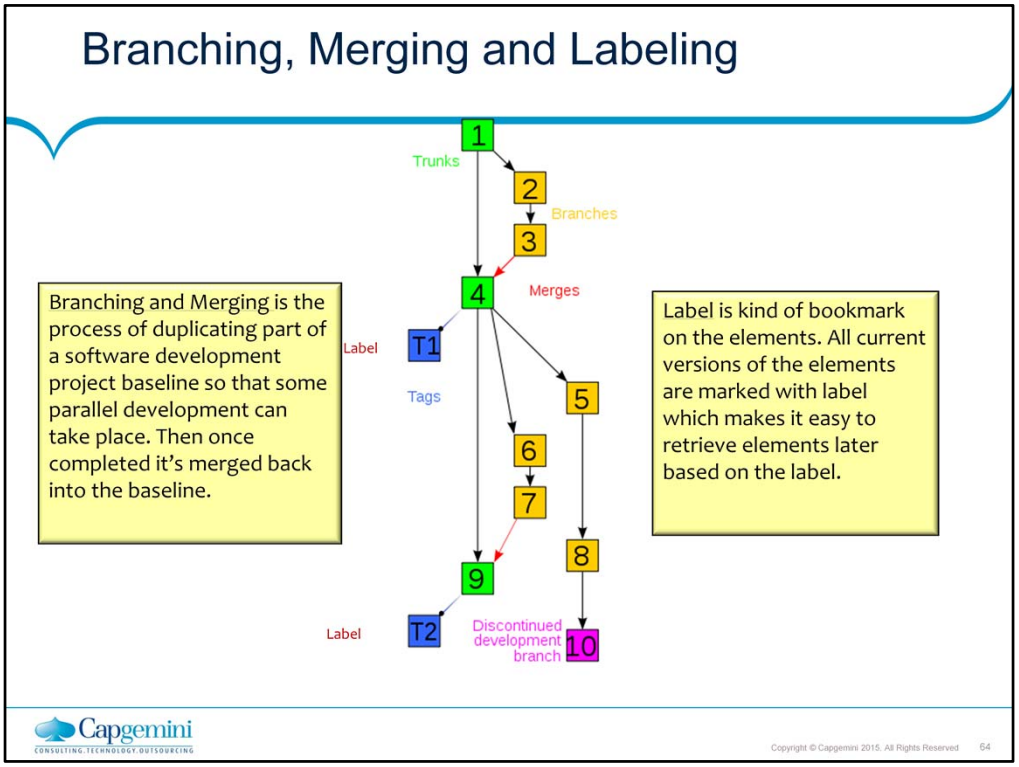
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 62



A baseline defines a set of files, each at a particular version. These need not be the latest (most recent) version. A baseline label uniquely identifies the configuration. Files may belong to one or more baselines.

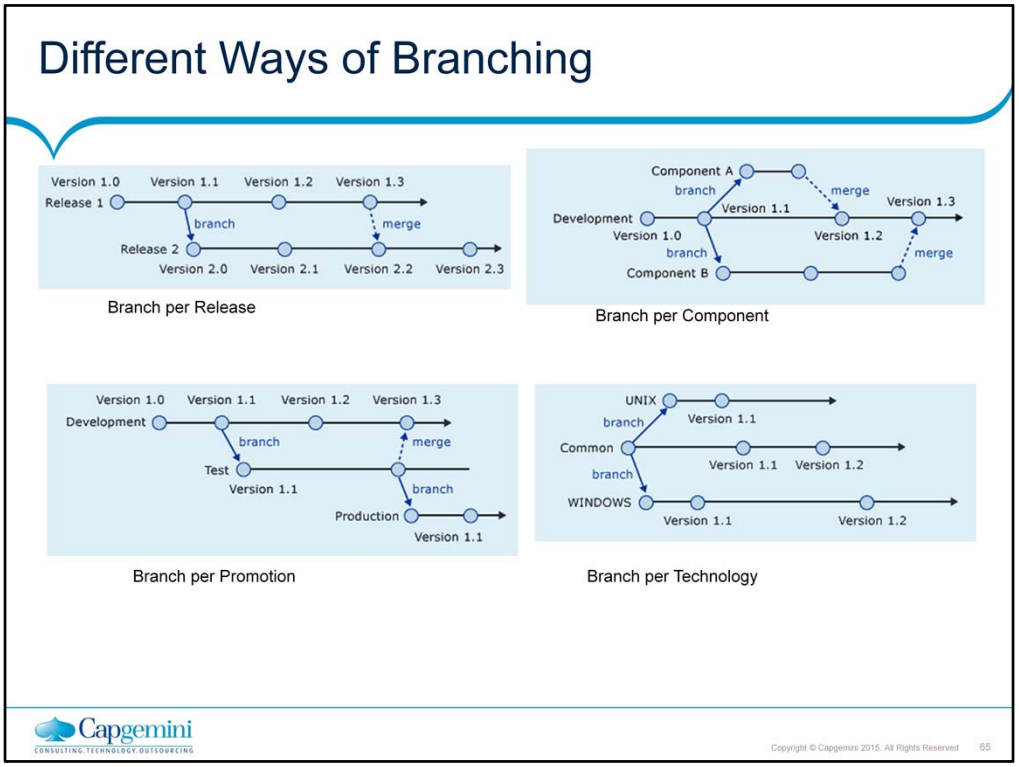
In the example of Figure 12 baseline BL1.0 is the first baseline recorded. It consists of seven artefacts, each at a unique revision number. For this example, assume that BL1 records the most recent versions of each artefact. As development progresses each artefact is modified as required (that is, some artefact are modified, some are not). At some time later another baseline is taken – BL2.0. In this case BL2.0 records the current latest revisions of each file. Notice that artefact F is unchanged, so F v1.0 is included in both baseline BL1.0 and BL2.0.
In general each successive baseline contains more recent versions of files (but not always).



Branching and Merging are two important aspects of version control. This concepts are extremely useful in parallel software development The two concepts are briefly explained below

Branch : It is a line of development that exists independently of another line, yet still shares a common history. For example assume we are developing a banking application for American customers. The same application can be used by Canadians with some customization. T he solution to this requirement can be achieved by creating a branch for the Canadian customers and incorporating the needed changes. Since the two branches are related , if any changes /bug fixes needed in both can be easily duplicated . The main line of development is called **trunk (shown in the diagram), whereas a branch is a side line of a development**

Merge : In simple terminologies a merge is basically "copying" the changes across branches . To quote an example , assume that we have started working on the next version of our product (version 4.0) . A critical bug and some minor customization is asked for . To accommodate we create a branch to incorporate change and deploy to the customer. Once the next release is ready we merge the branch completely so as to incorporate the changes done in the branch in the new version



Branch per Release
Every release is a new branch; common changes are merged between the releases. Branches are killed off only when the releases are no longer supported.

Branch per Component
Each architectural component of the system is a new, independent branch. Components are merged into the main branch as they are completed.

Branch per Promotion

Every tier is a permanent branch. As changes are completed and tested, they pass the quality gate and are "promoted" as merges into successive tiers.

Branch per Technology
Each technology platform is a permanent branch. Common parts of the codebase are merged between each platform.

SCM Tools

- Tools help in managing projects better as it reduces a lot of manual effort
- Early SCM tools had facilities for controlling and managing CI , but now it comes with a lot of features like
  - Build and Release management
  - Defect Management and tracking
  - Packaging control etc ..
- Major advantages of SCM tools are
  - Sharing of project information across teams with accurate and reliable information,
  - Flexible in supporting parallel development
  - Provide better decision-making capability by managing and tracking

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 66

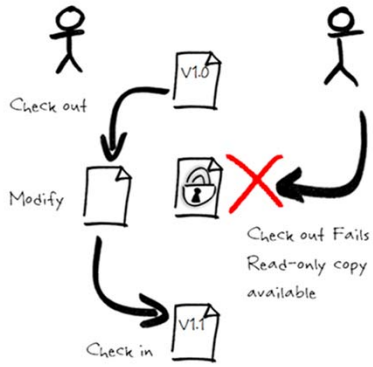
Different Roles and Accesses in SCM Tool

- Administrator
  - All access to all project folders
  - Can add users, create and manage projects and modify their access levels
- Configuration Manager
  - Greater access than all team-members.
  - Creates the basic environment for his projects configuration management
  - Responsible for moving files across projects, establishing baselines, adding requirements files, preparing guidelines, etc
- Team-member
  - Varying access depending on their responsibilities. For e.g. PM gets add/modify access to Management project

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 67

Check –in and Check -Out



A check-out is the act of creating a local working copy from the repository. A user may specify a specific revision or obtain the latest.

A check - in is the action of writing or merging the changes made in the working copy back to the repository.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 68

File locking
In a file locking system only one developer has write access to the artifact Other developers will have read-only access to the current (stored) version. The file is only available again once it is checked back in.

Test your understanding

- SRS (System Requirement Specification) are prepared in which phase ?
- A software can be built using more than 1 life cycle model . (T/F)
- Name some non functional requirements
- Architecture focuses on functional requirement T/F ?
- Alpha testing and beta testing happens in which phase ?

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 69

Test Your Understanding!

- In which phase UNIT Test Plan is written ?
- White box testing includes
  - Unit Testing
  - System Testing
  - Operations Acceptance Testing
  - Integration testing
- A single baseline may contain many files.(T/F)
- A Tester can test in the development library.(T/F)

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 70

Introduction to Software Testing Life Cycle

What is Software Quality?

- Quality is the totality of features and characteristics of a product or service that bear on its ability to satisfy stated and implied needs
- Software Quality means:
  - Total conformance to user requirements and specifications
  - Absence of Defects
  - Fitness of Use

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 72

Other definitions

- Errors:
  - Mistakes we make
- Defects:
  - Defects are the results of errors
- Failures:
  - Software failures are the results of Defects, observed while running the software

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 73


Quality Assurance & Quality Control
Quality Control
Ensures that the product developed is correct
Detection and correction activity
E.g. Reviews and Testing
Carried out after initiation
Quality Assurance
Ensures that the product is developed correctly
Prevention activity
E.g. Process definition, planning
Done at the beginning of project
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING
Copyright © Capgemini 2015. All Rights Reserved
74

Importance of QA/QC

- "All QC/QA activities are important, because all lead to quality software, if practiced correctly. However, if you allow me to choose one only, my choice is Formal Technical Reviews. If conducted properly, they are the single-most effective ways to uncover and fix defects while they are still inexpensive to find and fix."

- Roger Pressman

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 75

What is V&V?

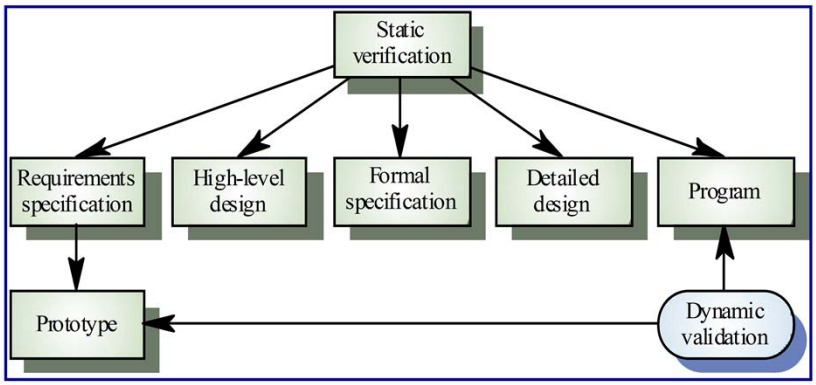
- Verification.
  - Determine whether the artifacts of each phase of SDLC fulfill the requirements of the previous phase.
  - Are we building the product correctly?
  - Also called as static technique or Review

- Validation
  - Determine whether the artifacts of each phase of the SDLC are in tune with the customer requirements.
  - Are we building the correct product?
  - Also called as dynamic technique or Testing

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

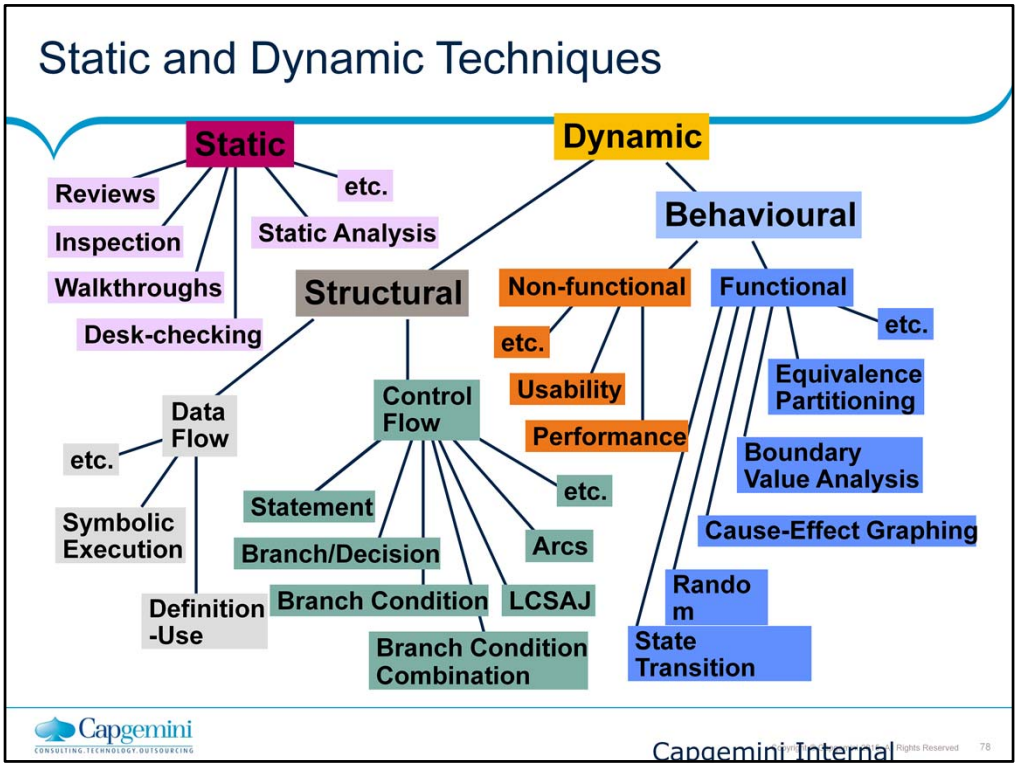
 76

Static and Dynamic V&V



Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING

 77


Static and Dynamic Techniques
Static
Dynamic
Reviews
Inspection
Walkthroughs
Desk-checking
etc.
Static Analysis
Structural
Behavioural
Non-functional
Functional
etc.
Usability
Performance
etc.
Equivalence Partitioning
Boundary Value Analysis
Cause-Effect Graphing
Rando m
State Transition
Data Flow
etc.
Symbolic Execution
Definition -Use
Control Flow
Statement
Branch/Decision
Branch Condition
Branch Condition Combination
Arcs
LCSAJ
etc.
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING
Capgemini Internal

Why V&V?

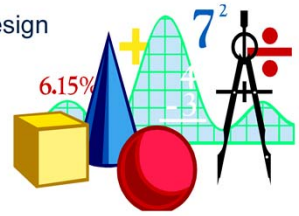
- V&V is required because :
  - Software construction is a complex task.
  - Software construction is a Human activity
  - Humans make mistakes
  - Discipline and Methodology for construction reduce mistakes but cannot eliminate mistakes
- V&V is about evaluating the products and processes used for producing the products
- V&V is NOT about evaluating the producers

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 79

Types of V&V

- Reviews
  - Project Plans, Schedules, and Costs
  - Requirements, Functional Specifications, and Design
  - Test Plans, Test Specifications, Test Scripts,
  - Source Code
- Testing of Software
  - Unit
  - Integration
  - System
  - Customer Acceptance



Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

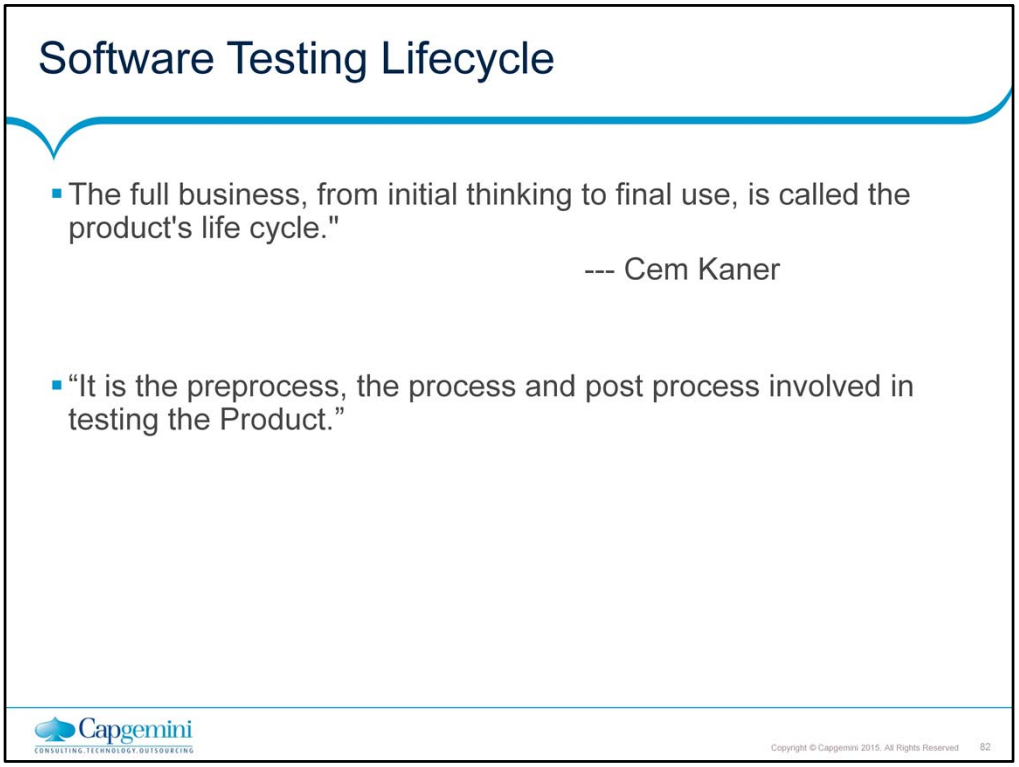
 80

What is Life Cycle ?

- Lifecycle in simple term refers to the sequence of changes from one form to other form
- Every entity has a lifecycle from its inception to retire
- In a similar way, Software is also an entity
- Just like developing software involves a sequences of steps, testing also has steps which should be executed in a definite sequence
- This phenomenon of executing the testing activities in a systematic and planned way is called testing life cycle

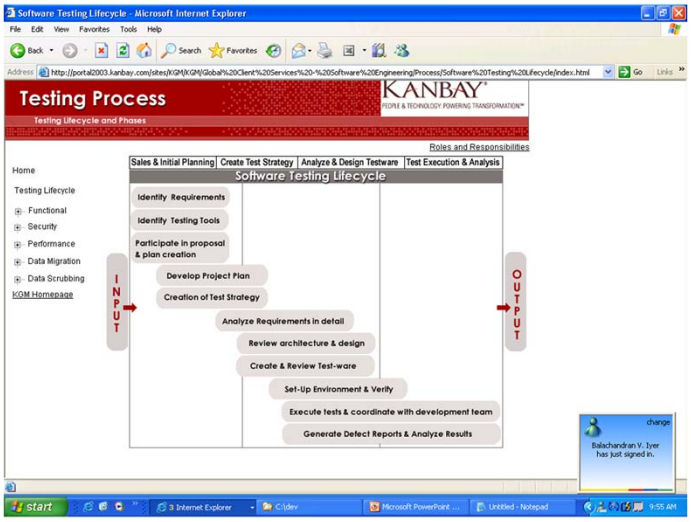
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

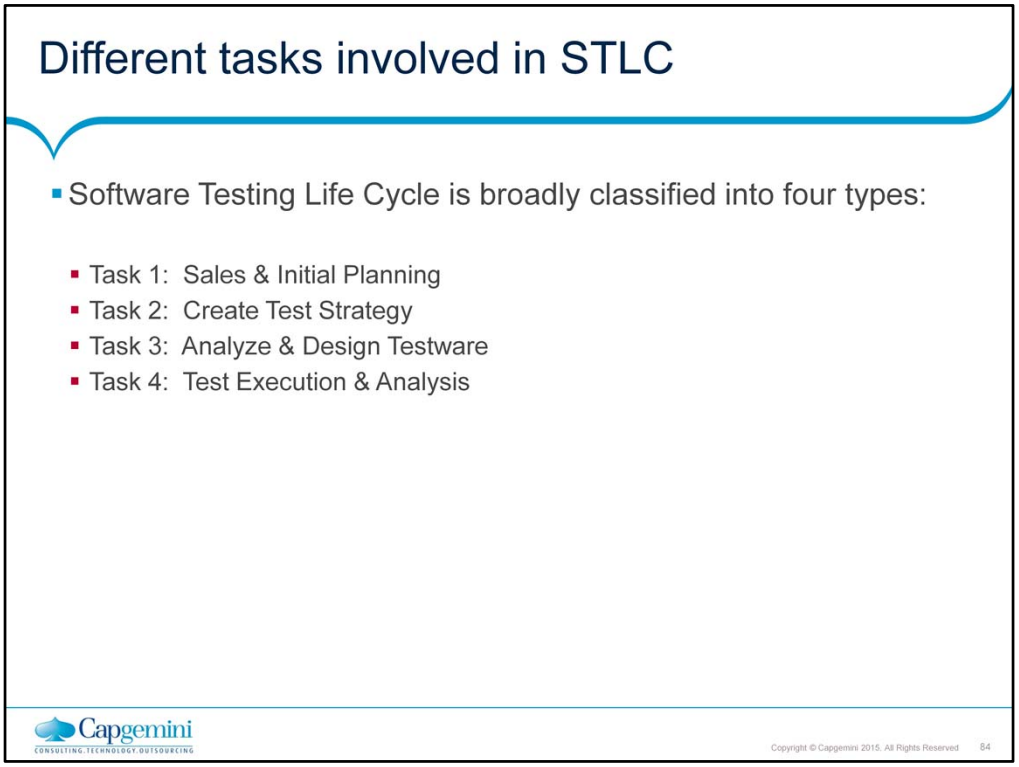
 81



Software Testing Life Cycle (STLC) is the testing process which is executed in systematic and planned manner. In STLC process, different activities are carried out to improve the quality of the product.

Software Test Life Cycle (STLC)

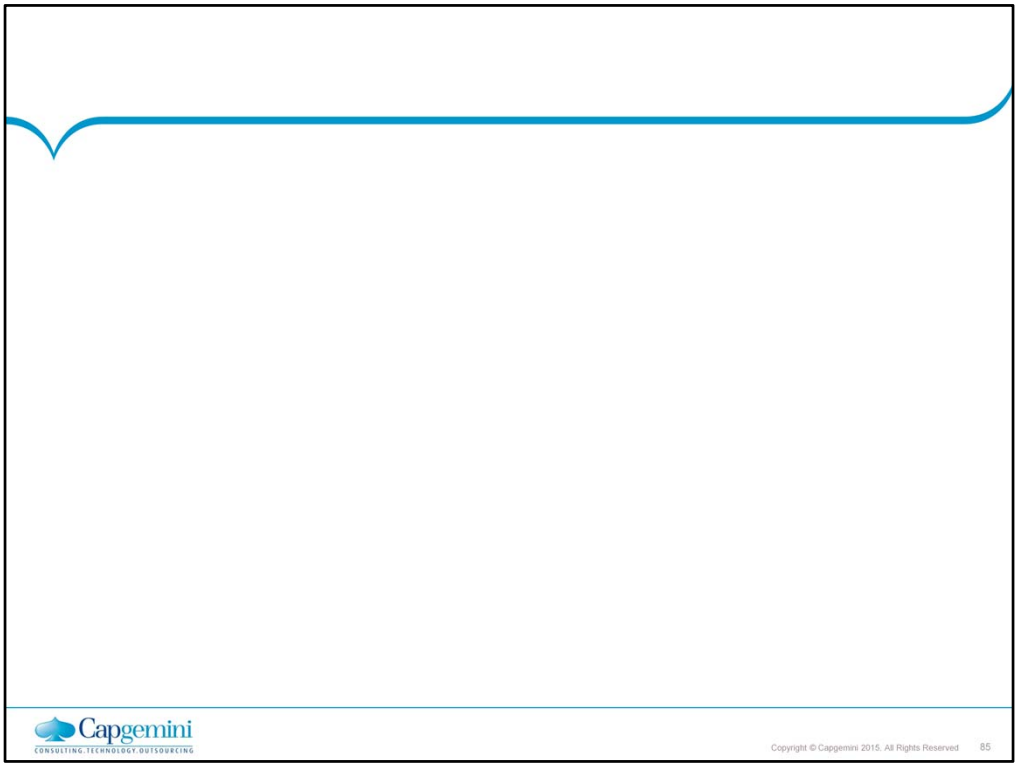




Task 1: Sales and initial planning
During this phase of STLC, analyze and study the requirements. Have brain storming sessions with other teams and try to find out whether the requirements are testable or not.
In practical scenarios, Test planning is the first step of the testing process. In this phase we identify the activities and resources which would help to meet the testing objectives. During planning we also try to identify the metrics, the method of gathering and tracking those metrics.
Identify the need and feasibility of tools to be used based on project profitability, enhanced delivery capability and ease & effectiveness of testing.

Task 2: Create test strategy
A Test Strategy document is a high level document and normally developed by project manager and defines "Software Testing Approach" to achieve testing objectives. It sets the standards for testing processes and activities and other documents such as the Test Plan draws its contents from those standards set in the Test Strategy Document.

Task 3: Analyze and Design Testware
Analyse: Defines "WHAT" to be tested.
Design Testware: Defines "How" to test.
This is the most important stage of the testing life cycle, in this section or phase of the testing part; the testers will develop the test cases based against the requirements of the customer. There are usually three levels of requirements, to be understood by the testers before they can proceed to write the test cases for the product
•HLI (High level Information)
•LLI / Use Cases (Low level Information)
•Snapshots (Prototype or images of a similar product or framework.)



Task 4: Test Execution & Analysis

Test Execution - This is the phase where the test engineers will prepare and execute the test cases. One of the best techniques is to refer the use cases, pick up the standard test case templates and prepare test cases based on different categories. While do remember to maintain Traceability matrix alongside test execution to avoid rework and confusions at a later stage.

Analysis- It concentrates on the exit criteria and reporting after execution of test cases. Call out the testing team member meeting & evaluate cycle completion criteria based on Test coverage, Quality, Cost, Time, Critical Business Objectives, and Software. Discuss what all went good, which area needs to be improve & taking the lessons from current STLC as input to upcoming test cycles, which will help to improve bottleneck in the STLC process. Test case & bug report will analyze to find out the defect distribution by type and severity. Once complete the test cycle then test closure report & Test metrics will be prepared. Test result analysis to find out the defect distribution by type and severity. So Higher management people can take some decisions by analyzing these reports.

Different activities in each tasks

- Lifecycle in simple term refers to the sequence of changes from one form to other form
- Every entity has a lifecycle from its inception to retire
- In a similar way, Software is also an entity
- Just like developing software involves a sequences of steps, testing also has steps which should be executed in a definite sequence
- This phenomenon of executing the testing activities in a systematic and planned way is called testing life cycle

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Task 1: Sales & Initial Planning

- 'Sales & Initial Planning' in tune has three major activities like
  - Identify Requirements
  - Identify Testing Tool
  - Participate in proposal & plan creation

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 87

Task 1: Sales & Initial Planning - Identify Requirements

- Details about this activity : This task ensure that requirements related to designing & performing software testing at various stages is adequately captured like,
  - Business and functional requirements
  - Non-functional requirements
  - Data related requirements
  - Environment related requirements

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 88

Task 1: Sales & Initial Planning - Identify Requirements

- Roles and Responsibilities
- Tasks to be perform by Test Manager with PM / Technical Architect / Business Analyst :
  - Identify testable and non testable items from requirements
  - Ensure all non functional requirements like Usability, Security, Reliability, Performance, Stress, transaction are captured
  - The data requirements for each level of testing and types of testing are captured
  - Verify if the production database at client has some sensitive data and information in it
  - Data structure & relationships are analyzed and the business rules associated with the tables & columns are to be captured
  - Identify how exceptions, errors and deviations are to be handled
  - Identify risks and assumptions and mitigation plan. Unanswered questions can be marked as pending until the project progresses
  - Obtain client sign-off on the requirements & baseline the requirement

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 89

Task 1: Sales & Initial Planning - Identify Testing Tool

- Identify Testing Tools : Identify the need and feasibility of tools to be used based on project profitability, enhanced delivery capability and ease & effectiveness of testing.

- Testing tools could be used for the following purposes:
  - Functional Testing
  - Performance Testing
  - Website Monitoring,
  - Data Generation
  - Data Upload
  - Defect Tracking
  - Application Security Testing
  - Data Migration testing

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Task 1: Sales & Initial Planning - Identify Testing Tool

- Details about this activity : Some of the tools should be chosen by default, e.g. Bug Tracking Tool, Unit Testing Tool, but the tool expert must consider the following:
  - Evaluate needs for testing tools
  - Impact of the tool usage on the cost/time and effort for the project
  - Infrastructure changes, licensing and availability of the tool to be accessed
  - Revise estimates for testing effort based on the tools selected
  - The deal team/ PM shall place a helpdesk request if tools have to be procured
  - Based on the timelines for availability of tools, project schedule & timelines may be revised

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 91

Task 1: Sales & Initial Planning - Participate in proposal & plan creation

- Details about this activity:
  - The testing activities, effort estimates, cost estimates & milestones are determined based on the previous activities
  - These are now factored into the proposal to the client & an initial project plan is developed

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Task 1: Sales & Initial Planning - Participate in proposal & plan creation

- Roles and Responsibilities :

- Tasks to be perform by test manager :
  - Estimate the testing effort & cost for the project and integrate the same with the overall project effort in the proposal
  - Identify the Size and roles of testing team, Major testing milestones, Hardware and Software requirements for testing and Data back-up & roll back facilities
  - Review the proposal using the proposal review checklist
  - Raise resource requisitions
  - Integrate all testing activities identified into the creation of phase one project plan

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 93

Task 2: Create Test Strategy

- 'Create Test Strategy' in tune has two major activities like :
  - Develop Detailed Project Plan
  - Creation of Test Strategy

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 94

Task 2: Create Test Strategy - Develop Detailed Project Plan

- Develop Detailed Project Plan : Once the deal for the approval to go ahead with the project is obtained, detail planning needs to be executed.
- Roles and Responsibilities :
- Tasks to be perform by test manager along with PM:
  - Confirm the resource requisitions and raise a helpdesk request for hardware, software requirements etc.
  - Update the project plan periodically as and when there is any changes

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 95

Task 2: Create Test Strategy - Creation of Test Strategy

- Details about this activity
- Based on the technology, the level/type of testing and the results of the initial requirements analysis, specific testing deliverables which will become the basis for creating a test strategy need to be defined and use the available Test Strategy Templates.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 96

Task 2: Create Test Strategy - Creation of Test Strategy

- Roles and Responsibilities :
- Some of the tasks to be perform like :
  - Set the testing objectives
  - Define scope based on the requirements
  - Identify the types of testing required
  - Determine the levels of testing involved in the project
  - Define Entry and Exit criteria
  - Define Test environment requirements with respect to test infrastructure requirements, test bed set up, configuration management process
  - Identify various cycles of testing
  - Identify the core functionality and the risk associated with it

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 97

Task 2: Create Test Strategy - Creation of Test Strategy

- Identify Test Data Requirements, Test Procedures, Special Considerations and time to prepare test cases from several test scenarios, for all types of tests
- Identify what and how the data will be scrubbed for functional and performance testing
- Plan for test data and it source and the method for replicating data from the production environment, tools required for data creation
- Identify the data requirements for integration and system, if required
- Identify the test stubs and harness required plus how they should be designed
- Identify how test results (test reports, defects raised) shall be reproduced

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 98

Task 3: Analyze and Design Testware

- 'Analyze and Design Testware' in tune has five major activities like :
  - Analyze and Detail Requirements
  - Review Architecture and Design
  - Design Testware
  - Review Design of Testware
  - Set Up Test Environment

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Task 3: Analyze and Design Testware - Analyze and Detail Requirements

- Details about this activity :
  - The main goal here is to understand and detail the requirements to be able to design and develop the Testware.
  - Any gaps in requirements, functional or non-functional should be identified & resolved at this stage

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

Task 3: Analyze and Design Testware - Analyze and Detail Requirements

- Roles and Responsibilities :
- Some tasks to be perform by Test Manager with PM / Technical Architect / Lead and Business Analyst :
  - Ensure that the requirements are complete, testable, comprehensive and consistent
  - Ensure that all aspects of requirements like functionality, usability, GUI, security, reliability, portability etc. are adequately detailed
  - Ensure that, based on the business processes, the performance testing team has created transaction and user profiles
  - Identifying the source, access, approximate volume and ways of generating data required, analyze the data requirements in detail for every level
  - Revisit the list of testable and non-testable items and make required changes if any

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 101

Task 3: Analyze and Design Testware - Analyze and Detail Requirements

- Revisit the list of risks and assumptions identified during the initial requirements analysis
- Identify gaps in the time, effort and cost estimation of the project between the initial requirement analysis and this detailed one
- Adequately detail the data related requirements and responsibility to create test data is identified
- Define methods/ life cycle to record, report & track test results. Define reports that will be generated on a periodic basis. Define approach to analyze the test results
- For security testing, understand the application from both functionality and security perspective. Decompose the application. Ensure that for security testing different user profiles of an application are available to the tester
- For data migration testing, data to be migrated has to be identified along with existing relationships & associated business rules

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 102

Task 3: Analyze and Design Testware - Review Architecture and Design

- Details about each activity:
- Review Architecture and Design
  - The objective is to ensure that the architecture & design are developed/provided and have adequately implemented all testing related requirements.
- Ensure that the design is testable
  - In case of Object Oriented design the appointed architect from the testing team should review the elements of object-oriented software design that may cause testing problems. e.g. to limit the use of class inheritance, to limit the state behavior of objects, to avoid overly complex classes.
  - In the case of performance testing the aim is to convert non-functional requirements get converted into the appropriate design.
  - In case of application security testing the architecture/design that has been developed/provided should be sufficient to list or decompose into low level components.
  - The test lead along with the technical architect/ lead shall perform formal architecture/ design walk through to ensure traceability of test requirements into design and that the design is testable.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 103

Task 3: Analyze and Design Testware - Design Testware

- Details about this activity : Based on the scope of the project, analyze the requirements and design the Testware accordingly.
- Key components of designing Testware include:
  - Design test cases, scripts & test scenario
  - Design test environment and libraries
  - Design test data and test stubs
  - Design test harness, if required Design methods to log, track, analyze and report test results

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 104

Task 3: Analyze and Design Testware - Design Testware

- Roles and Responsibilities :
- Tasks to be perform by test manger along with test lead :
  - Design test cases, scripts & scenario for Functional Testing, Performance Testing, Security Testing and Data Migration testing
  - Design test environment & libraries
  - Design test data and test stubs for different levels (Unit, Integration, System, UAT) and modes (manual/automated) of testing
  - Impact Analysis in case of existing Testware.
  - If the Testware already exists identify the impact of the new functionality on the existing Testware

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 105

Task 3: Analyze and Design Testware - Review Design of Testware

- Details about this activity:
  - At least one level of formal review/walkthrough of all components of the Testware should be performed to ensure adequacy of test design & reduce rework
  - The peer review by the Business Analyst or the Domain expert as applicable. For Security testing the Team Lead would be responsible for reviewing the Testware
  - After all internal reviews have been completed get the test cases along with the test data reviewed by the client

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 106

Task 3: Analyze and Design Testware - Set Up Test Environment

- Details about this activity:
- Set Up Test Environment :
  - The configuration of test environment is detailed in the test strategy. Hardware is set up as per specifications, specific versions of software are installed and Databases are set up with initial data to be used for testing. All this is done to perform various levels & types of testing

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 107

Task 3: Analyze and Design Testware - Set Up Test Environment

- Roles and Responsibilities :
- Tasks perform by IT support and test lead :
  - Review & clarify the specifications for different test environment at various project locations (onsite/ offshore)
  - Install the necessary hardware and software
  - Set up databases as per specifications
  - Upload any initial test data if required
  - Set up testing tools, defect tracking tools
  - Test the setup of the environment at each location & address any issues

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 108

Task 4: Test execution and analysis

- 'Test execution and analysis' in tune has three major activities like :
  - Execute tests & coordinate with development team
  - Defect Reporting & Analyze Test Result
  - Considerations for independent Test Assignments

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 109

Task 4: Test execution and analysis - Execute tests & coordinate with development team

- Details about this activity :
- Execute tests & coordinate with development team :
  - Testing may be performed at various levels (unit, integration, system, UAT) at various project locations and with various types (functional, performance, usability, and security).
  - Typically unit testing is performed by the development team. Other types of testing may be performed by a separate testing team.
  - For such a situation formal procedures for handover should be defined and implemented. Similarly, a formal procedure should be defined and implemented for handover of the tested code, along with test results and status of defects detected.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 110

Task 4: Test execution and analysis - Execute tests & coordinate with development team

- Roles and Responsibilities :
- Tasks to be perform by testers, test lead / manager & the team lead :
  - The team lead shall complete release notes and handover to the test lead to identify modules/ code to be tested
  - The test lead shall allocate tasks to the testers as per plan
  - Testers shall prepare for execution of the test. Specific data & further detailed test scenarios, scripts, test cases would have to be created. Cases may be prioritized for testing
  - Tester shall review the environment installation for completion
  - Testers shall execute the test, record the results and log defects using agreed upon tools

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 111

Task 4: Test execution and analysis - Execute tests & coordinate with development team

- The team lead shall review the test results, analyze the code, and assign the open defects to developers. After the bug fixes, testing team have to perform another iteration of testing
- The test lead shall verify the test results for accuracy, complete the release note and bug report. These are then handed over to the development team (client or Kanbay). If the reports have to be submitted to the client, then additional report providing the analysis of the test results shall be created.
- All the above steps to be repeated until all tests are executed successfully with either zero defects or with an acceptable level of defects.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 112

Task 4: Test execution and analysis - Defect Reporting & Analyze Test Result

- Details about this activity:
  - Analysis in functional testing refers to identifying the defects, their causes, studying the defect patterns and analyzing them.
  - Consolidation of defects in the defect reports would primarily be the responsibility of the testing team while analyzing the causes of defects would be the responsibility of the development team.
  - Defect reports or Client reports also provide information to the development team/client on the quality of the product, stating the risks and issues faced during testing and making recommendations on the product quality and future testing process improvements.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 113

Task 4: Test execution and analysis - Defect Reporting & Analyze Test Result

- Roles and Responsibilities:

- Tasks to be perform by Test Manager / Performance Analyst :
  - For Performance Testing – Analyze the Response Time per transaction, Transaction Response Time Distribution, Total Hits/Second and related graphs.
  - For Functional Testing -- Analyze the Test Results to determine defects, their priority and severity. If the testing activity is a part of development project being executed by Kanbay, root causes of defects should be determined along with the development team and fixes identified. In other cases this could be joint activity with the client team.
  - For Application security testing -- The threat maps will be populated that will show the coverage of threats tested for. A security threat report will be prepared which will list the vulnerabilities identified as well as some mitigation suggestions.
  - For Data Migration Testing -- Data in the target and source database are compared & validated post migration.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 114

Task 4: Test execution and analysis - Considerations for independent Test Assignments

- Details about this activity :
- Considerations for independent Test Assignments :
  - For independent functional testing projects, performance testing projects or application security testing projects all the above workflows and phases are applicable. Certain upstream activities such as reviewing design, reviewing sample code to meet performance and security requirements and performing application tuning would be out of scope.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 115

Test Your Understanding!

- Question 1: Identify Requirements , Identify Testing Tool and Participate in proposal & plan creation are the activities of Create Test strategy.
  - True/ False
- Question 2: Which of the following tasks to be performed by Test Manager with PM / Technical Architect / Lead and Business Analyst in analyze and Design Testware?
  - a) Ensure that the requirements are complete, testable, comprehensive and consistent
  - b) Set the testing objectives
  - c) Estimate the testing effort & cost for the project and integrate the same with the overall project effort in the proposal
  - d) All of the mentioned

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 116

Test Your Understanding!

- Match the following :

1. Sales and Initial planning	A. Set-up Environment
2. Create Test Strategy	B. Execute tests and coordinate with development team
3. Analyze & Design Testware	C. Identify Requirements
4. Test Execution & Analysis	D. Develop Project Plan & create test strategy

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 117