6.	SCM procedures for managing change
7.	Methods for maintaining SQA related records
8.	Organizational roles and responsibilities

The SQA plan is a document aimed to give confidence to developers and customers that the specified requirements will be met and final product will be a quality product.

Review Question

1. *Give the template for SQA Plan.*

❑❑❑

8 Software Maintenance and Configuration Management

Syllabus

Types of software maintenance, Re-engineering, Reverse engineering, Forward engineering, The SCM process, Identification of objects in the software configuration, Version control and change control.

Contents

8.1 Software Maintenance ..... **Summer-2018,** ..... Marks 7

8.2 Re-Engineering ..... **Summer-2016,** ..... **Winter-2017, 2018,** ..... Marks 7

8.3 Reverse Engineering ..... **Summer-2013, 2018, 2019,** ..... **Winter-2017, 2018,** ..... Marks 7

8.4 Forward Engineering ..... **Summer-2019, Winter-2019,** Marks 4

8.5 Introduction to Software Configuration Management (SCM) ..... **Summer-2016, Winter-2018,** Marks 7

8.6 Software Configuration Items ..... **Winter-2011,** ..... Marks 7

8.7 SCM Process ..... **Summer-2013, 2018, 2019,** ..... **Winter-2019,** ..... Marks 7

8.1 Software Maintenance

GTU : Summer-2018, Marks 7

- Software maintenance is an activity in which program is modified after it has been put into use.
- In software maintenance usually it is not preferred to apply major software changes to system's architecture.
- Maintenance is a process in which changes are implemented by either modifying the existing system's architecture or by adding new components to the system.

8.1.1 Need for Maintenance

The software maintenance is essential because of following reasons :

1. Usually the system **requirements are changing** and to meet these requirements some changes are incorporated in the system.
2. There is a strong relationship between system and its environment. When a system is installed in an environment, it changes that environment. This ultimately changes the system requirements.
3. The maintained system remains useful in their working environment.

- Maintenance is applicable to software developed using any software life cycle model. The system changes and hence maintenance must be performed in order to :
  a) Correct faults.
  b) Improve the design.
  c) Implement enhancement.
  d) Interface with other systems.
  e) Adoption of environment (different hardware, software, system features etc.).
  f) Migrate legacy software.
  g) Replacement of old software by new software.
- In software maintenance report four key characteristics should be mentioned.
  i) Maintaining control over the software's day to day functions.
  ii) Maintaining control over software modification.
  iii) Reparing of functions.
  iv) Performance degradation should be avoided.

8.1.2 Types of Software Maintenance

Various types of software maintenance are

1. **Corrective maintenance** - Means the maintenance for correcting the software faults.

2. **Adaptive maintenance** - Means maintenance for adapting the change in environment (different computers or different operating systems).
3. **Perfective maintenance** - Means modifying or enhancing the system to meet the new requirements.
4. **Preventive maintenance** - Means changes made to improve future maintainability.

Review Question

1. *Explain software maintenance.* GTU : Summer-2018, Marks 7

8.2 Re-Engineering

GTU : Summer-2016, Winter-2017, 2018, Marks 7

Software re-engineering means re-structuring or re-writing part or all of the software engineering system.

The software re-engineering is needed for the applications which require frequent maintenance.

Advantages of software re-engineering

1. **Reduced risk** – The re-engineering allows the developer to eliminate certain constraints on the system. This helps in reducing the risks of failures.
2. **Reduced cost** – The cost of re-engineering is often significantly less than the costs of developing new software.

Re-engineering process activities

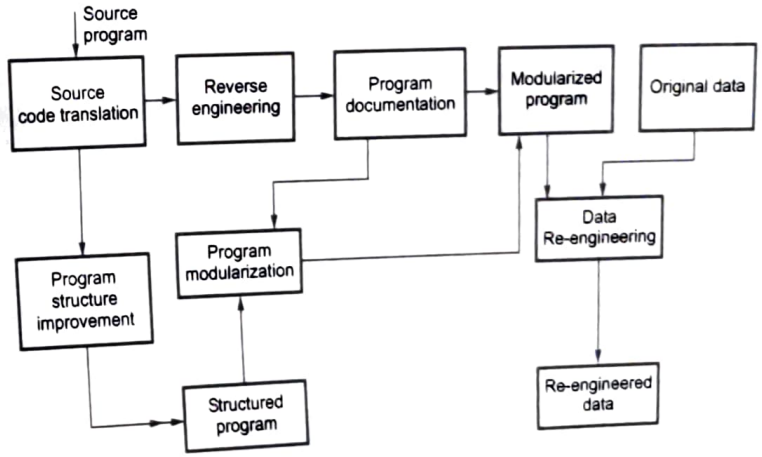

Fig. 8.2.1 Re-engineering process activities

- Source code translation : In this phase the code is converted to a new language.
- Reverse engineering : Under this activity the program is analysed and understood thoroughly.
- Program structure improvement : Restructure automatically for understandability.
- Program modularization : The program structure is reorganized.
- Data re-engineering : Finally clean-up and restructure system data.

Review Questions

1. *Explain the software re-engineering activities.*
2. *Explain software re-engineering process model.* **GTU : Summer-2016, Marks 7**
3. *Write short notes on re-engineering.* **GTU : Winter-2017, 2018, Marks 3**

8.3 Reverse Engineering

GTU : Summer-2013, 2018, 2019, Winter-2017, 2018, Marks 7

Reverse engineering is the process of design recovery. In reverse engineering the **data, architectural** and **procedural information** is extracted **from a source code.**

There are three **important issues** in reverse engineering.

1. **Abstraction level :** This level helps in obtaining the design information from the source code. It is expected that abstraction level should be high in reverse engineering. High abstraction level helps the software engineer to understand the program.
2. **Completeness level :** The completeness means detailing of abstract level. The completeness decreases as abstraction level increases.

   For example - From a given source code listing one can easily develop a complete procedural design representation. But it is very difficult to develop complete set of data flow diagrams or entity relationship diagram. The completeness in reverse engineering develops the interactivity. The term interactivity means the degree to which the human is integrated with automated tools to create effective reverse engineering process. As the abstraction level increases the interactivity must increase to bring the completeness.
3. **Directionality level :** Directionality means extracting the information from source code and give it to software engineer. The directionality can be one way or two way. The one way directionality means extracting all the information from source code and give it to software engineer. The two way directionality means the information taken from source code is fed to a re-engineering tool that attempts to restrcture or regenerate old programs.

8.3.1 Reverse Engineering Process

Initially the dirty source code or unstructured source code is taken and processed and the code is restructured. After restructuring process the source code becomes clean. The core to reverse engineering is an activity called extract abstractions.

In extract abstraction activity, the engineer must evaluate older programs and extract information about procedures, interfaces, data structures or databases used.

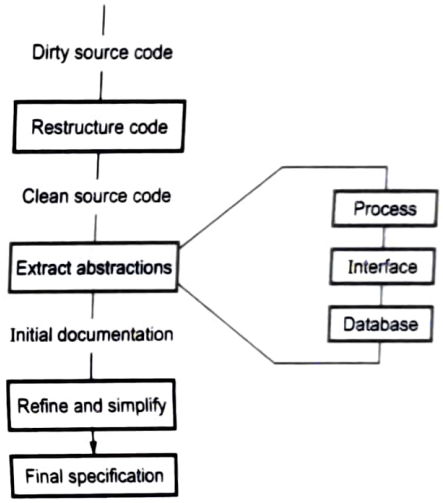

Fig. 8.3.1 Reverse engineering

The output of reverse engineering process is a clear, unambiguous final specification obtained from unstructured source code. This final specification helps in easy understanding of source code.

Difference between Software Engineering and Reverse Engineering

Sr. No.	Software engineering	Reverse engineering
1.	Software engineering is a discipline in which theories, methods and tools are applied to develop a professional software product.	Reverse engineering is a process in which the dirty or unstructured code is taken, processed and it is restructured.
2.	Initially only user requirements are available for software engineering process.	A dirty or unstructured code is available initially.
3.	This process starts by understanding user requirements.	This process starts by understanding the existing unstructured code.

4.	The software engineering is conducted using, requirement gathering, analysis, design, implementation and testing.	The reverse engineering is conducted using restructuring the code, cleaning it, by extracting the abstractions. After refinement and simplification of the code final code gets ready.
5	It is simple and straightforward approach.	It is complex because cleaning the dirty or unstructured code requires more efforts.
6.	Documentation or specification of the product is useful to the end-user.	Documentation or specification of the product is useful to the developer.

Review Questions

1. *Write short note on : Reverse engineering* **GTU : Summer-2013, Marks 7**
2. *Write short notes on reverse engineering.* **GTU : Winter-2017, 2018, Summer-2019, Marks 4**
3. *Explain the following term in brief - 1) Re-Engineering, 2) Reverse Engineering.* **GTU : Summer-2018, Marks 4**

8.4 Forward Engineering

GTU : Summer-2019, Winter-2019, Marks 4

If the poorly designed and implemented code is to be modified then following alternatives can be adopted -

1. Make lot of modifications to implement the necessary changes.
2. Understand inner workings of the program in order to make the necessary modifications.
3. Redesign, recode and test small modules of software that require modifications.
4. Completely redesign, recode and test the entire program using re-engineering tool.

Definition : Forward engineering is a process that makes use of software engineering principles, concepts and methods to re-create an existing application. This re-developed program extends the capabilities of old programs.

8.4.1 Forward Engineering for Client Server Architectures

- During re-engineering process, many mainframe applications are modified to accommodate the client server architecture.
- Following are the features of this re-engineering process -
  1. The application functionality is transferred to each client computer from centralized resources.
  2. The new GUI are developed for each client workstation.

3. The database functionalities are handed over to servers.
4. The responsibility of some specialized functionality can also be handled by Servers.
5. New communication techniques are incorporated in the architecture.
6. New security mechanisms are also established both at client and server side.

- The forward engineering for mainframe to client server architecture requires both business and software re-engineering activities.
- The **database** transactions and queries are handled by server applications. At the same time these transactions must be controlled within the context of business rules.
- That means, make sure that these transactions are executed in consistent manner such that all updates are performed by authorized users.
- During forward engineering, there exists a **business layer** present at both client and server. The task of this layer is to control and coordinate the tasks of transactions and query handling. The communication among desktop applications is controlled by business rules layer.
- There is **client applications layer** which implements the business functions that are required by specific group of end users.

8.4.2 Forward Engineering for Object Oriented Architectures

- Forward engineering is a process of re-engineering conventional software into the object oriented implementation.
- Following are the steps that can be applied for forward engineering the conventional software -
  1. Existing software is reverse engineered in order to create data, functional, and behavioral models.
  2. If existing system extends the functionality of original application then **use cases** can be created.
  3. The data models created in this process are used to create the base for **classes.**
  4. Class Hierarchies, object-relationship models, object behavioral models, and subsystems are defined.
- During this forward engineering process, algorithms and data structures are reused from existing conventional application.

Difference between Forward and Reverse Engineering

- Forward engineering is a process of constructing a system for specific purpose.
- Reverse engineering is a process of de-constructing a system in order to extend the functionalities or in order to understand the working of the system.

Review Questions

1. *Difference between reverse engineering and forward engineering.*
2. *Write short note on - Forward Engineering.* GTU : Summer-2019, Marks 4 GTU : Winter-2019, Marks 4

8.5 Introduction to Software Configuration Management (SCM)

GTU : Summer-2016, Winter-2018, Marks 7

Definition : Software configuration management is a set of activities carried out for identifying, organizing and controlling changes throughout the lifecycle of computer software.

During the development of software change must be managed and controlled in order to improve quality and reduce error.

8.5.1 Need for SCM

The software configuration management is concerned with managing the changes in the evolving software. If the changes are not controlled at all then this stream of uncontrolled change can cause the well-running software project into chaos. Hence it is essential to perform following activities -

i) Identify these changes
ii) Control the changes
iii) Ensure that the changes are properly implemented and
iv) Report these changes to others.

The software configuration management may be seen as part of quality management process.

Review Questions

1. *Explain software configuration management.* GTU : Summer-2016, Marks 7
2. *Discuss software configuration management in detail.* GTU : Winter-2018, Marks 7

8.6 Software Configuration Items

GTU : Winter-2011, Marks 7

A Software Configuration Item (SCI) is information that is created as part of the software engineering process.

Examples of Software Configuration Items are

- Computer programs
  - Source programs
  - Executable programs

- Documents describing the programs
  - Technical manual
  - Users manual
- Data
  - Program components or functions
  - External data
  - File structure

For each type of item, there may be a large number of different individual items produced. For instance there may be many documents for a software specification such as project plan, quality plan, test plan, design documents, programs, test reports, review reports.

These SCI or items will be produced during the project, stored, retrieved, changed, stored again, and so on.

Each configuration item must have a unique name, and a description or specification which distinguishes it from other items of the same type.

Review Question

1. *What do you mean by software configuration ? What is meant by software configuration management ?* GTU : Winter-2011, Marks 7

8.7 SCM Process

GTU : Summer-2013, 2018, 2019, Winter-2019, Marks 7

The primary objectives of Software Configuration Management process (SCM) are -

1. **Configuration Identification :** Identify the items that define the software configuration.
2. **Change Control :** Manage changes to one or more items.
3. **Version Control :** Facilitate to create different versions of the application.
4. **Configuration Authentication :** To ensure that the quality of the software is maintained as the configuration evolves over the time.

The SCM process must be developed in such a way that the software team must answer the following set of questions -

1. How does the software team identify the software configuration items ?
2. How does the software team control the changes in the software before and after delivering it to the customer ?

3. How does the software team manage the versions of the programs in the software package ?
4. How does team get ensured that the changes are made properly ?
5. Who is responsible for approving the changes in the software ?

The answers to these questions lead the definition of five tasks of SCM and those are - Identification, change control, version control and configuration audit and status reporting.

8.7.1 Identification of Objects in Software Configuration

- The software configuration items must be separately named and identified as object.
- These objects must be arranged using **object oriented approach**.
- There are two categories of objects - **basic objects** and **aggregate objects**.
- The **basic object** is unit of information created during requirements analysis, design, coding or testing. For example basic object can be part of **source code**.
- Aggregate object is a collection of basic objects and other aggregate objects. For example SRS or data model can be aggregate object.
- Each object can be uniquely identified because it has got -
  1. **Name** : The name of the object is nothing but the collection of characters(string) or some text. It is unique.
  2. **Description** : For describing the object, the object description can be given. This description contains document, program or some other description such as project identifier or version information.
  3. **List of resources** : The resources are the entities that are used for accessing, referencing and processing of objects. The data types and functionalities can serve as a resource.
  4. **Realization or identification** : It is pointer to object.
- The configuration object identification can also consider relationships that exist between the named objects.
- If a change is made to one configuration object it is possible to determine which other configuration objects in the repository are affected by the change.
- Basically object evolve throughout the software process. During the process of object identification the evolution of objects along with its process must be identified.
- Major modifications in the object must be noted.

8.7.2 Change Control

Changes in any software projects are vital. Sometimes, introducing small changes in the system may lead to big problems in product. Similarly, introducing some changes may enhance the capabilities of the system. According to **James Bach** too little changes may create some problems and too big changes may create another problems.

For a large software engineering project, uncontrolled change creates lot of chaos. For managing such changes, human procedures or automated tools can be used.

The change control process is shown by following Fig. 8.7.1. (See Fig. 8.7.1 on next page.)

Step 1 : First of all there arises a need for the change.

Step 2 : The change request is then submitted by the user

Step 3 : Developers evaluate this request to assess technical merits, potential side effects, and overall impact on system functions and cost of the project.

Step 4 : A change report is then generated and presented to the Change Control Authority **(CCA)**.

Step 5 : The change control authority is a person or a group of people who makes a final decision on status or priority of the change.

Step 6 : An Engineering Change Order **(ECO)** is generated when the change gets approved. In ECO the change is described, the restrictions and criteria for review and audit are mentioned.

Step 7 : The object that needs to be changed is checked out of the project database.

Step 8 : The changes are then made on the corresponding object and appropriate SQA activities are then applied.

Step 9 : The changed object is then checked in to the database and appropriate version control is made to create new version.

The checked in and checked out mechanisms require two important elements -

- Access control
- Synchronization control

The **access control** mechanism gives the authority to the software engineer to access and modify the specific configuring object. The **synchronization control** mechanism allows to make parallel changes or the changes made by two different people without overwriting each other's work.

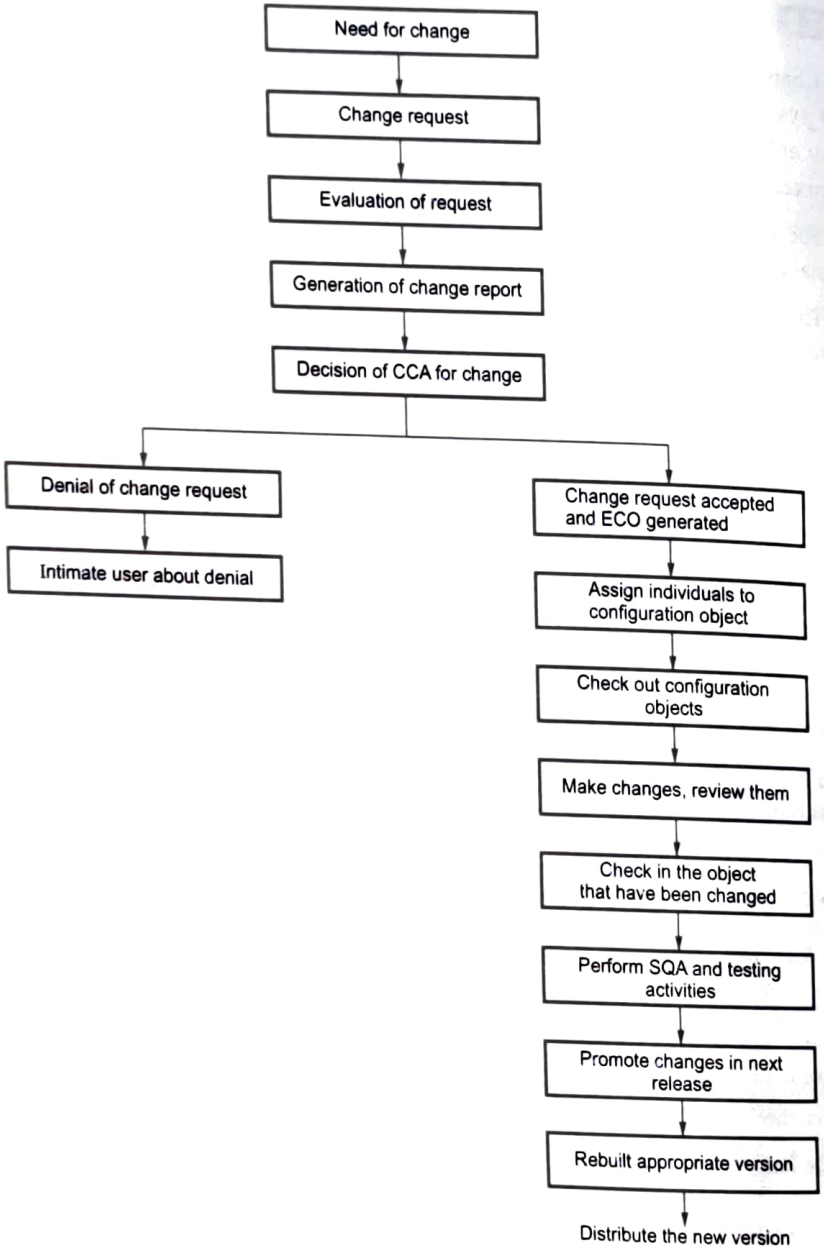

Fig. 8.7.1 Change control process

8.7.3 Version Control

Version is an instance of a system which is functionally distinct in some way from other system instances.

Version control works to help manage different versions of configuration items during the development process.

The configuration management allows a user to specify the alternative configurations of the software system by selecting appropriate version.

Certain attributes are associated with each software version. These attributes are useful in identifying the version. For example : The attribute can be 'date', 'creator', 'customer', 'status'.

In practice the version needs an associated name for easy reference.

Different versions of a system can be shown by an evolution graph as

Each version of software system is a collection of software configuration items.

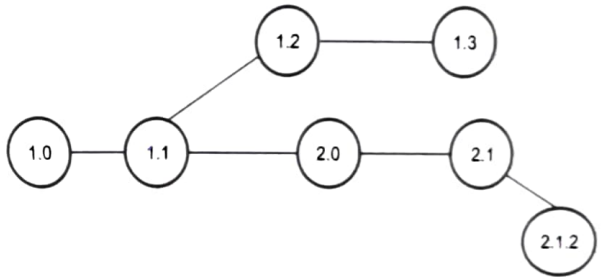

Fig. 8.7.2 Version numbering in evolution graph

8.7.4 Configuration Audit

- In order to ensure that the change has been properly implemented or not two activities are carried out.
  1. Formal Technical Review (FTR)
  2. Software Configuration Audit
- In Formal Technical Review, the correctness of configuration object is identified and corrected. It is conducted by technical reviewer.
- The software configuration audit assess the configuration object for the characteristics that are not reviewed in formal technical review. It is conducted by software quality assurance group.
- Following are some primary questions that are asked during configuration audit -
  1. Whether FTR is conducted to assess the technical correctness ?
  2. Whether or not the change specified by ECO has been made ?

3. If additional changes need to be made or not ?
4. Whether the software engineering standards are properly followed ?
5. Do the attributes of configuration objects reflect the change ?
6. Whether all the SCI are updated properly ?
7. Whether the SCM process(object identification, change and version control, configuration audit and status reporting) are properly followed ?

- The above questions can be asked as a part of formal technical review.

8.7.5 Status Reporting

The status reporting focuses on communication of changes to all people in an organization that involve with changes.

During status reporting following type of questions were asked.

1. **What happened ? :** What are the changes that are required ?
2. **Who did it ? :** Who will be handling these changes ?
3. **When did it happen ? :** The time at which these changes are arised.
4. **What else will be affected ? :** The objects or part of the software that might be reflected due to these changes.

Review Questions

1. *Define software configuration management. Explain change control management and version control management.*
2. *Explain software configuration management and change control management in detail.*
3. *Explain "How to manage the different versions that get created and how to maintain the quality of code under changing conditions ?".* **GTU : Summer-2013, Marks 7**
4. *Explain version and change control management.* **GTU : Summer-2018, 2019, Winter-2019, Marks 4**
5. *Explain SCM process in details.* **GTU : Summer-2019, Marks 7**

❑❑❑

9 DevOps

Syllabus

Product lifetime : Independent product Vs. continues, Improvement, Software as a service, SaaS architecture.

Contents

9.1 Overview
9.2 Problem Case Definition
9.3 Benefits of Fixing Application Development Challenges
9.4 DevOps Adoption Approach through Assessment
9.5 Solution Dimensions
9.6 What is DevOps?
9.7 DevOps Importance and Benefits
9.8 DevOps Principles and Practices
9.9 The 7 C's of DevOps Lifecycle for Business Agility
9.10 DevOps and Continuous Testing
9.11 How to Choose Right DevOps Tools
9.12 Challenges with DevOps Implementation
9.13 Must Do Things for DevOps
9.14 Mapping My App to DevOps
9.15 Assessment, Definition, Implementation, Measure and Feedback