TRINITY INSTITUTE OF TECHNOLOGY AND RESEARCH BHOPAL

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Subject Name: **Software Engineering & Project Management**

Subject Code: **CS-403**

Semester: **4th**

UNIT-V

Software Maintenance

Software Maintenance is the process of modifying a software product after it has been delivered to the customer. The main purpose of software maintenance is to modify and update software application after delivery to correct faults and to improve performance.

Need for Maintenance –
Software Maintenance must be performed in order to:

- Correct faults.
- Improve the design.
- Implement enhancements.
- Interface with other systems.
- Accommodate programs so that different hardware, software, system features, and telecommunications facilities can be used.
- Migrate legacy software.
- Retire software.

Categories of Software Maintenance –
Maintenance can be divided into the following:

1. **Corrective maintenance:**
   Corrective maintenance of a software product may be essential either to rectify some bugs observed while the system is in use, or to enhance the performance of the system.
2. **Adaptive maintenance:**
   This includes modifications and updations when the customers need the product to run on new platforms, on new operating systems, or when they need the product to interface with new hardware and software.
3. **Perfective maintenance:**
   A software product needs maintenance to support the new features that the users want or to change different types of functionalities of the system according to the customer demands.

4. **Preventive maintenance:**
   This type of maintenance includes modifications and updations to prevent future problems of the software. It goals to attend problems, which are not significant at this moment but may cause serious issues in future.

Reverse Engineering –

Reverse Engineering is processes of extracting knowledge or design information from anything man-made and reproducing it based on extracted information. It is also called back Engineering.

Software Reverse Engineering –

Software Reverse Engineering is the process of recovering the design and the requirements specification of a product from an analysis of it's code. Reverse Engineering is becoming important, since several existing software products, lack proper documentation, are highly unstructured, or their structure has degraded through a series of maintenance efforts.

Why Reverse Engineering?

- Providing proper system documentatiuon.
- Recovery of lost information.
- Assisting with maintenance.
- Facility of software reuse.
- Discovering unexpected flaws or faults.

Used of Software Reverse Engineering –

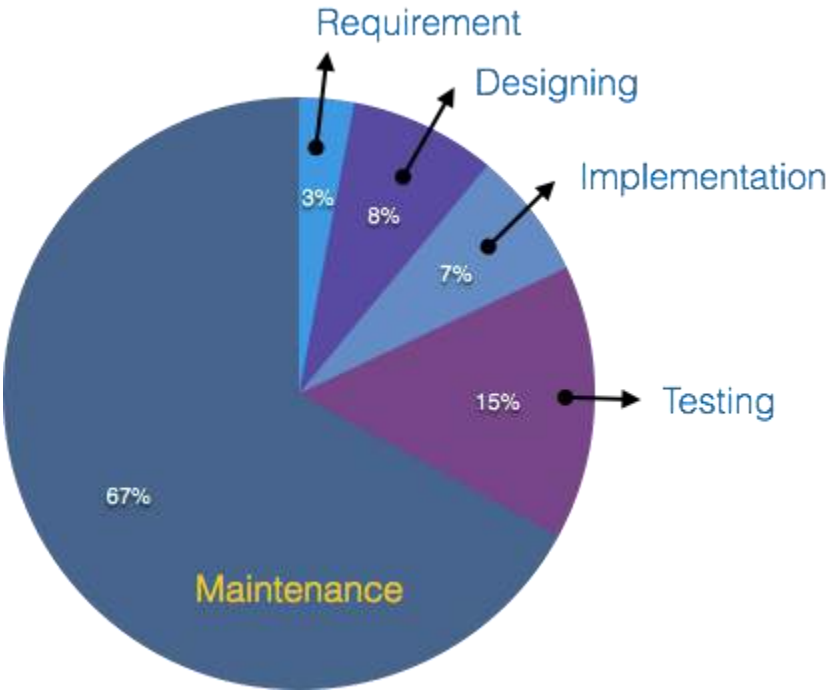
- Software Reverse Engineering is used in software design, reverse engineering enables the developer or programmer to add new features to the existing software with or without knowing the source code.
- Reverse engineering is also useful in software testing, it helps the testers to study the virus and other malware code .

Cost of Maintenance

Reports suggest that the cost of maintenance is high. A study on estimating software maintenance found that the cost of maintenance is as high as 67% of the cost of entire software process cycle.

TRINITY INSTITUTE OF TECHNOLOGY AND RESEARCH BHOPAL

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



On an average, the cost of software maintenance is more than 50% of all SDLC phases. There are various factors, which trigger maintenance cost go high, such as:

Real-world factors affecting Maintenance Cost

- The standard age of any software is considered up to 10 to 15 years.
- Older softwares, which were meant to work on slow machines with less memory and storage capacity cannot keep themselves challenging against newly coming enhanced softwares on modern hardware.
- As technology advances, it becomes costly to maintain old software.
- Most maintenance engineers are newbie and use trial and error method to rectify problem.
- Often, changes made can easily hurt the original structure of the software, making it hard for any subsequent changes.
- Changes are often left undocumented which may cause more conflicts in future.

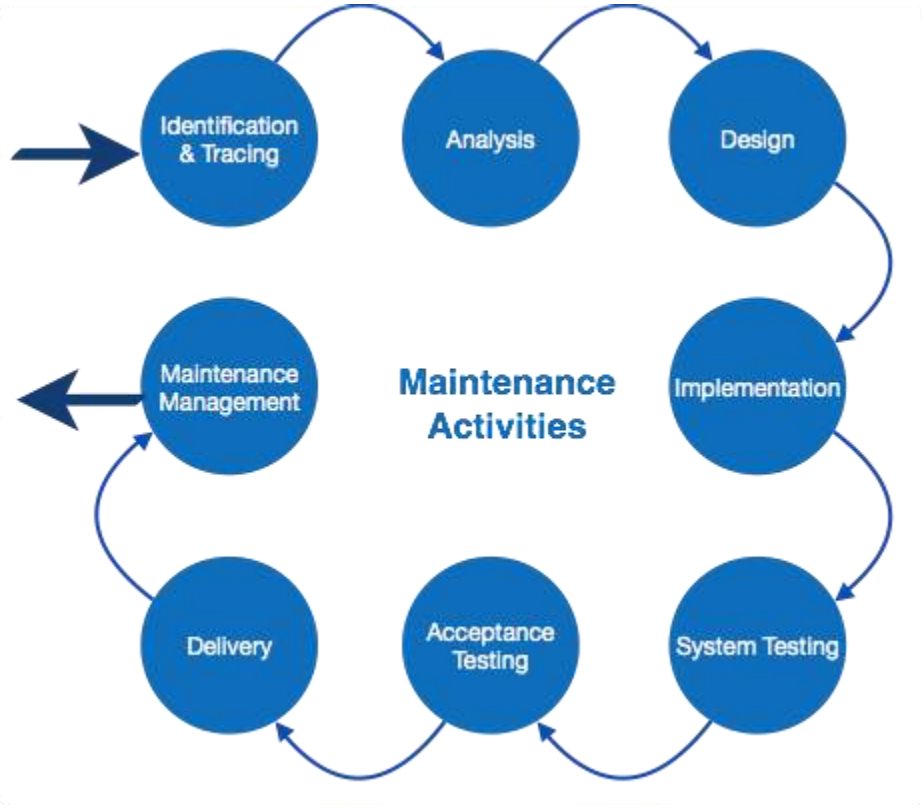
Software-end factors affecting Maintenance Cost

- Structure of Software Program
- Programming Language
- Dependence on external environment
- Staff reliability and availability

Prepared by : URMILA MAHOR

Maintenance Activities

IEEE provides a framework for sequential maintenance process activities. It can be used in iterative manner and can be extended so that customized items and processes can be included.



These activities go hand-in-hand with each of the following phase:

- **Identification & Tracing** - It involves activities pertaining to identification of requirement of modification or maintenance. It is generated by user or system may itself report via logs or error messages.Here, the maintenance type is classified also.
- **Analysis** - The modification is analyzed for its impact on the system including safety and security implications. If probable impact is severe, alternative solution is looked for. A set of required modifications is then materialized into requirement specifications. The cost of modification/maintenance is analyzed and estimation is concluded.
- **Design** - New modules, which need to be replaced or modified, are designed against requirement specifications set in the previous stage. Test cases are created for validation and verification.
- **Implementation** - The new modules are coded with the help of structured design created in the design step.Every programmer is expected to do unit testing in parallel.

- **System Testing** - Integration testing is done among newly created modules. Integration testing is also carried out between new modules and the system. Finally the system is tested as a whole, following regressive testing procedures.
- **Acceptance Testing** - After testing the system internally, it is tested for acceptance with the help of users. If at this state, user complaints some issues they are addressed or noted to address in next iteration.
- **Delivery** - After acceptance test, the system is deployed all over the organization either by small update package or fresh installation of the system. The final testing takes place at client end after the software is delivered.

  Training facility is provided if required, in addition to the hard copy of user manual.
- **Maintenance management** - Configuration management is an essential part of system maintenance. It is aided with version control tools to control versions, semi-version or patch management.

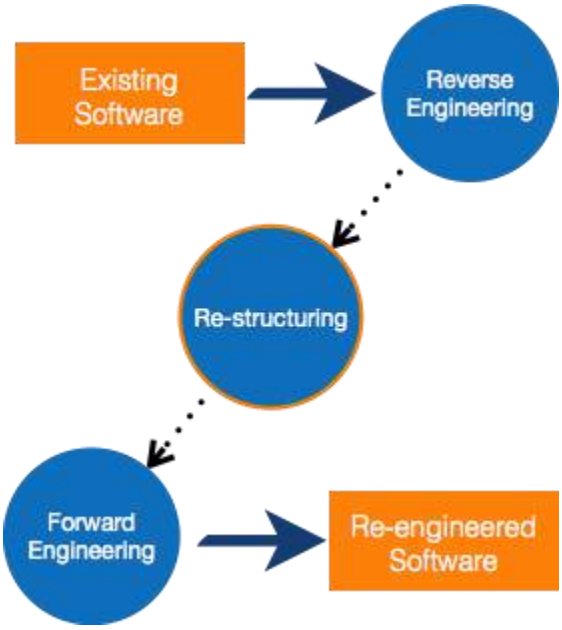
Software Re-engineering

When we need to update the software to keep it to the current market, without impacting its functionality, it is called software re-engineering. It is a thorough process where the design of software is changed and programs are re-written.

Legacy software cannot keep tuning with the latest technology available in the market. As the hardware become obsolete, updating of software becomes a headache. Even if software grows old with time, its functionality does not.

For example, initially Unix was developed in assembly language. When language C came into existence, Unix was re-engineered in C, because working in assembly language was difficult.

Other than this, sometimes programmers notice that few parts of software need more maintenance than others and they also need re-engineering.



Re-Engineering Process

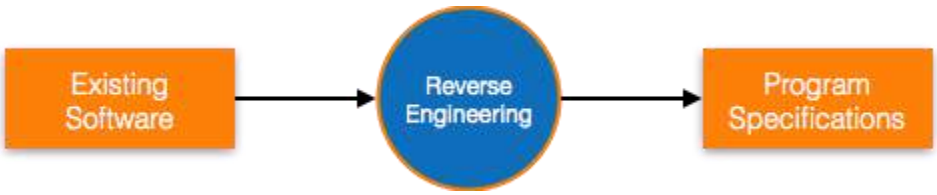
- **Decide** what to re-engineer. Is it whole software or a part of it?
- **Perform** Reverse Engineering, in order to obtain specifications of existing software.
- **Restructure Program** if required. For example, changing function-oriented programs into object-oriented programs.
- **Re-structure data** as required.
- **Apply Forward engineering** concepts in order to get re-engineered software.

There are few important terms used in Software re-engineering

Reverse Engineering

It is a process to achieve system specification by thoroughly analyzing, understanding the existing system. This process can be seen as reverse SDLC model, i.e. we try to get higher abstraction level by analyzing lower abstraction levels.

An existing system is previously implemented design, about which we know nothing. Designers then do reverse engineering by looking at the code and try to get the design. With design in hand, they try to conclude the specifications. Thus, going in reverse from code to system specification.



Program Restructuring

It is a process to re-structure and re-construct the existing software. It is all about re-arranging the source code, either in same programming language or from one programming language to a different one. Restructuring can have either source code-restructuring and data-restructuring or both.

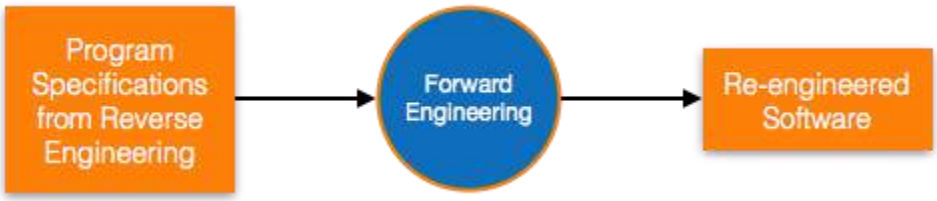
Re-structuring does not impact the functionality of the software but enhance reliability and maintainability. Program components, which cause errors very frequently can be changed, or updated with re-structuring.

The dependability of software on obsolete hardware platform can be removed via re-structuring.

Forward Engineering

Forward engineering is a process of obtaining desired software from the specifications in hand which were brought down by means of reverse engineering. It assumes that there was some software engineering already done in the past.

Forward engineering is same as software engineering process with only one difference – it is carried out always after reverse engineering.



Component reusability

A component is a part of software program code, which executes an independent task in the system. It can be a small module or sub-system itself.

Example

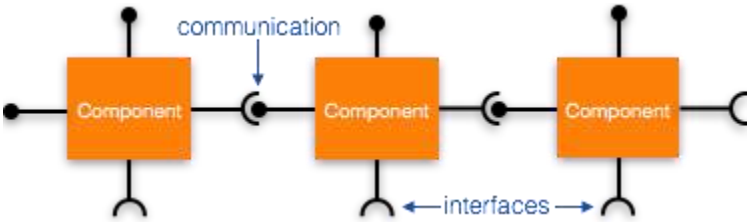
The login procedures used on the web can be considered as components, printing system in software can be seen as a component of the software.

Components have high cohesion of functionality and lower rate of coupling, i.e. they work independently and can perform tasks without depending on other modules.

In OOP, the objects are designed are very specific to their concern and have fewer chances to be used in some other software.

In modular programming, the modules are coded to perform specific tasks which can be used across number of other software programs.

There is a whole new vertical, which is based on re-use of software component, and is known as Component Based Software Engineering (CBSE).



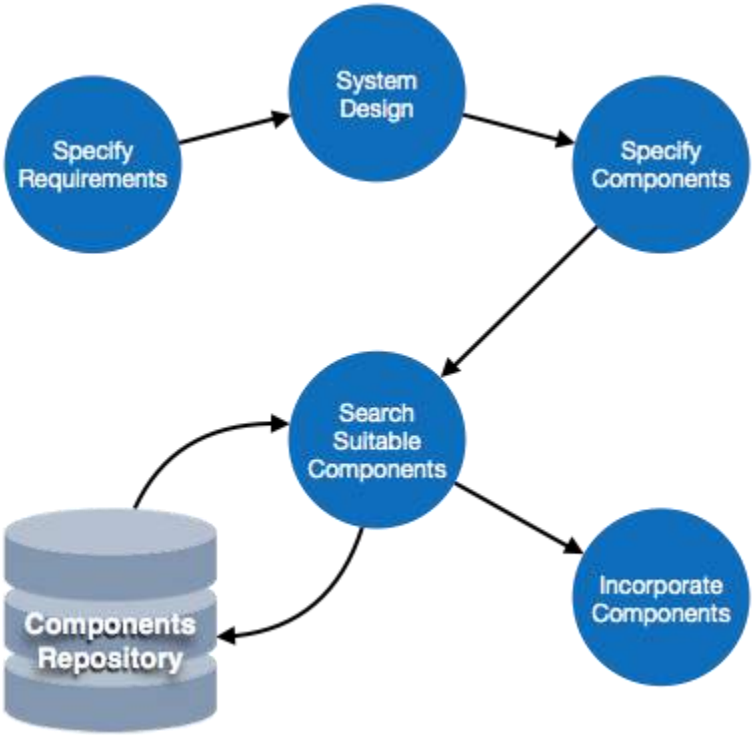
Re-use can be done at various levels

- **Application level** - Where an entire application is used as sub-system of new software.
- **Component level** - Where sub-system of an application is used.
- **Modules level** - Where functional modules are re-used.

  Software components provide interfaces, which can be used to establish communication among different components.

Reuse Process

Two kinds of method can be adopted: either by keeping requirements same and adjusting components or by keeping components same and modifying requirements.



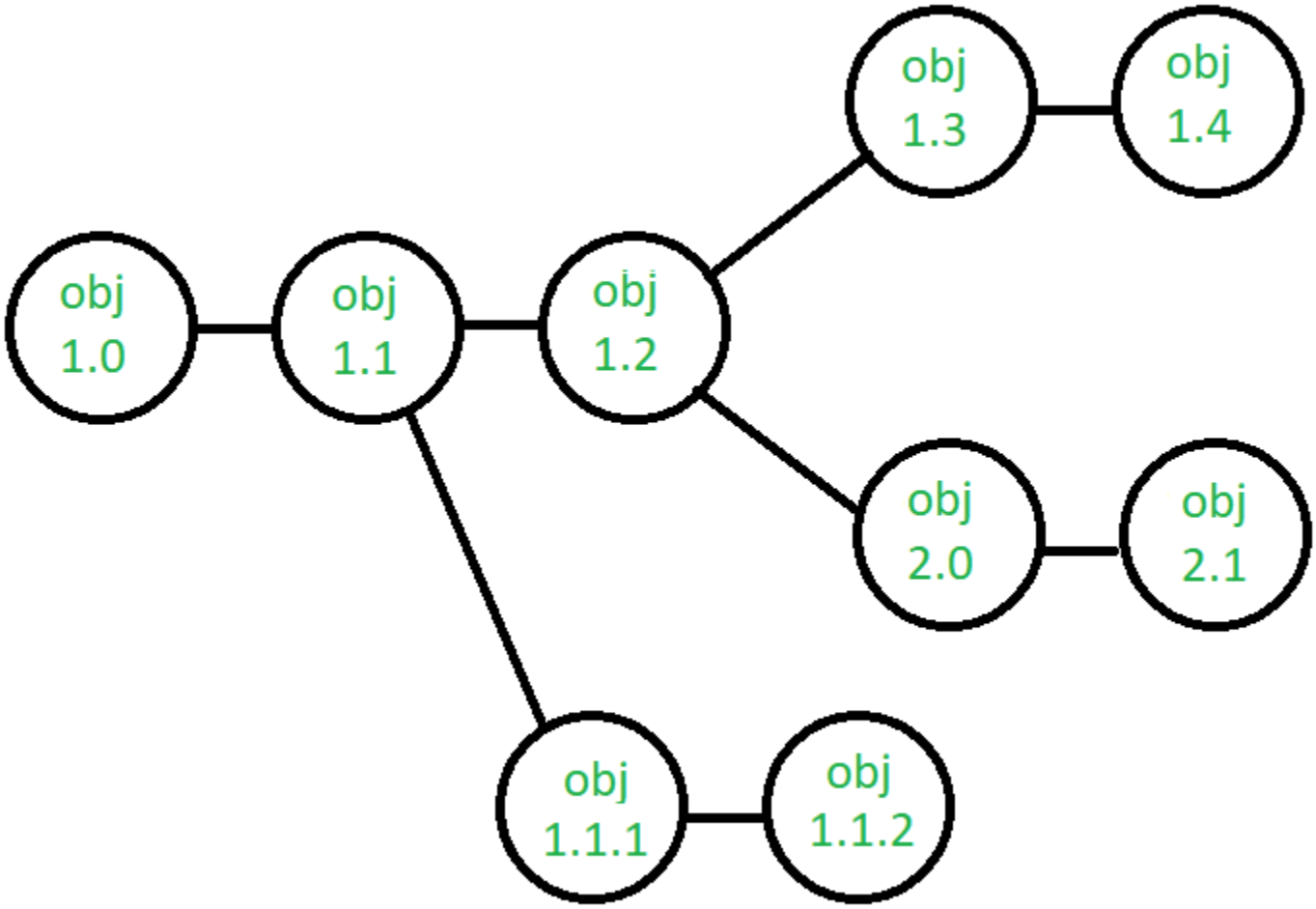
- **Requirement Specification** - The functional and non-functional requirements are specified, which a software product must comply to, with the help of existing system, user input or both.
- **Design** - This is also a standard SDLC process step, where requirements are defined in terms of software parlance. Basic architecture of system as a whole and its sub-systems are created.
- **Specify Components** - By studying the software design, the designers segregate the entire system into smaller components or sub-systems. One complete software design turns into a collection of a huge set of components working together.
- **Search Suitable Components** - The software component repository is referred by designers to search for the matching component, on the basis of functionality and intended software requirements..
- **Incorporate Components** - All matched components are packed together to shape them as complete software.

System Configuration Management (SCM) is an arrangement of exercises which controls change by recognizing the items for change, setting up connections between those things, making/characterizing instruments for overseeing diverse variants, controlling the changes being executed in the current framework, inspecting and revealing/reporting on the changes made. It is essential to control the changes in light of the fact that if the changes are not checked legitimately then they may wind up undermining a well-run programming. In this way, SCM is a fundamental piece of all project management activities.

Processes involved in SCM –
Configuration management provides a disciplined environment for smooth control of work products. It involves the following activities:

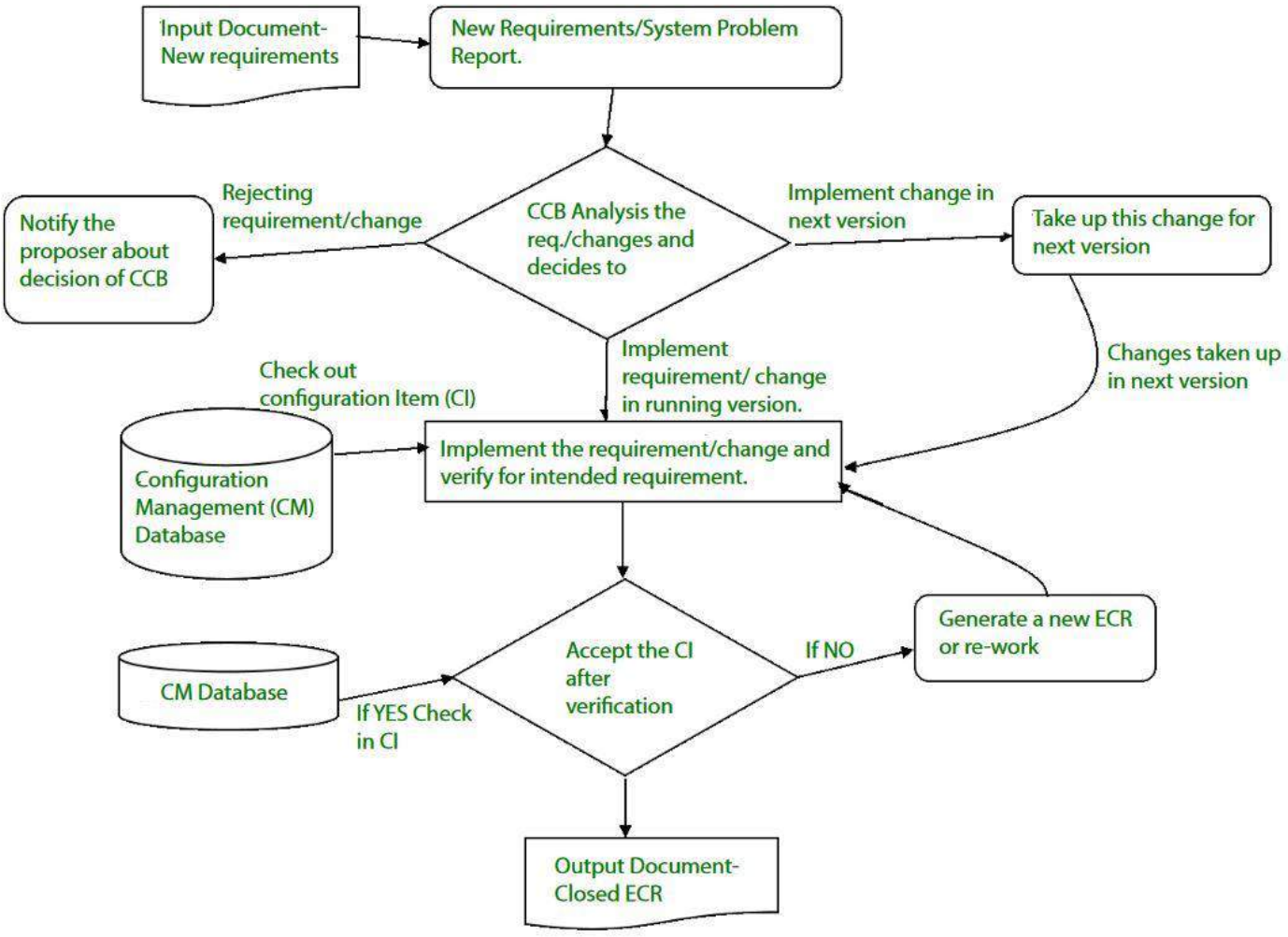
Identification and Establishment – Identifying the configuration items from products that compose baselines at given points in time (a baseline is a set of mutually consistent Configuration Items, which has been formally reviewed and agreed upon, and serves as the basis of further development). Establishing relationship among items, creating a mechanism to manage multiple level of control and procedure for change management system.

Version control – Creating versions/specifications of the existing product to build new products from the help of SCM system. A description of version is given below:



Suppose after some changes, the version of configuration object changes from 1.0 to 1.1. Minor corrections and changes result in versions 1.1.1 and 1.1.2, which is followed by a major update that is object 1.2. The development of object 1.0 continues through 1.3 and 1.4, but finally, a noteworthy change to the object results in a new evolutionary path, version 2.0. Both versions are currently supported.

 Change control – Controlling changes to Configuration items (CI). The change control process is explained in Figure below:



A change request (CR) is submitted and evaluated to assess technical merit, potential side effects, overall impact on other configuration objects and system functions, and the projected cost of the change. The results of the evaluation are presented as a change report, which is used by a change control board (CCB) —a person or group who makes a final decision on the status

and priority of the change. An engineering change Request (ECR) is generated for each approved change.

Also CCB notifies the developer in case the change is rejected with proper reason. The ECR describes the change to be made, the constraints that must be respected, and the criteria for review and audit. The object to be changed is "checked out" of the project database, the change is made, and then the object is tested again. The object is then "checked in" to the database and appropriate version control mechanisms are used to create the next version of the software.

 Configuration auditing – A software configuration audit complements the formal technical review of the process and product. It focuses on the technical correctness of the configuration object that has been modified. The audit confirms the completeness, correctness and consistency of items in the SCM system and track action items from the audit to closure.
 Reporting – Providing accurate status and current configuration data to developers, tester, end users, customers and stakeholders through admin guides, user guides, FAQs, Release notes, Memos, Installation Guide, Configuration guide etc .

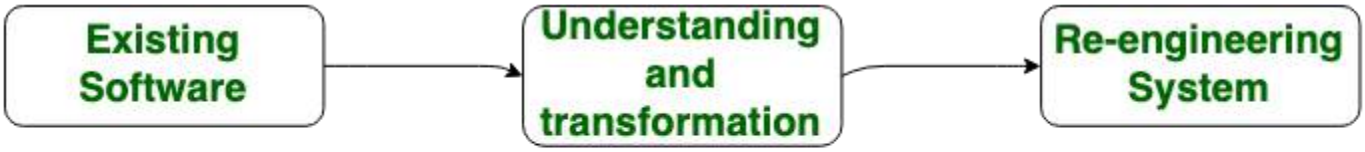
SCM Tools –
Different tools are available in market for SCM like: CFEngine, Bcfg2 server, Vagrant, SmartFrog, CLEAR CASETOOL (CC), SaltStack, CLEAR QUEST TOOL, Puppet, SVN-Subversion, Perforce, TortoiseSVN, IBM Rational team concert, IBM Configuration management version management, Razor, Ansible, etc. There are many more in the list.
It is recommended that before selecting any configuration management tool, have a proper understanding of the features and select the tool which best suits your project needs and be clear with the benefits and drawbacks of each before you choose one to use.

Difference between Forward Engineering and Reverse Engineering

Forward Engineering:
Forward Engineering is a method of creating or making an application with the help of the given requirements. Forward engineering is also known as **Renovation and Reclamation**. Forward engineering is required high proficiency skill. It takes more time to construct or develop an application.

Forward Engineering

Reverse Engineering:
Reverse Engineering is also known as backward engineering, is the process of forward engineering in reverse. In this, the information are collected from the given or exist application. It takes less time than forward engineering to develop an application. In reverse engineering the application are broken to extract knowledge or its architecture.



Reverse Engineering

Difference between Forward Engineering and Reverse Engineering:

S.NO	FORWARD ENGINEERING	REVERSE ENGINEERING
1.	In forward engineering, the application are developed with the given requirements.	In reverse engineering or backward engineering, the information are collected from the given application.

2.	Forward Engineering is high proficiency skill.	Reverse Engineering or backward engineering is low proficiency skill.
3.	Forward Engineering takes more time to develop an application.	While Reverse Engineering or backward engineering takes less time to develop an application.
4.	The nature of forward engineering is Prescriptive.	The nature of reverse engineering or backward engineering is Adaptive.
5.	In forward engineering, production is started with given requirements.	In reverse engineering, production is started by taking existing product.
6.	The example of forward engineering are construction of electronic kit, construction DC MOTOR etc.	The example of backward engineering are research on Instruments etc.

Software Re-engineering

Software Re-engineering is a process of software development which is done to improve the maintainability of a software system. Re-engineering is the examination and alteration of a system to reconstitute it in a new form. This process encompasses a combination of sub-processes like reverse engineering, forward engineering, reconstructing etc.
Re-engineering is the reorganizing and modifying existing software systems to make them more maintainable.

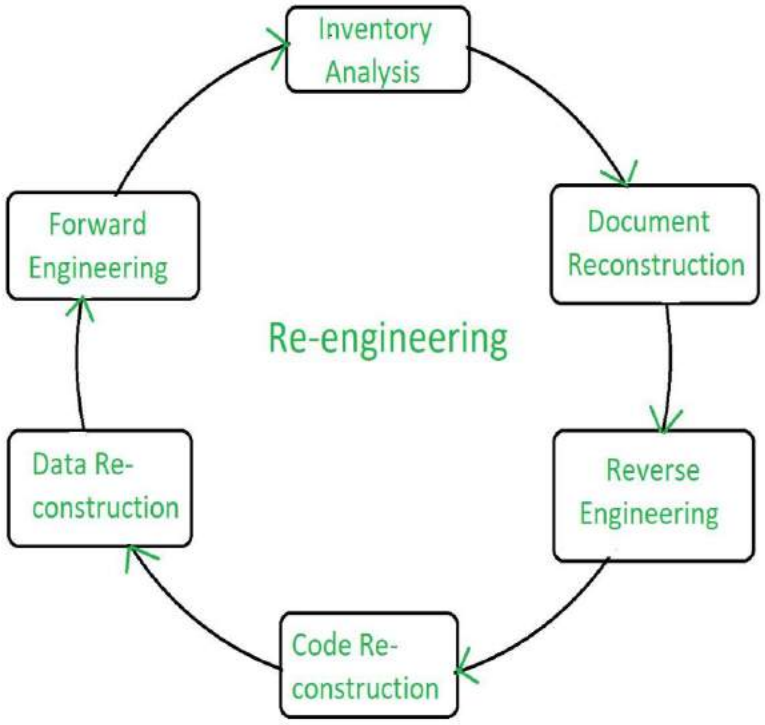
Objectives of Re-engineering:

- To describe a cost-effective option for system evolution.
- To describe the activities involved in the software maintenance process.
- To distinguish between software and data re-engineering and to explain the problems of data re-engineering.

Steps involved in Re-engineering:

1. Inventory Analysis
2. Document Reconstruction
3. Reverse Engineering
4. Code Reconstruction
5. Data Reconstruction
6. Forward Engineering

Diagrammatic Representation:



Re-engineering Cost Factors:

- The quality of the software to be re-engineered
- The tool support available for re-engineering
- The extent of the required data conversion
- The availability of expert staff for re-engineering

Advantages of Re-engineering:

- **Reduced Risk:**
  As the software is already existing, the risk is less as compared to new software development. Development problems, staffing problems and specification problems are the lots of problems which may arise in new software development.
- **Reduced Cost:**
  The cost of re-engineering is less than the costs of developing new software.

Software Engineering:

Software engineering covers not only the technical aspects of building software systems, but also management issues, such as directing programming teams, scheduling, and budgeting.

Software engineering is an engineering branch associated with development of software product using well-defined scientific principles, methods and procedures. The outcome of software engineering is an efficient and reliable software product.

Software project management has wider scope than software engineering process as it involves communication, pre and post-delivery support etc.

The need of software engineering arises because of higher rate of change in user requirements and environment on which the software is working.

- **Large software-** It is easier to build a wall than to a house or building, likewise, as the size of software become large engineering has to step to give it a scientific process.
- **Scalability-** If the software process were not based on scientific and engineering concepts, it would be easier to re-create new software than to scale an existing one.
- **Cost-** As hardware industry has shown its skills and huge manufacturing has lower down the price of computer and electronic hardware. But the cost of software remains high if proper process is not adapted.
- **Dynamic Nature-** The always growing and adapting nature of software hugely depends upon the environment in which user works. If the nature of software is always changing, new enhancements need to be done in the existing one. This is where software engineering plays a good role.
- **Quality Management-** Better process of software development provides better and quality software product.

RE Engineering:

• Restructuring or rewriting part or all of a system without changing its functionality

• Applicable when some (but not all) subsystems of a larger system require frequent maintenance

• Reengineering involves putting in the effort to make it easier to maintain

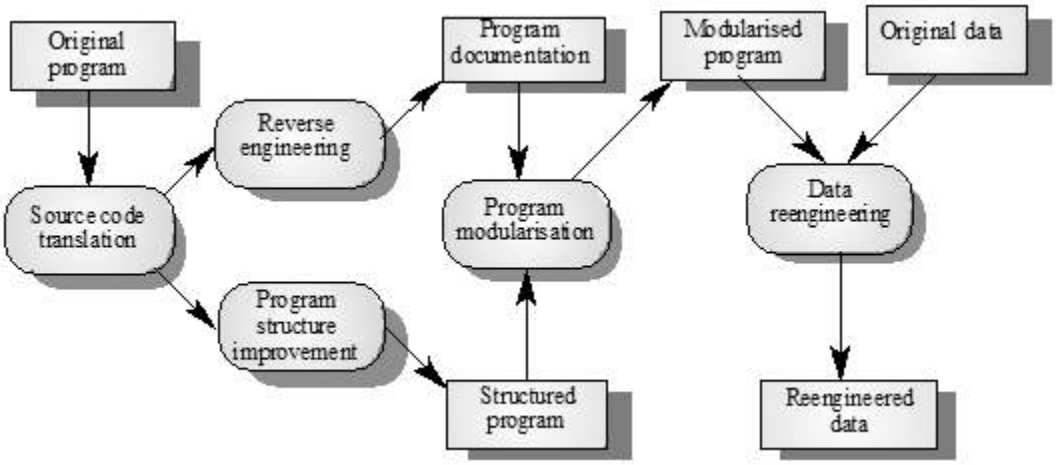
• The reengineered system may also be restructured and should be re-documented

When do you decide to reengineer?

• When system changes are confined to one subsystem, the subsystem needs to be reengineered

• When hardware or software support becomes obsolete

• When tools to support restructuring are readily available



Economics of Reengineering:

• **Cost of maintenance:** cost annual of operation and maintenance over application lifetime

• **Cost of reengineering:** predicted return on investment reduced by cost of implementing changes and engineering risk factors

• **Cost benefit:** Cost of re engineering - Cost of maintenance

Re-engineering advantages:

Reduced risk

There is a high risk in new software development. There may be development problems, staffing problems and specification problems

Reduced cost

The cost of re-engineering is often significantly less than the costs of developing new software

The complete Software Re-Engineering lifecycle includes:

Product Management: Risks analysis, root cause analysis, business analysis, requirements elicitation and management, product planning and scoping, competitive analysis

Research and Innovation: Definition of a problem, data gathering and analysis, identifying a solution and developing best-of-breed or innovative algorithms, verification of quality for data and results, patent preparation

Product Development: Technology analysis and selection, software architecture and design, data architecture, deployment architecture, prototyping and production code development, comprehensive software testing, data quality testing, and product packaging and deployment preparation

Product Delivery and Support: Hardware/Platform analysis and selection, deployment and release procedures definition, installations and upgrades, tracking support issues, organizing maintenance releases.

Project Management: Brings efficiency and productivity to your software re-engineering project by utilizing modern, practical software project management, software quality assurance, data quality assurance, and advanced risk management techniques.

Reverse Engineering:

Reverse engineering is taking apart an object to see how it works in order to duplicate or enhance the object. The practice, taken from older industries, is now frequently used on computer hardware and software. Software reverse engineering involves reversing a program's machine code (the string of 0s and 1s that are sent to the logic processor) back into the source code that it was written in, using program language statements.

Reverse-engineering is used for many purposes: as a learning tool; as a way to make new, compatible products that are cheaper than what's currently on the market; for making software interoperate more effectively or to bridge data between different operating systems or databases; and to uncover the undocumented features of commercial products.

Following are reasons for reverse engineering a part or product:

1. The original manufacturer of a product no longer produces a product
2. There is inadequate documentation of the original design
3. The original manufacturer no longer exists, but a customer needs the product
4. The original design documentation has been lost or never existed
5. Some bad features of a product need to be designed out. For example, excessive wear might indicate where a product should be improved
6. To strengthen the good features of a product based on long-term usage of the product

7. To analyze the good and bad features of competitors' product
8. To explore new avenues to improve product performance and features
9. To gain competitive benchmarking methods to understand competitor's products and develop better products
10. The original CAD model is not sufficient to support modifications or current manufacturing methods
11. The original supplier is unable or unwilling to provide additional parts
12. The original equipment manufacturers are either unwilling or unable to supply replacement parts, or demand inflated costs for sole-source parts
13. To update obsolete materials or antiquated manufacturing processes with more current, less-expensive technologies.

Software Project Management (SPM)

Software Project Management (SPM) is a proper way of planning and leading software projects. It is a part of project management in which software projects are planned, implemented, monitored and controlled.

Need of Software Project Management:

Software is an non-physical product. Software development is a new stream in business and there is very little experience in building software products. Most of the software products are made to fit client's requirements. The most important is that the basic technology changes and advances so frequently and rapidly that experience of one product may not be applied to the other one. Such type of business and environmental constraints increase risk in software development hence it is essential to manage software projects efficiently.

It is necessary for an organization to deliver quality product, keeping the cost within client's budget constrain and deliver the project as per scheduled. Hence in order, software project management is necessary to incorporate user requirements along with budget and time constraints.

Software Project Management consists of several different type of managements:

1. **Conflict Management:**
   Conflict management is the process to restrict the negative features of conflict while increasing the positive features of conflict. The goal of conflict management is to improve learning and group results including efficacy or performance in an organizational setting. Properly managed conflict can enhance group results.

2. **Risk Management:**
   Risk management is the analysis and identification of risks that is followed by synchronized and economical implementation of resources to minimize, operate and control the possibility or effect of unfortunate events or to maximize the realization of opportunities.

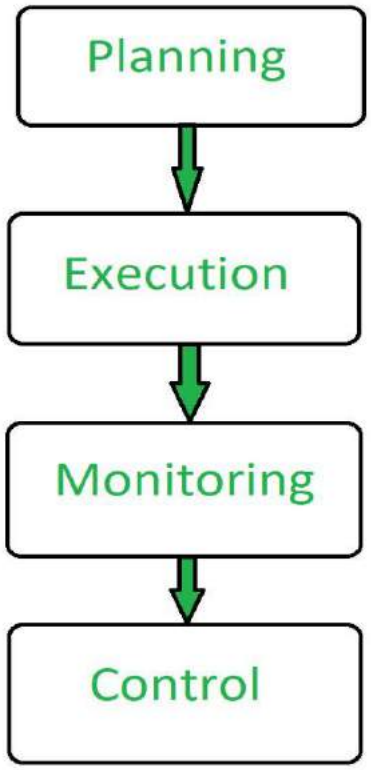
3. **Requirement Management:**
   It is the process of analyzing, prioritizing, tracing and documenting on requirements and then supervising change and communicating to pertinent stakeholders. It is a continuous process during a project.

4. **Change Management:**
   Change management is a systematic approach for dealing with the transition or transformation of an organization's goals, processes or technologies. The purpose of change management is to execute strategies for effecting change, controlling change and helping people to adapt to change.

5. **Software Configuration Management:**
   Software configuration management is the process of controlling and tracing changes in the software, part of the larger cross-disciplinary field of configuration management. Software configuration management include revision control and the inauguration of baselines.

6. **Release Management:**
   Release Management is the task of planning, controlling and scheduling the build in deploying releases. Release management ensures that organization delivers new and enhanced services required by the customer, while protecting the integrity of existing services.

Aspects of Software Project Management:



Advantages of Software Project Management:

- It helps in planning of software development.
- Implementation of software development is made easy.
- Monitoring and controlling are aspects of software project management.
- It overall manages to save time and cost for software development.

Feasibility is defined as the practical extent to which a project can be performed successfully. To evaluate feasibility, a feasibility study is performed, which determines whether the solution considered to accomplish the requirements is practical and workable in the software. Information such as resource availability, cost estimation for software development, benefits of the software to the organization after it is developed and cost to be incurred on its maintenance are considered during the feasibility study. The objective of the feasibility study is to establish the reasons for developing the software that is acceptable to users, adaptable to change and conformable to established standards. Various other objectives of feasibility study are listed below.

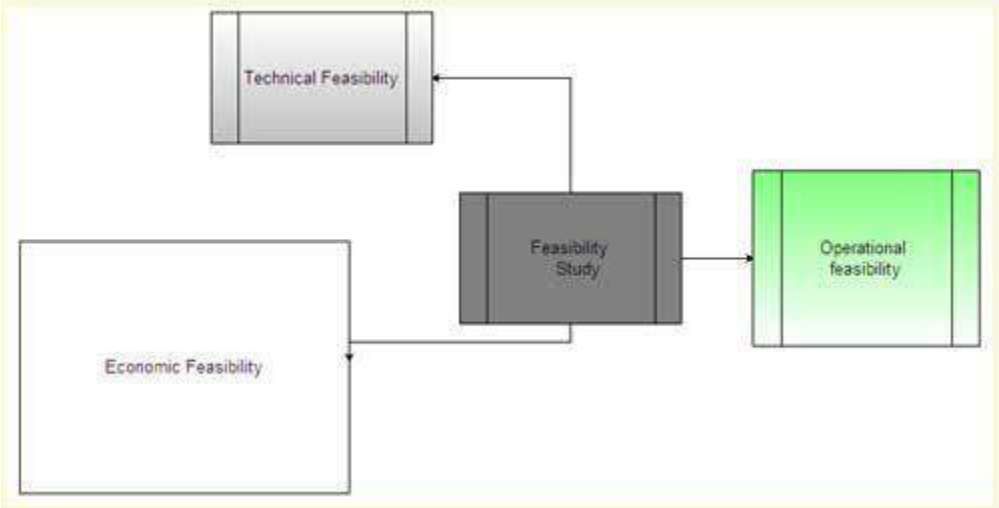
• To analyze whether the software will meet organizational requirements.

• To determine whether the software can be implemented using the current technology and within the specified budget and schedule.

• To determine whether the software can be integrated with other existing software.

Types of Feasibility

Various types of feasibility that are commonly considered include technical feasibility, operational feasibility, and economic feasibility.



Technical feasibility assesses the current resources (such as hardware and software) and technology, which are required to accomplish user requirements in the software within the allocated time and budget. For this, the software development team ascertains whether the current resources and technology can be upgraded or added in the software to accomplish specified user requirements. Technical feasibility also performs the following tasks.

• Analyzes the technical skills and capabilities of the software development team members.

• Determines whether the relevant technology is stable and established.

• Ascertains that the technology chosen for software development has a large number of users so that they can be consulted when problems arise or improvements are required.

Operational feasibility assesses the extent to which the required software performs a series of steps to solve business problems and user requirements. This feasibility is dependent on human resources (software development team) and involves visualizing whether the software will operate after it is developed and be operative once it is installed. Operational feasibility also performs the following tasks.

• Determines whether the problems anticipated in user requirements are of high priority.

• Determines whether the solution suggested by the software development team is acceptable.

• Analyzes whether users will adapt to a new software.

• Determines whether the organization is satisfied by the alternative solutions proposed by the software development team.

Economic feasibility determines whether the required software is capable of generating financial gains for an organization. It involves the cost incurred on the software development team, estimated cost of hardware and software, cost of performing feasibility study, and so on. For this, it is essential to consider expenses made on purchases (such as hardware purchase) and activities required to carry out software development. In addition, it is necessary to consider the benefits that can be achieved by developing the software. Software is said to be economically feasible if it focuses on the issues listed below.

• Cost incurred on software development to produce long-term gains for an organization.

• Cost required to conduct full software investigation (such as requirements elicitation and requirements analysis).

• Cost of hardware, software, development team, and training.

Feasibility Study Process

Feasibility study comprises the following steps.

• **Information assessment:** Identifies information about whether the system helps in achieving the objectives of the organization. It also verifies that the system can be implemented using new technology and within the budget and whether the system can be integrated with the existing system.

• **Information collection:** Specifies the sources from where information about software can be obtained. Generally, these sources include users (who will operate the software), organization (where the software will be used), and the software development team (which understands user requirements and knows how to fulfill them in software).

• **Report writing:** Uses a feasibility report, which is the conclusion of the feasibility study by the software development team. It includes the recommendations whether the software development should continue. This report may also include information about changes in the software scope, budget, and schedule and suggestions of any requirements in the system.

• **General information:** Describes the purpose and scope of feasibility study. It also describes system overview, project references, acronyms and abbreviations, and points of contact to be used. **System overview** provides description about the name of the organization responsible for the software development, system name or title, system category, operational status, and so on. **Project references** provide a list of the references used to prepare this document such as documents relating to the project or previously developed documents that are related to the project. **Acronyms and abbreviations** provide a list of the terms that are used in this document along with their meanings. **Points of contact** provide a list of points of organizational contact with users for information and coordination. For example, users require assistance to solve problems (such as troubleshooting) and collect information such as contact number, e-mail address, and so on.

Management summary: Provides the following information.

• **Environment:** Identifies the individuals responsible for software development. It provides information about input and output requirements, processing requirements of the software and the interaction of the software with other software. It also identifies system security requirements and the system's processing requirements

• **Current functional procedures:** Describes the current functional procedures of the existing system, whether automated or manual. It also includes the data-flow of the current system and the number of team members required to operate and maintain the software.

• **Functional objective:** Provides information about functions of the system such as new services, increased capacity, and so on.

• **Performance objective:** Provides information about performance objectives such as reduced staff and equipment costs, increased processing speeds of software, and improved controls.

• **Assumptions and constraints:** Provides information about assumptions and constraints such as operational life of the proposed software, financial constraints, changing hardware, software and operating environment, and availability of information and sources.

• **Methodology:** Describes the methods that are applied to evaluate the proposed software in order to reach a feasible alternative. These methods include survey, modeling, benchmarking, etc.

• **Evaluation criteria:** Identifies criteria such as cost, priority, development time, and ease of system use, which are applicable for the development process to determine the most suitable system option.

• **Recommendation:** Describes a recommendation for the proposed system. This includes the delays and acceptable risks.

• **Proposed software:** Describes the overall concept of the system as well as the procedure to be used to meet user requirements. **In** addition, it provides information about improvements, time and resource costs, and impacts. Improvements are performed to enhance the functionality and performance of the existing software. Time and resource costs include the costs associated with software development from its requirements to its maintenance and staff training. Impacts describe the possibility of future happenings and include various types of impacts as listed below.

• **Equipment impacts:** Determine new equipment requirements and changes to be made in the currently available equipment requirements.

• **Software impacts:** Specify any additions or modifications required in the existing software and supporting software to adapt to the proposed software.

• **Organizational impacts:** Describe any changes in organization, staff and skills requirement.

• **Operational impacts:** Describe effects on operations such as user-operating procedures, data processing, data entry procedures, and so on.

• **Developmental impacts:** Specify developmental impacts such as resources required to develop databases, resources required to develop and test the software, and specific activities to be performed by users during software development.

• **Security impacts:** Describe security factors that may influence the development, design, and continued operation of the proposed software.

• **Alternative systems:** Provide description of alternative systems, which are considered in a feasibility study. This also describes the reasons for choosing a particular alternative system to develop the proposed software and the reason for rejecting alternative systems.

Project Planning

Once a project is found to be possible, computer code project managers undertake project designing. Project designing is undertaken and completed even before any development activity starts. Project designing consists of subsequent essential activities:

Estimating the subsequent attributes of the project:

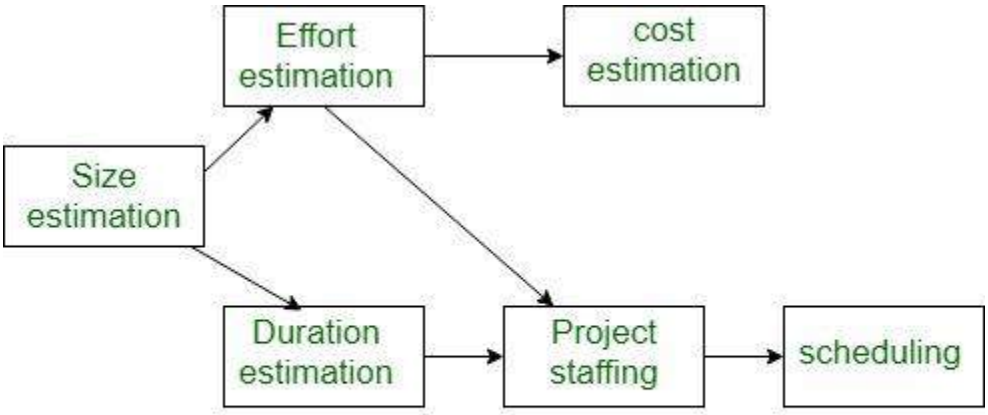
- **Project size:**
  What's going to be downside quality in terms of the trouble and time needed to develop the product?
- **Cost:**
  What proportion is it reaching to value to develop the project?
- **Duration:**
  However long is it reaching to want complete development?
- **Effort:**
  What proportion effort would be required?

The effectiveness of the following designing activities relies on the accuracy of those estimations.

- planning force and alternative resources
- workers organization and staffing plans
- Risk identification, analysis, and abatement designing
- Miscellaneous arranges like quality assurance plan, configuration, management arrange, etc.

Precedence ordering among project planning activities:

The different project connected estimates done by a project manager have already been mentioned. The below diagram shows the order during which vital project coming up with activities is also undertaken. It may be simply discovered that size estimation is that the 1st activity. It's conjointly the foremost basic parameter supported that all alternative coming up with activities square measure dispensed, alternative estimations like the estimation of effort, cost, resource, and project length also are vital elements of the project coming up with.



Precedence ordering among planning activities

Sliding Window Planning:
Project designing needs utmost care and a spotlight since commitment to unrealistic time and resource estimates end in schedule slippage. Schedule delays will cause client discontent and adversely have an effect on team morale. It will even cause project failure.
However, project designing could be a terribly difficult activity. particularly for giant comes, it's pretty much troublesome to create correct plans. A region of this issue is thanks to the actual fact that the correct parameters, the scope of the project, project workers, etc. might amendment throughout the span of the project. So as to beat this drawback, generally project managers undertake project designing little by little. Designing a project over a variety of stages protects managers from creating huge commitments too early. This method of staggered designing is thought of as window designing. Within the window technique, beginning with associate initial set up, the project is planned additional accurately in sequential development stages.

At the beginning of a project, project managers have incomplete information concerning the main points of the project. Their info base step by step improves because the project progresses through completely different phases. When the completion of each section, the project managers will set up every ulterior section additional accurately and with increasing levels of confidence.

Software Project Manager

A software project manager is a person who undertakes the responsibility of executing the software project. Software project manager is thoroughly aware of all the phases of SDLC that

the software would go through. Project manager may never directly involve in producing the end product but he controls and manages the activities involved in production.

A project manager closely monitors the development process, prepares and executes various plans, arranges necessary and adequate resources, maintains communication among all team members in order to address issues of cost, budget, resources, time, quality and customer satisfaction.

Let us see few responsibilities that a project manager shoulders -

Managing People

- Act as project leader
- Liaison with stakeholders
- Managing human resources
- Setting up reporting hierarchy etc.

Managing Project

- Defining and setting up project scope
- Managing project management activities
- Monitoring progress and performance
- Risk analysis at every phase
- Take necessary step to avoid or come out of problems
- Act as project spokesperson

Software Management Activities

Software project management comprises of a number of activities, which contains planning of project, deciding scope of software product, estimation of cost in various terms, scheduling of tasks and events, and resource management. Project management activities may include:

- **Project Planning**
- **Scope Management**
- **Project Estimation**

Project Planning

Software project planning is task, which is performed before the production of software actually starts. It is there for the software production but involves no concrete activity that has any direction connection with software production; rather it is a set of multiple processes, which facilitates software production. Project planning may include the following:

Scope Management

It defines the scope of project; this includes all the activities, process need to be done in order to make a deliverable software product. Scope management is essential because it creates boundaries of the project by clearly defining what would be done in the project and what would not be done. This makes project to contain limited and quantifiable tasks, which can easily be documented and in turn avoids cost and time overrun.

During Project Scope management, it is necessary to -

- Define the scope
- Decide its verification and control
- Divide the project into various smaller parts for ease of management.
- Verify the scope
- Control the scope by incorporating changes to the scope

Project Estimation

For an effective management accurate estimation of various measures is a must. With correct estimation managers can manage and control the project more efficiently and effectively.

Project estimation may involve the following:

- **Software size estimation**

  Software size may be estimated either in terms of KLOC (Kilo Line of Code) or by calculating number of function points in the software. Lines of code depend upon coding practices and Function points vary according to the user or software requirement.

- **Effort estimation**

  The managers estimate efforts in terms of personnel requirement and man-hour required to produce the software. For effort estimation software size should be known. This can either be derived by managers' experience, organization's historical data or software size can be converted into efforts by using some standard formulae.

- **Time estimation**

  Once size and efforts are estimated, the time required to produce the software can be estimated. Efforts required is segregated into sub categories as per the requirement specifications and interdependency of various components of software. Software tasks are divided into smaller tasks, activities or events by Work Breakthrough Structure (WBS). The tasks are scheduled on day-to-day basis or in calendar months.

  The sum of time required to complete all tasks in hours or days is the total time invested to complete the project.

- **Cost estimation**

  This might be considered as the most difficult of all because it depends on more elements than any of the previous ones. For estimating project cost, it is required to consider -

  - Size of software
  - Software quality
  - Hardware
  - Additional software or tools, licenses etc.
  - Skilled personnel with task-specific skills
  - Travel involved
  - Communication
  - Training and support

Project Estimation Techniques

We discussed various parameters involving project estimation such as size, effort, time and cost.

Project manager can estimate the listed factors using two broadly recognized techniques –

Decomposition Technique

This technique assumes the software as a product of various compositions.

There are two main models -

- **Line of Code** Estimation is done on behalf of number of line of codes in the software product.
- **Function Points** Estimation is done on behalf of number of function points in the software product.

Empirical Estimation Technique

This technique uses empirically derived formulae to make estimation.These formulae are based on LOC or FPs.

- **Putnam Model**

  This model is made by Lawrence H. Putnam, which is based on Norden's frequency distribution (Rayleigh curve). Putnam model maps time and efforts required with software size.

- **COCOMO**

  COCOMO stands for COnstructive COst MOdel, developed by Barry W. Boehm. It divides the software product into three categories of software: organic, semi-detached and embedded.

Project Scheduling

Project Scheduling in a project refers to roadmap of all activities to be done with specified order and within time slot allotted to each activity. Project managers tend to define various tasks, and project milestones and arrange them keeping various factors in mind. They look for tasks lie in critical path in the schedule, which are necessary to complete in specific manner (because of task interdependency) and strictly within the time allocated. Arrangement of tasks which lies out of critical path are less likely to impact over all schedule of the project.

For scheduling a project, it is necessary to -

- Break down the project tasks into smaller, manageable form
- Find out various tasks and correlate them
- Estimate time frame required for each task
- Divide time into work-units
- Assign adequate number of work-units for each task
- Calculate total time required for the project from start to finish

Resource management

All elements used to develop a software product may be assumed as resource for that project. This may include human resource, productive tools and software libraries.

The resources are available in limited quantity and stay in the organization as a pool of assets. The shortage of resources hampers the development of project and it can lag behind the schedule. Allocating extra resources increases development cost in the end. It is therefore necessary to estimate and allocate adequate resources for the project.

Resource management includes -

- Defining proper organization project by creating a project team and allocating responsibilities to each team member
- Determining resources required at a particular stage and their availability
- Manage Resources by generating resource request when they are required and de-allocating them when they are no more needed.

Project Risk Management

Risk management involves all activities pertaining to identification, analyzing and making provision for predictable and non-predictable risks in the project. Risk may include the following:

- Experienced staff leaving the project and new staff coming in.
- Change in organizational management.
- Requirement change or misinterpreting requirement.
- Under-estimation of required time and resources.
- Technological changes, environmental changes, business competition.

Risk Management Process

There are following activities involved in risk management process:

- **Identification -** Make note of all possible risks, which may occur in the project.
- **Categorize -** Categorize known risks into high, medium and low risk intensity as per their possible impact on the project.
- **Manage -** Analyze the probability of occurrence of risks at various phases. Make plan to avoid or face risks. Attempt to minimize their side-effects.
- **Monitor -** Closely monitor the potential risks and their early symptoms. Also monitor the effects of steps taken to mitigate or avoid them.

Project Execution & Monitoring

In this phase, the tasks described in project plans are executed according to their schedules.

Execution needs monitoring in order to check whether everything is going according to the plan. Monitoring is observing to check the probability of risk and taking measures to address the risk or report the status of various tasks.

These measures include -

- **Activity Monitoring -** All activities scheduled within some task can be monitored on day-to-day basis. When all activities in a task are completed, it is considered as complete.
- **Status Reports -** The reports contain status of activities and tasks completed within a given time frame, generally a week. Status can be marked as finished, pending or work-in-progress etc.
- **Milestones Checklist -** Every project is divided into multiple phases where major tasks are performed (milestones) based on the phases of SDLC. This milestone checklist is prepared once every few weeks and reports the status of milestones.

Project Communication Management

Effective communication plays vital role in the success of a project. It bridges gaps between client and the organization, among the team members as well as other stake holders in the project such as hardware suppliers.

Communication can be oral or written. Communication management process may have the following steps:

- **Planning** - This step includes the identifications of all the stakeholders in the project and the mode of communication among them. It also considers if any additional communication facilities are required.
- **Sharing** - After determining various aspects of planning, manager focuses on sharing correct information with the correct person on correct time. This keeps every one involved the project up to date with project progress and its status.
- **Feedback** - Project managers use various measures and feedback mechanism and create status and performance reports. This mechanism ensures that input from various stakeholders is coming to the project manager as their feedback.
- **Closure** - At the end of each major event, end of a phase of SDLC or end of the project itself, administrative closure is formally announced to update every stakeholder by sending email, by distributing a hardcopy of document or by other mean of effective communication.

After closure, the team moves to next phase or project.

Configuration Management

Configuration management is a process of tracking and controlling the changes in software in terms of the requirements, design, functions and development of the product.

IEEE defines it as "the process of identifying and defining the items in the system, controlling the change of these items throughout their life cycle, recording and reporting the status of items and change requests, and verifying the completeness and correctness of items".

Generally, once the SRS is finalized there is less chance of requirement of changes from user. If they occur, the changes are addressed only with prior approval of higher management, as there is a possibility of cost and time overrun.

Baseline

A phase of SDLC is assumed over if it baselined, i.e. baseline is a measurement that defines completeness of a phase. A phase is baselined when all activities pertaining to it are finished and well documented. If it was not the final phase, its output would be used in next immediate phase.

Configuration management is a discipline of organization administration, which takes care of occurrence of any change (process, requirement, technological, strategical etc.) after a phase is baselined. CM keeps check on any changes done in software.

Change Control

Change control is function of configuration management, which ensures that all changes made to software system are consistent and made as per organizational rules and regulations.

A change in the configuration of product goes through following steps -

- **Identification** - A change request arrives from either internal or external source. When change request is identified formally, it is properly documented.
- **Validation** - Validity of the change request is checked and its handling procedure is confirmed.
- **Analysis** - The impact of change request is analyzed in terms of schedule, cost and required efforts. Overall impact of the prospective change on system is analyzed.
- **Control** - If the prospective change either impacts too many entities in the system or it is unavoidable, it is mandatory to take approval of high authorities before change is incorporated into the system. It is decided if the change is worth incorporation or not. If it is not, change request is refused formally.
- **Execution** - If the previous phase determines to execute the change request, this phase take appropriate actions to execute the change, does a thorough revision if necessary.
- **Close request** - The change is verified for correct implementation and merging with the rest of the system. This newly incorporated change in the software is documented properly and the request is formally is closed.

Project Management Tools

The risk and uncertainty rises multifold with respect to the size of the project, even when the project is developed according to set methodologies.

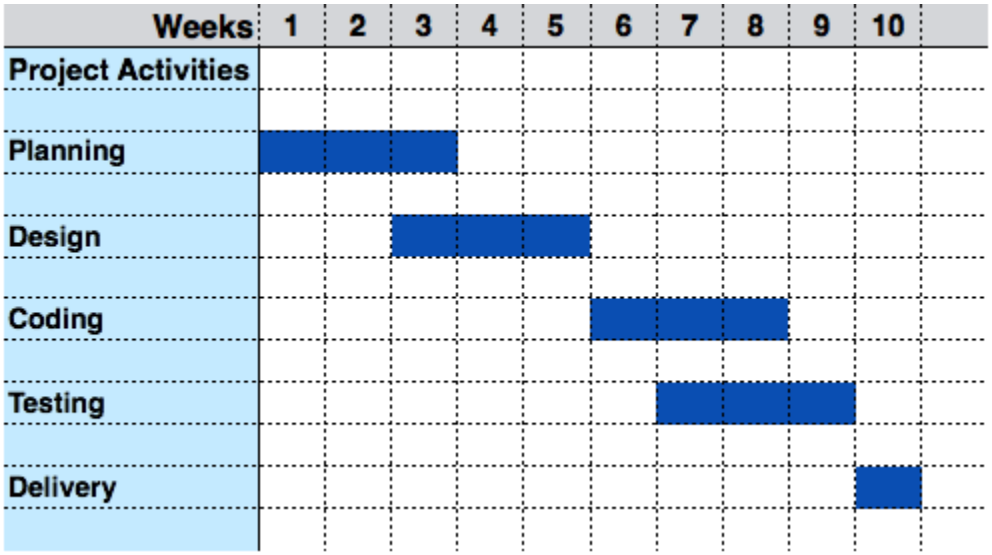
There are tools available, which aid for effective project management. A few are described -

Gantt Chart

Gantt charts was devised by Henry Gantt (1917). It represents project schedule with respect to time periods. It is a horizontal bar chart with bars representing activities and time scheduled for the project activities.

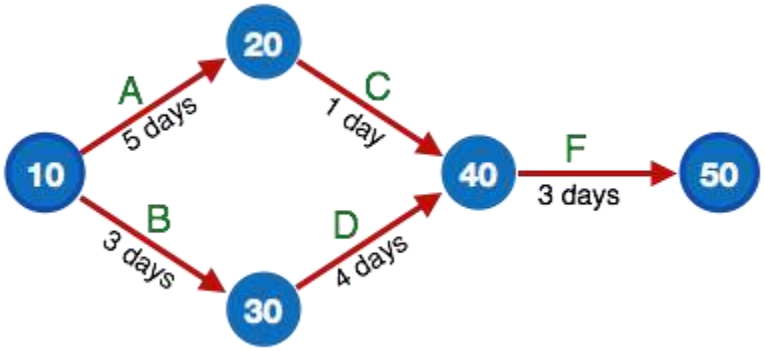
TRINITY INSTITUTE OF TECHNOLOGY AND RESEARCH BHOPAL

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



PERT Chart

PERT (Program Evaluation & Review Technique) chart is a tool that depicts project as network diagram. It is capable of graphically representing main events of project in both parallel and consecutive way. Events, which occur one after another, show dependency of the later event over the previous one.



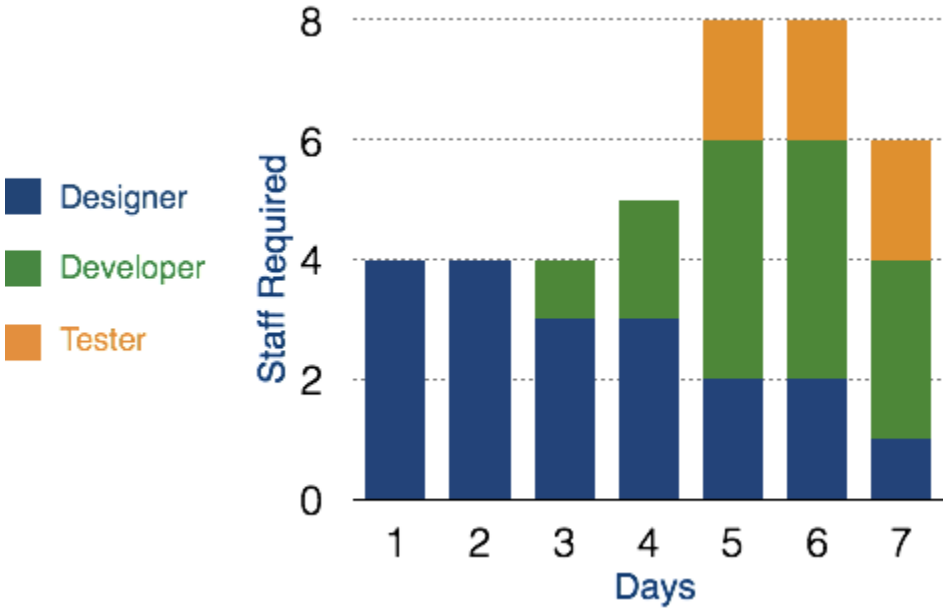
Events are shown as numbered nodes. They are connected by labeled arrows depicting sequence of tasks in the project.

Resource Histogram

This is a graphical tool that contains bar or chart representing number of resources (usually skilled staff) required over time for a project event (or phase). Resource Histogram is an effective tool for staff planning and coordination.

Prepared by : URMILA MAHOR

Staff	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7
Designer	4	4	3	3	2	2	1
Developer	0	0	1	2	4	4	3
Tester	0	0	0	0	2	2	2
Total	4	4	4	5	8	8	6



Critical Path Analysis

This tools is useful in recognizing interdependent tasks in the project. It also helps to find out the shortest path or critical path to complete the project successfully. Like PERT diagram, each event is allotted a specific time frame. This tool shows dependency of event assuming an event can proceed to next only if the previous one is completed.

The events are arranged according to their earliest possible start time. Path between start and end node is critical path which cannot be further reduced and all events require to be executed in same order.

Software Engineering | Risk Management

Risk Management:
A computer code project may be laid low with an outsized sort of risk. so as to be ready to consistently establish the necessary risks which could have an effect on a computer code project, it's necessary to reason risks into completely different categories. The project manager will then examine the risks from every category square measure relevant to the project.
There square measure 3 main classes of risks that may have an effect on a computer code project:

1. **Project Risks:**
   Project risks concern varies sorts of monetary fund, schedule, personnel, resource, and customer-related issues. a vital project risk is schedule slippage. Since computer code is intangible, it's terribly tough to observe and management a computer code project. it's terribly tough to manage one thing that can not be seen. For any producing project, like producing cars, the project manager will see the merchandise taking form.

   For example, see that the engine is fitted, at the moment the doors area unit fitted, the automotive is obtaining painted, etc. so he will simply assess the progress of the work and management it. The physical property of the merchandise being developed is a vital reason why several computer codes comes to suffer from the danger of schedule slippage.
2. **Technical Risks:**
   Technical risks concern potential style, implementation, interfacing, testing, and maintenance issues. Technical risks conjointly embody ambiguous specifications, incomplete specification, dynamic specification, technical uncertainty, and technical degeneration. Most technical risks occur thanks to the event team's lean information concerning the project.
3. **Business Risks:**
   This type of risk embodies the risks of building a superb product that nobody needs, losing monetary fund or personal commitments, etc.

Mitigating options include:

- **Accept:** Acknowledge that a risk is impacting the project. Make an explicit decision to accept the risk without any changes to the project. Project management approval is mandatory here.
- **Avoid:** Adjust project scope, schedule, or constraints to minimize the effects of the risk.
- **Control:** Take action to minimize the impact or reduce the intensification of the risk.
- **Transfer:** Implement an organizational shift in accountability, responsibility, or authority to other stakeholders that will accept the risk.
- **Continue Monitoring:** Often suitable for low-impact risks, monitor the project environment for potentially increasing impact of the risk.

Software Engineering | Software Quality Assurance

Software Quality Assurance (SQA) is simply a way to assure quality in the software. It is the set of activities which ensure processes, procedures as well as standards suitable for the project and implemented correctly.
Software Quality Assurance is a process which works parallel to development of a software. It focuses on improving the process of development of software so that problems can be prevented before they become a major issue. Software Quality Assurance is a kind of an Umbrella activity that is applied throughout the software process.

Software Quality Assurance have:

1. A quality management approach
2. Formal technical reviews
3. Multi testing strategy
4. Effective software engineering technology
5. Measurement and reporting mechanism

Major Software Quality Assurance Activities:

1. **SQA Management Plan:**
   Make a plan how you will carry out the sqa through out the project. Think which set of software engineering activities are the best for project.check level of sqa team skills.
2. **Set The Check Points:**
   SQA team should set checkpoints. Evaluate the performance of the project on the basis of collected data on different check points.
3. **Multi testing Strategy:**
   Do not depend on single testing approach. When you have lot of testing approaches available use them.
4. **Measure Change Impact:**
   The changes for making the correction of an error sometimes re introduces more errors keep the measure of impact of change on project. Reset the new change to change check the compatibility of this fix with whole project.
5. **Manage Good Relations:**
   In the working environment managing the good relation with other teams involved in the project development is mandatory. Bad relation of sqa team with programmers team will impact directly and badly on project. Don't play politics.

Benefits of Software Quality Assurance (SQA):

1. SQA produce high quality software.
2. High quality application saves time and cost.
3. SQA is beneficial for better reliability.

4. SQA is beneficial in the condition of no maintenance for long time.
5. High quality commercial software increase market share of company.
6. Improving the process of creating software.
7. Improves the quality of the software.

Disadvantage of SQA:

There are a number of disadvantages of quality assurance. Some of them include adding more resources, employing more workers to help maintain quality and so much more.

Software Measurement and Metrics

Software Measurement: A measurement is an manifestation of the size, quantity, amount or dimension of a particular attributes of a product or process. Software measurement is a titrate impute of a characteristic of a software product or the software process. It is an authority within software engineering. Software measurement process is defined and governed by ISO Standard.

Need of Software Measurement:

Software is measured to:

1. Create the quality of the current product or process.
2. Anticipate future qualities of the product or process.
3. Enhance the quality of a product or process.
4. Regulate the state of the project in relation to budget and schedule.

Classification of Software Measurement:

There are 2 types of software measurement:

1. **Direct Measurement:**
   In direct measurement the product, process or thing is measured directly using standard scale.
2. **Indirect Measurement:**
   In indirect measurement the quantity or quality to be measured is measured using related parameter i.e. by use of reference.

Metrics:

A metrics is a measurement of the level that any impute belongs to a system product or process. There are 4 functions related to software metrics:

1. Planning
2. Organizing
3. Controlling
4. Improving

Classification of Software Metrics:

There are 2 types of software metrics:

1. **Product Metrics:**
   Product metrics are used to evaluate the state of the product, tracing risks and

undercovering prospective problem areas. The ability of team to control quality is evaluated.

2. **Process Metrics:**
   Process metrics pay particular attention on enhancing the long term process of the team or organisation.
3. **Project Metrics:**
   Project matrix is describes the project characteristic and execution process.
   - Number of software developer
   - Staffing pattern over the life cycle of software
   - Cost and schedule
   - Productivity