UNIT - V

# 5 Software Maintenance and Software Project Measurement

## 5.1 : Need and Types of Maintenance

**Q.1 What is software maintenance?**

**Ans. :** • 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.

**Q.2 Explain the need for software maintenance**

**Ans. :** 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.

These are the factors that makes cost high.

**Q.3 What are different types of software maintenance ?**

**Ans. :**

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.

**Q.4 What is software maintenance ? Differentiate between perfective maintenance and preventive maintenance** ☞ **[RGPV : June.-10, Marks 10]**

**Ans. : Software maintenance** - Refer Q.1.

| Perfective Maintenance | Preventive Maintenance |
|---|---|
| Identify the injection points and build new functionality from this point. | This is kind of maintenance technique, in which defects are removed before the failure. |
| This technique focuses on organize and simplify the functionalities. | This technique focus on guessing and eliminating the faults. |

**Q.5 Describe the components of software maintenance process.**

**OR What are various software maintenance activities involved throughout its cycle ?**

☞ **[RGPV : Dec.-16, Marks 7]**

Ans. : The software maintenance process can be as shown Fig. Q.5.1.

1. **Change Request** - In the maintenance process initially the request for change is made.
2. **Change management** - In this phase the status of all the change requests is identified, described.
3. **Impact analysis** - Following activities are performed in this phase.
   i) Identify all systems and system products affected by a change request.
   ii) Make an estimate of the resources needed to effect the change.
   iii) Analyze the benefits of the change.
4. **System release planning** - In this phase the schedule and contents of software release is planned. The changes can be made to all types of software maintenance.
5. **Change implementation** - The implementation of changes can be done by first designing the changes, then coding for these changes and finally testing the changes. Preferably the regression testing must be performed while testing the changes.
6. **System release** - During the software release
   i) Documentation ii) Software iii) Training
   iv) Hardware changes
   v) Data conversion should be described.

**Q.6 Explain various problems that may arise during software maintenance process**

Ans. :

1. Changes in the software are not well documented. Hence it becomes difficult to trace the evolution of software through many versions or releases.
2. Sometimes it becomes difficult to trace the process through which the software was created.
3. While maintaining the software it is difficult to understand someone else's program.
4. Most of the software is not designed for a change. In other words the change in the software is not at all possible and needs to redesign it.
5. The software posses unstructured code.
6. Maintenance programmers have insufficient knowledge of the system or problem domain.
7. Appropriate documentation does not exists.
8. Developers do not like maintenance.

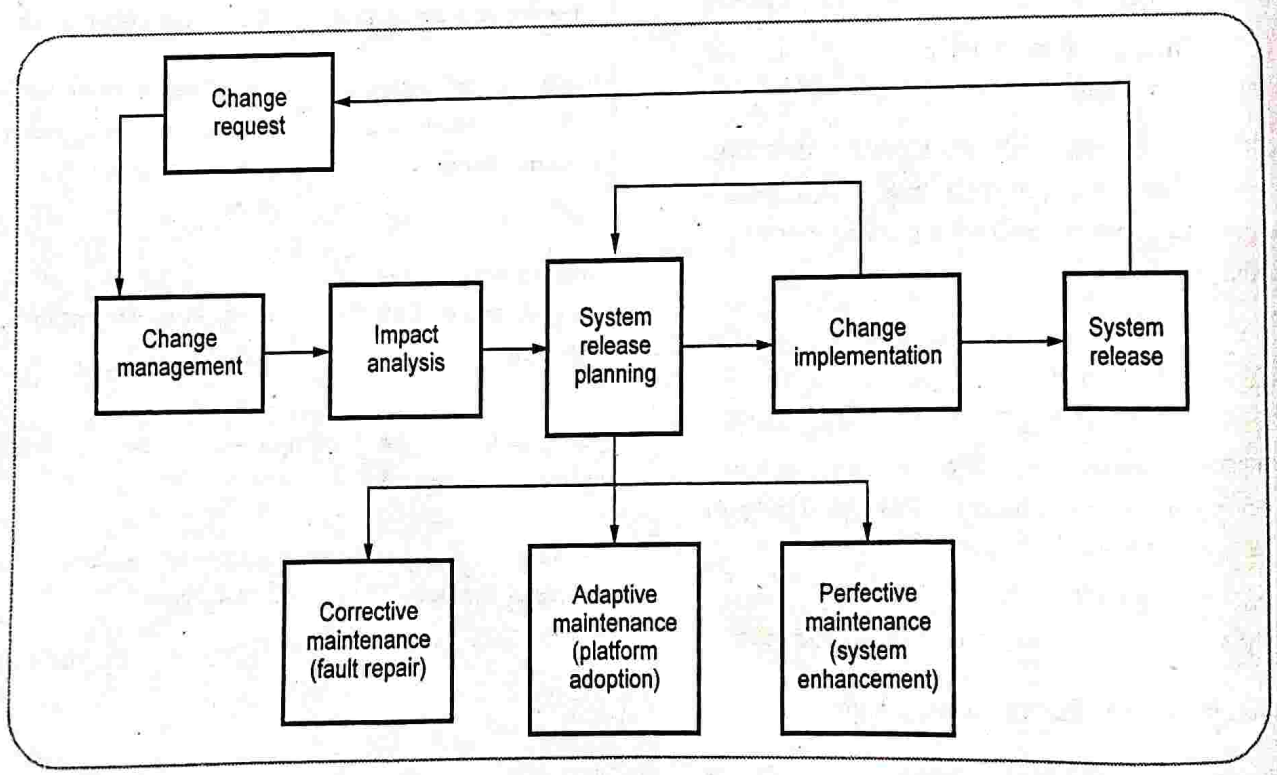



**Fig. Q.5.1 Maintenance process**

## 5.2 : Software Configuration Management (SCM)

**Q.7 What are the objectives and features supported by software configuration management ?**

☞ [RGPV : June-16, Marks 3]

**Ans. :** • 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 i) **Identify** these changes ii) **Control** the changes iii) Ensure that the changes are properly **implemented** and iv) **Report** these changes to others. And to carry out this set of activities the software configuration is essential.

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

**Q.8 Write short note - SCM.**

☞ [RGPV : Dec.-18, Marks 7]

**Ans. :** Software configuration management (SCM) is a set of activities carried out for identifying, organizing and controlling changes throughout the life cycle of computer software.

**SCM Activities**

1. Identify change 2. Control change
3. Ensure that change is being properly implemented
4. Report change to others who may have an interest

Software configuration management is a set of tracking and control activities that begin when a software development project begins and terminates when the software is taken out of operation.

**Q.9 What is SCM ? Explain the concept of baseline and SCM items in brief.**

☞ [RGPV : May-18, Marks 7]

**Ans. :** SCM : Refer Q.8

**BaseLine :** • The IEEE (IEEE Std. No. 610.12-1990) defines a baseline as :

"A specification or product that has been formally reviewed and agreed upon, that thereafter serves as the basis for further development, and that can be changed only through formal change control procedures".

• A baseline is a milestone in the development of software that is marked by the delivery of one or more software configuration items and the approval of them is obtained through formal technical review

• The baseline is the shared project database. It is an SCM task to maintain the integrity of the set of artifacts.

• The elements of a design model have been first documented and reviewed. From this design model errors are identified and corrected. Once all parts of the model have been reviewed, corrected and then approved, the design model becomes a baseline.

• Further changes to the program architecture (which is actually documented in the design model) can be made only after each has been evaluated and approved.

**SCM Items :**

**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

## 5.3 : Software Change Management

**Q.10 Brief the software configuration management and its process.** ☞ [RGPV : Dec.-17, Marks 7]

**Ans. :** Various tasks that are carried out in SCM process are -

**1. Identification**

• Each SCI must be named and identified as objects.

- Software configuration item can be organized to form a database or repository of configuration objects or basic objects
- The configuration object consists of
  1. Name and Description
  2. Attributes and reference pointer to the object in the database
  3. Relationships to other configuration 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.

**2. Version control**

- The configuration objects become the baseline, at baseline it becomes
  version 1.0. Subsequent changes in baseline become new versions.
- The version control combines procedures and tools to manage different versions of configuration objects

**3. Change control**

Change control is an essential step in software lifecycle. The change control can be carried out using following steps

1. A change request initiates a change
2. The configuration object is 'checked out' of the database.
3. The changes are applied to the object
4. The object is then 'checked in' to the database where automatic version control is applied.

**4. Configuration audit**

In order to ensure change has been properly implemented or not two activities are carried out

1. Formal Technical Review
2. Software Configuration Audit

In the formal technical review, the correctness of configuration object identified and corrected. It is conducted by technical reviewer.

The software configuration audit assesses the configuration object for the characteristics that are not reviewed in formal technical review. It is conducted by the Software Quality Assurance group.

**5. Status reporting**

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

## 5.4 : Version Control, Change Control and Reporting

**Q.11 What are the problems encountered with uncontrolled change management ?**

☞ [RGPV : June-17, Marks 2]

**Ans. :** Following problems are encountered with uncontrolled change management -

1) There might be the need to create support processes for changes.
2) The user need to be trained for the uncontrolled changes
3) The rollback plan must be prepared.
4) Sometimes there is a need to make conversion of huge data.

**Q.12 What is version control in project ?**

☞ [RGPV : Dec.-15, Marks 2]

**OR Explain version control in detail**

**Ans. :** 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 shown in Fig. Q.12.1.

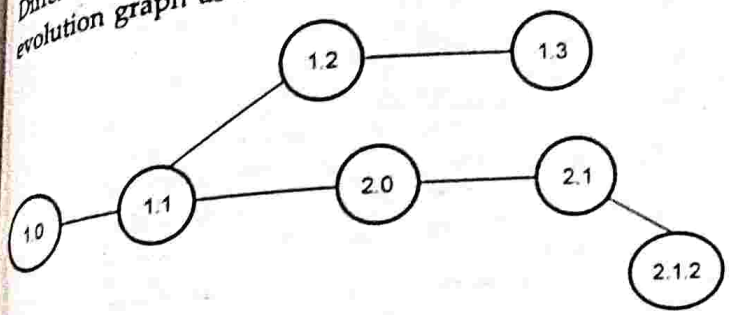



**Fig. Q.12.1 Version numbering in evolution graph**

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

## 5.5 : Program Comprehension Techniques

**Q.13 Describe various program comprehension techniques.** [RGPV : Dec.-16, Marks 7]

**Ans. :** Program comprehension is the process of developing mental models of a software systems intended architecture, meaning, and behavior.

The purpose of program comprehension is to recover high-level information about a system including :

1) Its structure (components and their interrelationships)
2) Its functionality (what operations are performed on what components)
3) Its dynamic behavior (how input is transformed to output)
4) Its rationale (how was the design process and what decisions have been taken)
5) Its construction, modules, documentation, and test suites.
6) Program comprehension does not change the subject system, nor create a new system. It is the process of examining and understanding the object system.

## 5.6 : Re-engineering

**Q.14 What do you mean by the term software re-engineering? Why it is required?**

[RGPV : June-12, Marks 10]

**Ans. :** 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.

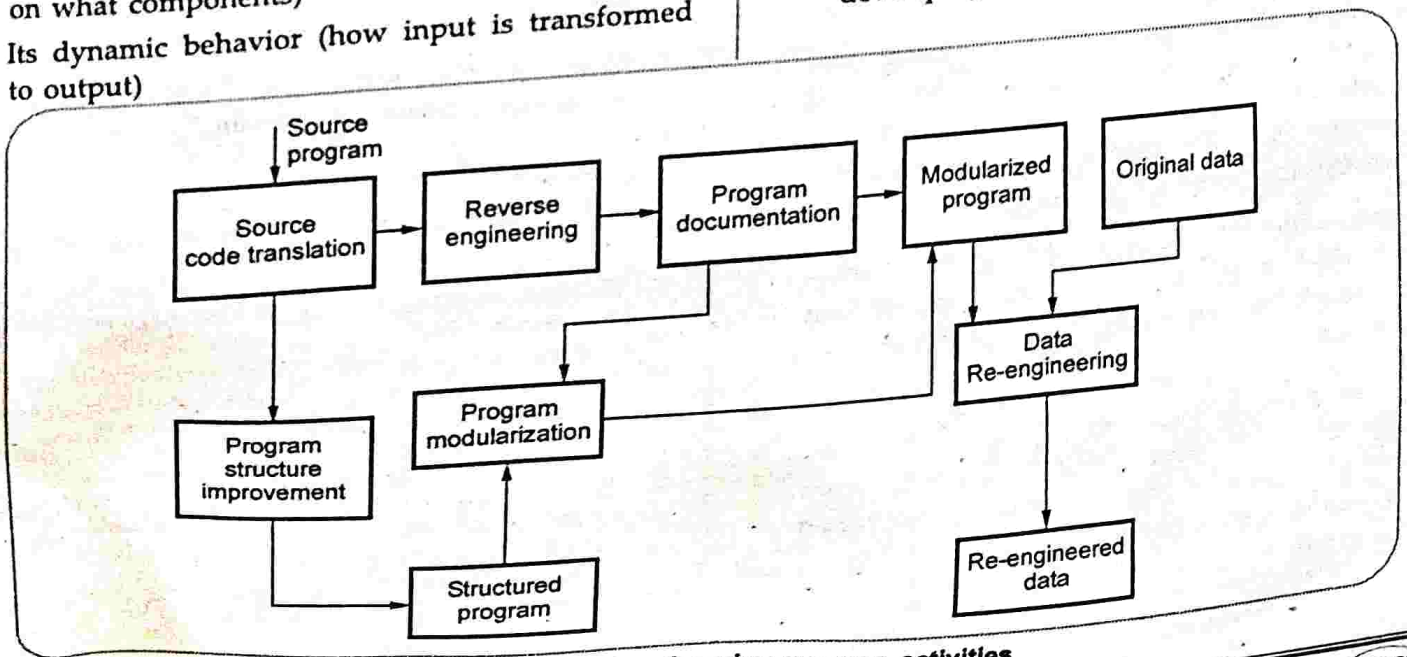



**Fig. Q.14.1 Re-engineering process activities**

**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.

## 5.7 : Reverse Engineering

**Q.15 Explain reverse engineering**

☞ [RGPV : June-16, Marks 2]

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

The reverse engineering is required because using this technique the dirty, ambiguous code can be converted to clear and unambiguous specification. This specification help in understanding the source code.

**Q.16 Write short note on - Reverse engineering**

☞ [RGPV : Dec.-18, Marks 7]

**Ans. :** 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.

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.

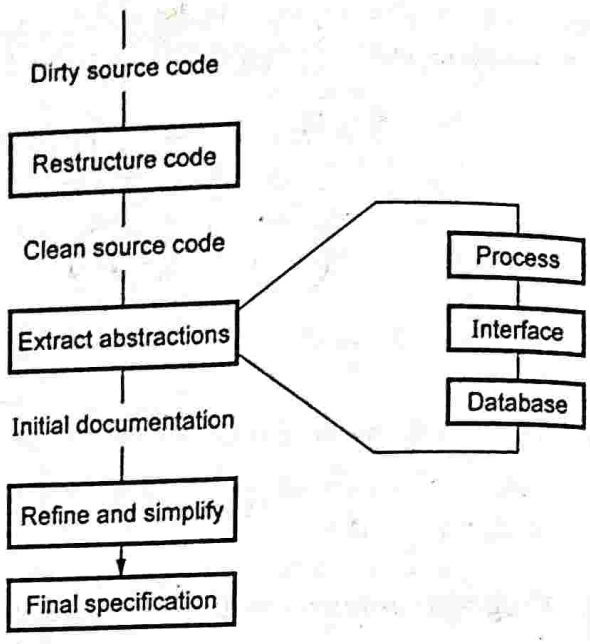



**Fig. Q.16.1 Reverse engineering**

**Q.17 What are the advantages of reverse engineering process?** ☞ [RGPV : June-15, 17, Marks 2]

**Ans. :**

1) The dirty code or outdated technology can be eliminated.
2) Improved version of the system can be created using reverse engineering.
3) Documentation can be made available, once the old system gets reverse engineered.
4) It is cheaper than developing the system from scratch.

**Q.18 What is the difference between forward engineering and reverse engineering?**

**Ans. :**

| Sr. No. | Forward engineering | Reverse engineering |
|---|---|---|
| 1. | Forward engineering is a process 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 forward 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 forward 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. |

**.19 How reverse engineering is different from engineering process? Explain with suitable xample** ☞ [RGPV : Dec.-16, Marks 7]

**ns. :** Refer Q.15

## 5.8 : Tool Support

**Q.20 Write a short note on - computer aided software engineering tools**

**Ans. :**

- The Computer Aided Software Engineering (CASE) tools automate the project management activities, manage all the work products. The CASE tools assist to perform various activities such as analysis, design, coding and testing.
- Software engineers and project managers make use of CASE tools.
- The use of CASE tools in the software development process reduces the significant amount of efforts.
- CASE is used in conjunction with the process model that is chosen.
- CASE tools help software engineer to produce the high quality software efficiently.

The CASE environment represents a framework in which a support for different CASE tools at various software development life cycle stages is shown. Following Fig. Q.20.1 depicts the typical CASE environment.

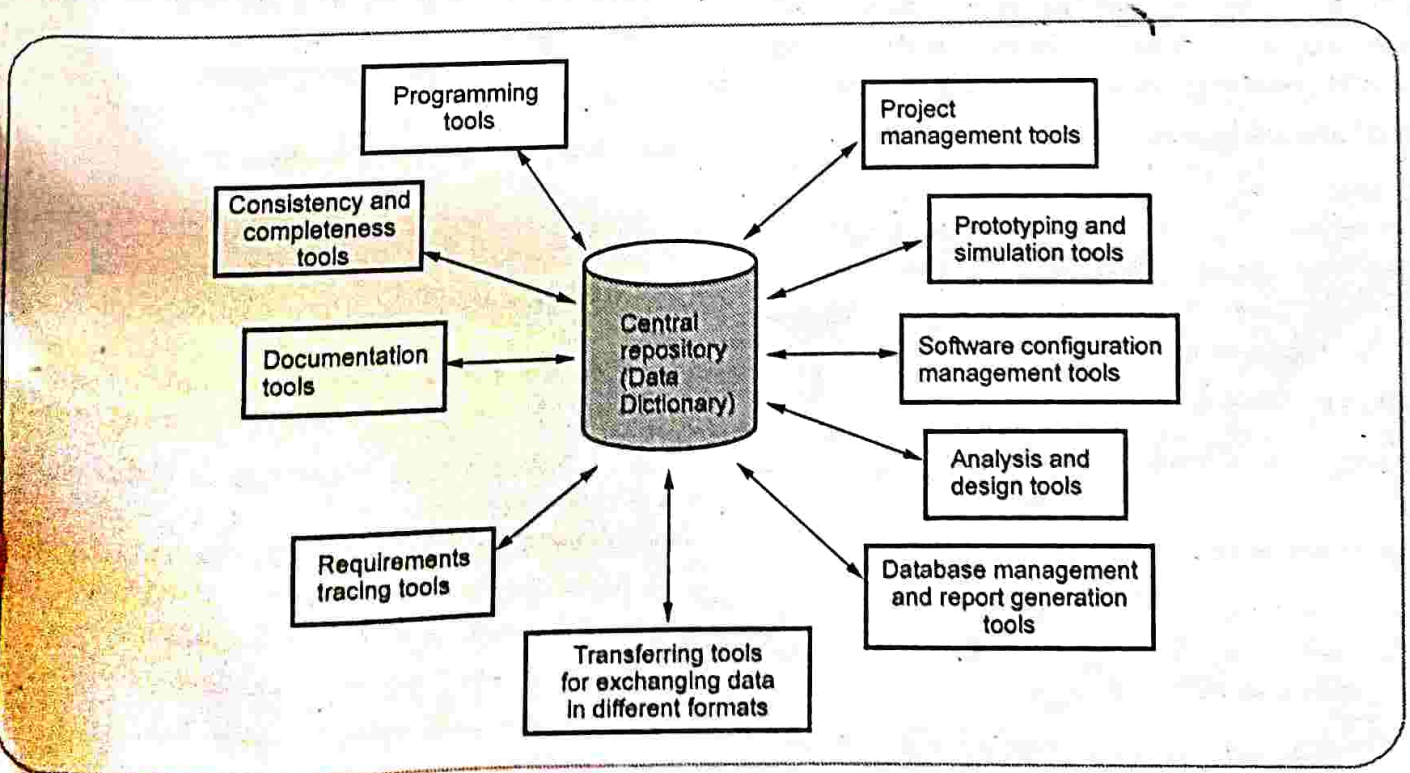



**Fig. Q.20.1 Block diagram for CASE environment**

**Q.21 Discuss the tools and techniques used in project management** [RGPV : Dec.-17, Marks 7]

Ans. :

**1) Process modeling and management tools**

It models software processes. First the processes need to be understood then only it could be modeled. This tool represents the key elements of the processes. Hence it is possible to carry out work tasks efficiently.

**2) Project planning tools**

These tools help to plan and schedule projects. Examples are PERT and CPM. The objective of this tool is finding parallelism and eliminating bottlenecks in the projects.

**3) Risk analysis tools**

It helps in identifying potential risks. These tools are useful for building the risk table and thereby providing detailed guidance in identification and analysis of risks. Using this tool one can categorize the risks as catastrophic, critical, marginal or negligible. A cost is associated with each risk which can be calculated at each stage of development.

**4) Project management tools**

These track the progress of the project. These tools are extension to the project planning tools and the use of these tools is to update plans and schedules.

**5) Metrics and management tools**

These tools assist to capture specific metrics that provide an overall measure of quality. These tools are intended to focus on process and product characteristics. For example "defects per function point", "LOC/person-month"

**6) Quality assurance tools**

These are actually metrics tools that audit source code to insure compliance with language standards.

**7) Database management tool**

It provides consistent interfaces for the project for all data, in particular the configuration objects are primary repository elements.

## 5.9 : Project Management Concepts

**Q.22 Explain activities covered by software project management** [RGPV : May-18, Marks 7]

Ans. : There are some typical activities that are normally carried out by the project manager -

- Writing the proposals for the project
- Project planning and scheduling
- Estimating the project cost
- Reviewing the project
- Project evaluation
- Report writing and presentation.

In order to gain the contract it is very much essential to write **proposal** for the project. The objective of the project should be clearly specified in this proposal. The proposal writing is a complex task and getting the contract for particular project mainly depends upon the proposal made for the project.

In project planning various project related **activities** are planned, **milestones** are determined. The project plan prepared in this phase serves as a guideline for various activities to be conducted in the project. The overall cost of the project is calculated tentatively.

## 5.10 : Feasibility Analysis

**Q.23 What is feasibility analysis ?**

[RGPV : June-16, Marks 2]

Ans. : *Definition : A feasibility study is a study made to decide whether or not the proposed system is worthwhile.*

**Q.24 Justify the importance of feasibility study in software development.** [RGPV : June-15, Marks 2]

Ans. : Following are benefits of feasibility study in software development -

1) It provides valuable information whether to develop the system or not. It identifies the a valid reason to undertake the project or not.
2) It narrows the business alternatives.
3) This improves the project team's focus.

**Q.25 How many types of feasibility analysis apply In a project ?** [RGPV : Dec.-15, Marks 2]

Ans. : **1. Technical Feasibility**

The technical feasibility is the study of configuration of the system. While studying the system configuration following things are to be studied,

- What is the exact configuration of the system ?
- How many workstations are required ?
- How different units are interconnected ? And are they working smoothly while interconnected ?
- What should be the speed at which the input is given and at what speed the output has to be generated ?

**2. Operational Feasibility**

The operational feasibility is based on the human factors and political aspects. The operational feasibility can be performed by answering following questions,

1. What change will be brought with the system ?
2. What are the factors that are disturbing organizational structure ?
3. Which are the new skills that are required for improvement in operations ?

**3. Economic Feasibility**

This kind of feasibility study is done for cost or benefit analysis. In this study the benefits of the proposed system are identified and the corresponding costs are determined. If the identified benefits are cost effectiveness then the decision is made whether or not to design and implement the system.

## 5.11 : Project and Process Planning

**Q.26 Write short note on - project planning.**

☞ **[RGPV : Dec-07, Marks 5]**

Ans. :

- The project manager should recognize the future problems in advance and should be ready with the **tentative solution** to those problems.
- A **project plan** must be prepared in advance from the available information.
- The project planning is an **iterative** process and it gets completed only on completion of the project.
- This process is iterative because new information gets available at each phase of project development. Hence the plan needs to be modified on regular basis for accommodating new requirements of the project.
- The **pseudo code** for the project planning process as given below.

```
Find out the constraints of the project.
Perform the initial assessment of the project parameters.
Define the project milestones and targets.
If project is cancelled then
  Exit
While project has not been completed
{
• Determine the project schedule.
• Start the project related activities as per the schedule
  { wait }
• Review the project progress.
• Revise the project estimates.
• Revise and modify the project schedule.
• Negotiate the project constraints
• If (problem occurs) then
    { Conduct the technical reviews}
}
```

**Q.27 What are milestones and deliverables in project planning activities.**

Ans. : When the project begins then it is expected that the project related activities must be initiated. In project planning, the series of milestones must be established. **Milestone** can be defined as the recognisable endpoint of the software project activity. At each milestone a report must be generated. Milestone is a distinct and logical stage of the project.

**Deliverable** is defined as the result of the project that can be submitted to the customer.

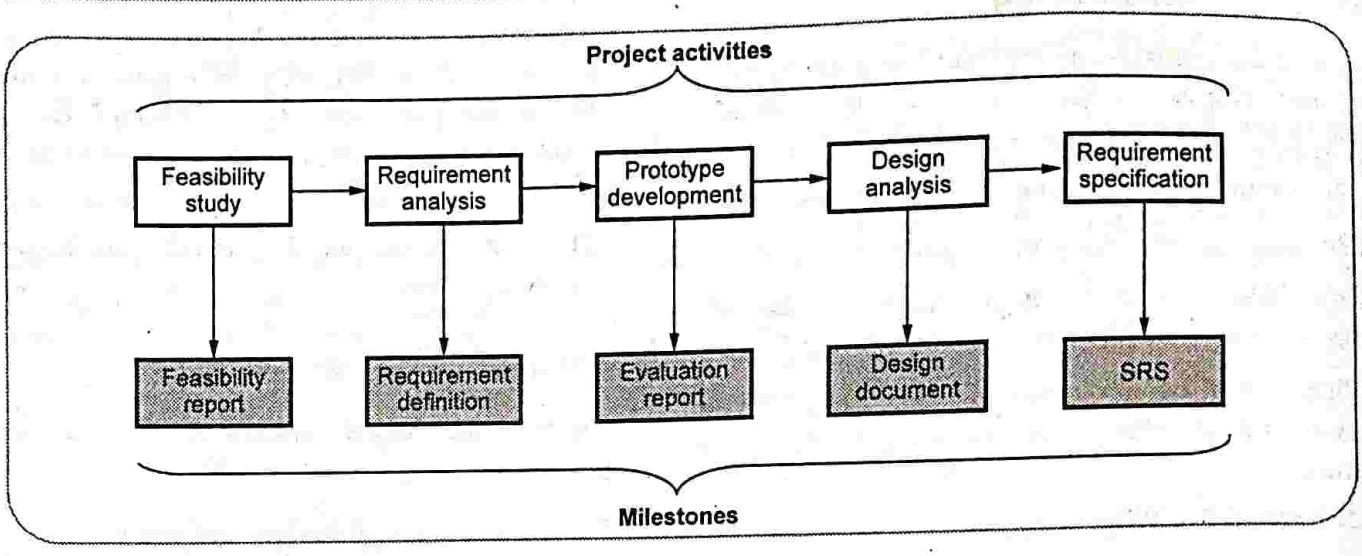



**Fig. Q.27.1 Milestones**

Deliverables are generally the milestones but it is not necessary that milestone is deliverable.

Fig. Q.27.1 shows various project activities and milestones.

## 5.12 : Resources Allocations

**Q.28 What are the types of resources used in project. Explain it with the help of resource pyramid.** **[RGPV : Dec.-11, Marks 10]**

**Ans. :** The resources can be denoted by a pyramid which is also known as **resource pyramid.** At the base of this pyramid is the hardware and software tools, then at the middle layer the reusable components are situated and at the top of the pyramid the human resource is available. It is as shown in Fig. Q.28.1.

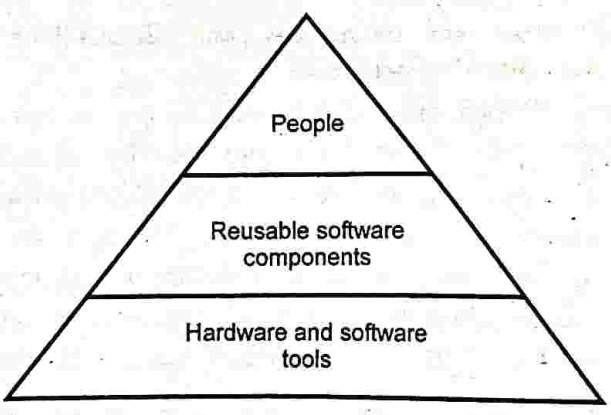



**Fig. Q.28.1 Resource pyramid**

When the software planner wants to specify the resources he specifies it using four characteristics -

- Description of resource
- Resource availability
- Time of resource when it will be available
- Duration of resource availability.

These resources can be described as follows -

- **Human resource**

In software development process people play an important role. In software industry people are assigned with some **organizational positions** such as manager, software developer, software testing engineer and so on. These positions are according to their **skills** and **speciality.** For a small project only single individual can perform all these roles, but for a large project a team of people work on it. Total number of people that are required for the project is estimated by calculating **development effort** interms of **person-months.**

- **Reusable components**

For bringing ease in software development process or to accelerate the development process software industry prefers to use some ready software components. The **components** can be defined as the software building blocks that can be created and reused in the software development process. The use of components emphasize on **reusability.** This is also termed as Component Based Software Engineering (CBSE).

## 5.13 : Schedule and Cost Estimations

**Q.29 Describe COCOMO model.**

**[RGPV : Dec.-14, June-16, Marks 7, Dec.-15, Marks 3]**

**Ans. :** This model is developed in 1981 by **Barry Boehm** to give an estimate of the number of man-months it will take to develop a software product. COCOMO predicts the efforts and schedule of a software product based on size of the software. COCOMO stands for "**CO**nstructive **CO**st **MO**del".

COCOMO has three different models that reflect the complexity -

1) **Basic Model :** The basic COCOMO model estimates the software development effort using only **Lines of Code**. Various equations in this model are -

$$E = a_b (KLOC)^{b_b}$$

$$D = C_b (E)^{d_b}$$

$$P = E/D$$

Where E is the **effort** applied in person-months.

D is the development time in chronological months.

KLOC means kilo line of code for the project.

P is total number of persons required to accomplish the project.

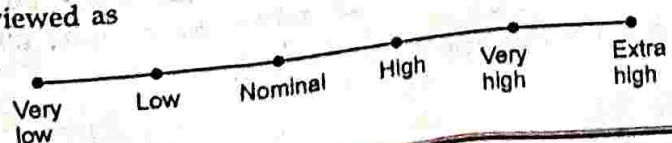
The coefficients $a_b$, $b_b$, $c_b$, $d_b$ for three modes are as given below.

| Software projects | $a_b$ | $b_b$ | $c_b$ | $d_b$ |
|---|---|---|---|---|
| Organic | 2.4 | 1.05 | 2.5 | 0.38 |
| Semi-detached | 3.0 | 1.12 | 2.5 | 0.35 |
| Embedded | 3.6 | 1.20 | 2.5 | 0.32 |

**Table Q.29.1**

2) **Intermediate Model :** This is an extension of Basic COCOMO model. This estimation model makes use of set of "Cost driver attributes" to compute the cost of software.

The cost driver attributes get a 6-point scale ranging from "very low" to "extra high". These ratings can be viewed as




The formula for effort calculation can be -

$$E = a_i (KLOC)^{b_i} \cdot EAF \text{ person-months}$$

The values for $a_i$ and $b_i$ for various class of software projects are -

| Software project | $a_i$ | $b_i$ |
|---|---|---|
| Organic | 3.2 | 1.05 |
| Semi-detached | 3.0 | 1.12 |
| Embedded | 2.8 | 1.20 |

**Table Q.29.2**

The duration and person estimate is same as in basic COCOMO model. i.e.

$$D = c_b (E)^{d_b} \text{ months}$$

i.e. use values of $c_b$ and $d_b$ coefficients that are in Table Q.29.1

$$P = E/D \text{ persons}$$

3) **Detailed COCOMO Model :** The detailed model uses the same equations for estimation as the Intermediate Model. But detailed model can estimate the effort (E), duration (D) and persons (P) of each of development phases, subsystems, modules.

The experimentation with different development strategies is allowed in this model.

Using the detailed cost drivers, an estimate is determined for each phase of the lifecycle.

**Q.30 Discuss various cost estimation models and compare them.** **[RGPV : June-15, Marks 7]**

**Ans. :** Various cost estimation models are -

1) **Empirical Estimation Technique :** In empirical estimation technique an educated guess of project parameters is made. Hence empirical estimation model is based on common sense. However as there are many activities involved in empirical estimation techniques, this technique is formalized. Hence **Delphi technique** is used as empirical estimation technique.

2) **Heuristic Technique :** In heuristic technique, the relationship among different project parameters is expressed using mathematical equations. The popular heuristic technique is given by **COCOMO Model.**

3) **Analytical Estimation Technique :** In analytical estimation technique, the results are derived by making certain basic assumptions about the project. Hence analytical estimation technique have some scientific basis. Halstead's software science is based on analytical estimation model.

**Q.31 A project size of 2000KLOC is to be developed. Software development team has average experience on similar types of projects. The project schedule is not very tight. Calculate the effort, development time, average staff size and productivity of the project.**

☞ [RGPV : Dec.-15, Marks 7]

**Ans. :** A semidetached mode of COCOMO model is the most appropriate mode. The constants of this mode are

| Semi-detached Mode | $a_b$ | $b_b$ | $c_b$ | $d_b$ |
|---|---|---|---|---|
| | 3.0 | 1.12 | 2.5 | 0.35 |

**i) Effort Estimation**

$$E = a_b(KLOC)^{b_b}$$

$$= 3.0(2000)^{1.12}$$

$$= 14937.39 \text{ Person-Month}$$

**ii) Duration Estimation**

$$D = c_b(E)^{d_b}$$

$$= 2.5(14937.39)^{0.35}$$

$$= 72.26 \text{ Months}$$

**iii) Average Staff Size**

E/D

$$= 14937.39/72.26$$

$$= 206.71 \text{ Persons}$$

**iv) Productivity**

KLOC/E

$$= 2000/14937.39$$

$$= 0.1338 \text{ KLOC/PM}$$

$$= 133 \text{ LOC/PM}$$

**Q.32 Suppose that the size of organic type product has been estimated to be 32000 lines of source code. Assume that the average salary of software engineer be ₹ 15,000/ per month. Determine the effort required to develop the software product and nominal development time.**

☞ [RGPV : Dec.-18, Marks 7]

**Ans. :** Given that,

Lines of source code is 32KLOC.

The type of software product is organic. Hence

| Organic Mode | $a_b$ | $b_b$ | $c_b$ | $d_b$ |
|---|---|---|---|---|
| | 2.4 | 1.05 | 2.5 | 0.38 |

**i) Effort Estimation**

$$E = a_b(KLOC)^{b_b}$$

$$= 2.4*(32)^{1.05}$$

$$= 91 \text{ Person-Month}$$

**ii) Nominal Development Time**

$$D = c_b(E)^{d_b}$$

$$= 2.5(91)^{0.38}$$

$$= 14 \text{ Months}$$

**iii) Cost required to develop the product = 14*15000**

**= ₹ 210000/-**

**Q.33 Compute the function point value for the project with the following information domain characteristics :**

**Number of user inputs - 32**

**Number of user output - 60**

**Number of user inquiries - 24**

**Number of files - 4**

**Number of external interfaces - 2**

**Assume all the complexity adjustment values are average. Assume that 14 algorithms have been counted. Compute the feature point under same conditions.** ☞ [RGPV : Dec.-03, Marks 10]

**Ans. :** Given that :

1. Number of user input = 32
2. Number of user output = 60
3. Number of user enquiries = 24
4. Number of files = 4
5. Number of external interfaces = 2

Complexity adjustment values are average.

| | Domain characteristics | Count | × | Average Value | |
|---|---|---|---|---|---|
| 1. | No. of user input | 32 | × | 4 | 128 |
| 2. | No. of user output | 60 | × | 5 | 300 |
| 3. | No. of user enquiries | 24 | × | 4 | 96 |
| 4. | No. of files | 4 | × | 10 | 40 |
| 5. | No. of external interface | 2 | × | 7 | 14 |
| | | | | | 578 |

$$\sum_{i=1}^{14} F_i = 14 \times 3 = 42$$

$$\therefore \quad FP = \text{Count total} \times [0.65 + 0.01 \times \sum F_i]$$

$$= 578 \times [0.65 + 0.01 \times 42]$$

$$= 618.46$$

**Q.34 Consider a project to develop a full screen editor. The major components identified are -**

**i) Screen edit**

**ii) command language interpreter**

**iii) File input and output iv) Cursor movement**

**v) Screen movement**

**The sizes estimated are 4 K, 2 K, 1 K, 2 K and 8K delivered source lines of code. Use COCOMO model to determine -**

**Overall cost and schedule etimates, assume effort adjustment factor being 1.21.**

| $\mu p$ (effort) | System Design | Detail Design | Module Code and test | Integration and Test |
|---|---|---|---|---|
| $\tau_p$ (time) | 0.16<br>0.19 | 0.26<br>0.24 | 0.42<br>0.39 | 0.16<br>0.18 |

**[RGPV : Dec.-04, 06, 07, Marks 10]**

**Ans. :** The size is KLOC can be

1) Screen edit = 4 KLOC

2) Command language interpreter = 2 KLOC

3) File input and output = 1 KLOC

4) Cursor movement = 2 KLOC

5) Screen movement = 3 KLOC

Total = 12 KLOC

The formula for effort calculation is

$$E = a_i (KLOC)^{b_i} \times EAF \text{ person - months}$$

For organic project $a_i = 3.2$ and $b_i = 1.05$

EAF is given as 1.21

$$\therefore \quad E = 3.2 * (12)^{1.05} * 1.21$$

$$= 43.48 * 1.21$$

$$= 52.61$$

The duration for completion of project can be calculated using following formula

$$D = C_b * (E)^{d_b} \text{ months}$$

The $C_b = 2.5$ and $d_b = 0.38$

$$\therefore \quad D = 2.5 * (52.61)^{0.38}$$

$$= 11.27 \text{ months}$$

ii) Effort and schedule estimate for different phases

**Effort estimation**

i) System design = 0.16 * 52.61 = 8.41 PM

ii) Detail design = 0.26 * 52.61 = 12.62 PM

iii) Module code and test = 0.42 * 52.61 = 20.51 PM

iv) Integration and test = 0.16 * 52.61 = 9.99 PM

**Schedule estimation**

i) System design = 0.19 * **11.27** = **2.14** months

ii) Detail design = 0.24 * 11.27 = 6.53

iii) Module code and test = 0.39 * 11.27 = 2.47

iv) Integration and test = 0.18 * 11.27 = 2.02

**Q.35 Explain Halsteads' software science.**

**Ans. :** Halstead's complexity measurement was developed to measure a program module's complexity directly from source code, with emphasis on computational complexity.

The Halstead's measures are based on four scalar numbers derived directly from a program's source code:

$n_1$ is number of distinct operators

$n_2$ is number of distinct operands

$N_1$ is total number of operators

$N_2$ is total number of operands

From these numbers, five measures are derived :

| Measure | Symbol | Formula |
|---|---|---|
| Program length | N | $N = N_1 + N_2$ |
| Program vocabulary | n | $n = n_1 + n_2$ |
| Volume | V | $V = N * (\log_2 n)$ |
| Difficulty | D | $D = (n_1/2) * (N_2/2)$ |
| Effort | E | $E = D * V$ |

Halstead's uses certain measures such as program length, program vocabulary, volume, difficulty and effort for the given algorithm. By this Halstead's is trying to show that the program length can be calculated, volume of the algorithm can be estimated. The above given table shows how actually these measures can be obtained.

The Halstead's measures are applicable to operational systems and to development efforts once the code has been written. Thus using Halstead's measurement experimental verification can be performed in software science.

**Program length**

The length of a program is total usage of operators and operands in the program.

$$\text{Length} = N_1 + N_2$$

**Program vocabulary**

The program vocabulary is the number of unique operators and operands used in the program.

$$\text{Vocabulary} \cdot n = n_1 + n_2$$

**Program volume**

The program volume can be defined as minimum number of bits needed to encode the program.

$$V = N \log_2 n$$

**Length estimation**

$$N = n_1 \log_2 n_1 + n_2 \log_2 n_2$$

**Q.36 Obtain Halstead's length and volume measure for following C function.**

```
void swap (int a [ ], int i)
{
   int temp ;
   temp = a[i] ;
   a[i] = a[i+1] ;
   a[i+1] = temp ;
}
```

**Ans :** We will first find out the operands and operators from above function along with their occurrences.

| Operands | Occurrences | Operators | Occurrences |
|---|---|---|---|
| swap | 1 | ( ) | 1 |
| a | 5 | { } | 1 |
| i | 5 | void | 1 |
| temp | 3 | int | 3 |
| 1 | 2 | [ ] | 5 |
| | | , | 1 |
| | | ; | 4 |
| | | = | 3 |
| | | + | 2 |
| $n_1 = 5$ | $N_1 = 16$ | $n_2 = 9$ | $N_2 = 21$ |

$N = N_1 + N_2 = 16 + 21 = 37 \quad n = n_1 + n_2 = 14$

Estimated length $= n_1 \log_2 n_1 + n_2 \log_2 n_2$

$= 5 \log_2 5 + 9 \log_2 9$

$= 5 * 2.32 + 9 * 2.19$

$= 11.60 + 19.77 = 31.37$

Volume $= N \times \log_2 n$

$= 37 * \log_2 (14)$

Volume $= 37 * 2.63 = 97.64$

**Q.37 For the following 'C' program estimate the Halstead's length and volume measures of size with LOC measure.**

```
/* Program to calculate GCD of two numbers */
int compute_gcd (x, y)
{
   while (x != y)
   if (x > y) then x = x - y ;
   else y = y - x ;
   return x ;
}
```

**Ans. :** We will find out operators and operands and their occurrences.

| Operator | Occurrences | Operands | Occurrences |
|---|---|---|---|
| int | 1 | compute_gcd | 1 |
| ( ) | 3 | x | 7 |
| { } | 1 | y | 6 |
| , | 1 | | |
| while | 1 | | |
| if | 1 | | |
| then | 1 | | |
| = | 2 | | |
| != | 1 | | |
| - | 2 | | |
| > | 1 | | |
| else | 1 | | |
| return | 1 | | |
| ; | 3 | | |
| $n_1 = 14$ | $N_1 = 20$ | $n_2 = 3$ | $N_2 = 14$ |

$N = N_1 + N_2$ and $n = n_1 + n_2$

i.e. $N = 20 + 14 = 34$

$n = 14 + 3 = 17$

Length $= n_1 \log_2 n_1 + n_2 \log_2 n_2$

$= 14 \log_2 14 + 3 \log_2 3$

$= 14 * 3.80 + 3 * 1.58 = 53.30 + 4.75$

$= 58$

Volume $= N \log_2 n$

$= 34 \log_2 17$

$= 34 * 4.08 = 139$

## 5.14 : Project Scheduling and Tracking

**Q.38 Write short note on- project scheduling.**

☞ **[RGPV : Dec.-14, Marks 7]**

**Ans. :** • While scheduling the project, the manager has to estimate the time and resource of the project.

- All the activities in the project must be arranged in coherent sequence.
- The schedules must be continually updated because some uncertain problems may occur during the project life cycle.
- For new projects initial estimates can be made optimistically.

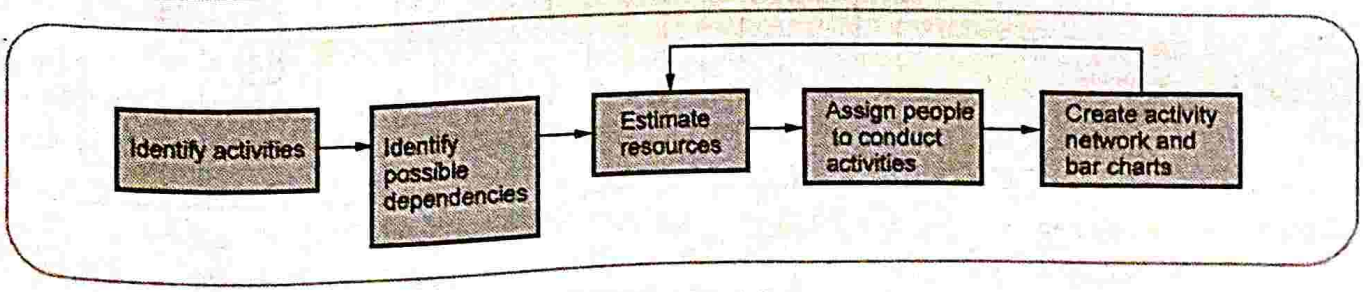



**Fig. Q.38.1 Project scheduling process**

- During the project scheduling the total work is separated into various small activities. And time required for each activity must be determined by the project manager. For efficient performance some activities are conducted in parallel.
- The project manager should be aware of the fact that : Every stage of the project may not be problem-free. Some of the typical problems in project development stage are :
  i) People may leave or remain absent.
  ii) Hardware may get failed.
  iii) Software resource may not be available.
- To accomplish the project within given schedule the required resources must be available when needed. Various resources required for the project are -
  i) Human effort
  ii) Sufficient disk space on server
  iii) Specialized hardware
  iv) Software technology
  v) Travel allowance required by the project staff.
- Project schedules are represented as the set of chart in which the work-breakdown structure and dependencies within various activities is represented.

**Q.39 What is Gnatt chart ? Give advantages of Gnatt charts in monitoring software project ?**

☞ [RGPV : June-15, Marks 3]

**Ans. :** • The purpose of Gnatt chart is to emphasize the scope of individual task. Hence set of tasks are given as input to the Gnatt chart.

- The Gnatt ine chart is also called as time line chart.
- The Gnatt chart can be developed for entire project or it can be developed for individual functions.
- In Gnatt chart

1) All the tasks are listed at the leftmost column.
2) The horizontal bars indicate the time required by the corresponding task.
3) When multiple horizontal bars occur at the same time on the calendar, then that means concurrency can be applied for performing the tasks.
4) The diamonds indicate the milestones.

- In most of the projects, after generation of time line chart the project tables are prepared. In project tables all the tasks are listed along with actual start and end dates and related information.

**Example**

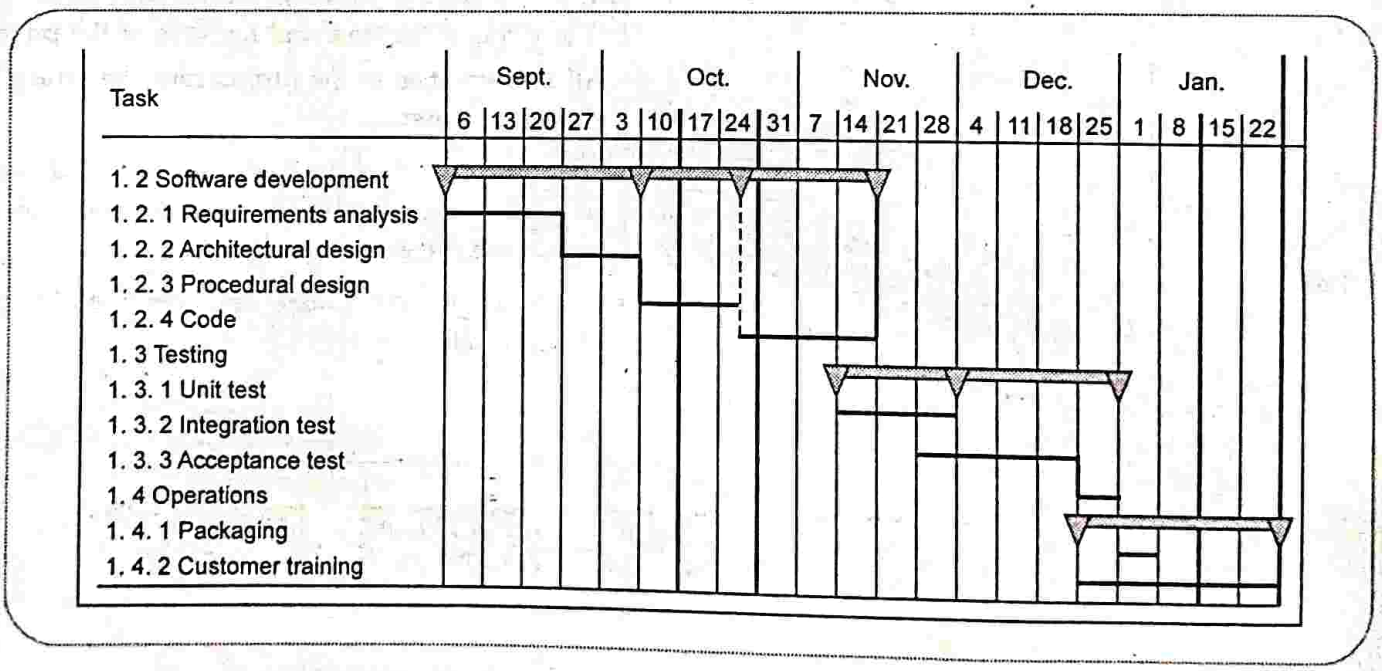



**Fig. Q.39.1 Gnatt chart**

**Q.40 Software project scheduling does not differ from scheduling any other multitask engineering project. Discuss.** [RGPV : June-17, Marks 2]

**Ans. :** • Scheduling of a software project does not differ greatly from scheduling of any multi-task engineering effort. Therefore, generalized project scheduling tools and techniques can be applied with little modification to software projects.

• Program evaluation and review technique (PERT) and critical path method (CPM) are two project scheduling methods that can be applied to software development.

• These project scheduling methods provide quantitative tools that allow the software planner to

(1) determine the critical path-the chain of tasks that determines the duration of the project;

(2) establish "most likely" time estimates for individual tasks by applying statistical models; and

(3) calculate "boundary times" that define a time window" for a particular task.

## 5.15 : Risk Assessment and Mitigation

**Q.41 Define software risk. What are the characteristics of software risk ?**

**Ans. :** • The software risk can be defined as a problem that can cause some loss or can threaten the success of the project.

• Risk management refers to the process of making decisions based on evaluation of the factors that threatens the business.

• There are two characteristics of the risks

1. The risk may or may not happen. It shows the uncertainty of the risks.
2. When risks occur, unwanted consequences or losses will occur.

**Q.42 Explain different types of risks**

**Ans. :** **1. Project risk**

Project risks arise in the software development process then they basically affect budget, schedule, staffing, resources, and requirements. When project risks become severe then the total cost of project gets increased.

**2. Technical risk**

These risks affect quality and timeliness of the project. If technical risks become reality then potential design, implementation, interface, verification and maintenance problems gets created. Technical risks occur when problem becomes harder to solve.

**3. Business risk**

When feasibility of software product is in suspect then business risks occur. Business risks can be further categorized as

i) Market risk - When a quality software product is built but if there is no customer for this product then it is called market risk (i.e. *no market for the product*).

ii) Strategic risk - When a product is built and if it is not following the company's business policies then such a product brings strategic risks.

iii) Sales risk - When a product is built but how to sell is not clear then such a situation brings sales risk.

iv) Management risk - When senior management or the responsible staff leaves the organization then management risk occurs.

v) Budget risk - Losing the overall budget of the project is called budget risk.

**Q.43 Explain the risk management process in detail.**

**Ans. :** • From Fig. Q.43.1 risk management is performed in following stages.

1. ***Risk identification :*** In this phase all possible risks are anticipated and a list of potential risks is prepared.
2. ***Risk analysis :*** The after-effects of the risks are obtained and a list is prepared in which, the risks that need to be handled are prioritised.
3. ***Risk planning :*** The risk avoidance or risk minimization plan is prepared in this phase.
4. ***Risk monitoring :*** Identified risks must be mitigated. Hence risk mitigation plan must be prepared once the risks are discovered.

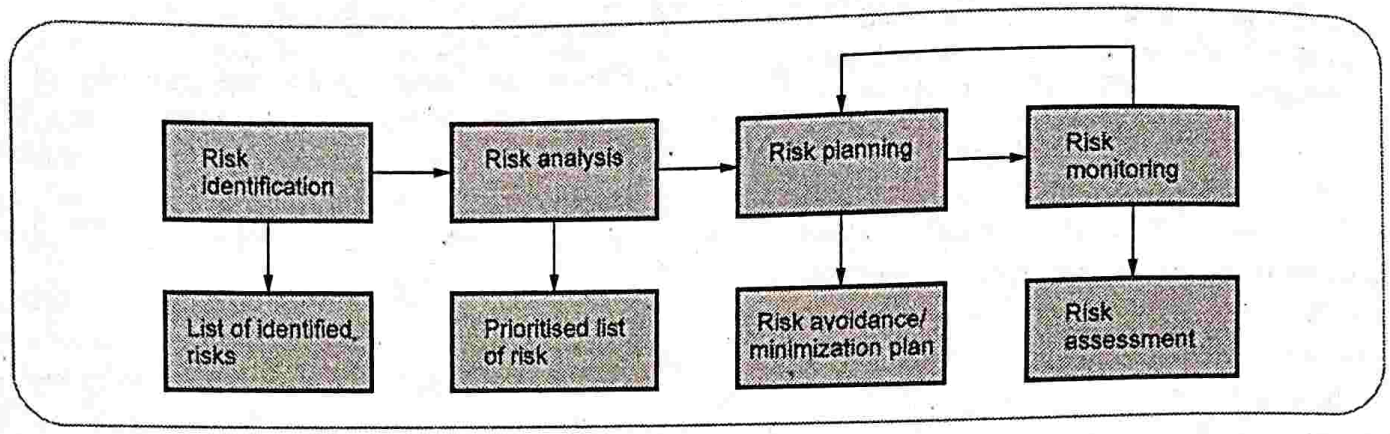



**Fig. Q.43.1 Risk management process**

**Q.44 What do you mean by risk mitigation ?**

**[RGPV : Dec.-14, Marks 7]**

**Ans :** Risk mitigation means preventing the risks to occur(risk avoidance). Following are the steps to be taken for mitigating the risks.

1. Communicate with the concerned staff to find of probable risk.
2. Find out and eliminate all those causes that can create risk before the project starts.
3. Develop a policy in an organization which will help to continue the project even though some staff leaves the organization.
4. Everybody in the project team should be acquainted with the current development activity.
5. Maintain the corresponding documents in timely manner. This documentation should be strictly as per the standards set by the organization.
6. Conduct timely reviews in order to speed up the work.
7. For conducting every critical activity during software development, provide the additional staff if required.

**Q.45 Explain risk assessment and mitigation.**

**[RGPV : June-16, Marks 7]**

**Ans. : Risk assessment :** The best approach is to prepare a set of questions that can be answered by project managers in order to assess the overall project risks. These questions can be

1. Will the project get proper support by the customer manager ?
2. Are the end-users committed to the software that has been produced ?
3. Is there a clear understanding of requirements?
4. Is there an active involvement of the customer in requirement definition ?
5. Is that the expectations set for the product are realistic ?
6. Is project scope stable ?
7. Are there team members with required skills ?
8. Are project requirements stable ?
9. Does the technology used for the software is known to the developers ?
10. Is the size of team sufficient to develop the required product ?
11. Is that all the customers know the importance of the product/ requirements of the system to be built ?

Thus the number of negative answers to these questions represents the severity of the impact of the risk on overall project.

**Risk Mitigation :** Refer Q.44

**Q.46 Suppose you are the project manager of a large software development project. List three common types of risks that your project might suffer. Point out main steps that you would follow to effectively manage risks in your project.**

**[RGPV : Dec.-15, Marks 7]**

**Ans. : Types of risks :** Refer Q.42

**Risk management :** Refer Q.43

## 5.16 Software Quality Assurance (SQA)

**Q.47 Define the tern - software quality**

**Ans. :** Software quality can be defined as "the conformance to explicitly stated functional and performance requirements, explicitly documented development standards, and implicit characteristics that are expected of all professionally developed software".

**Q.48 What is software quality assurance? Give the goals of software quality assurance(SQA).**

**Ans. : Definition of quality assurance** : *It is planned and systematic pattern of activities necessary to provide a high degree of confidence in the quality of a product. It provides quality assessment of the quality control activities and determines the validity of the data or procedures for determining quality.*

- The quality assurance consists of set of reporting and auditing functions.
- These functions are useful for assessing and controlling the effectiveness and completeness of quality control activities.
- The goal of quality assurance is to ensure the management of data which is important for product quality.

**Q.49 What are the activities performed during SQA ?** **[RGPV : Dec.-14, Marks 7]**

**Ans. :** Let us list out the SQA activities conducted by SQA group

**1. Prepare an SQA plan for a project.**

A SQA plan is developed while planning the project. Quality assurance activities are conducted that are indicated in this plan.

(Refer Fig. Q.49.1)

This plan basically

- Identifies evaluations to be performed.
- Audits and reviews to be performed, standards that should be adopted for the project.
- Procedures for error reporting and tracking.
- It also specifies documents to be produced by SQA group.
- Amount of feedback provided to the software project team.

**2. Participates in the development of the project's software process description.**

The process selected by the software team is reviewed by the SQA group. This review is for

- Process description to ensure that it follows the organizational policy.
- Internal software standards.
- Some standards that are adopted by the organization.

**3. Reviews software engineering activities to verify compliance with the defined software process**

The SQA group identifies and documents the processes. The group also verifies the correctness of software process.

**4. Audits designated software work products to verify compliance with those defined as part of the software process.**

The SQA group performs following tasks -

- Reviews selected work product
- Identifies the process
- Documents them

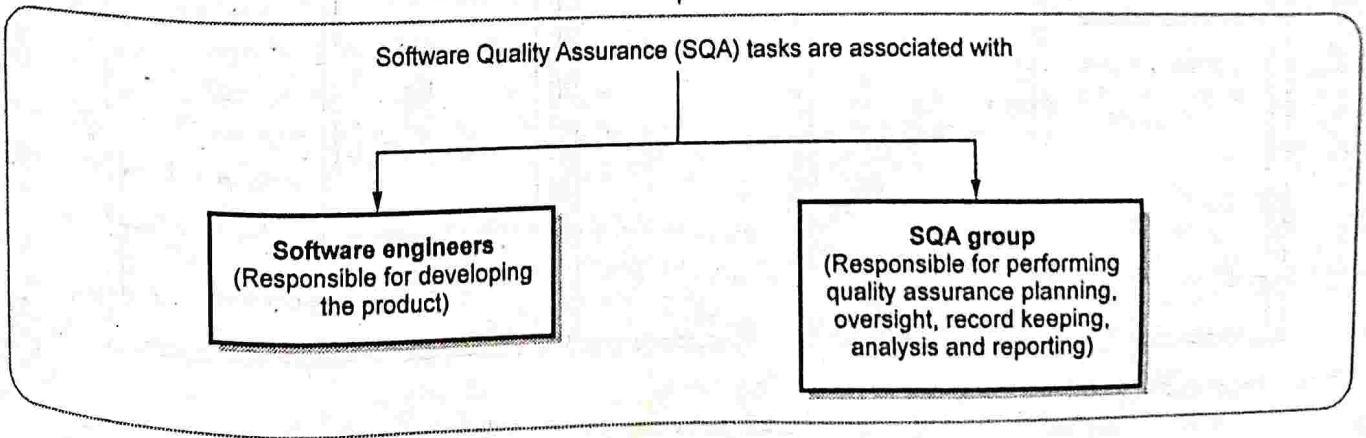



**Fig. Q.49.1**

- Tracks deviations
- Verifies the correctness made in the processes
- Regular reporting of results of its work to the project manager.

**5. Ensure the deviations in software work. These work products are documented and handled according to documented procedure.**

The deviations in software work are identified from project plan. These processes are identified and handled according to documented procedure.

**6. Records any noncompliance and reports to senior management**

Non compliance items are identified and pursued until they get resolved. The periodic reporting about it is done to project manager.

**Q.50 Describe Software Quality Assurance(SQA).**

☞ **[RGPV : Dec.-16, Marks 6, Dec.-18, Marks 7]**

**Ans. :** Refer Q.48

**Q.51 What is software quality assurance ? What are the measures of software quality ?**

☞ **[RGPV : Dec-17, Marks 7]**

**Ans. : Software Quality Assurance :** Refer Q.48

**Measures of Software Quality :**

Various measures of software quality are -

1) **Reliability :** Reliability includes aspects such as availability, accuracy, and recoverability. For example, recoverability of the system from shut-down failure is a reliability measure.
2) **Performance :** Performance means measuring throughput of the system using system response time, recovery time and start up time.
3) **Functionality :** The functionality represents that the system is satisfying the main functional requirements
4) **Supportability :** There are number of other requirement which the software system must satisfy. These include - testability, adaptability, maintainability, scalability and so on.
5) **Usability :** It is a capability of software system that it is easy to understand by its users. It includes aesthetic, consistency, documentation and responsiveness.

**Q.52 What are various quality concepts of SQA ?**

☞ **[RGPV : Dec.-17, May-18, Marks 7]**

**Ans. :** Refer Q.51

**Q.53 Explain various SDLC activities as outlined by ISO 12207 with a neat diagram**

☞ **[RGPV : May-18, Marks 7]**

**Ans. :** • The ISO 12207 is an international standard for life cycle processes. It aims to define primary standard that defines all the processes required for developing and maintaining software systems.

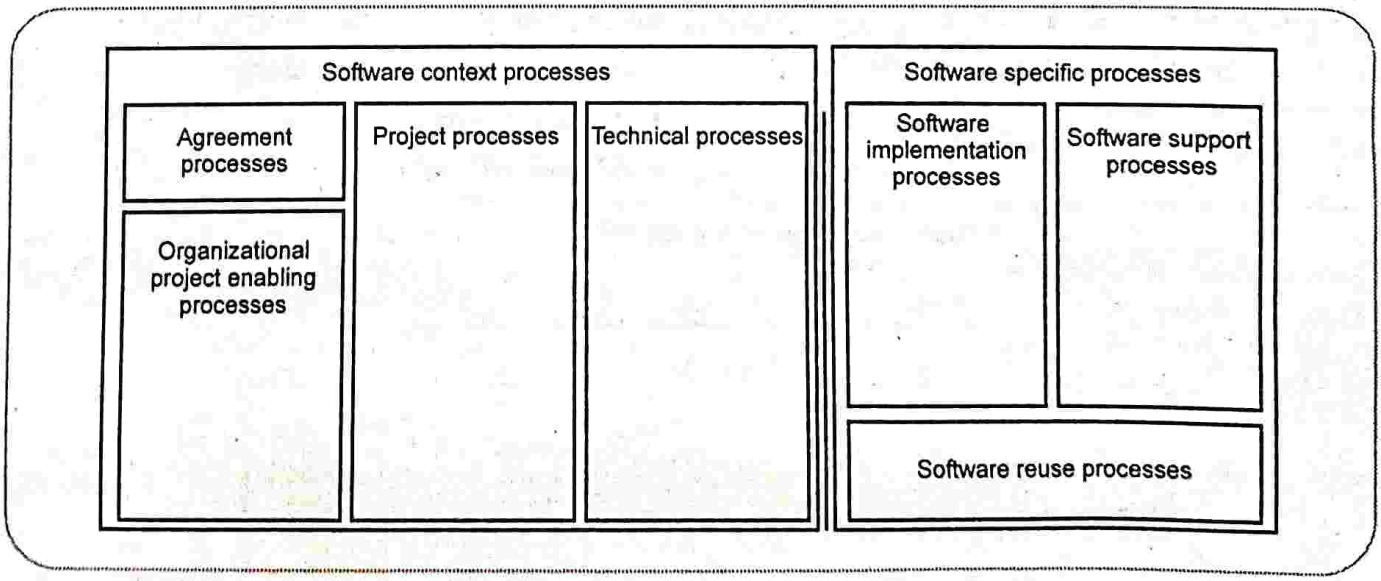



**Fig. Q.53.1 SDLC activities of ISO 12207**

- The software development Life cycle (SDLC) activities of ISO 12207 are as follows -

1. **Agreement processes :** These processes are carried out for establishing the agreement between two organizations. In this set of processes requirements analysis, defining the scope of the system, finding out the constraints for the system development all such tasks are carried out.
2. **Organizational project enabling processes :** These are set of processes that define and maintain the life cycle models and procedures for use by the organization with respect to the scope of international standard.
3. **Project processes :** In this phase, the project is described concerning with planning, assessment and control. There are two categories of processes - Project management processes and project support processes. This is a phase in which decision management processes, risk management processes, configuration management processes work.
4. **Technical processes :** This is a phase in which set of processes work to define the requirements for the system, to analyse the requirements, and to transform the requirements into effective design
5. **Software implementation processes :** In this phase, the set of processes are used to produce a specified system element implemented in software. Thus a software product or a service is developed in this phase.
6. **Software support processes :** This phase contains a set of processes which include documentation management process, quality assurance process, software verification process and so on. These processes serve as the support processes for developed software system.
7. **Software reuse processes :** There are certain processes such as domain engineering processes, reuse program management processes that serve as software reuse processes.

## 5.17 : Project Plan

**Q.54 Give the sample template for project plan document.**

**Ans. :** Following are the sections in the project plan -

**1.1 Introduction**

The goals, objectives and constraints of the project must be described in this section.

**1.2 Project organization**

The organization of development team should be described. The number of people involved along with their roles must be described in detail.

**1.3 Risk analysis**

All possible risks should be identified. Not only this but, the possible risk reduction strategies must be decided.

**1.4 Hardware and software requirements**

This section specifies the required hardware and software.

**1.5 Work breakdown**

Various project activities are grouped together to define the project work breakdown. Deciding the project milestone and deliverables is an important task in defining the work breakdown.

**1.6 Project schedule**

The tentative schedule of project activities must be determined.

**1.7 Report generation**

The structure of the project report, when it should be generated must be decided.

## 5.18 : Project Metrics

**Q.55 What is software measurement ?**

**Ans. :** Software measurement means deriving a numeric value for an attribute of a software product or process.

**Q.56 Define the terms - Measure, metrics, indicators.** ☞ [RGPV : June-06, Marks 6]

**Ans. : Measure** : It is a quantitative indication of the extent, amount, dimension, or size of some attribute of a product or process.

**Metrics** : It is the degree to which a system, component, or process possesses a given attribute. The software metrics relate several measures. For example - average number of errors found per review.

**Indicators** : Indicators mean combination of metrics that provides insight into the software process, project or product.

**Q.57 What is size oriented metrics ?**

**Ans :** • Size oriented measure is derived by considering the size of software that has been produced.

- The organization builds a simple record of size measure for the software projects. It is built on past experiences of organizations.
- It is a direct measure of software.

| Project | LOC | Effort | Cost($) | Doc.(pgs.) | Errors | Defects | People |
|---|---|---|---|---|---|---|---|
| ABC | 10,000 | 20 | 170 | 400 | 100 | 12 | 4 |
| PQR | 20,000 | 60 | 300 | 1000 | 129 | 32 | 6 |
| XYZ | 35,000 | 65 | 522 | 1290 | 280 | 87 | 7 |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |

**Table Q.57.1 Size measure**

- A simple set of size measure that can be developed is as given below -
  - Size = Kilo Lines of Code (KLOC)
  - Effort = Person/month
  - Productivity = KLOC/person-month
  - Quality = Number of faults/KLOC
  - Cost = $/KLOC

  Documentation = Pages of documentation/KLOC
- The size measure is based on the lines of code computation. The lines of code is defined as one line of text in a source file.
- While counting the lines of code the simplest standard is
  - Don't count blank lines
  - Don't count comments
  - Count everything else
- The size oriented measure is not universally accepted method.

**Q.58 Give advantages and disadvantages of size oriented metrics**

**Ans. : Advantages**

1. Artifact of software development which is easily counted.
2. Many existing methods use LOC as a key input.
3. A large body of literature and data based on LOC already exists.

**Disadvantages**

1. This measure is dependent upon the programming language.
2. This method is well designed but shorter program may get suffered.
3. It does not accommodate non procedural languages.
4. In early stage of development it is difficult to estimate LOC.

**Q.59 What is function oriented metrics? Explain in detail.**

**Ans. :** • The function point model is based on functionality of the delivered application.

- These are generally independent of the programming language used.
- This method is developed by Albrecht in 1979 for IBM.
- Function points are derived using
  1. Countable measures of the software requirements domain
  2. Assessments of the software complexity.

**How to calculate function point ?**

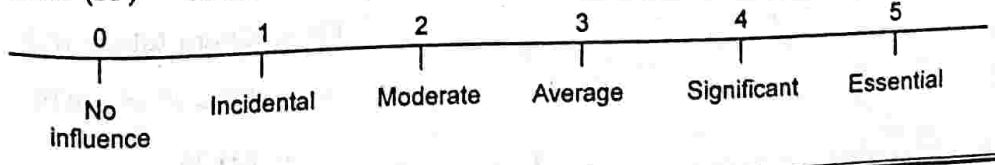
- The data for following information domain characteristics are collected
  1. Number of user inputs - Each user input which provides distinct application data to the software is counted.
  2. Number of user outputs - Each user output that provides application data to the user is counted, e.g. screens, reports, error messages.
  3. Number of user inquiries - An on-line input that results in the generation of some immediate software response in the form of an output.

4. Number of files - Each logical master file, i.e. a logical grouping of data that may be part of a database or a separate file.
5. Number of external interfaces - All machine-readable interfaces that are used to transmit information to another system are counted.

- The organization needs to develop criteria which determine whether a particular entry is simple, average or complex.
- The weighting factors should be determined by observations or by experiments.

| Domain Characteristics | Count | | Weighting factor | | | Count |
|---|---|---|---|---|---|---|
| | | | Simple | Average | Complex | |
| Number of user input | | X | 3 | 4 | 6 | |
| Number of user output | | X | 4 | 5 | 7 | |
| Number of user inquiries | | X | 3 | 4 | 6 | |
| Number of files | | X | 7 | 10 | 15 | |
| Number of external interfaces | | X | 5 | 7 | 10 | |
| **Count Total** | | | | | | |

- The count table can be computed with the help of above given table.
- Now the software complexity can be computed by answering following questions. These are complexity adjustment values.
  1. Does the system need reliable backup and recovery?
  2. Are data communications required?
  3. Are there distributed processing functions?
  4. Is performance of the system critical?
  5. Will the system run in an existing, heavily utilized operational environment?
  6. Does the system require on-line data entry?
  7. Does the on-line data entry require the input transaction to be built over multiple screens or operations?
  8. Are the master files updated on-line?
  9. Are the inputs, outputs, files or inquiries complex?
  10. Is the internal processing complex?
  11. Is the code which is designed being reusable?
  12. Are conversion and installation included in the design?
  13. Is the system designed for multiple installations in different organizations?
  14. Is the application designed to facilitate change and ease of use by the user?
- Rate each of the above factors according to the following scale :
- Function Points (FP) = Count total x $(0.65 + (0.01 \times \text{Sum}(F_i))$.



- Once the functional point is calculated then we can compute various measures as follows -
  - Productivity = FP/person-month
  - Quality = Number of faults/FP
  - Cost = $/FP
  - Documentation = Pages of documentation/FP.

**Q.60 Discuss the advantages and disadvantages of function oriented metrics**

**Ans. : Advantages**

1. This method is independent of programming languages.
2. It is based on the data which can be obtained in early stage of project.

**Disadvantages**

1) This method is more suitable for business systems and can be developed for that domain.
2) Many aspects of this method are not validated.
3) The functional point has no significant meaning. It is just a numerical value.

**Q.61 What is the difference between size oriented metrics and function oriented metrics.**

**Ans. :**

| Sr. No. | Size oriented metrics | Function oriented metrics |
|---|---|---|
| 1. | Size oriented software metrics is by considering the size of the software that has been produced. | Function oriented metrics use a measure of functionality delivered by the software. |
| 2. | For a size oriented metric the software organisation maintains simple records in tabular form. The typical table entries are : Project name, LOC, efforts, pages of documents, errors, defects, total number of people working on it. | In this metrics, computation of function point (FP) is done. The computation of FP is based on characteristics of softwares information domain and complexity. |

**Q.62 Compute the function point value for a project with the following information domain characteristic :**

**i) No. of external inputs 32**

**ii) No. of external outputs 60**

**iii) No. of external enquiries 24**

**iv) No. of internal logic files 08**

**v) No. of external interfaces file 02**

**Assume that all the complexity adjustment values are average.** [RGPV : June-15, Marks 7]

**Ans. : Given that**

1. Number of user input = 32
2. Number of user output = 60
3. Number of user enquiries = 24
4. Number of files = 8
5. Number of external interfaces = 2

Complexity adjustment values are average. The 14 algorithms are used.

| | Domain characteristics | Count | × | Weighing Factor | |
|---|---|---|---|---|---|
| 1. | No. of user input | 32 | × | 4 | 128 |
| 2. | No. of user output | 60 | × | 5 | 300 |
| 3. | No. of user inquiry | 24 | × | 4 | 96 |
| 4. | No. of files | 8 | × | 10 | 80 |
| 5. | No. of external interface | 2 | × | 7 | 14 |
| | | | | | 618 |

As complexity adjustment values are average

$$\sum_{i=1}^{14} F_i = 14 \times 3 = 42$$

$$\therefore \quad FP = \text{Count total} \times [0.65 + 0.01 \times \sum F_i]$$

$$= 618 \times [0.65 + 0.01 \times 42]$$

$$= 661.26$$

**Q.63 Consider a project with the following functional unit :**

**Number of user inputs = 50**

**Number of user outputs = 40**

**Number of user enquires = 35**

**Number of user files =06**

**Number of external interfaces =04**

**Assume all complexity adjustment factors and weighting factors are averages. Compute the function points for the project.**

☞ **[RGPV : Dec.-15, Marks 7]**

**Ans. :** Assume complexity adjustment values are average. The 14 algorithms are used.

| Sr. No. | Domain Characteristic | Count | × | Weighing factor | |
|---|---|---|---|---|---|
| 1 | No. of user input | 50 | × | 4 | 200 |
| 2 | No. of user output | 40 | × | 5 | 200 |
| 3 | No. of user enquiry | 35 | × | 4 | 140 |
| 4 | No. of user files | 06 | × | 10 | 60 |
| 5 | No. of external interfaces | 04 | × | 7 | 28 |
| | | | | | 628 |

As complexity adjustment values are average

$$\sum_{i=1}^{14} F_i = 14 \times 3 = 42$$

$$FP = \text{Count total} \times [0.65 + 0.01 \times \sum F_i]$$

$$= 628 \times [0.65 + 0.01 \times 42]$$

$$FP = 671.96$$

*END...*