1. Question 1

   Describe the main parameters that you should use when computing the costs of a software development project?

   Your Answer

   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

2. Question 2

   Elaborate the concept of modularity in software development? Why is it needed? What is its strength?

   Your Answer

   Size Oriented Metrics derived by normalizing quality and productivity Point Metrics measures by considering size of the 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.

   This metrics is one of simplest and earliest metrics that is used for computer program to measure size. Size Oriented Metrics are also used for measuring and comparing productivity of programmers. It is a direct measure of a Software. The size measurement is based on lines of code computation. The lines of code are defined as one line of text in a source file.

   While counting lines of code, simplest standard is:

   Don’t count blank lines
   Don’t count comments
   Count everything else
   The size-oriented measure is not a universally accepted method.
   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 / KLOCSize oriented software metrics are derived by normalizing quality and/or productivity measures by considering the size of the software that has been produced.
If a software organization maintains simple records, a table of size-oriented measures
Size-oriented metrics are not universally accepted as the best way to measure the software process but most of the controversy swirls around the use line of code as key measure.Function-Oriented Metrics is a method that is developed by Albrecht in 1979 for IBM (International Business Machine). He simply suggested a measure known as Function points that are derived using an empirical relationship that is based on countable measures of software's information or requirements domain and assessments of the complexity of software.

Function-Oriented Metrics are also known as Function Point Model. This model generally focuses on the functionality of the software application being delivered. These methods are actually independent of the programming language that is being used in software applications and based on calculating the Function Point (FP). A function point is a unit of measurement that measures the business functionality provided by the business product.

To determine whether or not a particular entry is simple, easy, average, or complex, a criterion is needed and should be developed by the organization. With the help of observations or experiments, the different weighing factors should be determined as shown below in the table. With the help of these tables, the count table can be computed.
Difference Between Size oriented metrics and Function oriented metrics

Size oriented -

1-Direct measure of software.

2-Attempt to measure the size of software.

3-It focuses on the lines of code.

4-It is dependent of programming language.

Function oriented -

1-Indirect measure of software.

2-Attempt to measure the functionality of software.

3-It focuses on function points.

4-It is independent of programming language

3. Question 3

Write a short note on the following:

0. a)Sizeoriented metrics and
1. b)Functionoriented metrics

How these two metrics are different from each other? Explain by captivating a suitable example.

Your Answer

Size-oriented Metrics
Attempt to quantify software projects by using the size of the project to normalize other quality measures

- Possible data to collect:
- number of lines of code
- number of person-months to complete
- cost of the project
- number of pages of documentation
- number of errors corrected before release
- number of bugs found post release

Function-Oriented Metrics

- Attempt to measure the functionality of a software system
- Use a unit of measure called function point
- Some possible function points:
- Internal data structures
- External data structures
- User inputs
- User outputs
- Transformations
- Transitions

How it is to be considered that software having good quality will have good functionality? Discuss the various parameters which play a keen role during quality assurances.

Your Answer

What is software quality?

The quality of software can be defined as the ability of the software to function as per user requirements. When it comes to software products it must satisfy all the functionalities written down in the SRS document.

Key aspects that conclude software quality include,

Good design – It's always important to have a good and aesthetic design to please users

Reliability – Be it any software it should be able to perform the functionality impeccably without issues

Durability- Durability is a confusing term, In this context, durability means the ability of the software to work without any issue for a long period of time.

Consistency – Software should be able to perform consistently over platform and devices

Maintainability – Bugs associated with any software should be able to capture and fix quickly and news tasks and enhancements must be added without any trouble

Value for money – customer and companies who make this app should feel that the money spent on this app has not gone to waste.

Question 2

Highlights the various types of risks associated during software development.

Your Answer

A software project can be concerned with a large variety of risks. In order to be adept to systematically identifying the significant risks which might affect a software project, it is essential to classify risks into different classes. The project manager can then check which risks from each class are relevant to the project.

There are three main classifications of risks that can affect a software project:

Project risks

Technical risks

Business risks

1. Project risks: Project risks concern different forms of budgetary, schedule, personnel, resource, and customer-related problems. A vital project risk is schedule slippage. Since the software is intangible, it is very tough to monitor and control a software project. It is very tough to control something which cannot be identified. For any manufacturing program, such as the manufacturing of cars, the plan executive can recognize the product taking shape.

2. Technical risks: Technical risks concern potential method, implementation, interfacing, testing, and maintenance issues. It also consists of an ambiguous specification, incomplete specification, changing specification, technical uncertainty, and technical obsolescence. Most technical risks appear due to the development team's insufficient knowledge about the project.

3. Business risks: This type of risk contain risks of building an excellent product that no one need, losing budgetary or personnel commitments, etc.

Another risk like .

1. Known risks: Those risks that can be uncovered after careful assessment of the project program, the business and technical environment in which the plan is being developed, and more reliable data sources (e.g., unrealistic delivery date)

2. Predictable risks: Those risks that are hypothesized from previous project experience (e.g., past turnover)

3. Unpredictable risks: Those risks that can and do occur, but are extremely tough to identify in advance.

Describe the software metrics for analysis and design models.

A software metric is a measure of software characteristics which are measurable or countable. Software metrics are valuable for many reasons, including measuring software performance, planning work items, measuring productivity, and many other uses.

Within the software development process, many metrics are that are all connected. Software metrics are similar to the four functions of management: Planning, Organization, Control, or Improvement.

Internal metrics: Internal metrics are the metrics used for measuring properties that are viewed to be of greater importance to a software developer. For example, Lines of Code (LOC) measure.

External metrics: External metrics are the metrics used for measuring properties that are viewed to be of greater importance to the user, e.g., portability, reliability, functionality, usability, etc.

Hybrid metrics: Hybrid metrics are the metrics that combine product, process, and resource metrics. For example, cost per FP where FP stands for Function Point Metric.

Project metrics: Project metrics are the metrics used by the project manager to check the project's progress. Data from the past projects are used to collect various metrics, like time and cost; these estimates are used as a base of new software. Note that as the project proceeds, the project manager will check its progress from time-to-time and will compare the effort, cost, and time with the original effort, cost and time. Also understand that these metrics are used to decrease the development costs, time efforts and risks. The project quality can also be improved. As quality improves, the number of errors and time, as well as cost required, is also reduced.

Software metrics can be classified into two types as follows:

1. Product Metrics: These are the measures of various characteristics of the software product. The two important software characteristics are:

Size and complexity of software.

Quality and reliability of software.

These metrics can be computed for different stages of SDLC.

2. Process Metrics: These are the measures of various characteristics of the software development process. For example, the efficiency of fault detection. They are used to measure the characteristics of methods, techniques, and tools that are used for developing software.