B.TECH.

IT-601/TIT-601

SIXTH SEMESTER EXAMINATION, 2006-2007

SOFTWARE PROJECT MANAGEMENT

Time: 3 *Hours* *Total Marks:* 100

Note: *Answer all questions. Make suitable assumption wherever necessary.*

Q.1. Attempt any two parts of the following: (10×2=20)

1. (*a*) What do you mean by Work Breakdown Structure (WBS) in context to software project and product? Discuss with examples.

Ans. A much favoured way of generating a task list is to create a WBS. This involves identifying the main tasks required to complete a project and then breaking each of these down into a set of lower level tasks.

The above figure shows a fragment of a WBS where the design task has been broken down into three tasks and one of these has been further decomposed.

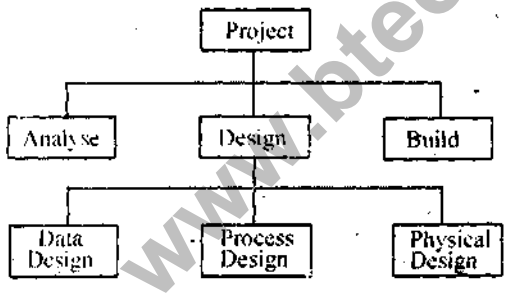

Fig.

While preparing a WBS, consideration must be given to the final level of detail or depth of the structure. Too great a depth will result in a large number of small tasks that will be difficult to manage, whereas a too shallow structure will provide insufficient detail for project control.

1. (*b*) End users demands are the source of most software project. Justify this statement. Also give various steps of a software project plan.

Ans. It is rigidly remarked that end user are the source for software project because they are the one whose demands are responsible for carrying at the projects. It is they who put forward their requirements are thus acts as client which are fulfilled by software companies. Hence, we can say that they are infact responsible for carrying out software project.

For various steps of software project plan: Please see Q.No. 1(*d*) of 2004-05.

1. (*c*) It has been said that the design can not be directed by technical experts alone. Do you agree with this statement? Write in detail to support your views. Give various responsibilities of a Project Management alongwith structure of personnel organisation.

Ans. Design can't be fulfilled alone by technical expert because we need input from other parts involved also. We also need functional experts to suggest guidelines as to how to arrange the flow of information in the project. We also need clients to give their valuable inputs that could be incorporated in the designing process. So, the above statement is rightly true.

Please See Q.No. 1(*b*) of 2003-04.

2. Attempt any two parts of the followings: (10×2=20)

2. (*a*) It seems odd that cost and schedule estimates are developed during software project planning before detailed software

requirement analysis or design has been conducted. Why do you think that this is to be done? Are there circumstances when it should not be done? What is cash flow forecasting?

Ans. For a given set of requirements it is desirable to know how much it will cost to develop the software to satisfy the given requirements and how much time development will take. These estimates are needed before development is initiated. The primary reason for cost and schedule estimation is to enable the client or developer to perform a cost benefit analysis and for project monitoring and central.

Cash flow forecast indicates when expenditure and income will take place. We need to spend money, such as staff wages, during development stages of a project.

2. (*b*) Discuss cost benefit analysis in details. What are the following terms: (*i*) Net profit value (*ii*) Return on investment (*iii*) Pay-back period.

Ans. (*i*) Net Profit: The net profit of the project is the difference total costs total income over the life of the project.

(*ii*) Payback Period: The payback period is the time taken to break even or payback the initial investment. Normally the project with shorest payback period is chosen.

(*iii*) Return on Investment (ROI): It is also known as accounting rate of return (ARR) It provides a way of comparing the net profitability with the investment required

$$\text{ROI} = \frac{\text{Avg. Annual Profit}}{\text{Total Investment}} \times 100$$

2. (*c*) Provide atleast five examples of software development projects that would require prototyping. Name two or three applications that would be more difficult to prototype.

Ans. The conventional life cycle model can allow for prototying within any of the phases. Unfortunately, this approach can be difficult to control when, for example, coding of a user interface takes place during requirements formulation and analysis. Hoffman provided a methodology to aid in this area.

Rapid Prototyping: The rapid prototyping model strives for demonstrating functionality early on the development of a system. This approach has been proposed for use in conjunction with risk management under a study sponsored by the US Air Force Electronic Systems Division. Examples of rapid prototyping were reported with notations that supported the rapid deployment of functionality.

Evolutionary Prototyping: In evolutionary prototyping the focuss is on achieving functionality for demonstrating a portion of the system to the end user for feedback and system growth. The prototype emerges as the actual system downstream in the life cycle. As with each iteration in development, functionality is added and then translated to an efficient implementation. Also of interest is functional programming and relational programming as a means for accomplishing evolutionary prototyping. Methods have been described for controlling prototyping from development through maintenance which are beneficial for incorporating this new technology.

Applications where it is difficult to implement: Prototyping has not been as successful as anticipated in some organizations for a variety of reasons. Training, efficiency, applicability, and behaviour can each have a negative impact on using software prototyping techniques.

1. Learning Curve: A common problem with adopting prototyping technology is high expectations for productivity with insufficient effort behind the learning curve. In addition to training for the use of a prototyping technique, there is an often overlooked need for developing corportate and project specific

underlying structure to support the technology. When this underlying structure is omitted, lower productivity can often result.

2. Tool Efficiency: Prototyping techniques outside the domain of conventional programming languages can have execution inefficiencies with the associated tools. One prominent prototyping technique was the basis for the ESPRIT funded SETL to Ada project as an effort to provide efficient transformations. The efficiency question was argued as a negative aspect of prototyping.

3. Applicability: Application domain has an impact on selecting a prototying technique. There would be limited benefit to using a technique not supporting real-time features in a process control system. The control room user interface could be described, but not integrated with senso monitoring deadlines under this approach. Goldsack and Finkelstein provide a discussion on various unique aspects of real time requirements.

3. Attempt any two parts of the following: **(10×2=20)**

3. (a) Discuss the role of cost estimation in a software development project. Briefly explain COCOMO model for cost estimation for all category of projects.

Ans. Software Effort Estimates Techniques :

1. *Algorithmic models :* Which use of effort presenting characteristics of the target system and the implementation environment to predict effort.
2. *Expert Judgement :* Where the advice of knowledge staff is solicited.
3. *Analogy :* Where a similar, completed, project is identified and its actual effort is used as a basis for the new project.
4. *Parkinson :* Which identifies the staff effort available to do a project and uses that as the 'estimates'.
5. *Price to Aim :* Where the estimate is a figure that appears to be sufficiently low to win a contract.
6. *Top down :* Where an overall estimate is formulated for the whole project and is then broken down into the effort required for component tasks.
7. *Bottom-up :* Where component task are identified and sized and these individual estimates are aggregated.

The basic COCOMO model is built around the equation, it is a parametric model, mainly used for cost based analysis. The abbreviation stands for constructive cost model.

It gives an equation effort = $C \times \text{size}^K$

where effort is measured in pm, or the number of person methods consisting of units of 152 working hours. Size is measured in kdsi, thousand of delivered source code instructions and C and K are constants.

The first step says to derive an estimate of the system size in term of kdsi. The constant C and K, depended on whether the system could be classified in Bohem's term as "organic", "semidenoted" or "embedded". These selected to the technical nature of the system and development environment.

Organic mode : This would typically be the care when relatively terms developed terms is highly familiar in house environment and when the system being developed was small and the interface requirement are flexible.

Embedded Mode : This means the product being developed hard to operate within very tight constraints and changes to the system were very costly. Semidetached mode this combineds the

elements of the organic and the embedded modes or characteristic that came between the two COCOMO constants.

System type	C	K
Organic	2.4	1.05
Semidetached	3.0	1.12
Embedded	3.6	1.20

The exponent value K when it is greater than 1, means larger projects and seen as requiring disproportionately more effort than smaller ones.

3. (*b*) Describe the function point analysis. How function points are used in estimation of cost and efforts using decomposition technique.

Ans. Function point analysis was developed by Allen Albrecht when he worked for IBM. The basis of FP is that computer based information systems comprise five major components or external user types that are benefit to the users.

These are namely external input types, external input types logical internal file types, external interface file types, external inquiry types.

The analyst has to identify each instance of each external user type in the projected system. Each component is the classified as having either high, average or low complexity.

3. (*c*) Is the critical path important if only one person is working on a software project? Discuss the concept of PERT/CPM in defining an optimal schedule.

Ans. CPM and PERT are Network planning modules Critical Path Method and Program Evaluation Review Techniques.

The answer should consists of :

1. Constructing CPM network
2. Using dummy activities
3. Rep. Lagged activities
4. Adding time aimensions
5. Forward pass
6. Backward pass
7. Identifying critical path
8. Activity float

Evolutionary risk to the schedule:

Using PERT to evaluate the efforts of uncertainty expected time and standard deviations.

4. Attempt any two parts of the followings: **(10×2=20)**

4. (*a*) Risk management is the area that tries to ensure that the impact of risk on cost, quality and schedule be minimal. Justify the statement. What is decision tree analysis in context to Risk Management?

Ans. A decision tree (or tree diagram) is a decision support tool that uses a graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. A decision tree is used to identify the strategy most likely to reach a goal. Another use of trees is as a descriptive means for calculating conditional probabilities.

In data mining and machine learning, a decision tree is a predictive model; that is, a mapping from observations about an item to conclusions about its target value. More descriptive names for such tree models are classification tree (discrete outcome) or regression tree (continuous outcome). In these tree structures, leaves represent classifications and branches represent conjunction of features that lead to those classifications. The machine learning technique for inducing a decision tree from data is called decision tree learning, or (colloquially) decision trees.

4. (*b*) List and discuss some of the points specific for identifying the risks during software development. Also give some of the category of Risks that are to be identified.

Ans. Factors that affect the risk identification procedure of any activity.

This includes the hazards that might affect the duration or resource cost of the project.

1. Application factor — Size and nature of the system e.g. is the proposed system data processing safety critical etc.
2. Staff factors — Experienced and skills of staff involved
3. Project factors — Project and objective well defined to the members.
4. Project method
5. Hardware/software factors.
6. Changeover factors
7. Supplier factors.
8. Environment factors
9. Health and safety factors

4. (*c*) As size is the main factor determining the cost of a project, an accurate size can be used to estimate the cost and schedule of the software project. Give your view in favour and in against of the statement. Also write in brief about Contract Management and Human Resource Management.

Ans. Accurate size can be used to estimate cost and schedule of the software project. It is the size of the project that matters. If the project is large, then definitely cost will be more and time taken to complete it will be more. So, we can say that size is the main determinant for estimating cost and schedule of the software project.

Contract Management : When the contract is being negotiated, certain key points in the project can be identified where customer approval is needed before the project can proceed. For each decision point, the deliverable to be presented by the suppliers, the decision to be made by the customer and the outputs from the decision point all need to be defined. These decision points have added significance if payments to the supplier are based on them.

When work is contracted out there will be a general concern about the quality of that work. The ISO 12207 standard envisages the possibility of there bing agents, employed independently of the supplies or customer, who will carry out verification, validation and quality assurance. It also allows for joint reviews of project processes and products, the nature of which needs to be clearly agreed when the contract is negotiated, otherwise the supplier might claim unwarranted interference in their work.

As the system is developed a need to change certain of the requirements often emerges. An effective change control procedure is therefore needed to record requests for changes, along with the supplier's agreement to them and any fees for the additional work improve motivation the manager might therefore do the following:

- *Set specific goals* these goals need to be demanding and yet acceptable to staff, involving staff in the setting of goals helps to gain acceptance for them.

- *Provide feedback* not only do goals have to be set, but staff have to have regular feedback about how they are progressing.
- *Consider job design* jobs can be altered to make them more interesting and give staff more feeling of responsibility.

Two measure are often used to enhance job design– *job enlargement* and *job enrichment.*

- *Job enlargement:* The person doing the job carries out a wider variety of activities. It is the opposite of increasing specialization. For example, a programmer in a maintenance group might be given responsibility for specifying minor amendments as well as carrying out the actual code changes. Couger and Zawacki found that programmer/analysts had higher job satisfaction than programmers.

5. Attempt any two parts of the followings: **(10×2=20)**

5. (*a*) Discuss about software quality factors and attributes.

Ans. The Importance of Software Quality : We would expect to be a concern of all producers of goods and services. However, the special characteristics of software, and in particular its intangibility and complexity, make special demands.

- *Increasing critically of software :* The final customer or user is naturally anxious about the general quality of software, especially its reliability. This is increasingly the case as organizations become more dependent on their oomputer systems and software is used more and more in areas which are safety critical, for example to control aircraft.
- *The intangibility of software :* This makes it difficult to know whether a particular task in a project has been completed satisfactorily. The results of these tasks can be made tangible by demanding that the developer produce 'deliverable' that can be examined for quality.
- *Accumulating errors during software development:* As computer system develo-pment is made up of a number of steps where the output from one step is the input to the next, the errors in the earlier deliverable will be added to those in the later steps leading to an accumulating detrimental effect. In general, the later in a project that an error is found the more expensive it will be to fix. In addition, because the number of errors in the system is unknown the debugging phases of a project are particularly difficult to control.

For these reasons quality management is an essential part of effective overall project management.

Defining Software Quality : Quality is a rather vague term and we need to define carefully what we mean by it. For any software system, there should be three specifications :

- *a functional specification* describing what the system is to do – methodologies such as SSADM are primarily concerned with this;
- *a quality (or attribute) specification* concerned with how well the functions are to operate;
- *a resource specification* concerned with how much is to be spent on the system.

At Brightmouth College, Brigette has to select the best off-the-shell payroll package of the college. How should she go about this in a methodical manner ?

One element of the approach could be the identification of criteria against which payroll packages are to be judged. What might these criteria be ? How could you check the extent to which packages match these criteria?

One attempt to identify specific product qualities that are appropriate to software has been that of

James A. McCall. He grouped software qualities into three sets the quality factors :

- product operations qualities;
- product revision qualities;
- product transition qualities.

The definition below are those given by McCall, but the reader may come across others. These are not all-bracing; sometimes other qualities might be of interest.

Product Operation Quality Factors :

- *Correctness:* The extent to which a program satisfies its specifications and fulfil the user's objectives.
- *Reliability:* The extent to which a program can be expected to perform intended function with required precision.
- *Efficiency:* The amounts of computer resources required by the software.
- *Integrity:* The extent to which access to software or data by unauthorized persons can be controlled.
- *Usability:* The effort required to learn, operate, prepare input and interpreter output.

Product Revision Quality Factors :

- *Maintainability:* The effort required to locate and fix an error in an operation program.
- *Testability:* The effort required to test a program to ensure it performs it intended function.
- *Flexibility:* The effort required to modify an operational program.

Product Transition Quality Factors

- *Portability:* The effort required to transfer a program from one hardware configuration and/or software system environment to another.
- *Reusability:* The extent to which a program can be used in other applications.
- *Interpretability:* The effort required to couple one system to another.

5. (*b*) Development our metrics for correctness, maintainability, integrity and usability of software. What is Statistical Quality Assurance (SQA)?

Ans. Please See Q.No. 5(*a*) of 2003-04.

5. (*c*) Write a short note on SEI Capability Maturity Model (CMM). How does it differ from ISO 9000?

Ans. Capability Process Models : Rather than just checking that a system is in place to detect faults, a customer might wish to check that a supplier is using software development methods and tools which are likely to produce good quality software. A customer might feel more confident, for instance, if they know that their software supplier is using structured methods. In the United States, an influential *capability maturity model* (CMM) has been developed at the Software Engineering Institutes (SEI), a part of the Carnegie-Mellon University. This attempts which indicate the sophisticated and quality of their software production practices. These levels are defined as follows :

- *Level 1 : Initial*–The procedures followed tend to be haphazard. Some projects might be successful, but this tends to be because of the skills of particular individuals including project managers. The is no level 0 and so any organization would be at this level by default.

- *Level 2: Repeatable–* Organizations at this level will have basic project management procedures in place. However, the way individual tasks are carried out will depend largely on the person doing it.
- *Level 3: Defined–* The organization has defined the way that each task in the software development life cycle should be done.
- *Level 4: Managed–* The products and processes involved in software development are subject to measurement and control.
- *Level 5: Optimizing–* Improvement in procedures can be designed and implemented using the data gathered from the measurement process.

For each of the levels, apart from the default level 1, *key process areas* (KPA) have been identified as distinguishing the current level from the lower ones. These are listed in the Table below

The assessment is done by a team of assessors coming into the organization and interviewing key staff about their practices using a standard questionnaire to capture the information. A key objective is not just to assess, but to recommend specific actions to bring the organization up to a higher level.

Level	*Key process areas*
1. Initial	Not applicable
2. Managed	Configuration management, quality assurance, subcontract management, project tracking and oversight, project planning
3. Defined	Peer reviews, inter-group coordination, software product engineering, integrated software management, training programme, organization process definition and focus.
4. Managed	Quality management, process measurement and analysis.
5. Optimizing	Process change management, technology innovation, defect prevention.