# Sardar Patel Institute of Technology

Bhavan's Campus, Munshi Nagar, Andheri (West), Mumbai-400058, India
(Autonomous College Affiliated to University of Mumbai)

**End Semester Examination**
Nov – Dec 2018

Max. Marks: 60 | Duration: 3Hrs
Class: T.E. | Semester:V
Course Code: IT51 | Branch:Information Technology
Name of the Course: Software Engineering

## SYNOPTIC

| QNo | | Marks | CO |
|---|---|---|---|
| Q1a) | Objective of the feasibility study(2 Marks)<br><br>The main focus of the feasibility study is to determine whether it would be financially and technically feasible to develop the software.<br>Data collected during feasibility study are analyzed to perform :<br>1) Development of an overall understanding of the problem.<br>2) Formulation of the various possible strategies for solving the problem.<br>3) Evaluation of the different solution strategies.<br><br>Types of feasibility: (3 Marks)<br><br>1) Operational feasibility<br>2) Technical feasibility<br>3) Economic feasibility<br>4) Schedule feasibility | 5 | 1 |
| b) | Extreme Programming<br>Diagram -2Marks<br>Description-5Marks<br>Key points:<br>1) User stories | 7 | 1 |

| | | | |
|---|---|---|---|
| | 2) CRC cards<br>3) Spike solution<br>4) Pair programming<br>5) Unit testing<br>6) Acceptance testing<br>7) Project velocity<br><br>OR<br>Use of a development process – (1Mark)<br>Key points:<br>Encourages development of software in a systematic and disciplined manner.<br><br>The following (or suitable alternatives) would be expected: (1mark for choosing appropriate process model and 1mark for justification)<br>Project1: Waterfall (No need for prototyping and requirements must be precisely specified)<br>Project 2: Evolutionary prototyping (requirements are vague and unstable, so there is a need for rapid prototyping leading to the final product)<br>Project 3: Prototyping 'merged' with waterfall (requirements are vague, so there is a need for prototyping, but they are stable, so the waterfall approach can be used – to develop the back end sub-system in particular) | | |
| Q2 a) | From the basic COCOMO estimation formula for organic software:<br>Effort = $2.4 \text{ X } (32)^{1.05}$<br>= 91 PM<br><br>Nominal development time = $2.5 \text{ X } (91)^{0.38}$<br>= 14 Months<br><br>Staff cost required to develop to develop the product<br>= 91 X Rs.15,000<br>= 1,465,000 Rs. | 6 | 3 |
| b) | Everyone (All Stakeholders) involved in the software engineering process is responsible for quality.(1Mark)<br>Objective of Formal Technical Review-(2 Marks)<br>1) To uncover the error | | 3 |

2) To verify software meets its requirements.
3) Ensure that the software has been represented according to predefined standards.
4) To make project more manageable.

Guidelines for Formal Technical review (3Marks)

1) Review the product not the producer.
2) Set an agenda and maintain it
3) Limit debate and rebuttal.
4) Enunciate problem areas but don't attempt to solve every problem noted.
5) Take written notes.
6) Limit the number of participants and insit upon advance preparation.
7) Develop a checklist.
8) Allocate resources and schedule.
9) Conduct meaningful training for all reviewers.
10) Review your early reviews.

**Q3 a)** a) Following are possible risks. (Identification of each risk carry 1 mark) — 2 — 3

1) Reusable software may contain defects that mean they cannot be reused as planned.
2) Development time is difficult to predict.

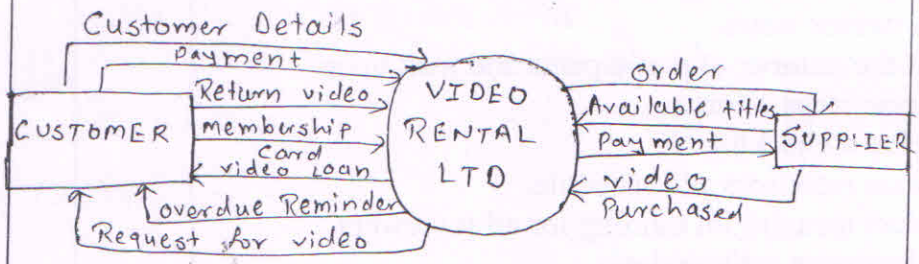
a) Risk checklist — 2

| | |
|---|---|
| Technology | Reusable software may contain defects that mean they cannot be reused as planned. |
| Estimation | Development time is difficult to predict. |

b) Mitigation action(2 Marks for each) — 4

| Risk | Mitigation Plan |
|---|---|
| Reusable software may contain defects that mean they cannot be reused as planned. | Replace potentially defective components with bought-in components of known reliability. |
| Development time is difficult to predict. | Adopt an agile development approach such as SCRUM. |

**b)** Baseline definition/purpose (2marks) — 4 — 3

| | | | |
|---|---|---|---|
| | A baseline is a software configuration management concept that helps to control changes without seriously impeding justifiable change.<br>Example (2marks) | | |
| Q4<br>a) | Context level DFD : (2Marks)<br>  <br><br>Level 1 DFD: (3Marks)<br>Further decomposition of context level DFD to show all the process and flow of information required.<br><br>Level 2: (3Marks)<br>Process identified at level 1 can be further decomposing to show detail design. | 8 | 2 |
| b) | It describe a design structure that solves a particular design problem within a specific context and amid "forces" that may have an impact on the manner in which the pattern is applied and used.(1Mark) | 4 | 2 |

| | | | |
|---|---|---|---|
| | Types of Design patterns (3Marks)<br>1) Architectural<br>2) Design<br>3) Idioms<br><br>OR<br>User interface design principal(Any Four)<br>1) User familiarity<br>2) Consistency<br>3) Minimal surprise<br>4) Recoverability<br>5) User guidance<br>6) User diversity | | |
| Q5 a) | Testability- How easily a computer program can be tested.<br><br>Characteristics-(Any Five )<br><br>1) Operability<br>2) Observability<br>3) Controllability<br>4) Decomposability<br>5) Simplicity<br>6) Stability<br>7) Understandability | 1<br><br>5 | 4 |
| b) | Real time system testing (2Marks)<br>1) Test case designer need to consider conventional test cases but also event handling, the timing of the data and the parallelism of the tasks that handles the data need to be considered.<br>2) The relationship between real time software and its hardware environment can also cause testing problem .Hence; the software test must consider the impact of hardware faults on software processing.<br><br>Strategy for testing a real time system(4Marks)<br>1) Task testing<br>2) Behavioral testing<br>3) Inter task testing<br>4) System testing | 2<br><br>4 | 4 |

| | | | |
|---|---|---|---|
| | OR<br><br>Verification: are we building the product right? (1Mark)<br><br>Verification has focus on checking that the product specification is being met through the functional and non-functional requirements. Software verification involves such things as testing and bug fixing, unit tests, integration tests etc. Satisfying requirements and design criteria.<br><br>Validation: are we building the right product? (1Mark)<br><br>Validation is checking that the product is actually what the client wants. Validation testing is ensuring the product conforms or meets customer expectations acceptance tests are part of validation testing etc.<br><br>Overall strategy for software testing.<br><br>Diagram:(2Marks)<br>Description :(2Marks)<br>Key points:<br>1) Unit testing<br>2) Integration testing<br>3) Validation testing<br>4) System testing<br>5) Code<br>6) Design<br>7) Requirement<br>8) System Engineering | 2<br><br>4 | |