B.Tech 4th Semester Examination, 2014
Model Answer

Subject:- Software Eng. **Paper Code:-** 05414

Sets (I) / (II)

Q1. a) C (b) b)B c)B d)D e)A f)B g)B h)D i)D j)A

Q2. What is software life cycle model? Discuss different phases of Waterfall Software Life Cycle Model. What are the major advantages and disadvantages of Waterfall Software Life Cycle Model?

Ans:

System Development Life Cycle (SDLC) is the overall process of developing information Systems through a multi-step process from investigation of initial requirements through Analysis, design, implementation and maintenance.

Modelling Phase
In this phase we view the software product as part of a larger system or organization where the product is required. This is basically a system view where all the system elements are created.

Software Requirements Analysis
Here we have a phase where the requirements are gathered. The information domain for the software is understood. The function, behaviour, performance and interfacing of the software are determined. The requirements of the software and the customer are decided upon.

Design
This determines the data structures, the software architecture, and the interface representations and the procedural (algorithmic) detail that goes into the software.

Code Generation
Here the actual programming is done to obtain the machine code; it is an implementation of the design.

Testing
The testing is a process that goes hand in hand with the production of the machine code. There are a number of testing strategies. First unit testing is done and then integration testing. Alpha testing is to see if the software is as per the analysis model whereas beta testing is to see if the software is what the customer wanted.

Installation
The software is released to the customer.

Maintenance
This is the largest phase of the software life cycle. Maintenance can be of different types: to modify the software as the requirements of the customer evolve, to remove the residual bugs in the software etc.

Benefits of waterfall model
- The waterfall model is simple to implement.
- For implementation of small systems waterfall model is useful.

Name of Setter: - A. Kumar Designation:- Asst. Prof.

Address:- NIT Patna Signature of Setter

Remarks: Answer of Q① (a) is taken as option (b) instead of option (c) given in this Memorandum.

24/8/14

SUMIT KUMAR
HOD, CSE DEPT. VVIT PURNEA
9334249879.

B.Tech 4th Semester Examination, 2014
Model Answer

Paper Code:-

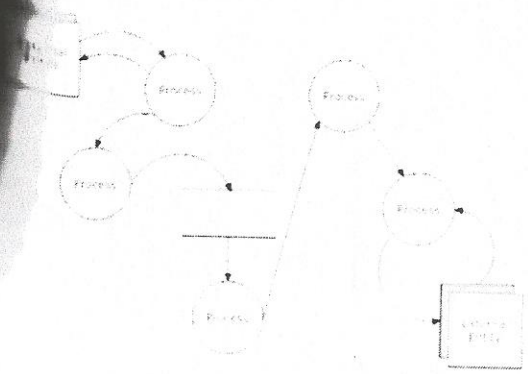
Sets (I) / (II)

model:
the sequential flow in software development process
is done initially and sometimes it is not possible to state all the requirements
This causes difficulty.
working model of the project only at the end. After reviewing of the working
dissatisfied then it cause serious problems.
model induces blocking states, because certain task may be dependent on
it is necessary to accomplish all the dependant tasks first. It may cause long

w diagrams and Entity-Relationship diagrams? What are purposes using
ules to be followed while creating Data Flow Diagram? Give an example

a means of representing a system at any level of detail with a graphic network
flows, data stores, data processes, and data sources/destinations.
w diagrams is to provide a semantic bridge between users and systems
re:
ing thousands of words;
ns - Modelling WHAT a system does, rather than physical models

ystems at any level of detail; and
er understanding and reviewing.



Below are the steps that required to be followed while creating Data Flow Diagram:

- No process can have only outputs or only inputs. The process must have both outputs and inputs
- the problem description can be identified as processes in the system.
- direct flow between data stores and external entity. This flow should go

Designation:-

Signature of Setter

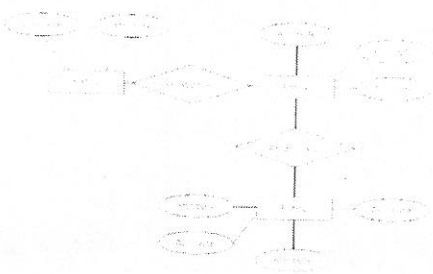
B.Tech 4th Semester Examination, 2014
Model Answer

Subject:- **Paper Code:-**

Sets (I) / (II)

- Data store labels should be noun phrases from problem description.
- No data should move directly between external entities. The data flow should go through a process.
- Generally source and sink labels are noun phrases.
- Interactions between external entities is outside scope of the system and therefore not represented in DFD.
- Data flow from process to a data store is for updation/insertion/deletion.
- Data flow from data store to process is for retrieving or using the information.
- Data flow labels are noun phrases from problem description

Entity Relationship Diagram is a graphical representation of the data layout of a system at a high level of abstraction. It defines data elements and their inter-relationships in the system.



Q4.

a. What are important characteristics of SRS? Discuss briefly.

Ans:

Some of important characteristics of SRS are as follows:

i. **Correct:** The SRS should be made up the date when appropriate requirements are identified.

ii. **Unambiguous:** When the requirements are correctly understood then only it is possible to write unambiguous software.

iii. **Complete**: To make SRS complete, its hold be specified what a software designer wants to create software.

iv. **Consistent:** It should be consistent with reference to the functionalities identified.

v. **Specific:** The requirements should be mentioned specifically.

vi. **Traceable:** What is the need for mentioned requirement? This should be correctly identified.

b. Differentiate between Software Quality Control and Software Quality Assurance. What steps are required to perform Statistical SQA?

Name of Setter: - Designation:-

Address:- Signature of Setter

B.Tech 4th Semester Examination, 2014
Model Answer

Subject:- **Paper Code:-**

Sets (I) / (II)

Ans:

Software Quality Control involves series of inspections, reviews and tests which are used throughout software process to ensure each work product meets requirements placed upon it.

Software Quality Assurance is a set of auditing and reporting functions that assess effectiveness and completeness of quality control activities.

Following are the steps required to perform for statistical SQA:

- Information about software defects is collected and categorized.
- Attempt is to trace each defect.
- Using Pareto principal, isolate 20%
- Once vital causes are identified, correct problems can be enforced to overcome.

Q5.

a. Explain the Capability Maturity Model.

Ans:

CMM model strives to achieve predictability and consistency as a precursor to continuous improvements by following a set of process in a well-defined framework.

- Level 1 is Initial Level
- Level 2 is repeatable which helps in achieving repeatability of performance and quality should the organizations undertake a similar project again.
- Level 3 is defined level.
- Level four is measured level.
- Level 5 is optimistic level, here people always work towards a target.

b. What is FP? How it is used for project estimation?

Ans:

Function Point: It is used as the estimation variable to size each element of the software. It requires considerably less details. It is estimated indirectly by estimating the number of inputs, outputs, data files, external interfaces.

c. What is LOC? How it is used for project estimation? Write the formula to calculate the effort in persons-months used in Dynamic multi variable Model?

Ans:

LOC: Lines of Code. It is used as estimation variable to size each element of the software. It requires considerable level of detail.

Name of Setter: Designation:

Address:- Signature of Setter

B.Tech 4th Semester Examination, 2014
Model Answer

Subject:- **Paper Code:-**

Sets (I) / (II)

Software Equation: E = [LOC' B^333/P]3 '(l/tj Where E is effort in person-months, t is project duration, B is special skills factor, P is productivity parameter.

Q6. What do you mean by cohesion and coupling? Discuss their types in brief.

Ans:

Cohesion is the indication of relative functional strength of a module. It is natural extension of Information Hiding and Performs a single task, requiring little integration with other components.

- Functional: Functional cohesion is when parts of a module are grouped because they all contribute to a single well-defined task of the module (e.g. tokenizing a string of XML).
- Logical: Logical cohesion is when parts of a module are grouped because they logically are categorized to do the same thing, even if they are different by nature (e.g. grouping all mouse and keyboard input handling routines).
- Communicational: Communicational cohesion is when parts of a module are grouped because they operate on the same data (e.g. a module which operates on the same record of information).
- Sequential: Sequential cohesion is when parts of a module are grouped because the output from one part is the input to another part like an assembly line (e.g. a function which reads data from a file and processes the data).
- Procedural: Procedural cohesion is when parts of a module are grouped because they always follow a certain sequence of execution (e.g. a function which checks file permissions and then opens the file).
- Temporal : When a module contains tasks that are related by the fact that all must be execute within the same span of time, then it is termed as temporal cohesion
- Utility.

Coupling is the quantitative measure of degree to which classes are connected to one another. Coupling should be kept as low as possible.

- Content Coupling Content occurs when one module modifies or relies on the internal workings of another module (e.g., accessing local data of another module).
- Common Coupling: When a number of modules reference a global data area, then the coupling is called common coupling
- Control Coupling: Control coupling is one module controlling the flow of another, by passing it information on what to do (e.g., passing a what-to-do flag).
- Stamp Coupling : When a portion of the data structure is passed via the module interface, then it is called as stamp coupling
- Data Coupling: Data coupling occurs when modules share data through, for example, parameters. Each datum is an elementary piece, and these are the only data shared (e.g., passing an integer to a function that computes a square root).

Name of Setter: - Designation:-

Address:- Signature of Setter

- External Coupling: External coupling occurs when two modules share an externally imposed data format, communication protocol, or device interface. This is basically related to the communication to external tools and devices.

Q7.

a. What is unit testing? Discuss its significance.

Ans:

In unit testing the individual components are tested independently to ensure their working as expected. The focus is to uncover the errors in design and implementation at unit level.The various tests that are conducted during the unit test are described as below.

- Module interfaces are tested for proper information flow in and out of the program.
- Local data are examined to ensure that integrity is maintained.
- Boundary conditions are tested to ensure that the module operates properly at boundaries established to limit or restrict processing.
- All the basis (independent) paths are tested for ensuring that all statements in the module have been executed only once.
- All error handling paths should be tested.
- Drivers and stub software need to be developed to test incomplete software. The "driver" is a program that accepts the test data and prints the relevant results and the "stub" is a subprogram that uses the module interfaces and performs the minimal data manipulation if required
-

b. What is user acceptance testing? Explain different testings in user acceptance testing. Why is it necessary?

Ans:

User Acceptance Testing is a phase of software development in which the software is tested in the "real world" by the intended audience /users.

Following are some of user acceptance testing:

- Alpha Testing

Alpha testing is the software prototype stage when the software is first able to run. It will not have all the intended functionality, but it will have core functions and will be able to accept inputs and generate outputs. An alpha test usually takes place in the developer's offices on a separate system.

- Beta Testing

The beta phase of software design exposes a new product, which has just emerged from in-house (alpha) testing, to a large number of real people, real hardware, and real usage. Beta testing is not a method of getting free software long-term, because the software expires shortly after the testing period.

User acceptance testing is used to know if the system is working or not (both clients & in-house)

Q8. What is ISO 9000 and ISO 9001? How the Registration process of ISO 9000 certification is done?

Ans:

ISO 9000 and 9001 are international set of standards for quality management. ISO 9000 is applicable to a range of organizations from manufacturing to service industries. ISO 9001 applicable to organizations which design, develop and maintain products. ISO 9001 is a generic model of the quality process that must be instantiated for each organization using the standard.

Below are steps involved in receiving ISO 9000 certifications:

1. Get top management commitment
    a. Top management considers ISO 9000 registration
    b. Quality steering committee meets to evaluate process
    c. Committee informs top management of ISO 9000 costs, schedule, etc.
    d. Top management commits to pursue ISO 9000 registration
2. Train personnel
    a. Hold basic quality and ISO 9000training for all employees
    b. Select and train personnel to be internal auditors
3. Prepare Quality Policy Manual
    a. Study and understand ISO 9000 requirements as they apply to your company
    b. Write (or re-write) company Vision and Mission statements
    c. Write basic Quality Policy Manual outline
    d. Complete first draft of Quality Policy Manual
    e. Send copy of manual to customer desiring ISO 9000 compliance (if necessary)
4. Prepare Operating Procedures
    a. Define responsibilities, using Quality Manual as a guide
    b. Have those responsible for functions outline their procedures
    c. Interview managers and fine-tune procedures
    d. Compare Operating Procedures with Quality Manual for consistency
5. Hold internal audit
    a. Hold internal audit of ISO 9000 manual vs. ISO 9000 compliance
    b. Implement corrective action items from audit
6. Select registrar
    a. Research registrars and their cost
    b. Qualify possible registrars
    c. Select third party registrar

Unsupported Personality: PCL

gistration process
y for registration and audits
e to audit process etc. with registrar
d pre-assessment audit
e any needed corrective action
e ISO 9000 registration audit
e any needed corrective action
udit as needed
e any needed corrective action
9000 registration
verifies that you operate your business in compliance to the ISO 9000
rements.

Between Known Risks and Predictable Risks.

t of risks that can be uncovered after careful evaluation of the project plan,
l environment in which the product is being developed. Example: Unrealistic

those risk which are extrapolated from past project experience. For example:

the advantages and disadvantages of size measure?

Advantages:

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

Disadvantages:

- This method is dependent upon the programming language.
- This method is well designed but shorter program may get suffered.
- commodate non procedural languages.
- f development it is difficult to estimate LOC.

c. **Discuss different phases of unified process.**

Ans:

Different phases of unified processes are:

- Inception Phase: Inception is the smallest phase in the project, and ideally it should be quite short. If the Inception Phase is long then it may be an indication of excessive up-front specification, which is contrary to the spirit of the Unified Process.
- Elaboration Phase: During the Elaboration phase the project team is expected to capture a healthy majority of the system requirements. However, the primary goals of Elaboration are to address known risk factors and to establish and validate the system architecture.
- Construction Phase: Construction is the largest phase in the project. In this phase the remainder of the system is built on the foundation laid in Elaboration. System features are implemented in a series of short, time-boxed iterations.
- Transition Phase: The final project phase is Transition. In this phase the system is deployed to the target users. Feedback received from an initial release (or initial releases) may result in further refinements to be incorporated over the course of several Transition phase iterations. The Transition phase also includes system conversions and user training.

Q10.

a. **Differentiate between Black and White Box Testing.**

Ans:

Black-box testing is a method of software testing that examines the functionality of an application (e.g. what the software does) without peering into its internal structures or workings (see white-box testing). This method of test can be applied to virtually every level of software testing: unit, integration, system and acceptance.

White-box is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality (i.e. black-box testing). In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs.

While white-box testing can be applied at the unit, integration and system levels of the software testing process, it is usually done at the unit level. It can test paths within a unit, paths between units during integration, and between subsystems during a system–level test. Though this method of test design can uncover many errors or problems, it might not detect unimplemented parts of the specification or missing requirements.

White-box test design techniques include:

- Control flow testing
- Data flow testing
- Branch testing
- Path testing
- Statement coverage
- Decision coverage

b. **What are different level of testing in Software Testing**

Ans:

i.) **Unit testing:** The individual components are tested in this type of testing.

ii.) **Module testing:** Related collection of independent components are tested.

iii.) **Sub-system testing:** This is a kind of integration testing. Various modules are integrated into a sub-system and the whole sub-system is tested.

iv.) **System testing**: The whole system is tested in this system.

v.) **Acceptance testing:** This type of testing involves testing of the system with customer data if the system behaves as per customer need then it is accepted.