First Semester MCA Degree Examination,Dec.2019/Jan.2020
Semester End Exam- Scheme and Solution

18MCA14-Software Engineering

Time: 3 hrs. Max.marks:100

Note: Answer any FIVE full questions, Choosing ONE full question from each module.

Module-1

1. a. What is software process model? Explain the essential attributes of good software (10Marks)

Solution:

A software process model is an abstract representation of a process. It presents a description of a process from some particular perspective 4 marks

Maintainability, Dependability, Efficiency, Acceptability, Usability, Reliability, Scalability, Portability, Reusability – 6marks

b. Discuss the various issues in software engineering professional responsibility and IEEE code of ethics (10Marks)

Solution:

*Issues in software engineering professional responsibility:*Confidentiality ,Competence, Intellectual property rights Computer misuse 5-marks

Software engineers shall adhere to the following Eight Principles: Public , Client And Employer ,Product Judgment ,Management Profession , Colleagues ,Self Learning 5-marks

OR

2. a. Discuss the process involved in waterfall and incremental development model with advantages and disadvantages. (10Marks)

Solution:

There are separate identified phases in the waterfall model: 3 marks

- Requirements analysis and definition
- System and software design
- Implementation and unit testing
- Integration and system testing
- Operation and maintenance

The incremental build model is a method of software development where the model is designed, implemented and tested incrementally until the product is finished. It involves both development and maintenance. The product is defined as finished when it satisfies all of its requirements. This model combines the elements of the waterfall model with the iterative philosophy of prototyping. 3 marks

Advantages: 2marks

Easy to understand, easy to use, especially by inexperienced staff

Generates working software quickly and early during the software life cycle.

Disadvantages: 2marks

All requirements must be known upfront – in most projects requirement change occurs after project start

Each phase of an iteration is rigid and do not overlap each other.

b. Explain in detail the principles of agile methods and discuss merits and users of extreme programming. (10Marks)

Solution:

The key principles, and how Agile Development fundamentally differs from a more traditional Waterfall approach to software development, are as follows: 5marks

- Active user involvement is imperative

- The team must be empowered to make decisions
- Requirements evolve but the timescale is fixed
- Capture requirements at a high level; lightweight & visual
- Develop small, incremental releases and iterate
- Focus on frequent delivery of products
- Complete each feature before moving on to the next
- Apply the 80/20 rule

Extreme programming expresses user requirements as stories, with each story written on a card. Discuss the advantages and disadvantages of this approach to requirements description. 5marks

The primary advantage of a user story is that it is short and concise and it expresses the requirement from the perspective of what the user business need is rather than attempting to tell a developer how to satisfy that need.

The disadvantage (if it is a disadvantage) is that it relies on direct communications to further elaborate the details of the requirement and some developers may not be very skilled in doing that.

Module-2

3. a. Describe requirements elicitation and analysis process with example. (10Marks)

Solution:

Requirements Elicitation & Analysis

It's a process of interacting with customers and end-users to find out about the domain requirements, what services the system should provide, and the other constrains.

Domain requirements reflect the environment in which the system operates so, when we talk about an application domain we mean environments such as train operation, medical records, e-commerce etc.

It may also involve a different kinds of stockholders; end-users, managers, system engineers, test engineers, maintenance engineers, etc.

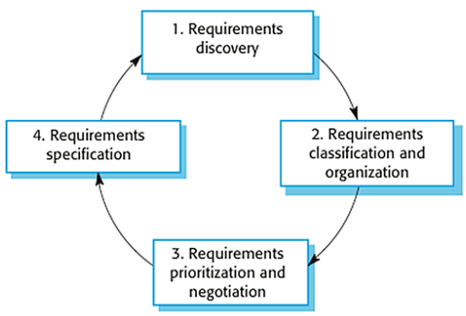
A stakeholder is anyone who has direct or indirect influence on the requirements.

The requirements elicitation and analysis has 4 main process

We typically start by gathering the requirements, this could be done through a general discussion or interviews with your stakeholders, also it may involve some graphical notation.

Then you organize the related requirements into sub components and prioritize them, and finally, you refine them by removing any ambiguous requirements that may raise from some conflicts.

Here are the 4 main process of requirements elicitation and analysis.



The process of requirements elicitation and analysis

It shows that it's an iterative process with a feedback from each activity to another. The process cycle starts with requirements discovery and ends with the requirements document. The cycle ends when the requirements document is complete.

b. Explain requirements validation techniques? Justify why should validate (10Marks)

Solution:

Requirements Validation

It's a process of ensuring the specified requirements meet the customer needs. It's concerned with finding problems with the requirements.

These problems can lead to extensive rework costs when these they are discovered in the later stages, or after the system is in service.

The cost of fixing a requirements problem by making a system change is usually much greater than repairing design or code errors. Because a change to the requirements usually means the design and implementation must also be changed, and re-tested.

During the requirements validation process, different types of checks should be carried out on the requirements. These checks include:

1. **Validity** *checks*: The functions proposed by stakeholders should be aligned with what the system needs to perform. You may find later that there are additional or different functions are required instead.
2. **Consistency** *checks*: Requirements in the document shouldn't conflict or different description of the same function
3. **Completeness** *checks*: The document should include all the requirements and constrains.
4. **Realism** *checks*: Ensure the requirements can actually be implemented using the knowledge of existing technology, the budget, schedule, etc.
5. **Verifiability**: Requirements should be written so that they can be tested. This means you should be able to write a set of tests that demonstrate that the system meets the specified requirements.

OR

4. a. Discuss the Component-Based Software Engineering (CBSE) process. (10Marks)
   b. Write short notes on:
   i. Ethnography
   ii. Use-case
   iii. Sequence diagram
   iv. Feasibility study
   v. Require change management

   Solution:

Ethnography: is the systematic study of people and cultures. It is designed to explore cultural phenomena where the researcher observes society from the point of view of the subject of the study. An ethnography is a means to represent graphically and in writing the culture of a group.

Use Case: In software and systems engineering, a use case is a list of actions or event steps typically defining the interactions between a role and a system to achieve a goal. The actor can be a human or other external system.

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

Feasibility Study is an assessment of the practicality of a proposed project or system.

Change management is the process, tools and techniques to manage the people side of change to achieve the required business outcome. Change management incorporates the organizational tools that can be utilized to help individuals make successful personal transitions resulting in the adoption and realization of change

Module-3

5. a. Define architectural design. Discuss the importance of behavioral model with example (10Marks)

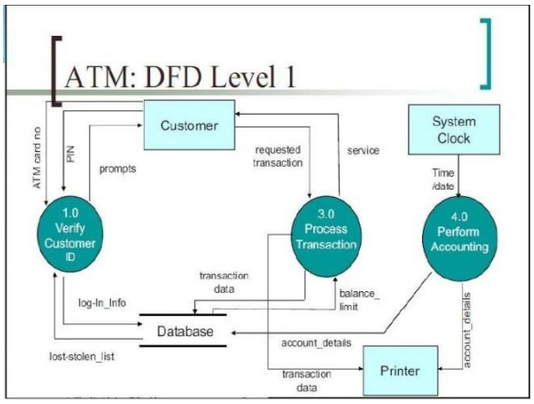
   Solution:

Architectural design encompasses both the data architecture and the program structure layers of the design model. This section defines the term "software architecture" as a framework made up of the system structures that comprise the software components, their properties, and the relationships among these components.
The following are the Taxonomy *(Classification)* of Architectural Style

- Data-centered architectures
- Data flow architectures
- Call and return architectures
- Object-oriented architectures
  Layered architectures

b. Draw a Data flow diagram (DFD) for Bank Transaction System and explain. (05Marks)

Solution:



c. Discuss the importance of the role of software architecture. (05Marks)

Solution:

Discuss the importance of the role of software architecture.

- ✧ Understanding and communication
- ✧ Reuse
- ✧ Construction and evolution
- ✧ Analysis
- ✧ Key technical consultant
- ✧ *Make decisions*
- ✧ Coach of development team
- ✧ Coordinate design
- ✧ Implement key parts

  Advocate software architecture

OR

6. a. Briefly explain different architecture styles for Components and Connectors view. (10Marks)

Solution:

- ✧ Two main elements – components and connectors
- ✧ Components: *Computational elements or data stores*
- ✧ Connectors: *Means of interaction between components*
- ✧ A C&C view defines the components, and which components are connected through which connector
- ✧ The C&C view describes a runtime structure of the system – what components exist at runtime and how they interact during execution
- ✧ Is a graph; often shown as a box-and-line drawing
- ✧ Most commonly used structure
- ✧ Units of computations or data stores
- ✧ Has a name, which represents its role, and provides it identity
- ✧ A comp may have a type; diff types rep by diff symbols in C&C view
- ✧ Comps use ports (or interfaces) to communicate with others
- ✧ An arch can use any symbols to rep components; some common ones are shown

Connectors:

- ✧ Interaction between components happen through connectors
- ✧ A connector may be provided by the runtime environment, e.g. procedure call
- ✧ But there may be complex mechanisms for interaction, e.g http, tcp/ip, ports,…; a lot of sw needed to support them
- ✧ Important to identify them explicitly; also needed for programming components properly
- ✧ Connectors need not be binary, e.g. a broadcast bus
- ✧ Connector has a name (and a type)
- ✧ Often connectors represented as protocol – i.e. comps need to follow some conventions when using the connector
- ✧ Best to use diff notation for diff types of connectors; all connectors should not be shown by simple lines

b. Discuss the methods of documenting architecture design with example. (10Marks)

Solution:

- ✧ While designing and brainstorming, diagrams are a good means
- ✧ Diagrams are not sufficient for documenting arch design
- ✧ An arch design document will need to precisely specify the views, and the relationship between them
- ✧ An architecture document should contain
- ✧ *System and architecture context*
- ✧ *Description of architecture views*
- ✧ *Across view documentation*
- ✧ A context diagram that establishes the sys scope, key actors, and data sources/sinks can provide the overall context
- ✧ A view description will generally have a pictorial representation, as discussed earlier
- ✧ Pictures should be supported by
- ✧ Element catalog: *Info about behavior, interfaces of the elements in the arch*
- ✧ Architectural rationale: *Reasons for making the choices that were made*
- ✧ Behavior: *Of the system in different scenarios (e.g. collaboration diagram)*
- ✧ Other Information: *Decisions which are to be taken, choices still to be made.*

Module-4

7. a. What is the relationship between function oriented design and detailed design? (05Marks)

Solution:

In the function-oriented designed approach, the system state is centralized and shared among different functions. On the other hand, in the object-oriented design approach, the system state is decentralized among the objects and each object manages its own state information.

b. Demonstrate how you can use complexity metrics for function oriented design that is easy to modify. (05Marks)

Solution:

Does the design implement the requirements Analysis for performance, efficiency, etc may also be done if formal languages used for design representation, tools can help Design reviews remain the most common approach for verification.

- ■ Basic purpose to provide a quantitative evaluation of the design (so the final product can be better)
- ■ Size is always a metric – after design it can be more accurately estimated
  - Number of modules and estimated size of each is one approach
- ■ Complexity is another metric of interest – will discuss a few metrics

Network Metrics:
Stability Metrics
Information Flow Metrics

c. Define coupling and cohesion. Explain in details types of coupling and justify which coupling is good. (10Marks)

Solution:

Coupling shows the relationships between modules. **Cohesion** shows the relationship within the module. **Coupling** shows the relative independence between the modules. **Cohesion** shows the module's relative functional strength.

Common **coupling** - modules share the same global data. External **coupling** - modules share an externally imposed data format, communication protocol or device interface. Control **coupling** - one module controls the flow of another, such as by passing it a flag or other information.

Good software will have ***low coupling*.** Where ***data coupling*** is most desirable. If the ***dependency between the modules*** is based on the fact that they communicate by passing only data, then the modules are said to be data coupled. In data coupling, the components are independent to each other and communicating through data. Module communications don't contain tramp data. *Example-customer billing system.*

OR

First Semester MCA Degree Examination,Dec.2019/Jan.2020
Semester End Exam- Scheme and Solution
18MCA14-Software Engineering

8. a. Explain in detail about different architectural patterns of distributed systems. (10Marks)

Solution:

- **Master-slave:** used in real-time systems for which guaranteed interaction response times are required.
  - Commonly used in real-time systems with separate processors associated with data acquisition, data processing, and computation and actuator management.
  - A **"Master"** process is usually responsible for computation, coordination, and communications; it controls the "slave" processes.
  - **"Slave"** processes are dedicated to specific actions, e.g., the acquisition of data from an array of sensors.

b. Describe in detail about software as a service in the distributed software engineering. (10Marks)

Solution:

Software as a service (SaaS) is a software distribution model in which a third-party provider hosts applications and makes them available to customers over the Internet. SaaS is one of three main categories of cloud computing.

SaaS is typically accessed by users using a thin client via a web browser. SaaS has become a common delivery model for many business applications, including office software, messaging software, payroll processing software, DBMS software, management software, CAD software , gamification, virtualization,[3] accounting, collaboration, customer relationship management (CRM), Management Information Systems (MIS), enterprise resource planning (ERP), invoicing, human resource management (HRM), talent acquisition, learning management systems, content management (CM), Geographic Information Systems (GIS), and service desk management. SaaS has been incorporated into the strategy of nearly all leading software companies.

Module-5

9. a. Describe in detail about Constructive Cost Model for effort estimation of planning software Project development. (10Marks)

The software development effort estimation is an essential activity before any software project initiation. We will illustrate how to easily estimate the software effort using known estimation techniques which are **Function Points Analysis (FPA)** and **Constructive Cost Model (COCOMO).**

Estimation Process

1- Scoping
2- Decomposition
3- Sizing
4- Expert and Peer Review
5- Estimation Finalization

b. Discuss in detail about project scheduling and staffing with example. (10Marks)

Project Scheduling in a project refers to roadmap of all activities to be done with specified order and within time slot allotted to each activity. Project managers tend to define various tasks, and project milestones and them arrange them keeping various factors in mind. They look for tasks lie in critical path in the schedule, which are necessary to complete in specific manner (because of task interdependency) and strictly within the time allocated. Arrangement of tasks which lies out of critical path are less likely to impact over all schedule of the project.

For scheduling a project, it is necessary to -

- Break down the project tasks into smaller, manageable form
- Find out various tasks and correlate them
- Estimate time frame required for each task
- Divide time into work-units
- Assign adequate number of work-units for each task
- Calculate total time required for the project from start to finish

OR

10. a. Explain the steps involved in the risk management process in detail. (10Marks)

Solution:

Risk is an expectation of loss, a potential problem that may or may not occur in the future. It is generally caused due to lack of information, control or time. A possibility of suffering from loss in software development process is called a software risk. Loss can be anything, increase in production cost, development of poor quality software, not being able to complete the project on time. Software risk exists because the future is uncertain and there are many known and unknown things that cannot be incorporated in the project plan. A software risk can be of two types (a) internal risks that are within the control of the project manager and (2) external risks that are beyond the control of project manager.
Risk management is carried out to:
Identify the risk
Reduce the impact of risk
Reduce the probability or likelihood of risk
Risk monitoring

b. Write in detail about white box testing techniques with suitable example. (10Marks)

Solution:

White box testing strategy deals with the internal logic and structure of the code. White box testing is also called as glass, structural, open box or clear box testing. The tests written based on the white box testing strategy incorporate coverage of the code written, branches, paths, statements and internal logic of the code etc.

In order to implement white box testing, the tester has to deal with the code and hence is needed to possess knowledge of coding and logic i.e. internal working of the code. White box test also needs the tester to look into the code and find out which unit/statement/chunk of the code is malfunctioning.