**Academic year: 2020-21**

**Sem-In Examinations-II,**

**B. Tech. (CSE), 2019 Batch**

**II/IV, 2nd Semester**

**19CS2211: Software Engineering**

**Scheme of Evaluation**

**Time: 2 hours Max. Marks: 50**

*(****Assume any missing data suitably and design adequate hypothesis, if required)***

|  |  |
| --- | --- |
| **CO3 Max.Marks 25** | |
| **Answer ALL Questions** | |
| 1. | Draw the Lifecycle of Scrum software development methodology 4.5 M  Scrum framework allows implementing Agile development methodology. Unlike the [waterfall model](https://xbsoftware.com/blog/software-development-life-cycle-waterfall-model/) of software development, Scrum enables an iterative and incremental development process. The project divides into several phases, each of which results in a ready-to-use product. At the end of each step (called sprint in Scrum terminology), a usable product is delivered to a customer. Customer feedback helps reveal possible problems or change the initial development plan if needed. If you want your project to strictly follow the main principles of the Agile manifesto, you can use the Scrum model and be sure that you’re on the right path. 2M  Here are the main roles involved in the development process, according to the Scrum model:   * The product owner takes care of the end user’s interests; * The Scrum master coordinates the whole development process. Another task is to make sure that Scrum is used properly and to hold regular Scrum meetings; * The Scrum team develops the product. Its main tasks are programming, analysis, testing, etc. 2.5M |
|  | (Or) |
| 2. | List out the steps that are needed for “creation” of a behavioral model 4.5 M  It regulates the visibility on documents and data according to the roles and profiles of the end users. It offers many advantages in a BI project, including: reducing the required number of analytical documents to be developed and maintained; coding visibility rules once only and apply them to several documents, each one with its own analytical logics; ensuring a uniform growth of the project over time; guaranteeing the respect of the visibility rules over time, with no limitation on the number of engines and analytical documents that can be added over time. 1M  The behavioural model is based on four main concepts:   * user profile, defining the user’s roles and attributes; * repository rights, defining the users’ rights in terms of document accessibility; * analytical drivers, defining which data of a document can be shown to the user; * presentation environment settings, defining how the user can reach and run his own documents. 2M   In other words, the behavioural model mainly answers the following questions:   * WHO uses the business intelligence solution (user profile); * WHAT is visible to users, in terms of documents and data (repository rights and analytical drivers); * HOW users work with their documents (analytical drivers and presentation environment settings).   Roles, users and attributes 1.5M |
| 3. | Identify five main classes in Requirements Modeling of the Output of the WebApp 8M   * An agile version of the generic software process can be applied when WebApps are engineered. * The process incorporates a communication activity that identifies   + Stakeholders and user categories,   + The business context,   + Defined informational and applicative goals,   + General WebApp requirements,   + Usage scenarios—   Each point carries one mark 7\*1==7M  and example carries one mark == 1M |
|  | (Or) |
| 4. | List out various values & practices in Scrum Meeting. 8M   1. Sprint Planning Meeting 2. Daily Scrum Meeting 3. Sprint Review Meeting 4. Sprint Retrospective Meeting 5. Backlog Refinement Meeting  Best Practices for Scrum Meetings  1. Define Clear Objectives   Scrum meetings have a tendency to quickly go off the rails if the objectives are not clear. For meetings to stay crisp, yet productive, it’s important to define a clear purpose and objective before the meeting is held.   Most of the times, team members dread [unplanned meetings](https://www.ntaskmanager.com/blog/why-unplanned-meetings-are-harmful/) if they’re not told beforehand what the meeting is all about. Therefore, before calling a meeting, identify the type of Scrum meeting it is and inform the team before the intended time. 2M    By following this practice, you ensure that your team members come prepared and time wastage is mitigated to the maximum.   2. Begin and End on Time   This particular Scrum meeting practice will also invoke a sense of punctuality in your team members the next time they join because it’ll be clear that meetings will always begin and end on time.    Another big no when it comes to Scrum meetings is to not try to fit in every team member’s individual feasibility of attending a meeting, because by doing so, you’ll quickly lose the overall decorum of meeting and you will have to shift the schedule multiple times- thus killing the productivity.   3. Daily Scrums for Monitoring Progress   The daily standups- for 15 minutes- should serve the intended purpose only, that is, to quickly address the 3 questions agenda and take a quick scan of the overall progress. The daily standups are not meant for rambling of team members- there should be other meetings for that. 2M    Try to strictly contain the discussion in 15 minutes timeframe and make it short. One way to do this is to make sure that the meeting is held while standing up- yes, get rid of the chairs, and ask the team members to stick to the three questions agenda only.    This Scrum meeting best practice shows that daily standups are a highly effective way of monitoring the overall progress of the development work in terms of Sprint goals.   4. Prepare Scrum Meeting Agenda 2M   Predefined Scrum meeting agenda helps in identifying only those elements which need to be discussed during a meeting, reducing the wandering off from the intended topic of discussion and saves significant time too.    A good practice of circulating the meeting agenda among team members before the meeting also makes the topic of discussion crystal clear and helps them to contribute effectively during the Scrum meeting. 5. Relevancy with the Sprint Goals Daily standups or daily Scrum meetings are a great way to take regular reports from the team members and help them in reviving the vision if they’re strayed off. Make this a habit during Scrum meetings to ask the team members about how they’re progressing towards the goals rather than how busy they are.   6. Continuous Improvement This is one of the most effective Scrum meeting best practices because it helps you in evaluating your actions and what you can do to improve them. 2M |
| (1 X 12.5M=12.5M) | |
| 5. | (a) Analyze the core practices of Scrum which help in the development of university management application and Explain in detail.  6.5M   Scrum simply provides a structure for delivery, but does not tell you how to do specific practices, leaving that to the team to determine. Exhibit 1 shows the basic Scrum framework.    Exhibit 1. The Original Scrum Framework  The project begins with a clear vision provided by the business, and a set of product features in order of importance. These features are part of the product backlog, which is maintained by the customer or customer representative referred to as the Product Owner. A time box commonly referred to as an iteration or sprint, is the set amount of time that the team has to complete the features selected. Sprints are generally from one to four weeks in length, and that length is maintained throughout the life of the project so as to establish a cadence. The team selects items from the product backlog that it believes can be completed in the sprint, and creates a sprint backlog consisting of the features and tasks as part of the sprint-planning meeting.  Once the team has committed to a sprint backlog, the task work begins. During this time in the sprint, the team is protected from interruptions and allowed to focus on meeting the sprint goal. No changes to the sprint backlog are allowed; however, the product backlog can be changed in preparation for the next sprint.  During the sprint, the team checks in daily with each other in the form of a 15-minute meeting known as a scrum. The team stands in a circle and each member states what they did yesterday, what they plan to do today, and what is getting in their way.  At the end of the sprint, the team demos the work they have completed to the stakeholders and gathers feedback that will affect what they work on in the next sprint. They also hold a retrospective to learn how to improve. This meeting is critical, as its focus is on the three pillars of Scrum: transparency, inspection, and adaptation.    (b) Explain how process framework activities helpful while developing the software. 6M  https://conceptdraw.com/a612c3/p1/preview/640/pict--uml-activity-diagram-uml-activity-diagram-cash-withdrawal-from-atm  Students can draw the diagram based on there perspective , but it should have proper meaning  One of the possible answer |
| (Or) | |
| 6. | (a) Summarize common mistakes and misunderstanding of Scrum 6.5M   * Expecting Transformation to Agile and Scrum to Be Easy * Doing the Practices Without the Principles * Complicating the Agile/Scrum Startup 2M * Leading a Scrum Team Like a Project Manager * An Un-Ready Product Backlog * Communicating “Through” the Scrum Master * . A Product Owner Who is Not Available Or Involved 2M * Not Conducting Retrospective Meetings After Every Sprint   For each Point with explanation **2.5M**    (b) For a context of Automotive application development, Draw the activity diagram for the Actuator-Sensor pattern. 6M    **Diagram-2M** |
| CO4 **Max.Marks 25** | |
| **Answer ALL Questions** | |
| 7 | Summarize various strategic approaches that are used in software testing  FACTORS TO CONSIDER IN CHOOSING SOFTWARE TESTING STRATEGIES  1. RISKS.  2. OBJECTIVES.  3. SKILLS.  4. PRODUCT. 2M  STRATEGIES IN SOFTWARE TESTING  1. Leave time for fixing.  2. Discourage passing the buck.  3. Manual testing has to be exploratory.  4. Encourage clarity.  5. Test often.  2.5M 4.5 M |
|  | (Or) |
| 8 | List out the elements of Design Modeling in the software process environment 4.5 M  1. Data design elements   * The data design element produced a model of data that represent a high level of abstraction. * This model is then more refined into more implementation specific representation which is processed by the computer based system. * The structure of data is the most important part of the software design.   2. Architectural design elements   * The architecture design elements provides us overall view of the system. * The architectural design element is generally represented as a set of interconnected subsystem that are derived from analysis packages in the requirement model.   The architecture model is derived from following sources:  The information about the application domain to built the software.  Requirement model elements like data flow diagram or analysis classes, relationship and collaboration between them.  The architectural style and pattern as per availability.  3. Interface design elements  The interface design elements for software represent the information flow within it and out of the system.  They communicate between the components defined as part of architecture.  Following are the important elements of the interface design: 1. The user interface 2. The external interface to the other systems, networks etc. 3. The internal interface between various components.  **Examples 0.5M** |
| 9 | Explain Capability Maturity Model Integration in detail.  8M  The Capability Maturity Model Integration (CMMI) is a capability maturity model developed by the Software Engineering Institute, part of Carnegie Mellon University in Pittsburgh, USA. The CMMI principal is that “the quality of a system or product is highly influenced by the process used to develop and maintain it”. CMMI can be used to guide process improvement across a project, a division, or an entire organization.  CMMI provides:   * Guidelines for processes improvement * An integrated approach to process improvement * Embedding process improvements into a state of business as usual * A phased approach to introducing improvements   **2M** |
|  | (Or) |
| 10 | Explain the Five different types of design classes and state their notations as per UML 8M Structural Things Graphical notations used in structural things are most widely used in UML. These are considered as the nouns of UML models. Following are the list of structural things.   * Classes * Object * Interface * Collaboration * Use case * Active classes * Components * Nodes  Class Notation UML class is represented by the following figure. The diagram is divided into four parts.   * The top section is used to name the class. * The second one is used to show the attributes of the class. * The third section is used to describe the operations performed by the class. * The fourth section is optional to show any additional components.   Class Notation  Classes are used to represent objects. Objects can be anything having properties and responsibility. Object Notation The object is represented in the same way as the class. The only difference is the name which is underlined as shown in the following figure.  Object Notation  As the object is an actual implementation of a class, which is known as the instance of a class. Hence, it has the same usage as the class. Interface Notation Interface is represented by a circle as shown in the following figure. It has a name which is generally written below the circle.  Interface Notation  Interface is used to describe the functionality without implementation. Interface is just like a template where you define different functions, not the implementation. When a class implements the interface, it also implements the functionality as per requirement. Collaboration Notation Collaboration is represented by a dotted eclipse as shown in the following figure. It has a name written inside the eclipse.  Collaboration Notation  Collaboration represents responsibilities. Generally, responsibilities are in a group.  **2M** |
| **(1 X 12.5M=12.5M)** | |
| 11. | (a) Examine the software design concepts that span both traditional and object-oriented software development. 6.5M  The following table explains how the object-oriented approach differs from the traditional structured approach −   |  |  | | --- | --- | | Structured Approach | Object Oriented Approach | | It works with Top-down approach. | It works with Bottom-up approach. | | Program is divided into number of submodules or functions. | Program is organized by having number of classes and objects. | | Function call is used. | Message passing is used. | | Software reuse is not possible. | Reusability is possible. | | Structured design programming usually left until end phases. | Object oriented design programming done concurrently with other phases. | | Structured Design is more suitable for offshoring. | It is suitable for in-house development. | | It shows clear transition from design to implementation. | Not so clear transition from design to implementation. | | It is suitable for real time system, embedded system and projects where objects are not the most useful level of abstraction. | It is suitable for most business applications, game development projects, which are expected to customize or extended. | | DFD & E-R diagram model the data. | Class diagram, sequence diagram, state chart diagram, and use cases all contribute. | | In this, projects can be managed easily due to clearly identifiable phases. | In this approach, projects can be difficult to manage due to uncertain transitions between phase |   (b) List out the Specific goals and generic goals of Capability Maturity Model Integration. 6M |
| (Or) | |
| 12. | (a) Compare and Contrast Acceptance Test Driven Development and Developer Test Driven Development 6M  **Test-driven development (TDD)** is a technique of using automated unit tests to drive the design of software and force decoupling of dependencies. The result of using this practice is a comprehensive suite of unit tests that can be run at any time to provide feedback that the software is still working.  The concept is to “get something working now and perfect it later.” After each test, refactoring is done and then the same or a similar test is performed again. The process is iterated as many times as necessary until each unit is functioning according to the desired specifications.  **ATDD** stands for **Acceptance Test Driven Development**, it is also less commonly designated as Storytest Driven Development (STDD). It is a technique used to bring customers into the test design process before coding has begun. It is a collaborative practice where users, testers, and developers define automated acceptance criteria. ATDD helps to ensure that all project members understand precisely what needs to be done and implemented. Failing tests provide quick feedback that the requirements are not being met. The tests are specified in business domain terms. Each feature must deliver real and measurable business value: indeed, if your feature doesn’t trace back to at least one business goal, then you should be wondering why you are implementing it in the first place.    (b) Examine Agile Model Driven Development (AMDD) and list out the Advantages of Test Driven Development. 6M |

Dr.K.V.D.Kiran

SE-Course Coordinator(19CS2211)