**NIIT UNIVERSITY DISCUSSION PORTAL**

**THEORY OF CORRESPONDANCE**

**PISE**

**DESCRIPTION OF EACH CYLE OF SOFTWARE DESIGN**

Members:

Utkarsh Srivastava, Shronit Bharagava, Yatham Praveen, Sourabh Kaushal, Siddhartha Kille

**Software development life cycle**

|  |  |
| --- | --- |
| **Key Concepts** | **Relevance** |
| * **Software development life cycle is also known as Application development lifecycle.** * **Software development life cycle includes the process of planning, creating, testing, and deploying an information system.** * **Different phases of SDLC are:**  1. **Requirement gathering and analysis** 2. **Design** 3. **implementation or coding** 4. **Testing** 5. **Deployment** 6. **Maintenance**  * **It describes the activities performed at each stage of a software development project.** * **Different approaches to life cycle methodology are: waterfall model, Prototyping Model, Spiral Model, Incremental Model RAD model.** * **This phase is important in getting to know how we will be proceeding with all the stages of the project, the software that we would be making.** | * **It is necessary for all developers to follow a lifecycle for their software. All professional software companies and developers are expected to follow this.** * **SDLC and its phases contribute to product high quality system that meets customer expectations, based on customer requirements, by delivering the system which move through each clearly defined phase, within scheduled time and estimated cost.** * **The software development lifecycle is not only a great way to ensure your app meets the needs of your business and customers, but it is also essential in supporting the app once it’s published.** |

|  |  |
| --- | --- |
| **Real World Contexts** | **Interdisciplinary Connections** |
| * **The systems development life-cycle concept is applicable to a range of hardware and software configurations, system can be composed of hardware only, software only, or a combination of both.** * **Following the assembly line manufacturing concept, an SDLC goals to produce high quality systems that meet customer expectations, based on customer requirements, by delivering systems which move through each clearly well-defined and planned phases, within fixed time schedules and estimated costs.** * Since long term goals are the most difficult to achieve, people seek to short term goals, which will slowly but surely help them to move in the right direction towards their goal. * SDLC helps us apply this idea to system development, giving the team working on the project a process. | * The SDLC process consists of six important phases that helps to manage a wide range of activity to conduct projects. * SDLC is not limited to technical activity but it actually begins with customer needs. * It evolves through different processes and user requirements to develop a solution to any problem. * The primary objective of implementing a standardized SDLC policy is to provide coordinated excellent service, at low costs, to support the activity of customers and users. |

|  |  |
| --- | --- |
| **Engage in critical thinking** | **Technology, Tools and Test Case** |
| **Critical thinking helps the in analyzing the requirements and mapping out the path of the project development, but might fail as members might not take it seriously.** | **Gmail**  **GitHub**  **Google Docs** |

|  |  |
| --- | --- |
| **Plan project management** | **Project specification** |
| **It depends on the type of software development phase we choose.** | * SDLC phases helped us in performing the actions before implementation of the code. We first identified and gathered our requirements and specifications and then analyzed them. * These phases behaved like a short term goals approaching us to our main goal making our customer satisfied. |

**Various Software Development Methodologies:**

|  |  |
| --- | --- |
| **Key Concepts** | **Relevance** |
| **It includes** Waterfall model, prototyping, interactive enhancement, spiral model. Role of Management in software development. Role of metrics and measurement. | **Waterfall Model:**  This model is one of the simplest classic life cycle models it is also known as the ‘linear-sequential’ life cycle model. In this model each phase must be completed before we move into another phase. At the end of each phase a review process is schedules to check project is on the right track.  **As the requirements changed, so waterfall is not the required model.**  **Prototype Model:​**  This model concentrates more on risk analysis. A preliminary design of the system is created.  **As no such prototype was build for the customer to evaluate, rather a clear guidance is taken from the customer and worked upon.**  **Spiral Model:​**  **As the constant interaction with the customer after every single progress is not performed, hence this model is not chosen.**  **Incremental Model:**  **​This is the most suitable as the software is made in increments .After completion of work on one part , the second system is handled and in this way the project progresses.**  **RAD Mode​l:** RAD stands for Rapid application development. This model is an incremental software development process model that concentrated more on short development cycle. |

|  |  |
| --- | --- |
| **Real World Contexts** | **Interdisciplinary Connections** |
| **In real world incremental model is the most suitable as the software is made in increments .After completion of work on one part , the second system is handled and in this way the project progresses.** | **The iterative procedure, the client involvement , the suggested requirements, the number of developers working and the time availability for the deployment are the connections for incremental model.** |

|  |  |
| --- | --- |
| **Engage in critical thinking** | **Technology, Tools and Test Case** |
| * **It involves heavy documentation.** * **Increments based on function and feature dependencies.** * **Requires more customer involvement. Partitioning the functions and features might be problematic·** * **Integration between iteration can be an issue.** | **No specific tools.** |

**Software Requirement Specification**

|  |  |
| --- | --- |
| **Key concepts** | **Relevance** |
| **It includes** Problem analysis, requirement specification, validation, metrics, monitoring and control. | **Functional Requirements :**  **These define how a product/service/solution. Describe the features and functions with which the end-user will interact directly.**  **Operational Requirements :**  **These define operations that must be carried out in the background to keep the product or process functioning over a period of time.**  **Technical Requirements :**  **These define the technical issues that must be considered to successfully implement the process or create the product.** |

|  |  |
| --- | --- |
| **Real World Contexts** | **Interdisciplinary Connections** |
| * **Each person considers the project from his or her individual perspective. We must understand these different perspectives and gather the different requirements to build a complete picture of what the project should achieve.** * **We should be clear about what the basic scope of the project is, and keep your discussions within this.** * **All sorts of functionality that your project was never designed to provide. If users have articulated these desires in detail, they may be disappointed when they are not included in the final specification.** | * **By connecting with end user, allows you to understand each person's specific views and needs.** * **Helps in resolving any conflicting requirements issues.** * **It will allow to explore how the proposed project would work in different possibilities and future use.** |

|  |  |
| --- | --- |
| **Engage in critical thinking** | **Technology, Tools and Test Case** |
| * **Critical thinking is one of the most useful modes of thought for analyzing.** * **Requirements analysis demands a combination of and solving complex problems encountered during requirements analysis.** * **This type of thinking matures with years of practice in real life situations in multiple disciplines** * **However since the scope of critical thinking transverses multiple disciplines and approaches to solving problems, no one definition has been agreed upon.** | * **Requirements analysis involves frequent communication with system users to determine specific feature expectations, resolution of conflict or ambiguity in requirements as demanded by the various users or groups of users, avoidance of feature creep and documentation of all aspects of the project development process from start to finish.** |

|  |  |
| --- | --- |
| **Plan project management** | **Project specification** |
| * **It includes planning**   + **Presentations**   + **Meetings**   + **Reviews**   + **Improvements from before feedbacks** | * **Separate functionality from implementation.** * **Develop a model of the desired behavior of a system. Establish the context in which software to be operated.** * **Define the environment in which the system operates.** |

**System Design**

|  |  |
| --- | --- |
| **Key concepts** | **Relevance** |
| * Abstraction, Modularity, Coupling, Cohesion, * Top-Down, Design. | * It is a design tool that gathers the information required to the user and rest of the information is ignored. * Applying it to the design makes the development in a proper manner. * Dependability among the modules is measured to greater extent. * Inter-dependability between the elements of the module is measured. * Easy to implement than bottom-up design. |

|  |  |
| --- | --- |
| **Real World Contexts** | **Interdisciplinary Connections** |
| * Useful for the developers to protect the unnecessary details. * Used effectively in testing and debugging. * Level of interaction among modules with each other is explained. * It is necessary as we know the requirements well before. | * Controls the difficulty of design process by proceeding from abstract design model to concrete model. * Maintenance of the design is simple without disturbing the functionality of the software. * When we need only one field of record then there is no need to pass all the records. * Proper knowledge of the working of each and every component is required. * Clarity to the developer is the primary objective for everyone to proceed. |

|  |  |
| --- | --- |
| **Engage in critical thinking** | **Technology, Tools and Test Case** |
| * No proper knowledge required on implementation. We can leave some desired components. * Increasing number of modules will create problem in integrating them. * Better coupling can be achieved by less coupling. * Higher the coupling , high the performance of the program. * Best results can be obtained for smaller programs. | * Argo UML, Star UML, Visual Studio, Photoshop |

|  |  |
| --- | --- |
| **Plan project management** | **Project specification** |
| * **The Project Manager is responsible for accountable for the successful execution**   **of the Design Phase.**  **The Project Manager is responsible for leading the team that accomplishes the phase activities and deliverables.**   * **The client may or may not take part in meetings.** | * **Define the system’s security architecture,**   **security policies, risk assessments.**   * **Describe planned activities to control access and protect the system and its information.** |

**Coding**

|  |  |
| --- | --- |
| **Key concepts** | **Relevance** |
| * Abstraction * top down programming * Structured programming * Information hiding | * Top level modules performs and allows other modules to execute particular tasks. * Performs repetitive tasks in a program. * Used to reduce the difficulty of different modules of software. |

|  |  |
| --- | --- |
| **Real World Contexts** | **Interdisciplinary Connections** |
| * Program is separated into several blocks in order to understand particular segment of code in a software code. * Entry points , multiple exit and statements are reduced in a program. * Non-essential details of the function are hidden as they are inaccessible to other components. | * Functions and procedures are globally visible. * Modification of the code is very easy. * After using Information hiding, modules are connected with a specific section of program. |

|  |  |
| --- | --- |
| **Engage in critical thinking** | **Technology, Tools and Test Case** |
| * Data structures are complex to execute as modules are dependent on each other. * Restricted to top-down approach of coding. * Information hiding doesn’t take part in creation of modules. Rely on other modules. | * Visual Studio, Github. * Self-Analysis. |

|  |  |
| --- | --- |
| **Plan project management** | **Project specification** |
| On deciding the language, we can plan according to it. | Depends on the language we choose. |

**Testing:**

|  |  |
| --- | --- |
| **Key points** | **Explore concepts significance and relevance** |
| * Software testing is done to uncover errors that were made during designing and construction. It includes test planning, test case design, test execution, data evaluation and collection. * It involves alpha testing , beta testing , class testing. * Project manager, software engineers, and testing specialists develops the strategy for software testing. * 4)Levels of testing: unit testing , integration testing ,system testing, acceptance testing. | * Testing tells about how to conduct the tests, how we should develop a plan for testing , and when to involve a customer. * The early testing focuses on a small group or single component of related components and testing is done to detect the errors in the data and processing logic . After components is done, they must remain integrated until the construction of the complete system is done. Here, a series of high-order tests are conducted to detect errors in assembling customer requirements. When errors are uncovered, they must be spotted and corrected by a process, called debugging * Testing often require more project effort than any other SE action. If it is conducted irregularly, unnecessary effort is expended, time is wasted, and even sometime errors remain undetected. Therefore, we should establish a systematic strategy for testing a software. * With the help of test specification document we can know the types of tests and specific testing steps that will be conducted. * Testing is a way of answering questions like How it works, how we can overcome any problem . when we take sports we require strategy for team work. Testing helps in making strategy and in foundation of tactics. |

|  |  |
| --- | --- |
| Find Real life contexts | Find Interdisciplinary connections |
| * unit , junit are some of the tools used for testing. * The tests can be non-functional or functional. Testing design techniques includes: Cause Effect Graphing,   Boundary Value Analysis, Equivalence partitioning. * some of the norms of software design are: there will be communicational, automotive and general. * **Testers mainly deal with 2 type of customers(companies).They are manufacturing and service based companies.** | * There is no one particular way to test a software. Not only we should understand various features of technology, but also we have to understand how that technology is applied as per different set of users and different perspectives. This requires technical skills, communications skills and reasoning skills. By this we can say that software testing is a interdisciplinary process. |

|  |  |
| --- | --- |
| **Engage in critical thinking** | **Technology, Tools, Techniques** |
| * There are mainly 2 different kind of thinking. System 1 thinking is helpful for making quick decisions about effortless matters while System 2 thinking is helpful where software testers want to focus their efforts on. This can be termed as critical thinking. * Critical thinking helps testers in remaining conscious about the software they're analyzing and when to start test runs and the nature of the defects they uncover. * It suggests, until and unless testers have a firms understanding of overall mission, they shouldn't conduct test. * Testers shouldn't take anything for granted. It is the job of the testers to notice the defect or performance issue. By this the underlying problem can be noticed, otherwise which would have gone unnoticed. | * HP Quick Test Professional, IBM Rational Functional Tester, Selenium, Test Complete , Load Runner are some of the tools used for testing. * Testing design techniques includes: Cause Effect Graphing,   Boundary Value Analysis, Equivalence partitioning. * There are mainly 3 type of testing: white box testing(in this tester is completely aware of internal working of the app), black box testing(in this tester is unaware of internal working of the app), grey box testing(in this tester has limited knowledge about internal working). * There are many type of technologies used for testing. The ones that create the most numerous and complex testing challenges are the most important technologies in the software quality .They are big data testing, mobile devices and cloud computing. * Testing [data warehouse systems](http://searchbusinessanalytics.techtarget.com/feature/Big-data-vendors-should-stop-dissing-data-warehouse-systems) and [big data](http://www.computerweekly.com/opinion/Leveraging-the-benefits-of-Big-Data) requires a whole new set of skills, so it has became challenging for software testers . The  [data warehouse testing](http://searchsoftwarequality.techtarget.com/tip/How-to-test-a-data-warehouse) is the transform ,[extract and load process](http://searchdatamanagement.techtarget.com/definition/extract-transform-load). The amount of data is huge, and it has been taken from many sources. Moreover, testers are challenged to work without a user interface. |

|  |  |
| --- | --- |
| **Plan project management** | **Project specification and project brief** |
| * Project plan management is one of the most important part. Inaccurate project planning can have a serious impact on the on the overall quality of the product as a whole and test execution of the system . * Firstly, we analyze the requirements and then **Scope of testing, designing the test strategy in accordance with the scope ,and the test effort and team defines the test schedule and then Identify** [test metrics](http://www.softwaretestinghelp.com/software-test-metrics-and-measurements/)**. These are the steps for effective plan and management of testing.** * **It includes distribution and allocation of tasks**,**Resource management, tools usage for reporting.** | * Test plan reflects the entire project testing approach and schedule. * Generally, a test plan includes  **tasks and  objectives, testing strategy (Unit Testing, System and Integration Testing, Performance and Stress Testing, User Acceptance Testing), test schedule, dependencies, risks/assumptions, tools and approvals. These are the specifications for testing.** |

Software project management:

|  |  |
| --- | --- |
| **Key points** | **Explore concepts, significance and relevance** |
| * Software project management starts with a set of activities that are known as project planning. * Cost estimation, Project scheduling, Staffing, Software configuration management, Quality assurance, Project Monitoring, Risk management, etc. * It helps in the estimation of determining how much money, resources, effort, and time it will take to make a software-based product or system. * Software project managers using information from software metrics data and project stake holders mainly do the software project management. | * cost estimation is an important aspect as any project cannot be made without it being financially. * It is necessary to come up with a feasible schedule. Hiring right and efficient people for the project is necessary for staffing. * Software project management helps in estimating the description of the scope of the problem. Then the problem is decomposed into a set of smaller problems, and by using experience as guides and historical data, each problem is estimated. * Problem risk and complexity are considered before a final estimate is made. * It also helps in estimating the tasks that we need to perform, how much we were about to spend, and the time line for the work to be managed. |

|  |  |
| --- | --- |
| **Find real life contexts** | **Find inter disciplinary connections** |
| * Software project management mainly focus on production aspects of software development, than the technical aspects, such as software tools. * It helps in understanding progress reports for stakeholders, projection and time tracking, percentage complete and quality assurance. * It also helps in communicating updates with local and distributed items, feedback, assignments and shared task lists. Identify and resolve projects obstacles , appraise financials and evaluate performance. | * Inter disciplinary features of software project management : * Provides a critical ,detailed review of software development processes and models. * Introduces key risk-reduction models and the basic software development process . * Explores recent swings in software process models and examines the subject of process improvement . * Explain how software tools influence problem-solving. * Describes how the focus of development has shifted to business contexts from technical. * Focuses on the economics and role of costs in software engineering. |

|  |  |
| --- | --- |
| **Engage in critical thinking** | **Technology , Tools and Techniques** |
| * The diversity of new approaches and techniques of contexts and clients in the software industry, requires software developers to have the ability to judge correctly and to discriminate successfully among these. traditional approaches in software engineering education, are not appropriate in equipping students with these diverse and unusual skills.   Critical thinking helps in: Understanding the Problem, Analysis and bringing together the Lists and Priorities. | Tools used:  Microsoft Visual Studio Team Foundation Server  Lean Kit  Plan box  Telerik Team Pulse.  Techniques:   * Agile Software Development * Crystal Methods * Dynamic Systems Development Model. |

|  |  |
| --- | --- |
| **Plan project management** | **Project specification and project brief** |
| * When compared to hardware production, software production is of low cost. To manage new development efforts, management required, but problems occurred during test runs, especially when confusion occurred between the delivered software and the user specifications . To avoid these problems, software project management methods focused on matching user requirements to delivered products. This can be done by software models like waterfall model, spiral model and agile model. * Analysis of software project management failures has shown that the following are the most common causes: involvement of insufficient end-user, Poor reporting of the project's status. | * Software project management is main important part of planning and leading software projects . * The main purpose of [project planning](https://en.wikipedia.org/wiki/Project_planning) is to recognize the scope of the project, [estimate](https://en.wikipedia.org/wiki/Estimation_in_software_engineering) the [work](https://en.wikipedia.org/wiki/Work_(project_management)) involved, and create a [project schedule](https://en.wikipedia.org/wiki/Gantt_chart). * Project planning starts with [requirements](https://en.wikipedia.org/wiki/Requirements_analysis) that defines the required software to be developed. Then the  [project plan](https://en.wikipedia.org/wiki/Project_plan" \o "Project plan) is developed to report the [tasks](https://en.wikipedia.org/wiki/Task_(project_management)) that will lead to completion. * Project controlling and monitoring involves status meetings to gather status from the team. When there is a need for changes, we should be up to date. |