Exam_Review

Software Development process model

- Waterfall - Spiral - Agile More info is found in [[Lecture_2]] & [[Lecture_3]]

Process Model - Waterfall

- Sequential model - 6 methods 1) Feasibility Planning 2) Requirements Definition 3) System and software design 4) implementation and unit testing 5) System testing and integration testing 6) Operation Maintenance - Not good for change

Process Model - Spiral

- 4 Stages 1) Plan Objectives 2) Risk Analysis and Resolution 3) Develop next version 4) Plan the next Phase - Risk management is crucial - Built off the previous version of the software system

Process Model - Agile

> Most Popular Model - Program specification and design are light and interleaved - The system is developed as a series of increments - Rapid development and delivery - Emerged in 1990s - 4 Values - Individuals and Interactions - Working Software - Customer Collaboration - Responding to Change

Extreme Programming

- Start with the simplest solution and evolve it as necessary - Pair programming - Collective ownership - Continuous integration - Sustainable pace - On-site customer

Practice	Description
Incremental planning	Plan in small parts and continue the cycle
Small releases	Release small portions at a time
Simple Design	Make each design a simplistic component that builds to a large complex one
Test-first development	Requirements being converted to test cases before software is fully developed
Refactoring	Make Changes Based on these

Scrum Programming

- Product Owner- Represents the stakeholder, voice of customer - Development Team- Self-organizing, and delivering the product. - Scrum Master- Facilitator who organizes meetings, tracks work and communicates with customers and management. - The starting point for planning is the Product Backlog - The selection phase involves the product team working with the customer to select the features and functionality developed during the sprint. - Meetings are short, describe progress since the last meeting and problems that have arisen, and plan to remedy this.

Version Control System

- Software tools to visualize changes over time - Track changes - Track errors - Etc - Centralized - Distributed

How Git Works - DCVS

- Working Directory - Local copy - Staging Index - Where the commit is prepared - Repository - Where the commit is stored - Atomic commits are small changes related to specific aspects of the projects - Example - Commit 1- Player Movement & Tile Map Wrong - Commit 1- Player Movement | Commit 2- Tile Map Right

What the Client Needs

- High-level requirements - give a general description of what the system should do - Low-level specifications - more specific and how they will operate - Functional requirements - specifies an action that the product must perform - Non-Functional requirements - Constraints, and things that are measurable

UML Diagram

- Actor - user is symbolized by a stick figure - The type of user needs to be identified - Could be an external system, database - Use case - represents functionality - One functional requirement - Linked to at least one actor, most of the time - (Not when triggered by another case) - Association - The participating actor is linked by this - Links elements together - Include - between two use cases - Extend - between two use cases

#incremental #development_models #cen4020 #software #engineering