Lecture_3

Agile Software Development

Program specification, design and implementation are light and interleaved. The system is developed as a series of versions or increments with clients involved in versions specification and evaluation.

Why is it popular?

- Businesses operate in a fast changing requirement and its practically impossible to produce a set of stable software requirements - Software has to evolve quickly to reflect changes that the business needs - Agile development emerged in the late 90's and the aim was to reduce delivery time for working software systems.

Agile Manifesto - 4 Values

- **Individual and interactions** over processes and tools - agile helps them collaborate and solve any problems that may arise - **Working Software** over comprehensive documentation - Agile values documentation, but values working software more - **Customer Collaboration** over contract negotiation - Agile enables the coordinated teams to align better with the customer requirements - **Responding to change** over following a plan - Agile adapt quickly in order to deliver a quality product and ensure their clients satisfaction

Extreme Programming (XP)

- A very influential agile method, developed in the late 1990's 1) Select user stories for this release 2) break down stories into tasks 3) plan

release 4) develop/ integrate / test 5) release software 6) evaluate system **Repeat the above proceess...**

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

A member of the end-user of the system should be available full time for advising the XP team

- Technical focus and is not easy to integrate with management practice in most organizations
- Consequently, when agile development uses practices from X, the method as originally defined is not widely used.
- User requirements are expressed as user stories or scenarios
- Written on cards and the development team break the down into implementation tasks. These tasks are the basis of schedule and cost estimates.

Refactoring

- Conventional wisdom in software engineering is to design for change. It is worth spending time and effort anticipating changes. - Programming team looks for possible improvements and make them where ever they are needed - Improves the understandability of the software and so reduces the need for documentation - Changes are

easier to make because the code is well-structured and clear - However, some changes requires architecture refactoring and this is much more expensive.

Examples of Refactoring

- Re-organizing a class hierarchy to remove duplicate code - Tidying up and renaming attributes and methods to make them easier to understand - The replacement of inline code with calls to methods that have been included in a program library

Test-First Development

Testing central to XP and XP has developed an approach where the program is tested after every change that has been made - Writing tests before code clarifies the requirements to be implemented - Tests are written as programs rather than data so that they can be executed automatically. The test includes a check that it has executed correctly. - All previous and new tests are run automatically when new functionality is added thus checking that the new functionality has not introduced errors.

Pair Programming

- Pair programming involves programmers working in pairs, developing code together. - This helps develop common ownership of code and spreads knowledge across the team - It serves as an informal review process - Swap frequently - Usually pair with someone of different experience level - Takes longer / more effort than single programming, however may yield higher quality code

Scrum

- Scrum is an agile framework that focus on managing iterative development - The scrum team consists of 3 role: 1) Product Owner 2) The Development Team 3) Scrum Master - **The Scrum Master** represents the products stakeholders and the voice of the customer. They create a prioritized, dynamic wish list (user stories) called a product backlog and is the sole person responsible for managing it - The starting point for planning is the **product backlog**. which is the list of work to be done for the project - The role of the Scrum Master is to protect the development team from external distractions - The team attends **Daily Meetings** where all team members share information, describe their progress since the last meeting, problems that have arisen and what is planned for the following day.