

Software Engineering

Prepared by Dr. Yasser Abdelhamid

IT 382
Software Engineering

Lecture 6





LECTURE OUTLINE

- ◉ **Testing in XP**
- ◉ **XP testing difficulties**
- ◉ **Pair programming**
- ◉ **Agile project management**
- ◉ **Scrum**

TESTING IN XP

XP TESTING FEATURES

- **Test-first development.**
- **Incremental test development from scenarios.**
- **User involvement in test development and validation.**
- **Automated test harnesses are used to run all component tests each time that a new release is built.**

TEST-FIRST DEVELOPMENT

- Writing tests before code clarifies the requirements to be implemented.

TEST-FIRST DEVELOPMENT

- 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.
 - Usually relies on a testing framework such as Junit.

TEST-FIRST DEVELOPMENT

- All previous and new tests are run automatically when new functionality is added.
- Checking that the new functionality has not introduced errors.

CUSTOMER INVOLVEMENT

- The role of the customer in the testing process is to **help develop acceptance tests** for the stories that are to be implemented in the next release of the system.

CUSTOMER INVOLVEMENT

- The customer who is part of the team writes tests as development proceeds.
- All new code is therefore validated to ensure that it is what the customer needs.

CUSTOMER INVOLVEMENT

- However, people adopting the customer role have limited time available and so cannot work full-time with the development team.
- They may feel that providing the requirements was enough of a contribution and so may be reluctant to get involved in the testing process.

TEST CASE DESCRIPTION FOR DOSE CHECKING

Test 4: Dose Checking

Input:

1. A number in mg representing a single dose of the drug.
2. A number representing the number of single doses per day.

Tests:

1. Test for inputs where the single dose is correct but the frequency is too high.
2. Test for inputs where the single dose is too high and too low.
3. Test for inputs where the single dose $\times$ frequency is too high and too low.
4. Test for inputs where single dose $\times$ frequency is in the permitted range.

Output:

OK or error message indicating that the dose is outside the safe range.

TEST AUTOMATION

- Test automation means that tests are written as executable components before the task is implemented
 - These testing components should be stand-alone,
 - should simulate the submission of input to be tested and
 - should check that the result meets the output specification.
- An automated test framework (e.g. Junit) is a system that makes it easy to write executable tests and submit a set of tests for execution.

TEST AUTOMATION

- As testing is automated, there is always a set of tests that can be quickly and easily executed
 - Whenever any functionality is added to the system, the tests can be run and problems that the new code has introduced can be caught immediately.

XP TESTING DIFFICULTIES

PROGRAMMERS PREFER PROGRAMMING TO TESTING

- Sometimes programmers take short cuts when writing tests.
- For example, they may write incomplete tests that do not check for all possible exceptions that may occur.

SOME TESTS CAN BE VERY DIFFICULT TO WRITE INCREMENTALLY

- For example, in a **complex user interface**, it is often difficult to write unit tests for the code that implements the 'display logic' and workflow between screens.

IT IS DIFFICULT TO JUDGE THE COMPLETENESS OF A SET OF TESTS

- Although you may have a lot of system tests, your test set **may not provide complete coverage.**

PAIR PROGRAMMING

PAIR PROGRAMMING

- In XP, programmers work in pairs, sitting together to develop code.
- This helps develop common ownership of code and spreads knowledge across the team.
- It serves as an informal review process as each line of code is looked at by more than 1 person.
- It encourages refactoring as the whole team can benefit from this.
- Measurements suggest that development productivity with pair programming is similar to that of two people working independently.

PAIR PROGRAMMING

- In pair programming, programmers sit together at the same workstation to develop the software.
- **Pairs are created dynamically** so that all team members work with each other during the development process.
- The **sharing of knowledge** that happens during pair programming is very important as it reduces the overall risks to a project **when team members leave**.
- Pair programming is not necessarily inefficient and there is evidence that a pair working together is more efficient than 2 programmers working separately.

ADVANTAGES OF PAIR PROGRAMMING

- It supports the idea of **collective ownership** and **responsibility** for the system.
 - Individuals are not held responsible for problems with the code. Instead, the team has collective responsibility for resolving these problems.

ADVANTAGES OF PAIR PROGRAMMING

- It acts as an **informal review process** because each line of code is looked at by at least two people.

ADVANTAGES OF PAIR PROGRAMMING

- It helps **support refactoring**, which is a process of software improvement.
 - Where pair programming and collective ownership are used, others benefit immediately from the refactoring so they are likely to support the process.

AGILE PROJECT MANAGEMENT

AGILE PROJECT MANAGEMENT

- The principal responsibility of software **project managers** is to manage the project so that the software is **delivered on time** and within **the planned budget** for the project.

AGILE PROJECT MANAGEMENT

- The standard approach to project management is **plan-driven**.
- Managers draw up a plan for the project showing **what should be delivered, when** it should be delivered and **who** will work on the development of the project deliverables.

AGILE PROJECT MANAGEMENT

- Agile project management requires a different approach, which is adapted to incremental development and the particular strengths of agile methods.

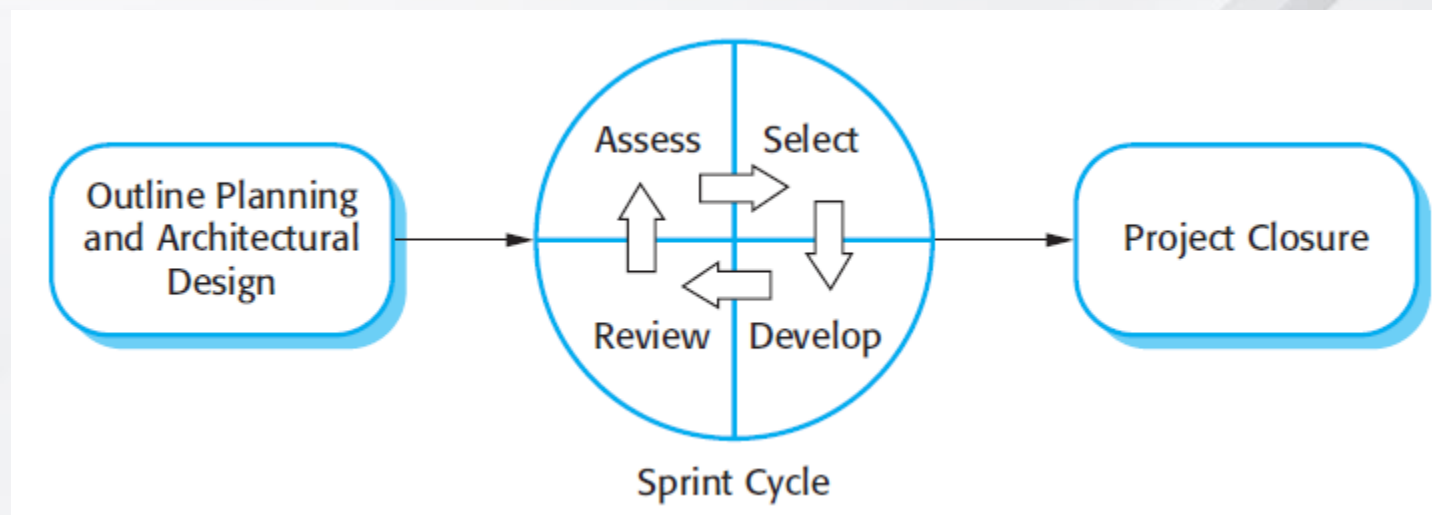
SCRUM

- The Scrum approach is a general agile method, but its **focus is on managing iterative development** rather than specific agile practices.

PHASES OF SCRUM

1. The initial phase is an **outline planning phase** where you establish the general objectives for the project and design the software architecture.
2. This is followed by a **series of sprint cycles**, where each cycle develops an increment of the system.
3. The **project closure phase wraps up the project**, completes required documentation such as system help frames and user manuals and assesses the lessons learned from the project.

THE SCRUM PROCESS



THE SPRINT CYCLE

- Sprints are fixed length, **normally 2–4 weeks**. They correspond to the development of a release of the system in XP.
- The starting point for planning is the **product backlog**, which is the **list of work to be done on the project**.
- The selection phase involves all of the project team who work with the customer to **select the features and functionality to be developed during the sprint**.

THE SPRINT CYCLE

- Once these are agreed, the team organize themselves to develop the software. During this stage, **the team is isolated from the customer and the organization**, with all communications channelled through the so-called '**Scrum master**'.
- The role of the Scrum master is to **protect** the development team from external distractions.
- At the end of the sprint, the work done is reviewed and presented to stakeholders. The next sprint cycle then begins.

SCRUM



SCRUM MASTER

- The 'Scrum master' is a facilitator who arranges daily meetings, tracks the backlog of work to be done, records decisions, measures progress against the backlog and communicates with customers and management outside of the team.

TEAMWORK IN SCRUM

- The whole team attends **short 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.
 - This means that everyone on the team knows what is going on and, if problems arise, can re-plan short-term work to cope with them.

SCRUM BENEFITS

- The product is broken down into a set of manageable and understandable **chunks**.
- **Unstable requirements** do not hold up progress.

SCRUM BENEFITS

- The whole team have **visibility of everything** and consequently team **communication is improved**.
- Customers see on-time delivery of increments and **gain feedback** on how the product works.

SCRUM BENEFITS

- **Trust between customers and developers is established** and a positive culture is created in which everyone expects the project to succeed.



SUMMARY

- A particular strength of extreme programming is the development of automated tests before a program feature is created. All tests must successfully execute when an increment is integrated into a system.
- The Scrum method is an agile method that provides a **project management framework**. It is centred round a set of sprints, which are fixed time periods when a system increment is developed.
- Scaling agile methods for large systems is difficult. Large systems need up-front design and some documentation.



Thank You

