Test Driven Development

Anil Sakala

Agenda

- ✓ Conventional way of developing software – Requirements , Code and Test
- ✓ Disadvantages with conventional approach
- ✓ Being test driven – TDD & ATDD
- ✓ Example using TDD
- ✓ Other topics

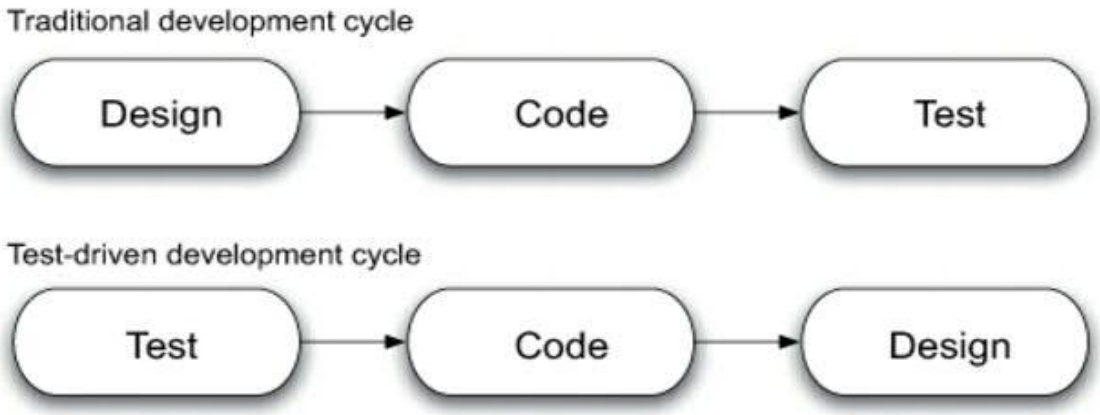
Conventional way of developing software - Disadvantages

- Conventional way – Requirements , Freeze Code and Test
- Disadvantages :
  - Quality is a concern :
    - Defects
    - Maintainenece (No safety net)
    - Solution : TDD
  - Changing requirements – Changing business rules
    - Better way of communicating between developers and testers
    - Solution : ATDD
- Above problems can be overcome very easily by being test driven

Being test driven

- Test Driven = TDD + ATDD
- TDD :
  - Lifecycle of TDD – Test , Code , Refactor
  - Only ever write a code to fix a failing test

Figure 1.3. TDD turns around the traditional design-code-test sequence. Instead, we test first, then write code, and design afterward.



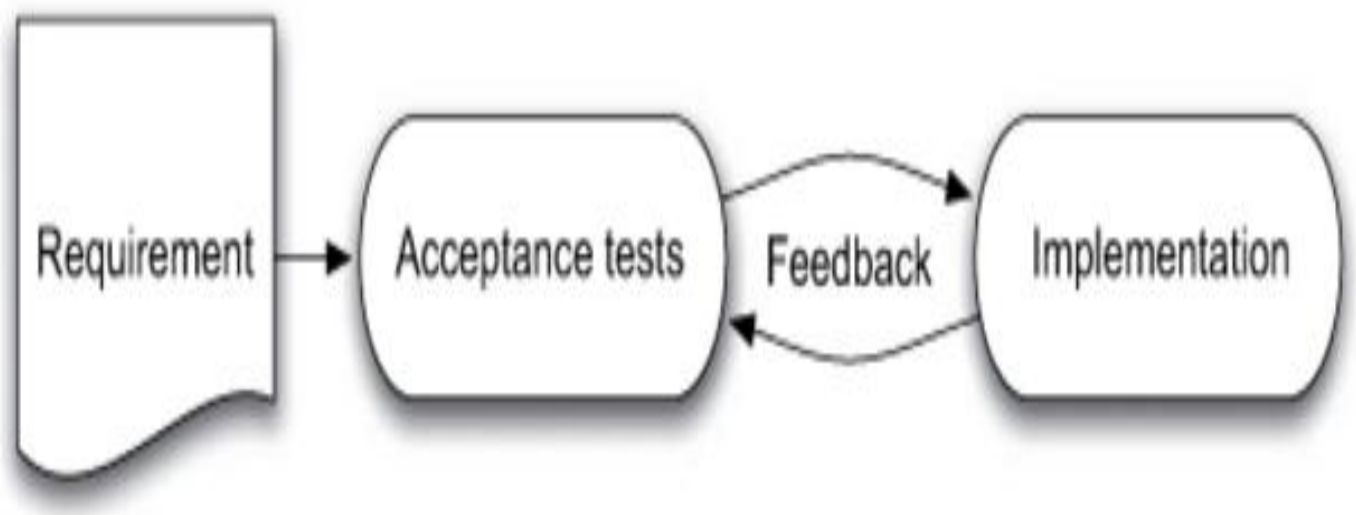
TDD

- Tools that help you writing this kind of code – Junit , TestNG , Mocking Frameworks
- Advantages :
  - Every object will have test object . This works like safety net
  - Thorough testing and immediate defects
  - Easy maintainenece
- Example : Check if a given String is palindrome or not

ATDD

- Conventional – Feature as requirement document , Code , Test , Customer acceptance test
- ATDD : Convert the requirement document to a set of acceptable tests and then build your system against these tests
- There are different tools available for this – Cucumber and Fit tool.
- How cucumber helps you in building
  - Living document
  - Reduces gap between developers and BA's
  - Trigger your test suite – Starts validating document against system
- Summary :
  - Write AT tests , Run Them , Build system to satisfy these tests

ATDD



Conclusion

- TDD and ATDD improve the quality of work
- **References**
  - **Test Driven: Practical TDD and Acceptance TDD for Java Developers**