FEABHAS

TDD for Embedded C++

Course category	Agile for Embedded
Training area	Agile for Embedded
Course code	TDDC++-301
Duration	3 days
Additional information	Available for on-site delivery only. Can be delivered remotely or Face-to-Face.

Agile for Embedded Training Course (updated Feb 21 see news**)**

One of the core Agile Practices is Test-Driven Development (TDD). For most software engineers TDD challenges the traditional approach to software development and testing (test-after-construction). TDD changes this model by using a Test-First approach.

However, almost all current books and training on TDD ignore the challenges of applying it in an embedded systems environment. Aspects such as testing on the target, hardware dependency and managing both host and cross-compiler toolchains are generally ignored or over-simplified.

The course covers testing embedded software across host, emulated and target environments.

Attendees perform practical TDD exercises based on real-world embedded requirements.

Course objectives:

- To understand the principles of Test-Driven Development
- How to select a suitable test framework
- To identify the required toolset to support TDD beyond the test framework
- The boundary between host-based testing and target-based testing
- The challenges and pitfalls of applying TDD to an embedded system

Delegates will learn:

- How to practice Test-Driven Development with version control (Git)
- The use of test harnesses for embedded systems (GoogleTest/doctest)
- API and Integration testing using test doubles (GoogleMock/FakeIt)
- The importance of architectural design to practical TDD
- What TDD doesn't address

Pre-requisites:

- Attendees should have a good working knowledge of the C++ language
- Understand the build process (preprocessor-compiler-linker-executable)

Who should attend:

The course is designed for software engineers working on an embedded C++ project.

The target audience is engineers working in, or looking to move to, an Agile project environment (e.g. Scrum) and who want practical experience of using TDD for a real embedded system.

Duration:

Three days

Course materials:

- Delegate manual
- Delegate workbook
- Delegate bootable Linux data key with all tools installed

Course workshop:

Approximately 50% of the course is given over to practical work.
The exercises use challenging real-world embedded problems with real-life code. Exercises cover code structuring and testing legacy code.

Agile Techniques and TDD

- Goals
- Where does TDD fit in?
- Agile onion
- Embedded TDD Strategy

TDD Foundations

- Unit tests
- Red-Green-Refactor (RGR)
- 3 rules of TDD
- Failing tests

- Mindset
- Mechanics

Test Construction

- Tests are FIRST
- File organisation
- Assertions
- Failure based tests
- Arrange-Act-Assert(AAA)/Given-When-Then

Design Principles

- Abstraction and encapsulation
- SOLID

Advanced Test Construction

- Fixture
- Weaknesses
- Classification Tree Method (CTM)
- Equivalence Class Testing (ECT)
- Boundary Value Analysis (BVA)
- Exceptions
- White Box Testing

Testing APIs

- UML Sequence Diagrams
- UML State Diagrams

Unit isolation

- Simple test doubles
- Dependency Challenges

Advanced Test Doubles

- Mocks
- Getting Mocks in place
- Design and change
- Strategies

- Issues

Legacy Code

- Object adaptor pattern
- Class adaptor pattern
- Legacy Code Change Policy

TDD and Threading

- GPOS threading vs. RTOS tasking
- Separate threading logic from application logic

Acceptance Tests and CI

- Containers and Docker
- Continuous Integration (CI) (Jenkins)
- Cloud-based CI builds (Github Actions, Bitbucket pipelines)

Incremental Design

- Walking skeleton
- Importance of architecture
- Version control and branching (git-workflow)

Target TDD

- Building for the target
- Testing using emulation (QEMU)
- Testing on the target