



TRAINING IN REAL-TIME
EMBEDDED DEVELOPMENT

SE-501 : Real-Time Software Engineering

Course Description:

This course is aimed at engineers new to developing software or new to developing real-time embedded software.

A range of skills are required to successfully develop real-time embedded software in a commercial environment, being able to program is simply the first step. The entire process involves design, documentation, reviews, quality control, configuration management and so on. This course answers the following key questions and more:

- Within these disciplines what are the most appropriate tools when the software being written is embedded?
- Which programming languages are suitable?
- Which design techniques lend themselves to difficult problems such as concurrency and distributed systems?
- How can we design for concurrency and distribution?
- Can real-time operating systems help and what are they?

The course has been written for Feabhas by the renowned real-time author Dr J. E. Cooling and is based upon his text book "Software Engineering for Real-Time Systems", 2003, Addison Wesley.

Overview:

A 5 day course introducing, at a basic level, the fundamental skills required to develop real-time embedded software in a commercial environment.

Course Objectives:

- Provide basic level information on all topics of software development e.g. programming, design, testing, documentation etc.
- Teach all these aspects in the context of real-time embedded software development.
- Give delegates the grounding required to start working in the development of real-time embedded software. Please note that delegates will need to attend further courses to become proficient in a particular programming language or design technique.

Delegates will learn:

- The characteristics of real-time systems
- Steps in developing software
- Programming issues – which languages?
- Why design and diagramming is so important
- Design basics - object oriented vs structured techniques
- Development tools
- Real-Time Operating Systems and what they do for us
- Documentation, coding and testing
- Safety and mission critical systems
- Performance engineering basics

Pre-requisites:

- A basic level of programming experience. (e.g. as a module on your degree course)

Who Should Attend:

This course is particularly suited to the following candidates who require a foundation in all aspects of embedded software development:

- Electronic engineers who are now moving into the field of software development.
- Graduates (including computer science graduates) who have not had experience of developing real-time embedded systems during their degree
- Engineers transferring to real-time embedded software development from other disciplines.

Duration:

Five days

Course Materials:

- "Software Engineering for Real-Time Systems", J.E. Cooling, Addison Wesley
- Delegate handbook

Related courses:

- C-501 C for Real-Time Developers
- C++-501 C++ for Embedded Developers
- OO-503 Real-Time Software Design with UML2.0
- RTOS-201 Fundamentals of Real-Time Operating Systems

Course Outline:

What is a real-time system?

- Characteristics of real-time systems
- Problems of real-time embedded development

Writing dependable software

- Why embedded software must be robust
- How errors are introduced

A process for software development

- Different software lifecycles
- The importance of requirements capture
- Fitting a process into your organisation

Design basics

- Design fundamentals
- Structured vs. OO techniques
- The importance of design reviews
- Design patterns – what and why?

Operating Systems for Real-Time

Applications

- Basic features of real-time operating systems.
- Scheduling
- Control of shared resources
- Task communication and synchronisation features.
- Memory management
- An introduction to Posix.

Design Notations

- Structured notation
- UML - the standard OO notation
- Extensions to notations for real-time
- Fitting diagrams into your design process

Programming Languages

- What languages are suitable for embedded development?
- A comparison of their strengths and weaknesses
- Code development and packaging
- Moving from design into code
- The importance of coding standards

Testing

- Unit, Module, Systems and Acceptance testing
- Static and Dynamic analysis of code
- Code walkthroughs
- White box and black box testing
- Code and design metrics

Development Tools

- Compilers & Debuggers
- Debugging on the host
- Debugging on the target
- Emulators & Probes
- Case tools
- Requirements tools
- Configuration management tools

Mission Critical and Safety Critical Systems

- System specification aspects.
- Application software aspects
- Real-world interfacing
- Operating systems aspects
- Numerical issues
- Processor problems
- Hardware-based fault-tolerance

Documentation

- What documents come from the design process
- User documentation
- Source code aspects
- Quality control
- A process for managing change
- Configuration management
- Library management

Software Re-use

- Can it be achieved?
- How can it be managed?
- Designing for re-use
- Testing re-used code

Continuous improvement

- Measuring the development process
- Quality standards
- ISO9001
- Tick IT
- The Capability Maturity Model CMM

FEABHAS

Feabhas Ltd

5, Lowesden Works
Lambourn Woodlands
Hungerford, Berkshire
RG17 7RY, UK

Tel: +44 (0) 1488 73050

Fax: +44 (0) 1488 73051

Email:

info@feabhas.com

Web:

www.feabhas.com