



TRAINING IN REAL-TIME  
EMBEDDED DEVELOPMENT

## Course OO-503 : Real-Time Software Design with UML 2.0

### Course description:

A detailed design course which focuses on designing a Real-Time Embedded System (RTES) using UML 2.0 notation to document the proposed design. This makes it significantly different from most UML courses which primarily focus on UML notation rather than design principles.

By using comprehensive real-world examples it also identifies the areas where UML 2.0 improves on UML 1.5, but also discusses its weaknesses (areas such as concurrency, multi-processing and distributed systems). The course is backed up by a comprehensive 'real-world' CASE study demonstrating how to apply UML 2.0 to a RTES.

This course has been developed by the leading author Dr. J. E. Cooling.

### Course objectives:

- To provide an understanding of the design principles of modern real-time software developments methods.
- To show how to develop real-time software in a rigorous and systematic manner.
- To enable attendees to develop their own practical design skills.
- To teach the UML 2.0 design notation for use on RTES.

### Delegates will learn:

- The fundamental concepts and terminology of real-time software techniques.
- The diagrammatic and modelling underpinnings provided by UML for OO development methods.
- How to apply the design principles in real-time applications.
- The basics of an integrated, traceable and consistent approach in the development of software for real-time systems.
- Where and how CASE tools can be used in the development process.

### Pre-requisites:

- Some understanding of technical software development methods and some knowledge of a high-level programming language.

### Who Should Attend:

- Designers new to the area of real-time software design.
- Developers embarking on projects using UML-based techniques for the first time.

### Duration:

Five days

### Course Materials:

- Delegate handbook
- All worked examples and solutions

### Related courses:

- OO-101 An Overview of UML for Real-Time Embedded Systems
- OO-301 Applying Real-Time UML
- SE-501 Real-Time Software Engineering
- SE-401 Systems Engineering using SysML

### Course Workshop:

Approximately 50% of the course involves practical application of the techniques discussed. Delegates work in small groups dealing with problems based on real-world systems.

The course specifically does not make use of a CASE tool. From our experience a CASE tool distracts delegates from learning design issues and UML. However, the workshops clearly demonstrate the benefits and disadvantages of CASE tools, thus aiding CASE tool selection.

### Course Outline:

#### Introduction to real-time systems design

- An overview of real-time systems
- Incremental design processes

#### Software machines - fundamentals

- Modularization principles and practices
- Diagramming techniques in program design - the UML activity diagram

#### Overview of OO design concepts

- Designing systems as sets of collaborating objects
- Classes, objects and their features
- Key design and build issues: software templates, encapsulation, interfacing, information hiding, inheritance and aggregation

#### Basic class and object modelling

- Design templates, encapsulation, interfacing and information hiding
- Class and object generalisation and specialisation: inheritance

#### Modelling dynamic behaviour

- State diagrams to define system and object dynamics
- The concepts, syntax and semantics of UML state machine diagrams

#### Developing the ideal object model

- Object relationships, communication and control in an ideal environment
- Scenarios, sequence diagrams and communication diagrams
- Responsibility-based design techniques
- Interfacing to the real-world

#### Developing the practical sequential (procedural) object model

- Controlling the collective behaviour of objects in sequential code: the coordinator object
- Specifying algorithmic (processing) operations using activity diagrams

#### Developing the specification model for small systems

- The role of the specification model
- Concurrent and non-concurrent structures - active and passive objects
- Specifying object structures - flat object structure

#### Concurrent systems and task-based design

- Problems with abstract software design
- Fundamentals of multitasking design

#### Developing the implementation (tasking) model for small systems

- Mapping the specification model to the tasking model.

#### Developing medium-sized systems

- Introduction to composite structures, ports and interfaces
- Generating the specification model for composite structures
- Specifying the implementation (tasking) model

#### Developing larger systems

- Key building blocks - packages and components
- Subsystem structuring of designs
- Implementation modelling

#### Multi-computer systems

- Multi-computer architectures for real-time systems
- Criteria for mapping software onto hardware
- A practical design technique

#### Use case analysis

- Use cases for organising and presenting requirements
- Scenarios and use cases
- System scope and direct and indirect actors.
- Modelling actor-system interactions graphically

#### Case study

- This is based on developing embedded software for a computer-controlled manufacturing test rig.

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

