



TRAINING IN REAL-TIME  
EMBEDDED DEVELOPMENT

FEABHAS

# Course DP-402: Design Patterns in C++ for Embedded Systems

## Course Description:

Everyone seems to be talking about *design patterns* these days. This course is designed to provide delegates with a basic understanding of design patterns how they can be applied to real-time C++ embedded systems.

It addresses the following questions:

- What are *design patterns* and why do I need to know about them, especially in embedded systems?
- Who are the “Gang of Four” and how is their work relevant?
- Which *patterns* are easy to work with and are most commonly found in embedded systems?
- Are there any specific patterns to embedded systems?
- What are anti-patterns?
- What is the difference between a pattern and an idiom?

## Overview:

This 4 day course will provide practical, hands-on experience with the core design patterns, uniquely addressing their suitability to a non-PC programming environment.

## Course Objectives:

- Provide a basic understanding of *Design Patterns* and how the language of *patterns* can aid designers and developers to be more productive.
- Provide practical experience of working with Design Patterns.
- Provide an understanding of the significant “Gang of Four” set of classical patterns and patterns associated specifically with multi-tasking embedded systems.
- Demonstrate how the patterns language can be used to document bad designs as, so called, “anti-patterns”.
- Gain the confidence to apply these new concepts to your next real-time project.
- Develop an understanding of Object Orientated principles through the lessons of others that are captured by *design patterns*.

## Pre-requisites:

- A good working knowledge of C++.
- An understanding of Object-Oriented principles.
- UML class modelling is useful, but not essential.

## Who Should Attend?

The course is aimed at software developers, designers, and architects wishing to make use of *design patterns* or software engineers in general who require a good briefing before deciding on their suitability to their projects.

Design Patterns aid considerably in collaborative working and the sharing of knowledge, at least initially by drawing on many external sources.

## Duration:

Four days.

## Course Materials:

Delegate handbook.

## Related Courses:

- Real-Time Software Design with UML 2.0 [OO-503]
- Advanced C++ for Embedded Systems [AC++-501]
- Embedded Software Testing [T-401]

## Course Workshop:

The course exercises are designed to foster an understanding of design patterns, object orientation, and C++. Ample opportunity is provided for delegates to consider the implications of patterns to the size and space concerns of embedded systems whilst reflecting on the broader quality issues that they directly address.

## Course Outline:

### Introduction

- History
- What is a design pattern?
- GoF design patterns
- Typical problems in embedded systems
- Design patterns in embedded systems
- Intention of this course

### Creational Patterns

- Singleton
- Abstract Factory
- Builder
- Prototype
- Factory

### Structural Patterns

- Bridge
- Smart Pointers
- Reference counting
- Adapter
- Decorator
- Façade
- Composite
- Proxy
- Flyweight

### Behavioural Patterns

- Chain of Responsibility
- Strategy
- Template method
- Interpreter
- Iterator
- Mediator

### Object Orientated Callbacks

- Observer
- C++ callbacks
- Command pattern
- Publish and Subscribe

### Anti- Patterns

- God class
- Lava flow
- Poltergeists
- Stove Pipe

### Real Time Patterns

- Threading
- Mutex
- Semaphores
- Thread Pools
- Barriers
- Futures
- Latches
- Exchangers
- Notifications
- Executors
- Transactions

### Real Time Anti-patterns

- Thread Affinity
- Race conditions
- Deadlock
- Livelock
- Priority inversion

## 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