

Design Patterns

Length: 24 hours, (+16 hours for the optional part related to concurrency)

Required facilities: VGA projector, white board, one workstation (per two programmers), development environment for an object oriented programming language (like C++, Java, C#, etc.).

Who can participate: programmers who want to take advantage in every day work of the experience and knowledge included in the design patterns.

Prerequisites: knowledge of an object oriented programming language, basics of object oriented programming principles, basics of Unified Modeling Language.

Course objectives: to show the place and the role of design patterns in software development, to study the key design patterns and learn to apply them.

The optional part is dedicated to design patterns which appear in concurrent context, specific to multi-threading programming.

The course might be seen as a good way to exercise the object oriented design principles; it offers a good opportunity to gain a valuable experience by covering diverse practical problems.

Related courses: Fundamentals of UML; Applying of OOP, UML and Design Patterns.

Attendees' evaluation: optional, during the training and/or a final test

Training written support: yes

Bibliography (minimal): Design Patterns – Elements of Reusable Object-Oriented Software, Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, Addison-Wesley, ISBN 0-201-63361-2

Description: this course discussed a set of design patterns which includes those originally defined by GoF (see the minimal bibliography) by examples and case studies. Besides understanding of the theory it is important how they are used: recognize the context of usage, variants, implementation particularities which depend of a programming language.

The course is highly interactive, it is an excellent opportunity to learn and exercise the principles of object oriented design.

The theoretical aspects are applied by solving practical problems, by particular implementations in Java, C# or C++.

The optional part – design patterns related to concurrency – shows specific constructions, problems and solutions to this zone.

Contents:

1. Introduction: definition, catalogs of design patterns, specific issues
2. **Principles of class design:** single responsibility, open/closed, Liskov substitution, dependency inversion, interface segregation
3. **Fundamental patterns:** delegation, interface, abstract superclass, immutable object, marker interface, proxy
4. **Creational patterns:** method factory, abstract factory, builder, prototype, singleton, object pool
5. **Partitioning patterns:** filter, composite object, read-only interface
6. **Structural patterns:** adapter, iterator, bridge, facade, flyweight, dynamic linkage, virtual proxy, decorator, cache management
7. **Behavioral patterns:** chain of responsibility, command, interpreter, mediator, snapshot, observer, state, strategy, null object, template method, visitor
8. **Concurrency patterns** (optional): critical zone, lock object, guarded suspension, balking, scheduler, read/write lock, producer/consumer, two step termination, double buffering, asynchronous processing, future, thread pool, double check locking, active object, monitor object, thread specific storage, leader/followers