



Software Quality Management

Introduction to Design Patterns

2004-2005

Marco Scotto (Marco.Scotto@unibz.it)



Outline

- What is a pattern?
- Why patterns?
- Software Patterns History
- Types Of Software Patterns
- GoF Design Patterns
- GoF Pattern Template
- Classification Of Design Patterns
- Benefits Of Design Patterns



What Is A Pattern? (1/2)

- Current use comes from the work of the architect Christopher Alexander
- Alexander studied ways to improve the process of designing buildings and urban areas
- “Each pattern is a three-part rule, which expresses a relation between a certain context, a problem and a solution.”



What Is A Pattern? (2/2)

- The common definition of a pattern is:
 - “A solution to a problem in a context.”
- Patterns can be applied to many different areas of human endeavor, including software development



Why Patterns? (1/2)

- "Designing object-oriented software is hard and designing reusable object-oriented software is even harder."
 - Erich Gamma
- Experienced designers reuse solutions that have worked in the past



Why Patterns? (2/2)

- Well-structured object-oriented systems have recurring patterns of classes and objects
- Knowledge of the patterns that have worked in the past allows a designer to be more productive and the resulting designs to be more flexible and reusable



Software Patterns History (1/2)

- 1987 - Cunningham and Beck used Alexander's ideas to develop a small pattern language for Smalltalk
- 1990 - The Gang of Four (Gamma, Helm, Johnson and Vlissides) begin work compiling a catalog of design patterns
- 1991 - Bruce Anderson gives first Patterns Workshop at OOPSLA



Software Patterns History (2/2)

- 1993 - Kent Beck and Grady Booch sponsor the first meeting of what is now known as the Hillside Group
- 1994 - First Pattern Languages of Programs (PLoP) conference
- 1995 - The Gang of Four (GoF) publish the *Design Patterns* book



Types Of Software Patterns

- Analysis
- Design
- Organizational
- Process
- Project Planning
- Configuration Management



Types Of Software Patterns

- Riehle and Zullighoven in "*Understanding and Using Patterns in Software Development*" mention three types of software patterns:
 - Conceptual Pattern
 - Pattern whose form is described by means of terms and concepts from the application domain
 - Design Pattern
 - Pattern whose form is described by means of software design constructs, such as objects, classes, inheritance and aggregation
 - Programming Pattern (Programming Idiom)
 - Pattern whose form is described by means of programming language constructs



Design Pattern Levels Of Abstraction

- Complex design for an entire application or subsystem
- Solution to a general design problem in a particular context
- Simple reusable design class such as a linked list, hash table, etc.



GoF Design Patterns

- The GoF design patterns are in the middle of these levels of abstraction
- “A design pattern names, abstracts, and identifies key aspects of a common design structure that makes it useful for creating a reusable object-oriented design”
- The GoF design patterns are “descriptions of communicating objects and classes that are customized to solve a general design problem in a particular context”



Classification Of Design Patterns (1/2)

- Purpose - what a pattern does
 - Creational Patterns
 - Concern the process of object creation
 - Structural Patterns
 - Deal with the composition of classes and objects
 - Behavioral Patterns
 - Deal with the interaction of classes and objects




Classification Of Design Patterns (2/2)

- Scope - what the pattern applies to
 - Class Patterns
 - Focus on the relationships between classes and their subclasses
 - Involve inheritance reuse
 - Object Patterns
 - Focus on the relationships between objects
 - Involve composition reuse

Design pattern space

		Purpose		
		Creational	Structural	Behavioral
Scope	Class	Factory Method	Adapter	Interpreter Template Method
	Object	Abstract Factory Builder Prototype Singleton	Adapter Bridge Composite Decorator Façade Proxy	Chain of Responsibility Command Iterator Mediator Memento Flyweight Observer State Strategy Visitor




GoF Essential Elements Of Design Patterns (1/2)

➤ Pattern Name

- Having a concise, meaningful name for a pattern improves communication among developers

➤ Problem

- What is the problem and context where we would use this pattern?
- What are the conditions that must be met before this pattern should be used?



GoF Essential Elements Of Design Patterns (2/2)

➤ Solution

- A description of the elements that make up the design pattern
- Emphasizes their relationships, responsibilities and collaborations
- Not a concrete design or implementation; rather an abstract description

➤ Consequences

- The pros and cons of using the pattern
- Includes impacts on reusability, portability, extensibility



GoF Pattern Template (1/3)

- Pattern Name and Classification
 - A good , concise name for the pattern and the pattern's type
- Intent
 - Short statement about what the pattern does
- Also Known As
 - Other names for the pattern
- Motivation
 - A scenario that illustrates where the pattern would be useful
- Applicability
 - Situations where the pattern can be used



GoF Pattern Template (2/3)

- Structure
 - A graphical representation of the pattern
- Participants
 - The classes and objects participating in the pattern
- Collaborations
 - How do the participants interact to carry out their responsibilities?
- Consequences
 - What are the pros and cons of using the pattern?
- Implementation
 - Hints and techniques for implementing the pattern



GoF Pattern Template (3/3)

- Sample Code
 - Code fragments for a sample implementation
- Known Uses
 - Examples of the pattern in real systems
- Related Patterns
 - Other patterns that are closely related to the pattern



Benefits Of Design Patterns

- Capture expertise and make it accessible to non-experts in a standard form
- Facilitate communication among developers by providing a common language
- Make it easier to reuse successful designs and avoid alternatives that diminish reusability
- Facilitate design modifications
- Improve design documentation
- Improve design understandability



Design Patterns Books (1/3)

- *Design Patterns: Elements of Reusable Object-Oriented Software*, Gamma, Helm, Johnson and Vlissides, Addison-Wesley, 1995
- *Design Patterns for Object-Oriented Software Development*, Wolfgang Pree, Addison Wesley/ACM Press, 1995
- *Patterns of Software: Tales From The Software Community*, Richard P. Gabriel, Oxford University Press, 1996
- *Pattern Oriented Software Architecture : A System of Patterns*, Frank Buschmann (Editor), Wiley, 1996
- *Analysis Patterns: Reusable Object Models*, Martin Fowler, Addison-Wesley, 1997
- *AntiPatterns*, Brown, Malveau, McCormick and Mowbray, Wiley, 1998



Design Patterns Books (2/3)

- *Pattern Hatching: Design Patterns Applied*, John Vlissides, Addison-Wesley, 1998
- *Patterns in Java Volume 1*, Mark Grand, Wiley, 2nd Ed., 2002
- *Patterns in Java Volume 2*, Mark Grand, Wiley, 1999
- *Java Enterprise Design Patterns: Patterns in Java Volume 3*, Mark Grand, Wiley, 2001
- *The Patterns Handbook*, edited by Linda Rising, Cambridge University Press, 1998
- *Java Design Patterns - A Tutorial*, James W. Cooper, Addison-Wesley, 2000



Design Patterns Books (3/3)

- *Design Patterns Explained*, Alan Shalloway and James R. Trott, Addison-Wesley, 2001
- *Design Patterns Java Workbook*, Steven John Metsker, Addison-Wesley, 2002
- *Applied Java Patterns*, Stephen Stelting and Olav Maassen, Prentice Hall, 2002
- *Core J2EE Patterns: Best Practices and Design Strategies*, Alur, Crupi and Malks, 2001
- *EJB Design Patterns: Advanced Patterns, Processes, and Idioms*, Floyd Marinescu, John Wiley, 2002
- *Head First Design Patterns*, Freeman and Freeman, O'Reilly, 2004