Software Architecture & Design SEC3071

Lecture No. 31

Muhammad Shahid
Department of Computer Science
National Textile University

shahid.abdullah@hotmail.com

Last Lecture Review

- SOLID Principles
- Dependency-Inversion Principle
- Code Examples

SOLID Principles – Review

- Single Responsibility Principle (SRP)
- Open/Closed Principle (OCP)
- Liskov Substitution Principle (LSP)
- Interface Segregation Principle (LSP)
- Dependency-Inversion Principle (DIP)

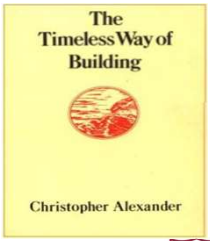
Agenda – What will you Learn Today?

Introduction to Software Design Patterns

Introduction to Design Patterns

A Brief History of Design Patterns

- 1979 - Christopher Alexander publishes: "The Timeless Way of Buildings"
  - Introduces the notion of pattern and a pattern language
  - It is a architecture book and not a software book
  - Alexander sought to define step-by-step rules for solving common engineering problems relevant to the creation of buildings and communities



A Brief History of Design Patterns

- 1936 Austria, B.Sc. architecture, M.Sc. in mathematics
- First Ph.D. in architecture ever awarded at Harvard University
- 1st book by author to influence CS: "Notes on the Synthesis of Form"
- Required reading for researchers in computer science in the '60s

A Brief History of Design Patterns

- Recommended by Marvin Minsky, founder of MIT's AI Lab, to his students.
- Heavy influence in the 60s & 70s:
  - Programming language design
  - Modular programming
  - Object-oriented programming
  - Software engineering
  - Other design methodologies
- ... and all of their books were about civil architecture

A Brief History of Design Patterns

- Kent Beck & Ward Cunningham at the OOPSLA-87 workshop on the Specification and Design for Object-Oriented Programming publish the paper: **"*Using Pattern Languages for Object-Oriented Programs*"**
- Discovered Alexander's work for software engineers by applying 5 patterns in Smalltalk

A Brief History of Design Patterns

- 1991 - **Erich Gamma** came up with an idea for a Ph.D. thesis about patterns, and by 1992, he had started collaborating with the other **GOF** members (Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides) on expanding this idea

A Brief History of Design Patterns

- 1993 - GOF submitted a catalog of Object Oriented Software Design Patterns to the European Conference of Object-Oriented Programming (ECOOP) in 1993 E. Gamma, R. Helm, R. Johnson, J. Vlissides. "***Design Patterns: Abstraction and Reuse of Object-Oriented Design***". ECOOP 97 LNCS 707, Springer, 1993

A Brief History of Design Patterns

- 1995 - GOF publishes : "***Design Patterns: Elements of Reusable Object-Oriented Software"***

- The most popular computer book ever published

- 1 million copies sold

What are Patterns?

- A pattern involves a general description of a recurring solution to a recurring problem with various goals and constraints. It identify more than a solution, it also explains why the solution is needed.

[Jim Coplien]

What are Patterns?

- ..describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice

[Christopher Alexander]

What are Design Patterns?

- "**Design patterns** are recurring solutions to software design problems you find again and again in real-world application development"

- Design Patterns not only allow you to reuse a code solution, but help provide extensibility, maintainability, and a way for fellow programmers to understand the solution

What are Design Patterns?

- "Some body has already solve your problem"

- A design pattern is a documented best practice or core of a solution that has been applied successfully in multiple environments to solve a problem that recurs in a specific set of situations.

- It is "a recurring solution to a common problem in a given context and system of forces."

DP – Language Independent

- A design pattern is an effective means to convey/communicate what has been learned about high-quality designs. The result is:
  - A shared language for communicating the experience gained in dealing with these recurring problems and their solutions
  - A common vocabulary of system design elements for problem solving discussions. A means of reusing and building upon the acquired insight resulting in an improvement in the software quality in terms of its maintainability and reusability

Essence of Design Patterns

- Knowing concepts like **abstraction**, **inheritance**, and **polymorphism** do not make you a good object oriented designer
- A **design GURU** thinks about how to create **flexible designs** that are **maintainable** and that can **cope with change**

Essence of Design Patterns

- The best way to use them is to load your mind with them and then identify places where you can apply them

- Instead of code reuse, patterns let you **experience reuse**

Formal Definition of Design Pattern

- The **design pattern** identifies classes and instances, their roles, collaborations and responsibilities. Each design pattern focuses on a particular object-oriented design problem or issue. It describes when it applies, whether it can be applied in the presence of other design constraints, and the consequences and trade-offs of its use."

Design Patterns Space

- **Creational Patterns**
  - Deal with initializing and configuring of classes and objects
- **Structural Patterns**
  - Deal with decoupling interface and implementation of classes and objects
- **Behavioral Patterns**
  - Deal with dynamic interactions among societies of classes and objects

Design Patterns Space

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

Design Patterns Space – Scope

- **Class**: Defined through inheritance between classes
- **Object**: Dynamically, defined through associations between objects

How Patterns are Described

- The GoF book describes a pattern using the following four attributes:

1. **Name**: The name to describes the pattern, its solutions and consequences in a word or two
2. **Problem**: Describes when to apply the pattern
3. **Solution**: Describes the elements that make up the design, their relationships, responsibilities, and collaborations
4. **Consequences**: The results and trade-offs in applying the pattern

Recap

- Introduction to Design Patterns
- Brief History of Design Patterns
- What are Patterns?
- What are Design Patterns?
- Essence of Design Patterns
- Design Patterns Space
- How Patterns are Described

Questions