Tecniche di Progettazione: Design Patterns

Laura Semini, Università di Pisa, Dipartimento di Informatica.


ATTENTION
BAD DESIGN
“CALATRAVA BRIDGE”

Pattern: the step

The rise is typically between 13 and 20 cm

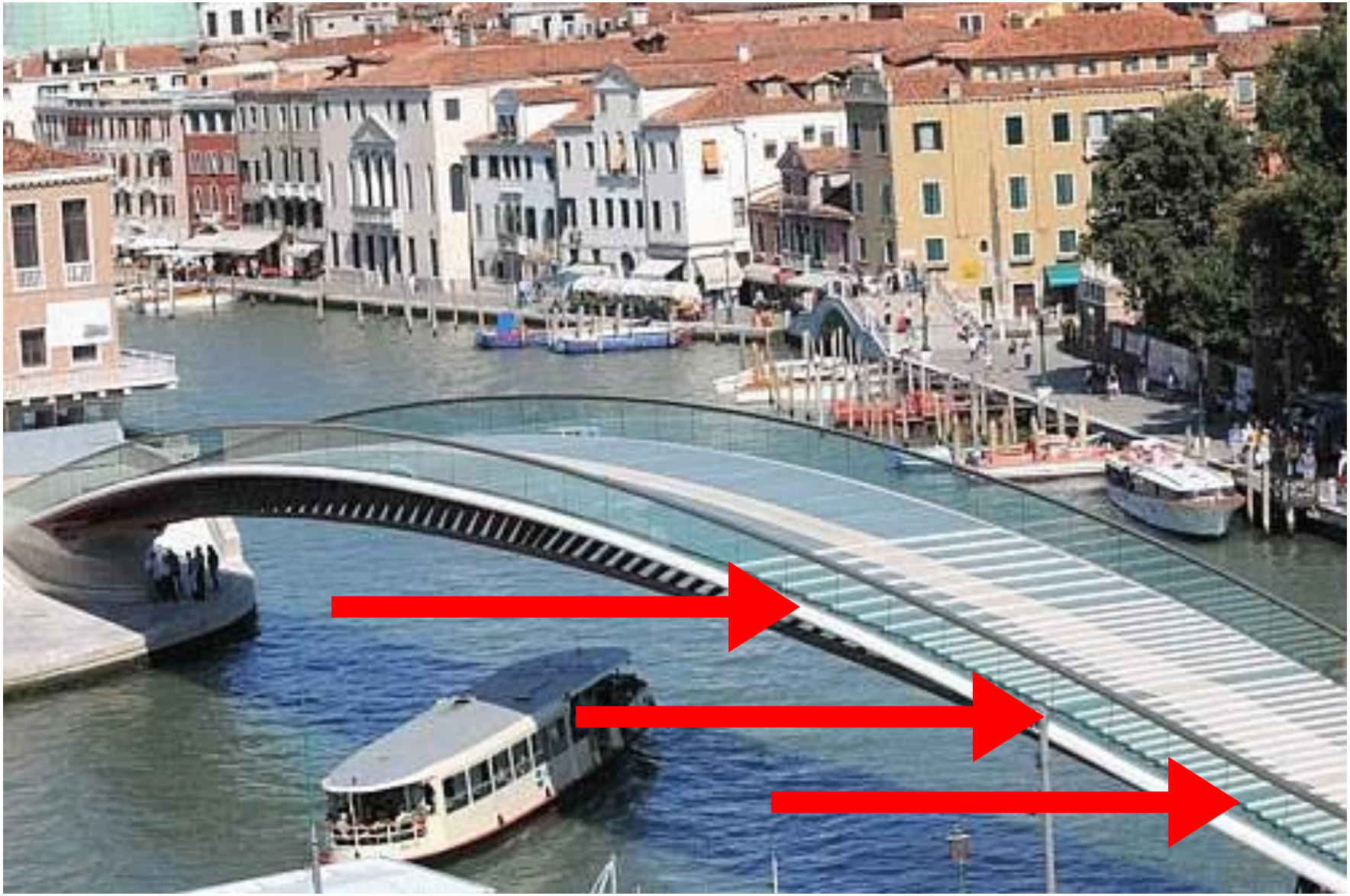
The run is calculated using:

Blondel formula: $2\text{Raise} + \text{Run} = 62 \div 64$ cm

By varying the slope, the user tends to change the length of the step so that the work done to overcome a step is equal to the work done to accomplish the same step on a plane.

Blondel formula: 2Raise + Run= 62 ÷ 64 cm

Run= 50, Raise = 8 → 50+2x8=66



Calatrava: no space to rest

Materials

The use of Istrian stone alternated with dark trachyte to mark the step is a good solution already identified by the Venetians in the fifteenth century.

Where is the step?

So what?

There are a set of *practical rules* the designer can follow to build a staircase:

- Rise/run ratio (Blondel)
- Materials…

These *practical rules* are the design patterns.

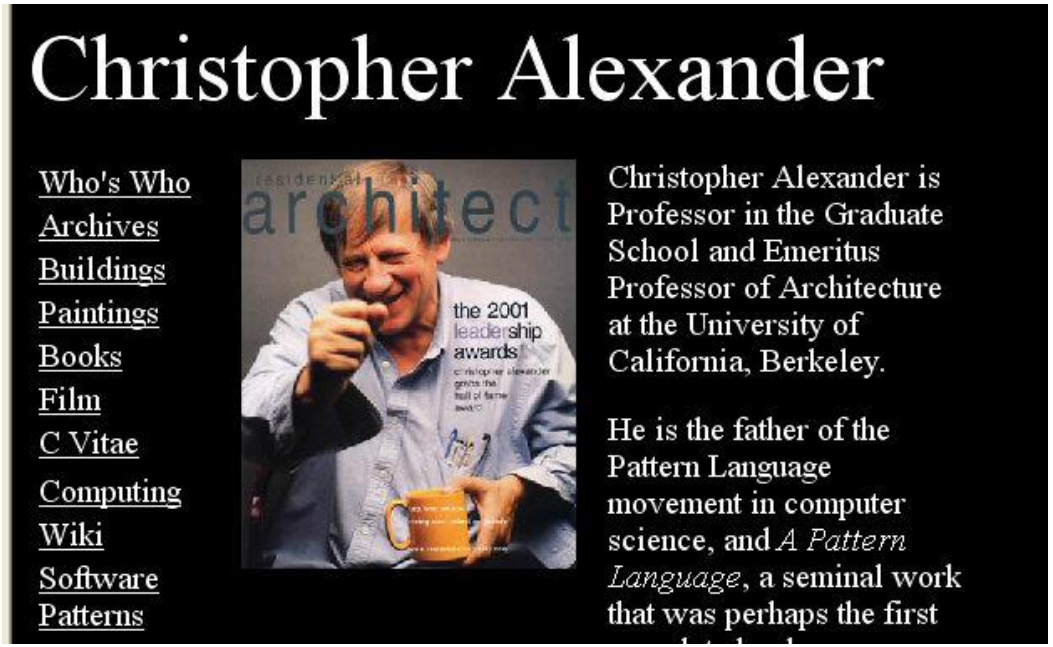
They are defined out of centuries of experience.

What is a (Design) Pattern?

"Each pattern 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

A Pattern Language, 1977

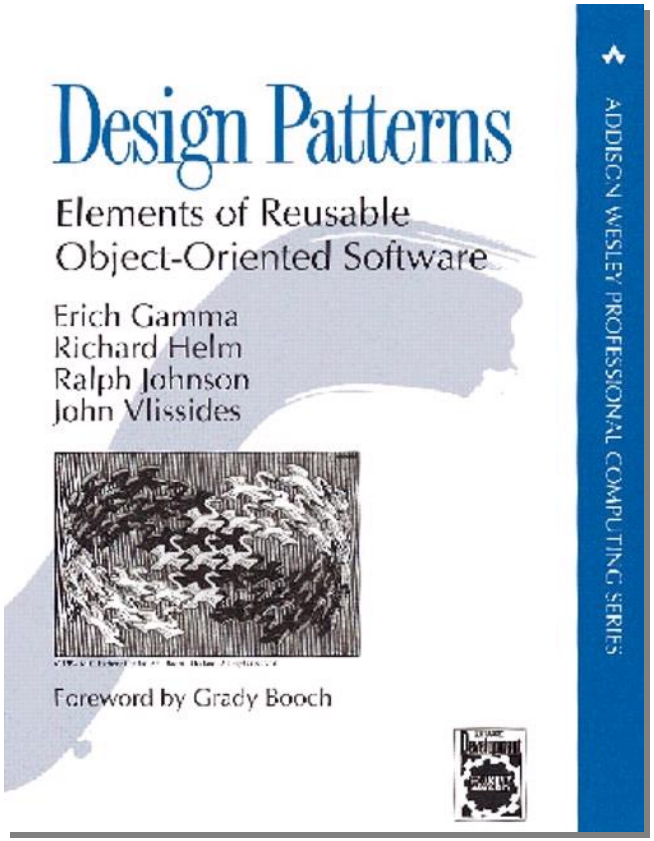


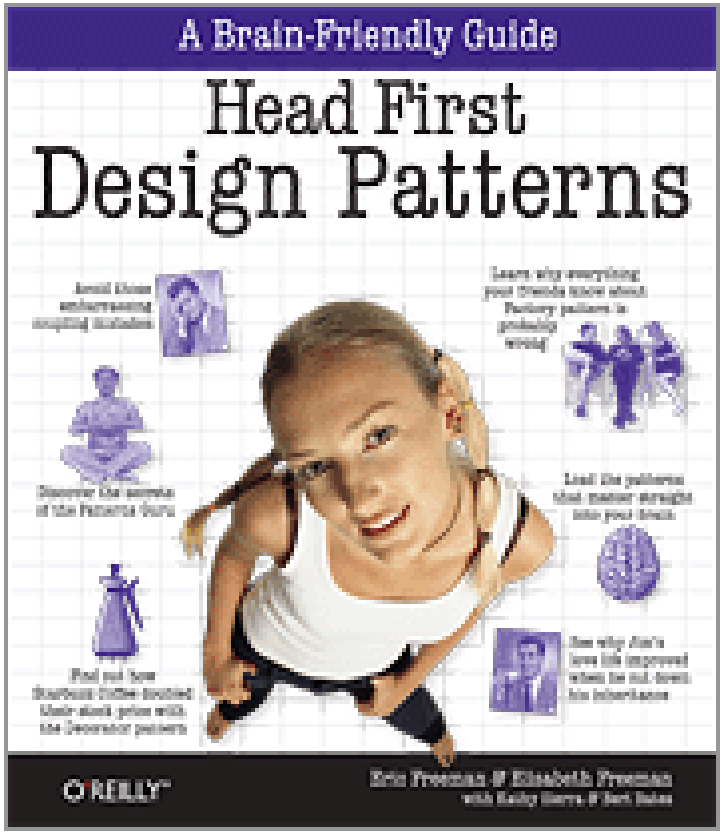
Design is not only a creative process

Carlo Scarpa told to a young architect:

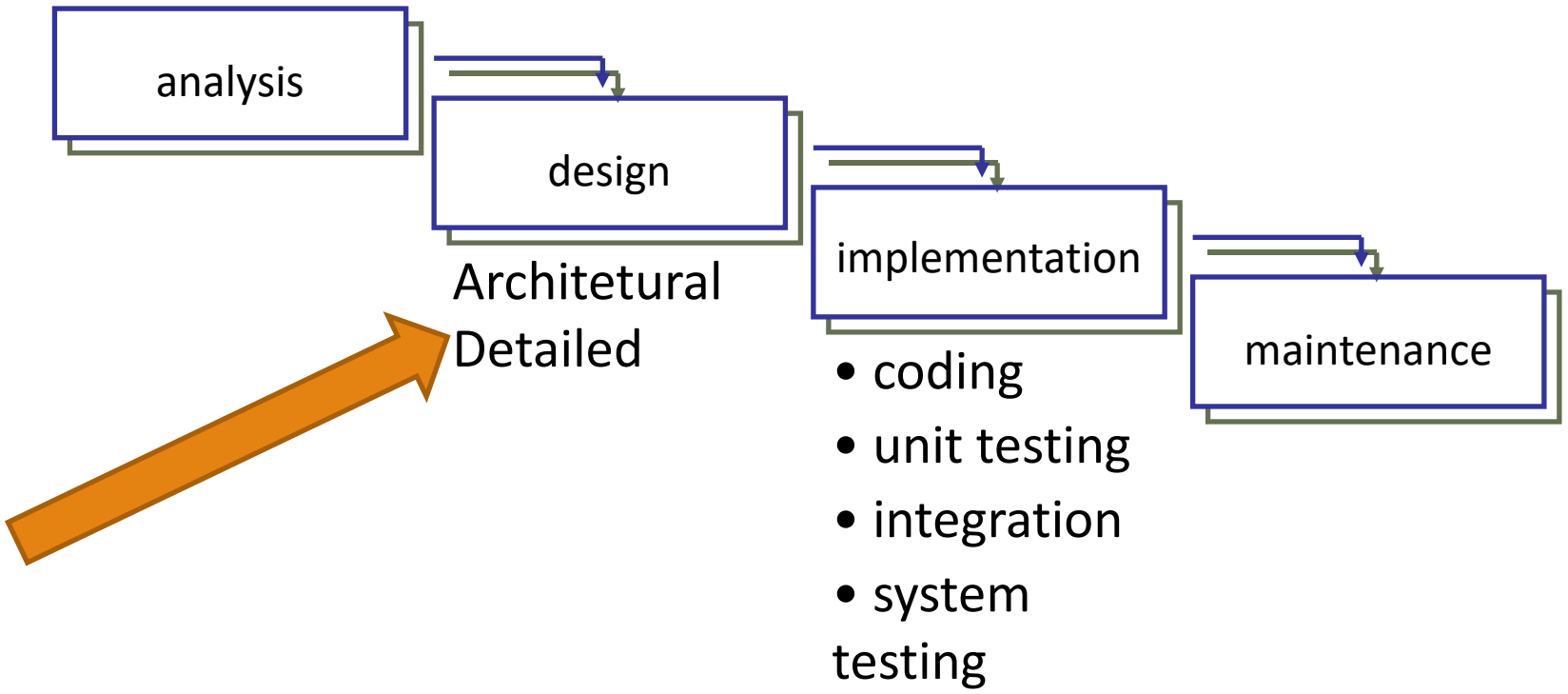
“Read a hundred pages of architecture per day”

libri suggeriti





In che fase si applicano



GoF Design Patterns

Sono 23 design pattern suddivisi in base al loro scopo

Creazionali:

- propongono soluzioni per creare oggetti

Comportamentali:

- propongono soluzioni per gestire il modo in cui vengono suddivise le responsabilità delle classi e degli oggetti

Strutturali:

- propongono soluzioni per la composizione strutturale di classi e oggetti

Why Patterns in software?

- "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.
- 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

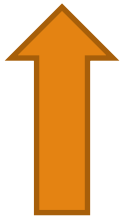
- 1987 - Cunningham and Beck used Alexander's ideas to develop a small pattern language for Smalltalk
- 1990 - The Gang of Four (Gamma, Helm, Johnson & Vlissides) begin compiling a catalog of design patterns
- 1991 - First Patterns Workshop at OOPSLA
- 1993 - Kent Beck and Grady Booch sponsor the first meeting of what is now known as the Hillside Group
- 1994 – 1st Pattern Languages of Programs (PLoP) conf.
- 1995 - The Gang of Four (GoF) *Design Patterns book*

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.



More Abstract

More Concrete

Architecture-Design-Code

Architectural Design Patterns

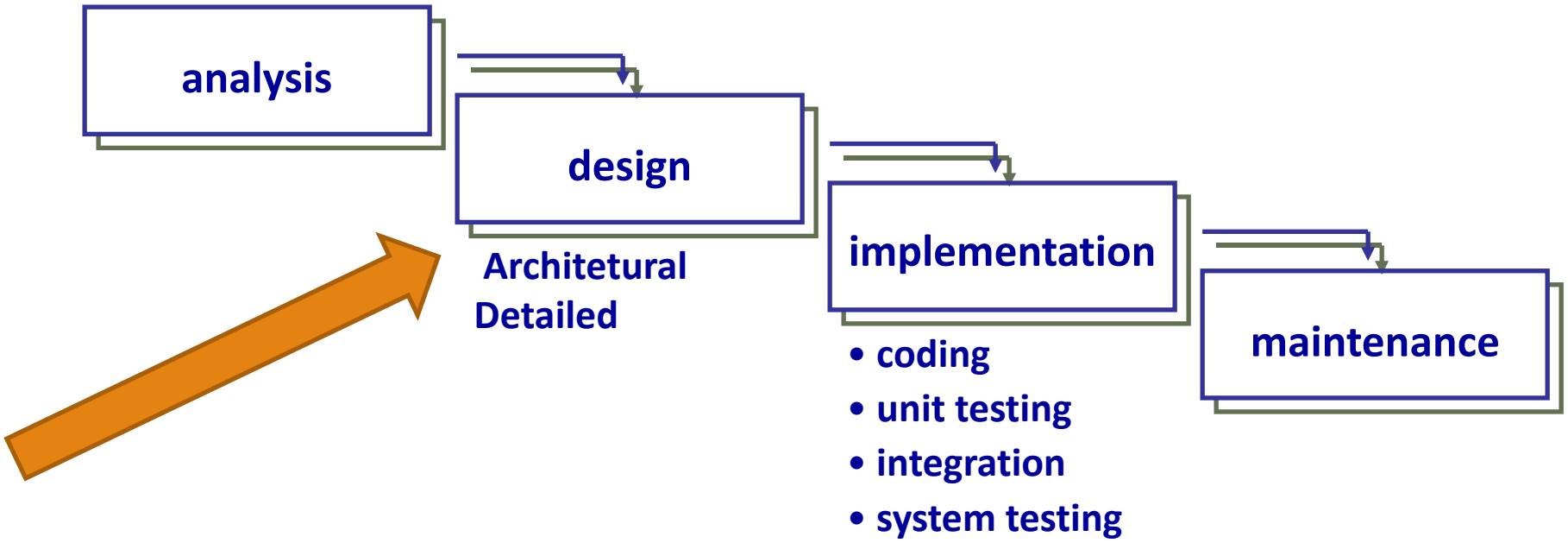
Design Patterns

Idioms o Coding Design Patterns

Architecture-Design-Code

Architectural Design Patterns

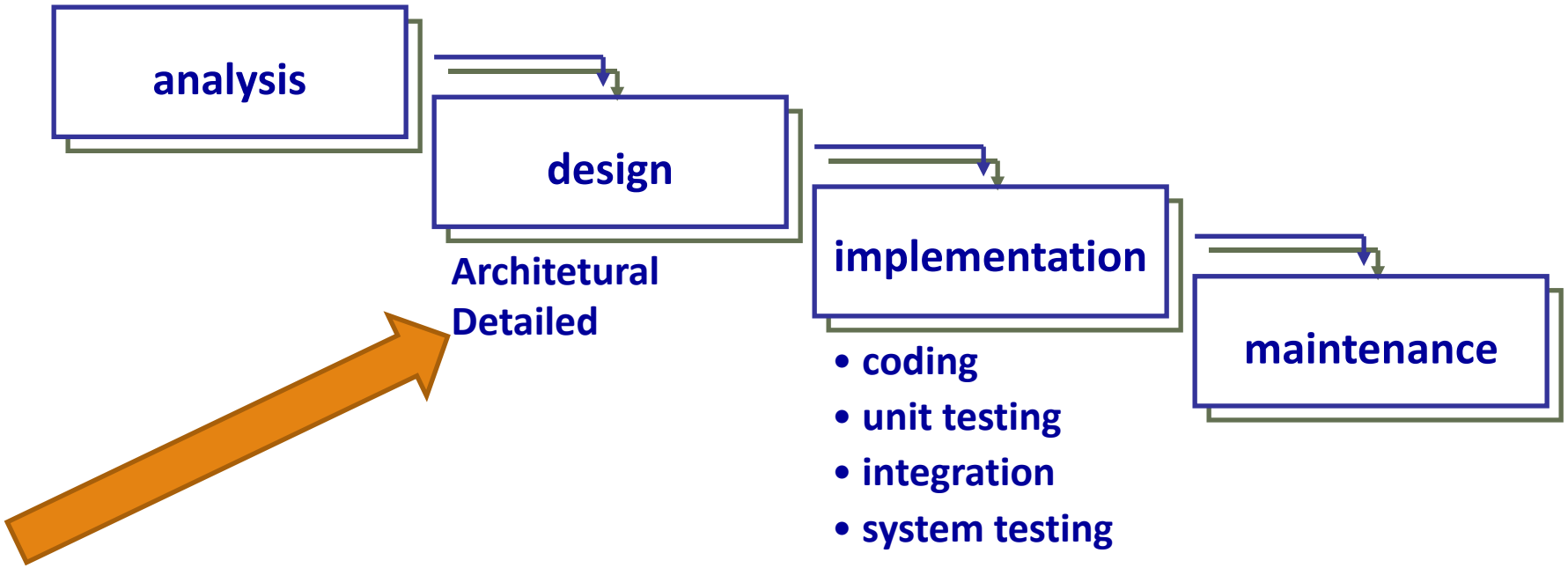
- Thay address the architecture of a sw system
- E.g. Layers, Pipes and Filters, Publish-Subsribe, Model-View-Controller, …



Architecture-**Design**-Code

Design Patterns

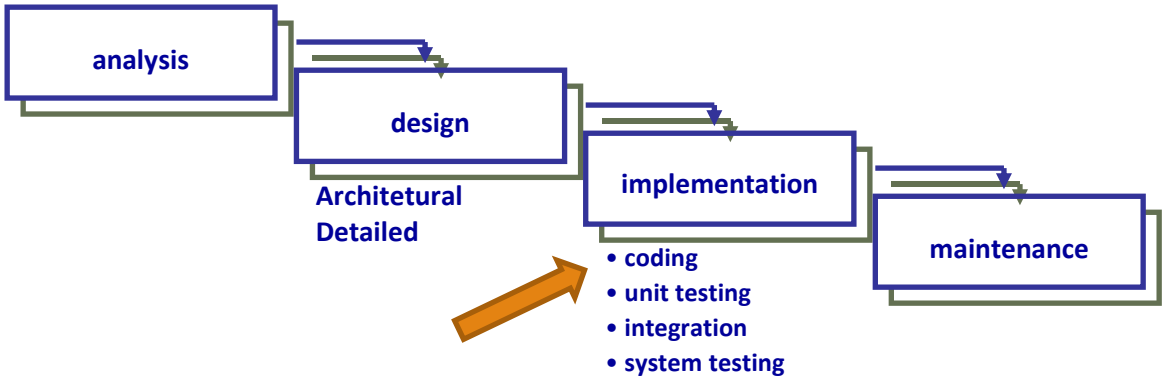
- They address the design and refinement of components.
- E.g. abstract factory, decorator, …



Architecture-Design-**Code**

Idioms o Coding Patterns

- Low-level patterns specific to a programming language.
- An Idiom is more restricted than a design pattern
  - Still describes a recurring problem
  - Provides a more specific solution, with fewer variations
  - Applies only to a narrow context
    - e.g., the C++ language
- E.g. Naming conventions, Source code formats, Memory management…



Best known families of patterns

- GRASP
  - General Responsibility Assignment Software Patterns (or Principles) [Graig Larman]
  - Information Expert, Creator, Controller, Low Coupling, High Cohesion, Polymorphism, Pure Fabrication, Indirection, Protected Variations
- SOLID
  - Single responsibility, Open-closed, Liskov substitution, Interface segregation and Dependency inversion
- GoF
  - 23 design patterns
- POSA
  - A System of Patterns: Pattern-Oriented Software Architecture
  - Volumes 1—5

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.”

GoF Classification Of Design Patterns

Purpose - what a pattern does

- Creational Patterns
  - Concern the process of object creation
  - *Abstract Factory, Builder, Factory Method, Prototype, Singleton.*
- Structural Patterns
  - Deal with the composition of classes and objects
  - *Adapter, Bridge, Composite, Decorator, Façade, Flyweight, Proxy.*
- Behavioral Patterns
  - Deal with the interaction of classes and objects
  - *Chain of responsibility, Command, Interpreter, Iterator, Mediator, Memento, Observer, State, Strategy, Template, Visitor.*

GoF Pattern Template

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 (Continued)

Structure

- A graphical representation of the pattern

Participants

- The classes and objects participating in the pattern

Collaborations

- How to 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 (Continued)

Sample Code

- Code fragments for a sample implementation

Known Uses

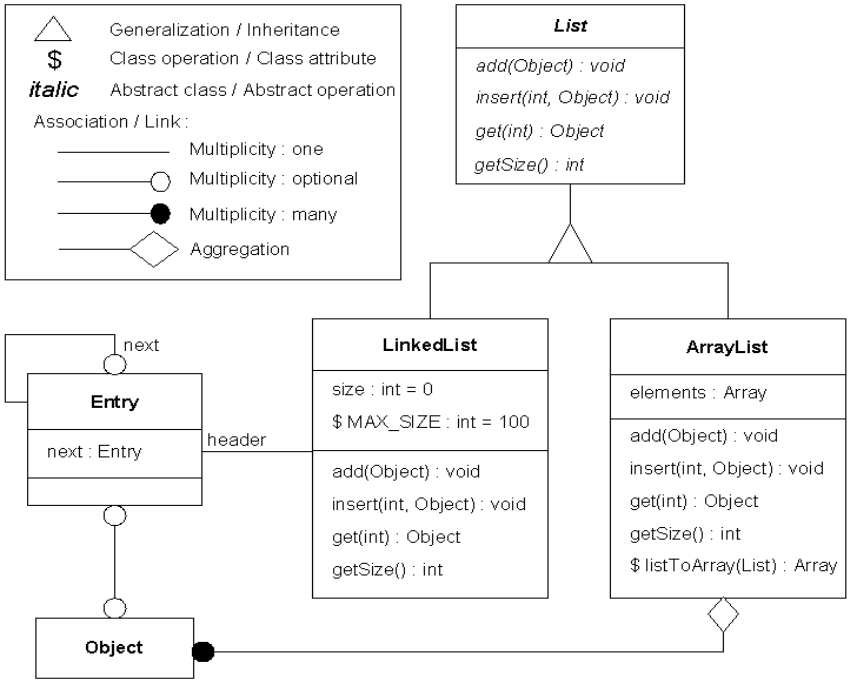
- Examples of the pattern in real systems

Related Patterns

- Other patterns that are closely related to the pattern

GoF Notation

The GoF book uses the Object Modeling Technique (OMT) notation for class and object diagrams



Head first uses UML