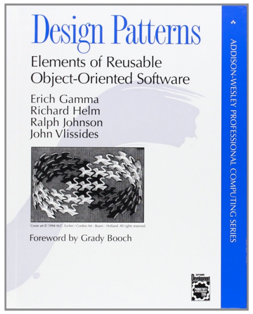
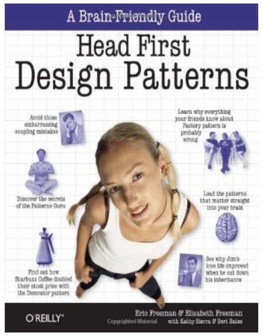
Design Patterns
General concepts

UA.DETI.PDS

José Luis Oliveira

Resources


Design Patterns
Elements of Reusable Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Foreword by Grady Booch
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES


A Brain-Friendly Guide
Head First
Design Patterns
O'REILLY
Eric Freeman & Elisabeth Freeman with Kathy Sierra & Bert Bates

- *Design patterns: elements of reusable object oriented software*. E. Gamma, R. Helm, R. Johnson, J. Vlissides. Addison Wesley, 1994.
- *Head first design patterns*. E. Freeman, E. Freeman, K. Sierra, B. Bates. O'Reilly, 2004.
- Also based on:
  - Object-Oriented Software Engineering, Glenn D. Blank, http://www.cse.lehigh.edu/~glennb/oose/oose.htm
  - Software Design, Joan Serrat, http://www.cvc.uab.es/shared/teach/a21291/web/

What are patterns?

- Principles and solutions codified in a structured format describing a problem and a solution
- A named problem/solution pair that can be applied in new contexts
- It is advice from previous designers to help designers in new situations
- The idea behind design patterns is simple:
  - Write down and catalog common interactions between objects that programmers have frequently found useful.
- Result:
  - Facilitate reuse of object-oriented code between projects and between programmers.

Some definitions of design patterns

- "Design patterns constitute a set of rules describing how to accomplish certain tasks in the realm of software development." (Pree, 1994)
- "Design patterns focus more on reuse of recurring architectural design themes, while frameworks focus on detailed design… and implementation." (Coplien & Schmidt, 1995).
- "A pattern addresses a recurring design problem that arises in specific design situations and presents a solution to it" (Buschmann, et. al. 1996)
- "Patterns identify and specify abstractions that are above the level of single classes and instances, or of components." (Gamma, et al., 1993)

Characteristics of Good patterns

- It solves a problem
- It is a proven concept
- The solution isn't obvious
- It describes a relationship
- The pattern has a significant human component

Types of patterns

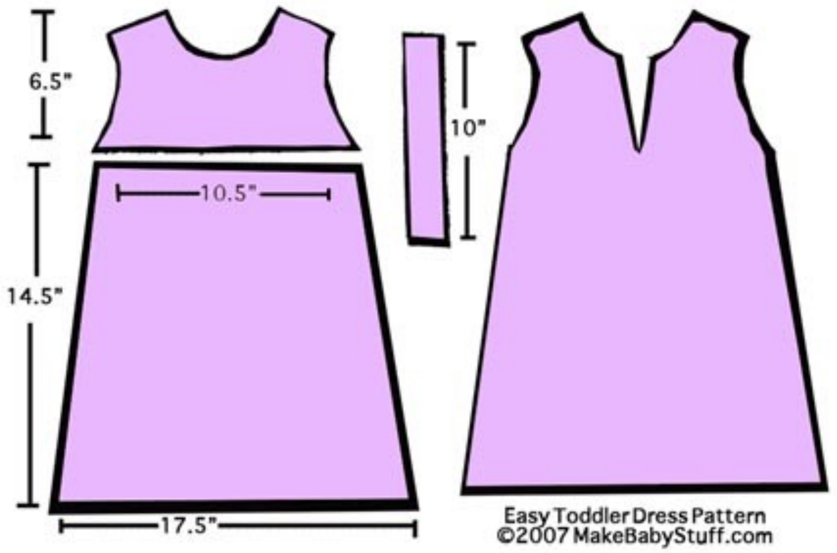
- Architectural Patterns
  - Expresses a fundamental structural organization or schema for software systems.
- Design Patterns
  - Provides a scheme for refining the subsystems or components of a software system, or the relationships between them.
- Idioms
  - An idiom describes how to implement particular aspects of components or the relationships between them using the features of the given language.

Design patterns in architecture

- A pattern is a recurring solution to a standard problem, in a context.
- Christopher Alexander, professor of architecture…
  - Why is what a prof of architecture says relevant to software?
  - "A 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."

Design and dress patterns

- Jim Coplein, a software engineer:
  - "I like to relate this definition to dress patterns …
  - I could tell you how to make a dress by specifying the route of a scissors through a piece of cloth in terms of angles and lengths of cut. Or, I could give you a pattern. Reading the specification, you would have no idea what was being built or if you had built the right thing when you were finished. The pattern foreshadows the product: it is the rule for making the thing, but it is also, in many respects, the thing itself."



Patterns in engineering

- How do other engineers find and use patterns?
  - Mature engineering disciplines have handbooks describing successful solutions to known problems
  - Automobile designers don't design cars from scratch using the laws of physics
  - Instead, they reuse standard designs with successful track records, learning from experience
  - Should software engineers make use of patterns? Why?
- Developing software from scratch is also expensive
  - Patterns support reuse of software architecture design

Gang of Four (GoF) Patterns

- Eric Gamma and colleagues published in 1995 the influential book Design patterns: *Elements of Reusable Object-Oriented Software*.
- Has a catalogue of 23 patterns. For each one, a template is followed:
  - Name
  - Intent : what it does and advantages 1–2 sentences
  - Motivation : example
  - Structure : template class diagram
  - Applicability : when to use it
  - Consequences : advantages and shortcomings
  - Implementation discussion, C++ sample code

Naming Patterns – important!

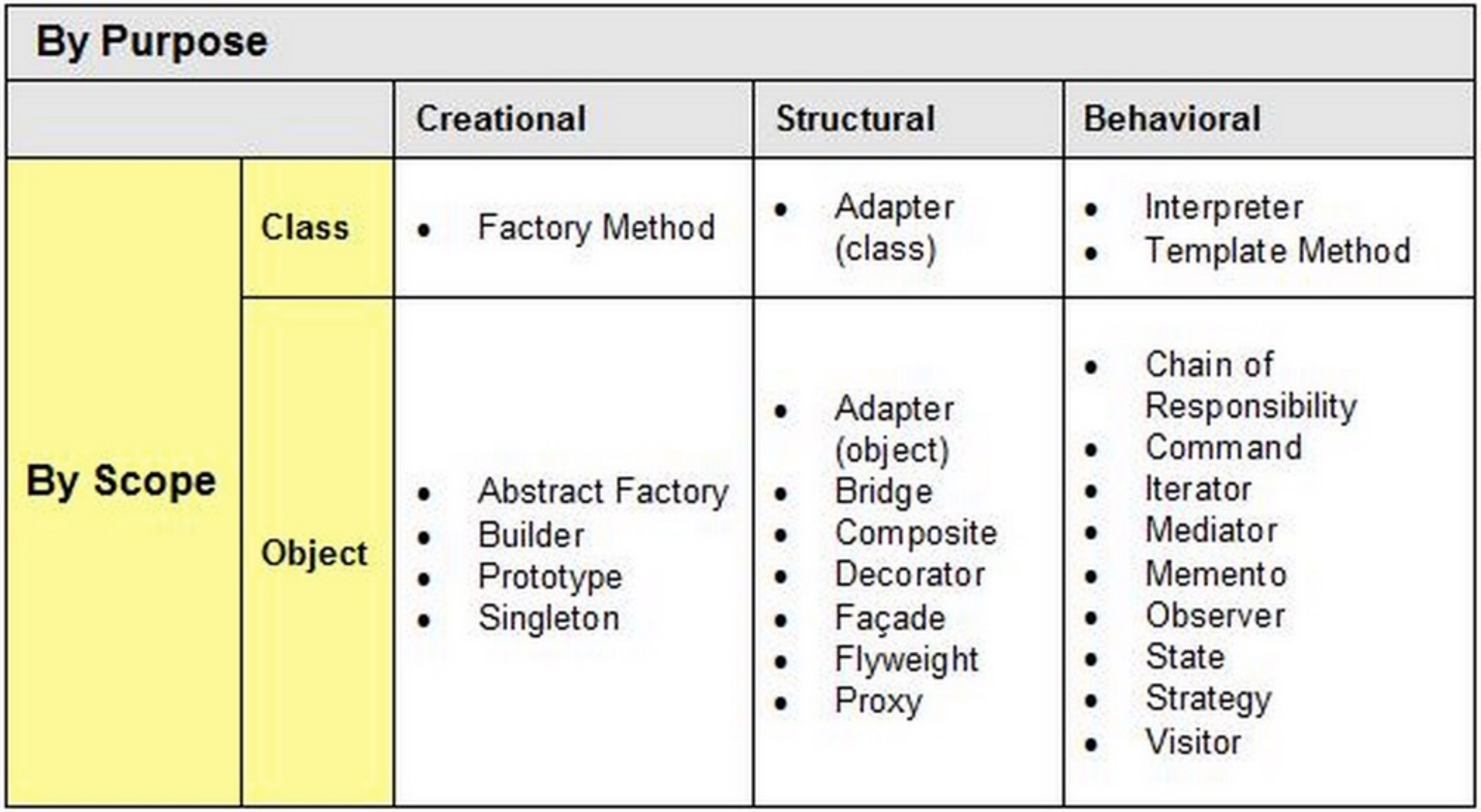
- Patterns have suggestive names:
  - Arched Columns Pattern, Easy Toddler Dress Pattern, etc.
- Why is naming a pattern or principle helpful?
  - It supports chunking and incorporating that concept into our understanding and memory
  - It facilitates communication

GoF Patterns

- Gamma et al. classify patterns into 3 groups:

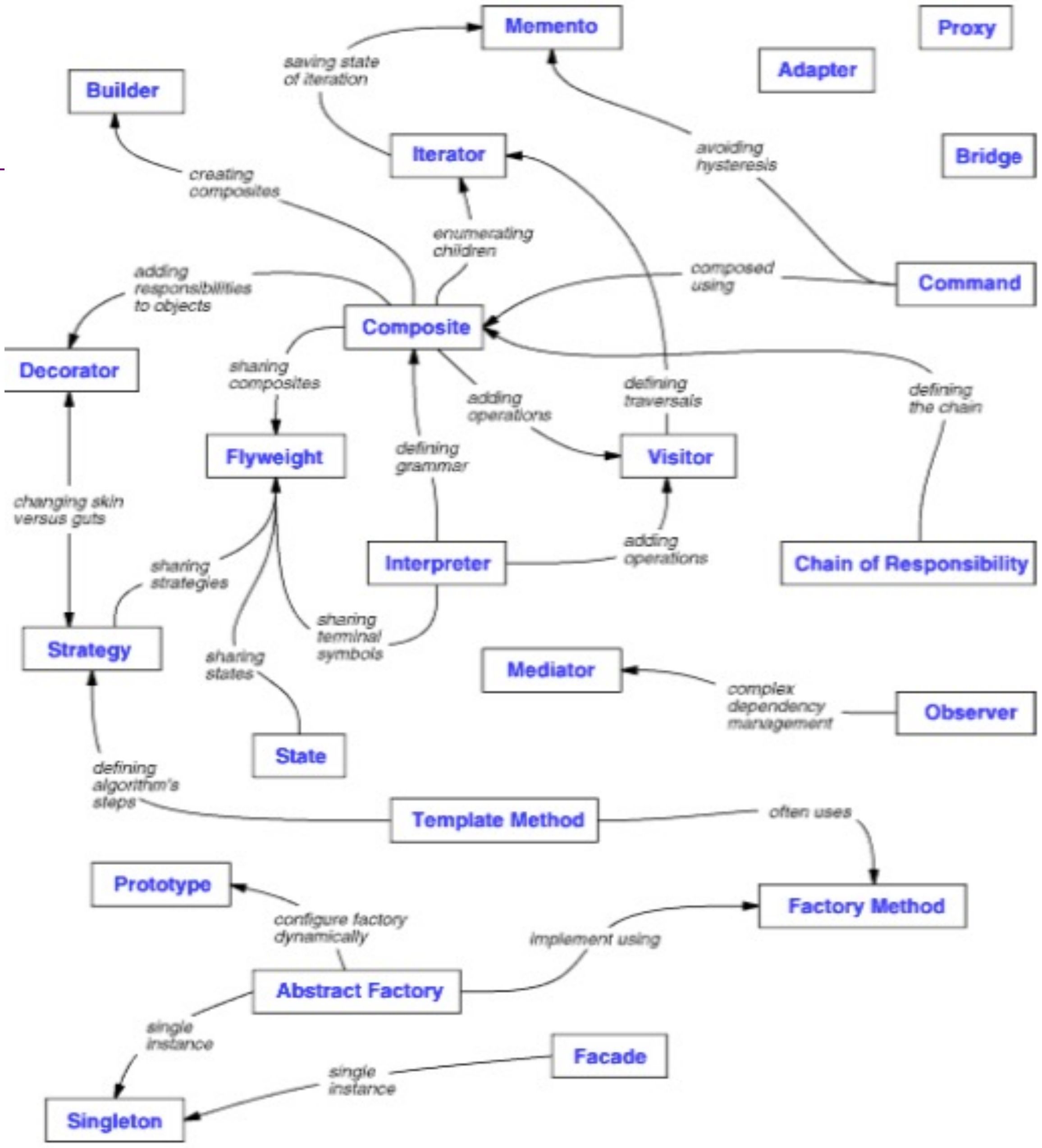
- Creational
  - patterns concern the process of object creation
- Structural
  - patterns deal with the composition of classes or objects
- Behavioral
  - patterns characterize the ways in which classes or objects interact and distribute responsibilities

GoF Patterns

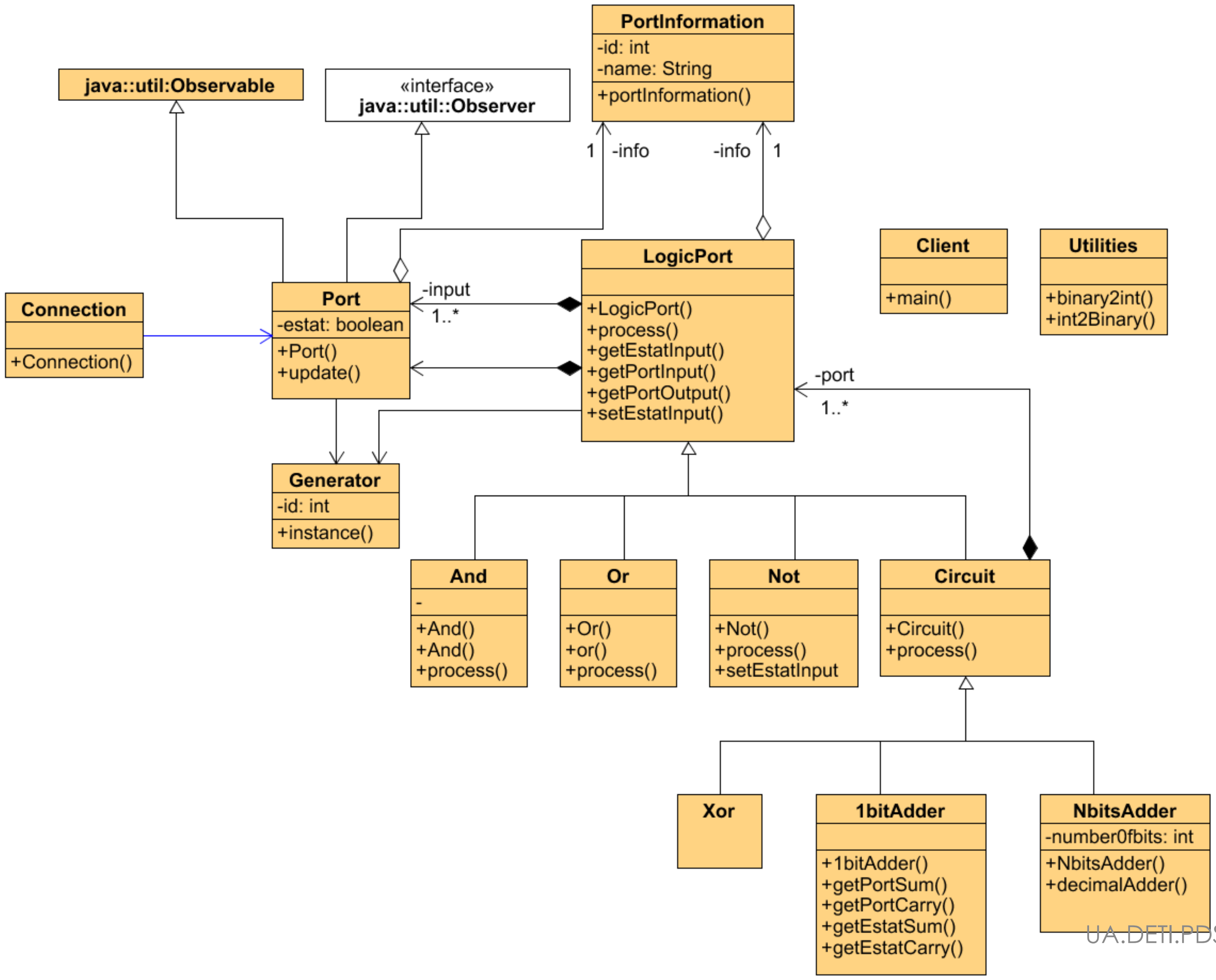


By Purpose				
		Creational	Structural	Behavioral
By Scope	Class	• Factory Method	• Adapter (class)	• Interpreter • Template Method
	Object	• Abstract Factory • Builder • Prototype • Singleton	• Adapter (object) • Bridge • Composite • Decorator • Façade • Flyweight • Proxy	• Chain of Responsibility •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

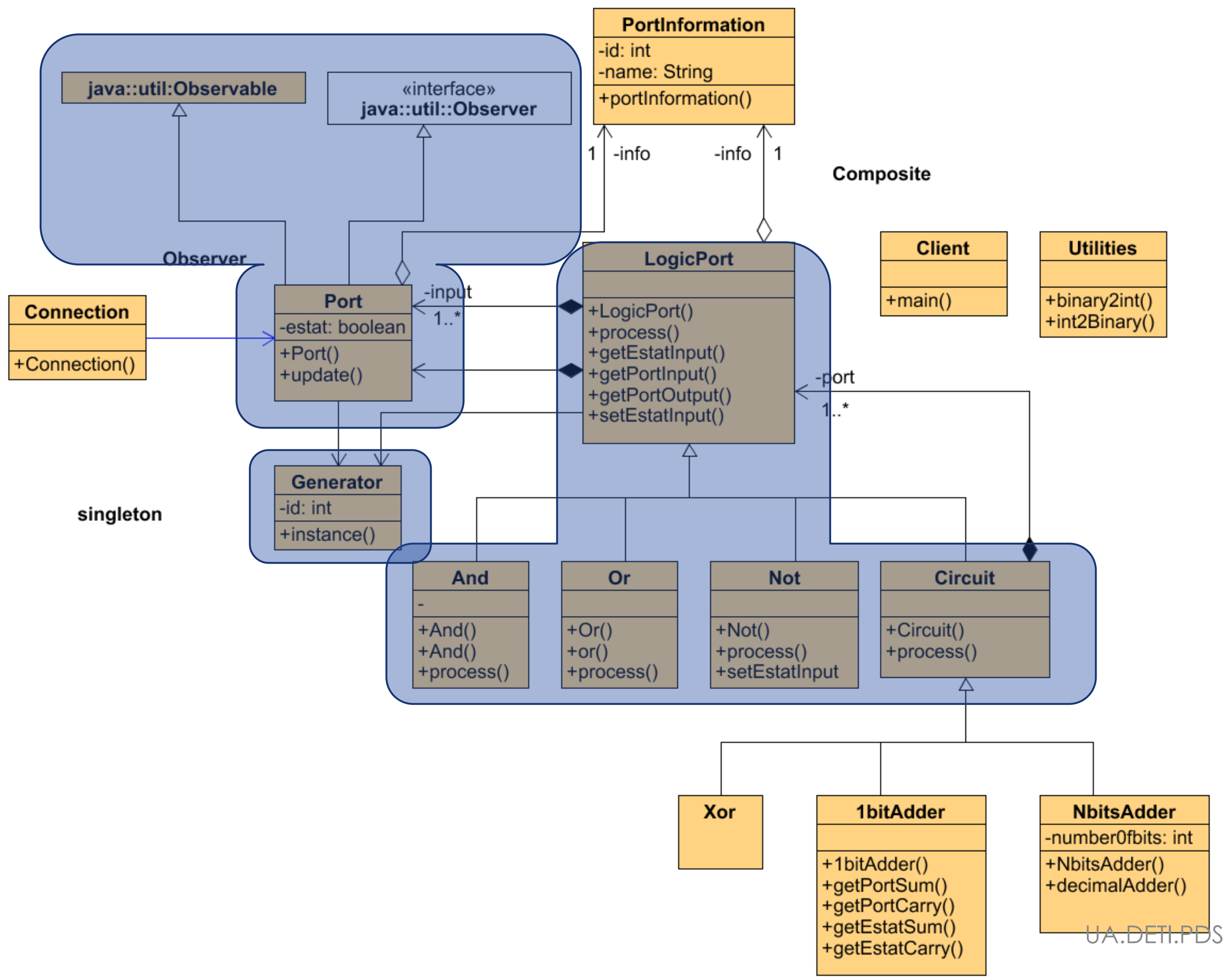
Relationships



Why patterns?



Why patterns?



Why patterns?

- A novice chess player knows
  - the game rules
  - the value of all pieces
- A good chess player knows
  - tactics: occupy central cells, …
  - strategies: immobilize, win with two bishops, …
  - apertures, famous matches
- A novice OO designer must know
  - inheritance, encapsulation, data abstraction . . .
  - UML notation
- An expert designer knows
  - object oriented principles
  - examples of good designs
  - design patterns

More on this in the next weeks…