

# Design patterns & Meta-model

A decorative horizontal bar consisting of a series of vertical rectangular segments in various colors including black, blue, light blue, teal, yellow, and dark blue, arranged in a slightly wavy pattern.

Pattern identification  
and code generation

# Us



- Hervé Albin-Amiot
  - 3<sup>rd</sup> year PhD student
  - Partly funded by Soft-Maint S.A. (France)



- Yann-Gaël Guéhéneuc
  - 2<sup>nd</sup> year PhD student
  - Partly funded by Object Technology International, Inc. (Canada)



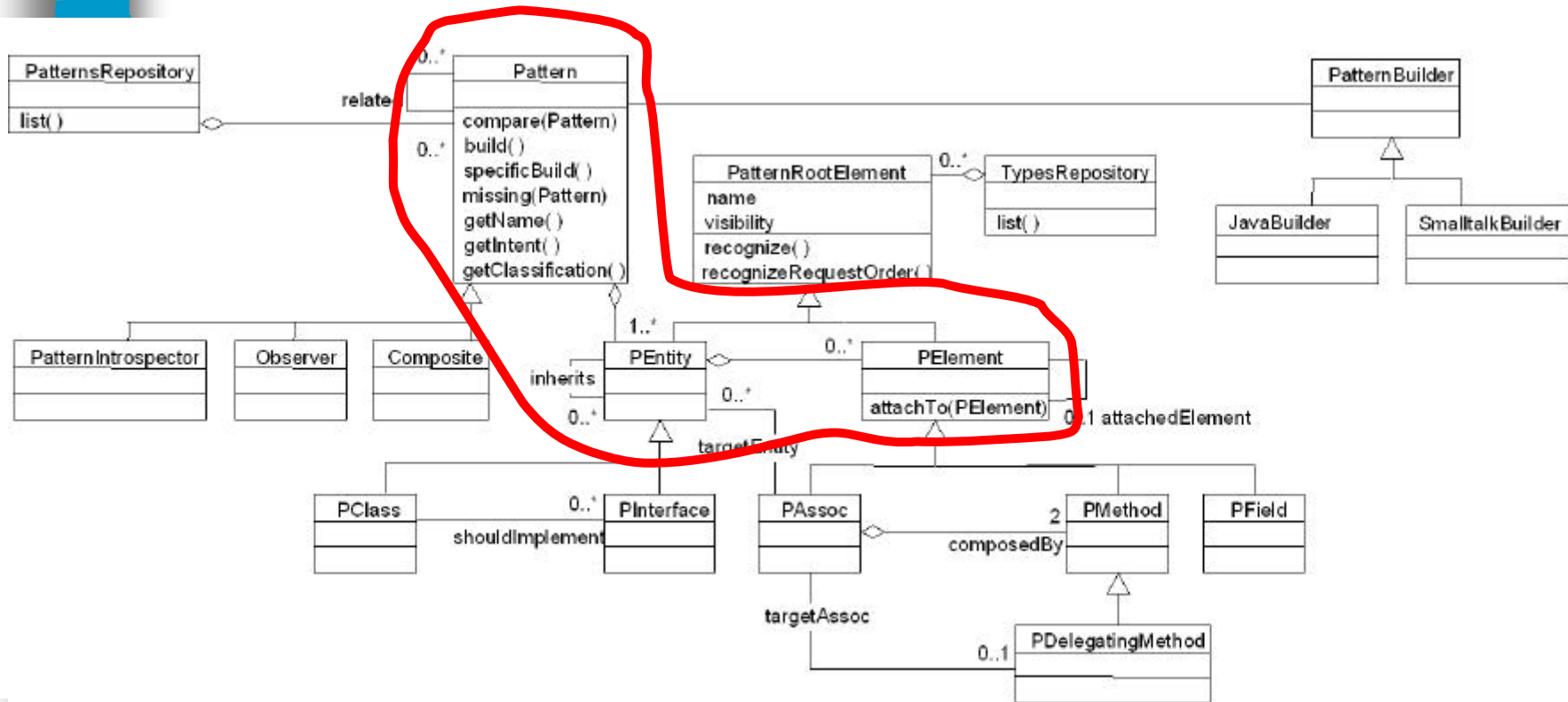
- PhDs hosted by the  
École des Mines de Nantes



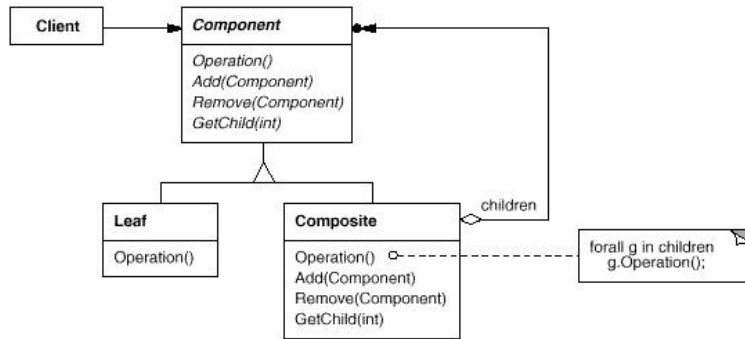
# Goals

- Formalize design patterns
- Describe design patterns
  - First-class entities
    - They know how to generate their source code
    - They know how to identify their occurrences
  - Manipulatable entities
    - We can reason about them
    - We can adapt them to specific contexts

# A solution: A meta-model



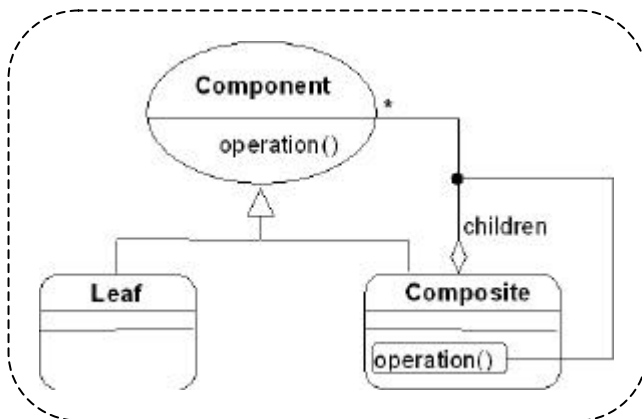
# A Basic Use



Informal descriptions from [GoF]



Translates into a design pattern model



## Legend



Instance of Pattern



Instance of PInterface



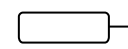
Instance of PClass



Instance of PAssoc



Instance of PDelegation



`name()`

Instance of PMethod



# Goals fulfillment

- A design pattern model is an object:
  - It has a structure composed by the set of the design pattern constituents
  - It answers to messages
    - Message `build()` generates code
    - Message `compare()` matches an instance of a design pattern model with some source code
    - Specific message `addLeaf()` for Composite design pattern model, ...



# Future

- To have two separate meta-models:
  - One specific for design patterns
  - One specific for source code
- To apply patterns to existing code with source-to-source transformation
- To define an operational semantics for association, aggregation, ... links
- To manage dynamic information