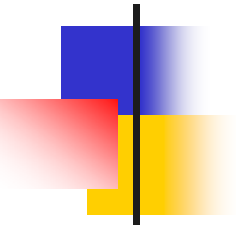
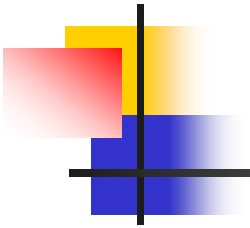


Presentation Models for Design Pattern Detection Tools

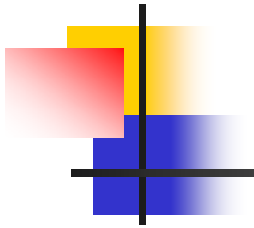
Shouzheng Yang

Oct. 26, 2011



Motivation

- How can we understand FINDER's results in a short time?
- Do the detected instances potentially contain some relations?
- Other design pattern detection tools also have this problem.
- As far as I know, nobody has done this in the literature.

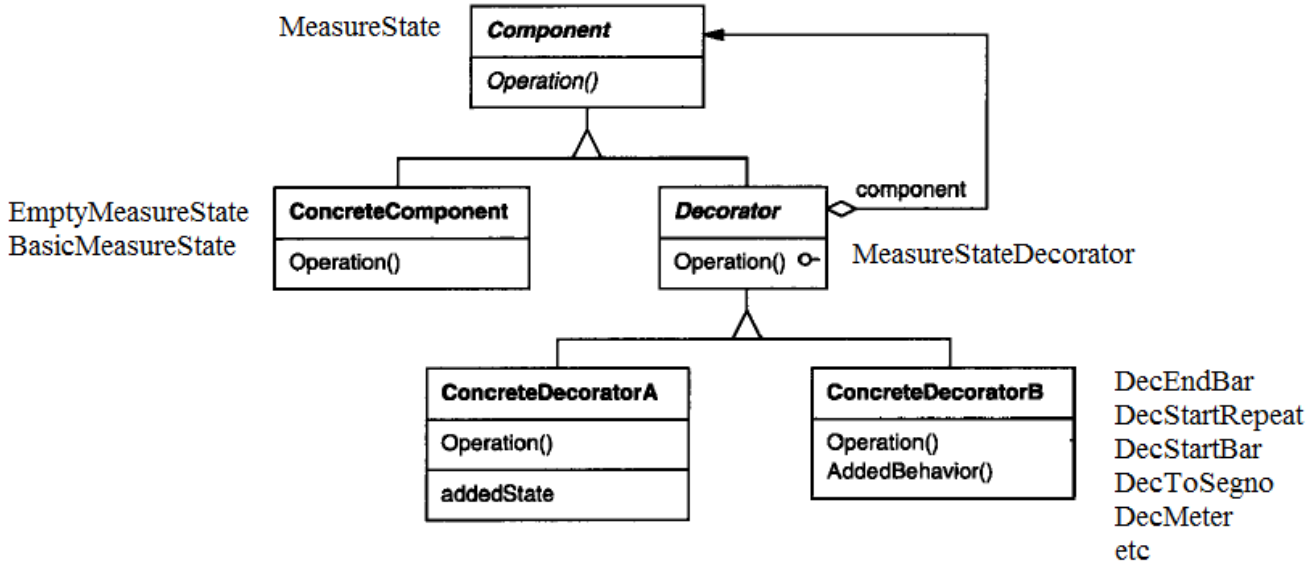


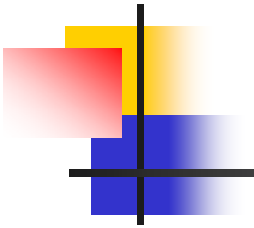
First Stage

- We noticed that design patterns roles are not necessary equally important.
- E.g. Decorator Pattern
  - More important: Component, Decorator
  - Less important: Concrete Decorator, Concrete Component, Client

First Stage

- Some role instances are often shared among multiple pattern instances.
- E.g. TAB2PS (Decorator)



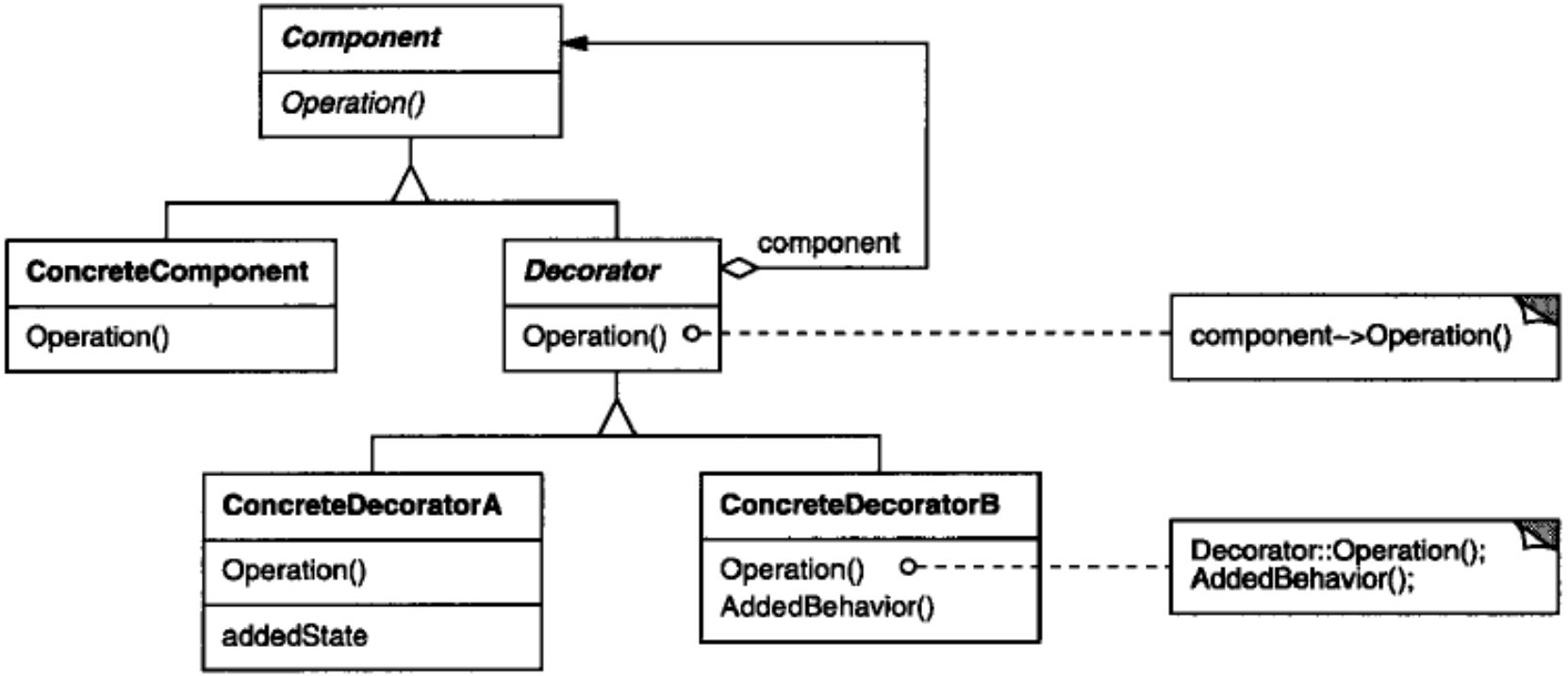


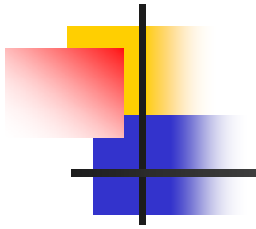
First Stage

- So we see the need for grouping pattern instances by the “shared roles”.
- Benefits:
  - Redundant information is removed.
  - Related pattern instances are able to be placed together for better overall comprehension.
  - It helps to quickly eliminate false positives.
  - It can potentially help to compare the results produced by different detection tools.

Second Stage

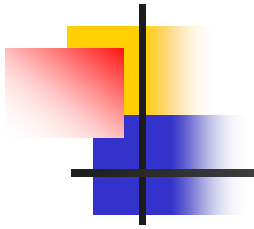
- We noticed some roles are independent of each other.





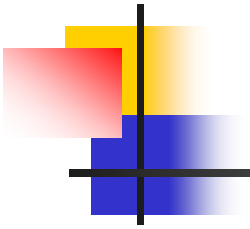
Second Stage

- In theory, all combinations of independent roles should be detected.
  - This can help verify the detection tools.
- We say there is a **cross-product** relation between independent roles.



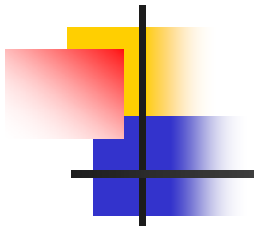
Second Stage

- So what about dependent relations?
  - One-to-one
  - One-to-many
  - Many-to-many
- In practice, many-to-many is often the case.



Second Stage

- What can we do for dependent relations?
  - Indicate the type of relation.
  - Interactively mark potential false positives.
  - Separate unrelated role instances.
    - E.g. (composite, componentArray)
    - A composite role could have multiple componentArrays.
    - Different composite roles should be separated as sub-groups.

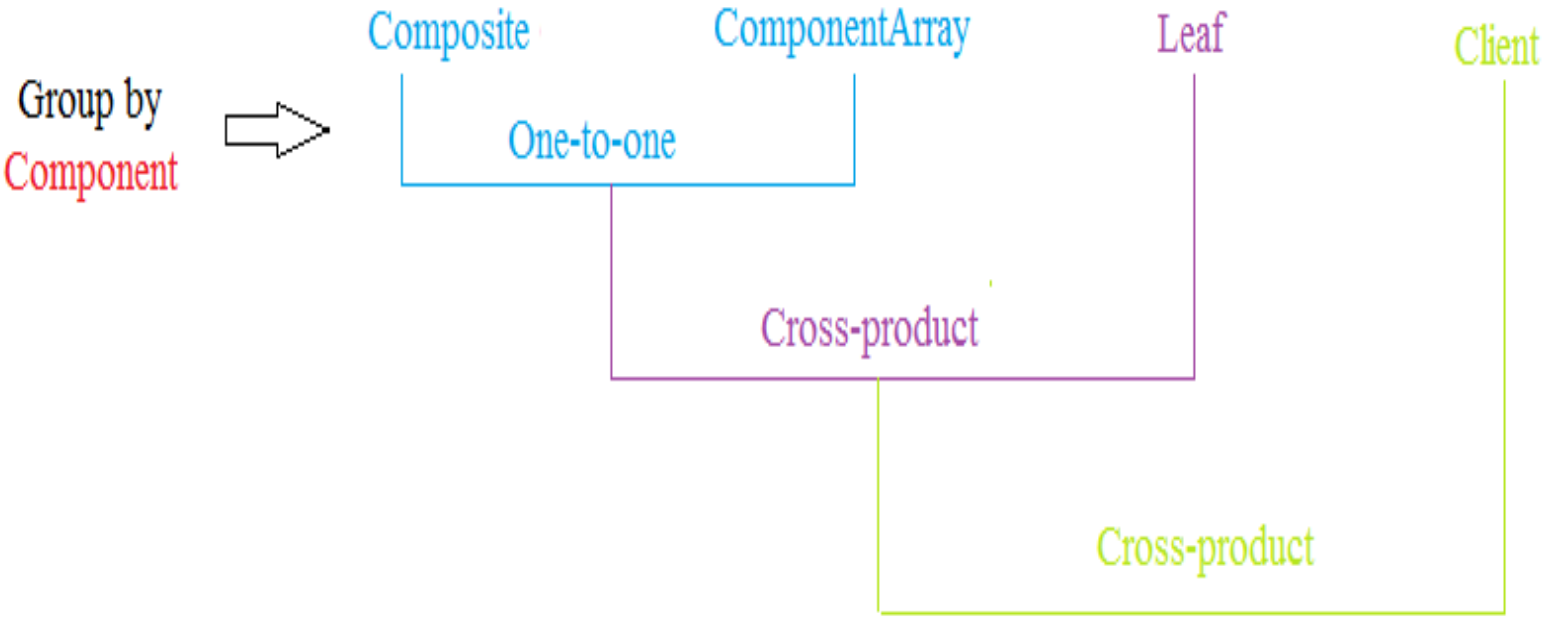


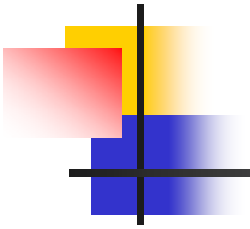
Presentation Model

- In order to realize the aforementioned ideas, we need
  - A model for the description of design pattern presentation constraints.
    - Group by “shared role(s)”.
    - Relationships between all non-shared role(s).

Presentation Model

- E.g. Composite Pattern





Presentation Model

- A pattern could have more than one models for different implementation variants, or just for different views, like including/excluding client.
  - This allows for the presentation of the results even if some roles are missing.

Overall Idea