ISP: The Interface Segregation Principle

- Depending on the modules that contain more than you need
  - may force unnecessary recompilation and redeployment
  - the same problem can exist at the architectural level
  -

DIP: The Dependency Inversion Principle

Minimize the source code dependencies

- Avoid depending on volatile concretions, favor the stable abstract interfaces
    - Don’t refer to volatile concrete classes
    - Don’t derive from volatile concrete classes
    - Don’t override concrete functions
    - Never mention the name of anything concrete and volatile
- Dependency rule
    - Abstract factory pattern
    - all dependencies points upon more abstract entities