Cascading Factories to Eliminate Dependencies

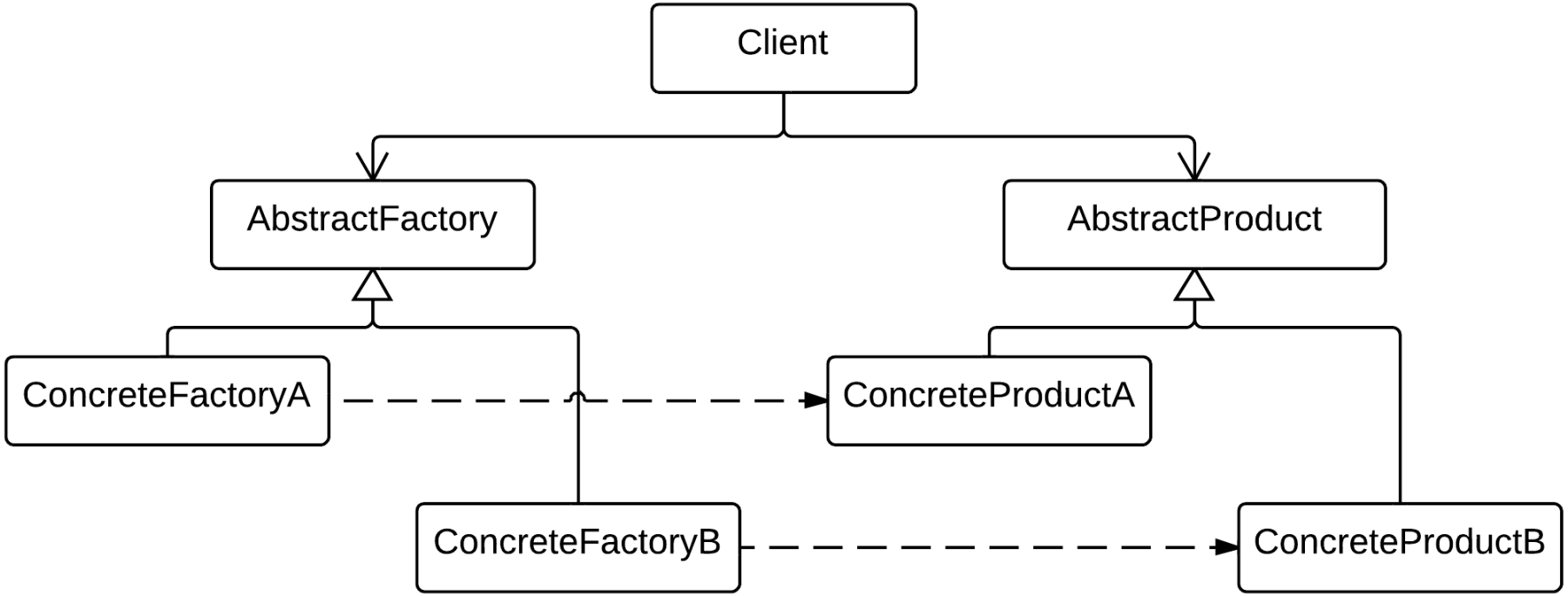
Zoran Horvat

@zoranh75 | www.codinghelmet.com

Definition of Abstract Factory

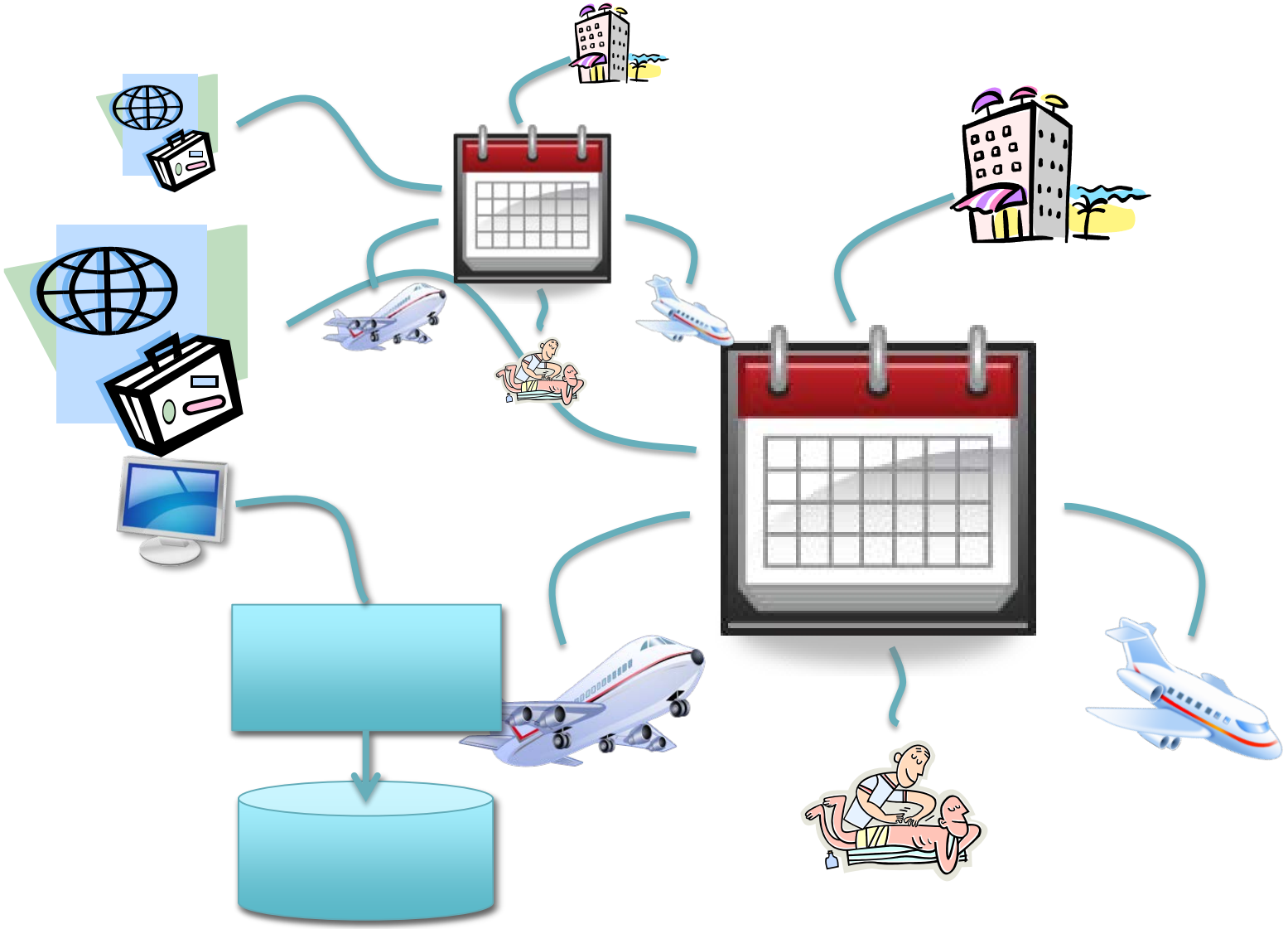
"Provide an interface for creating families of related or dependent objects without specifying their concrete classes."

Gama et al., Design Patterns: Elements of Reusable Object-Oriented Software

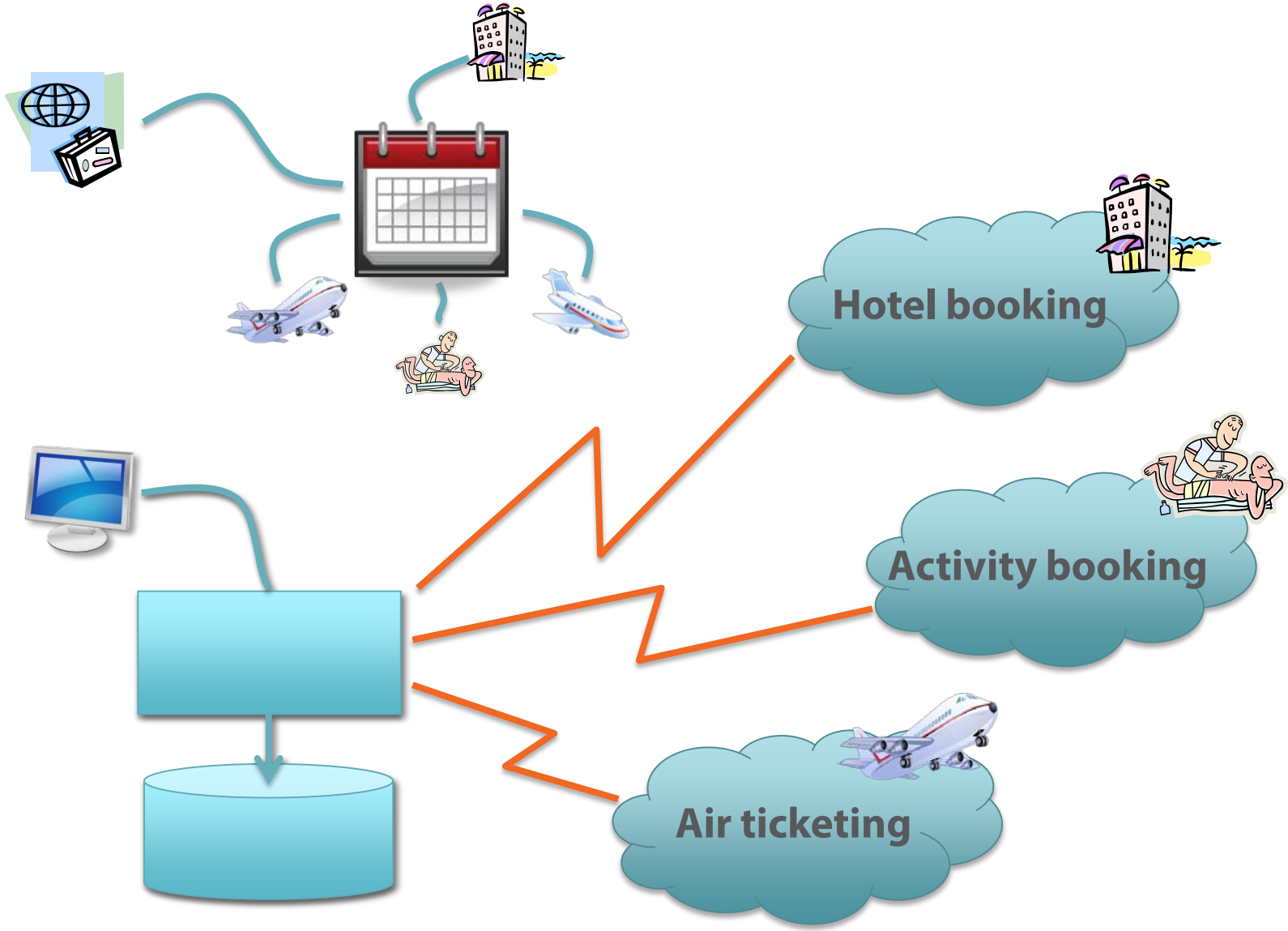


Example: Booking Application

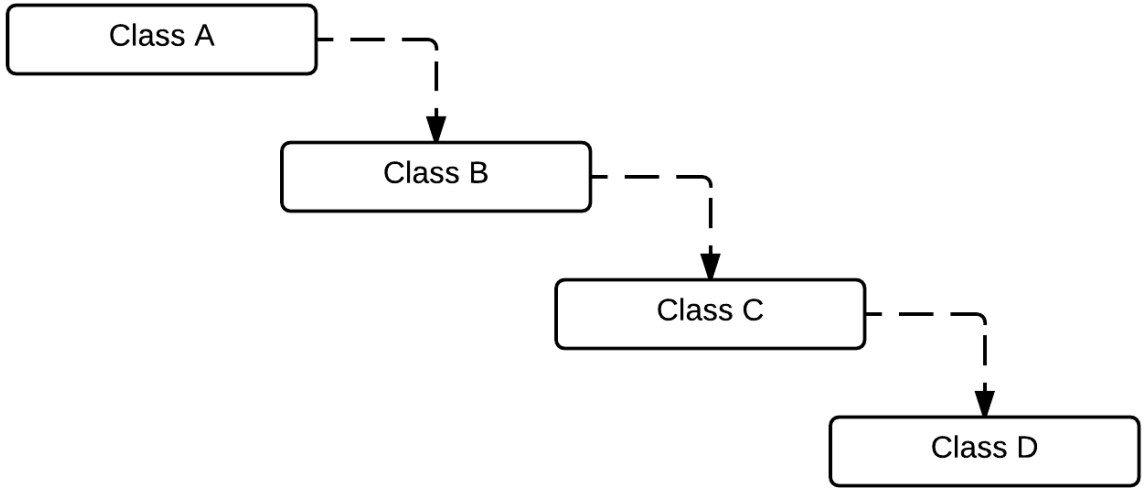
Example: Booking Application



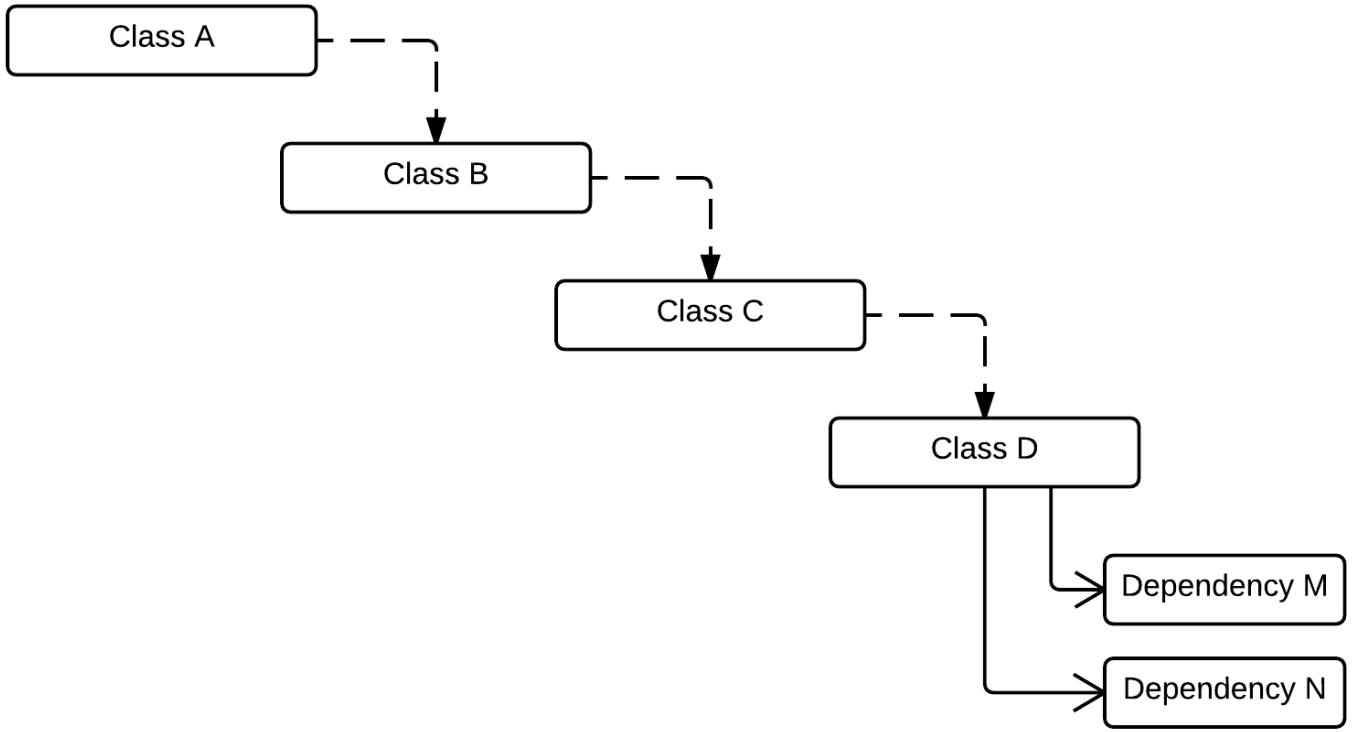
Example: Booking Application



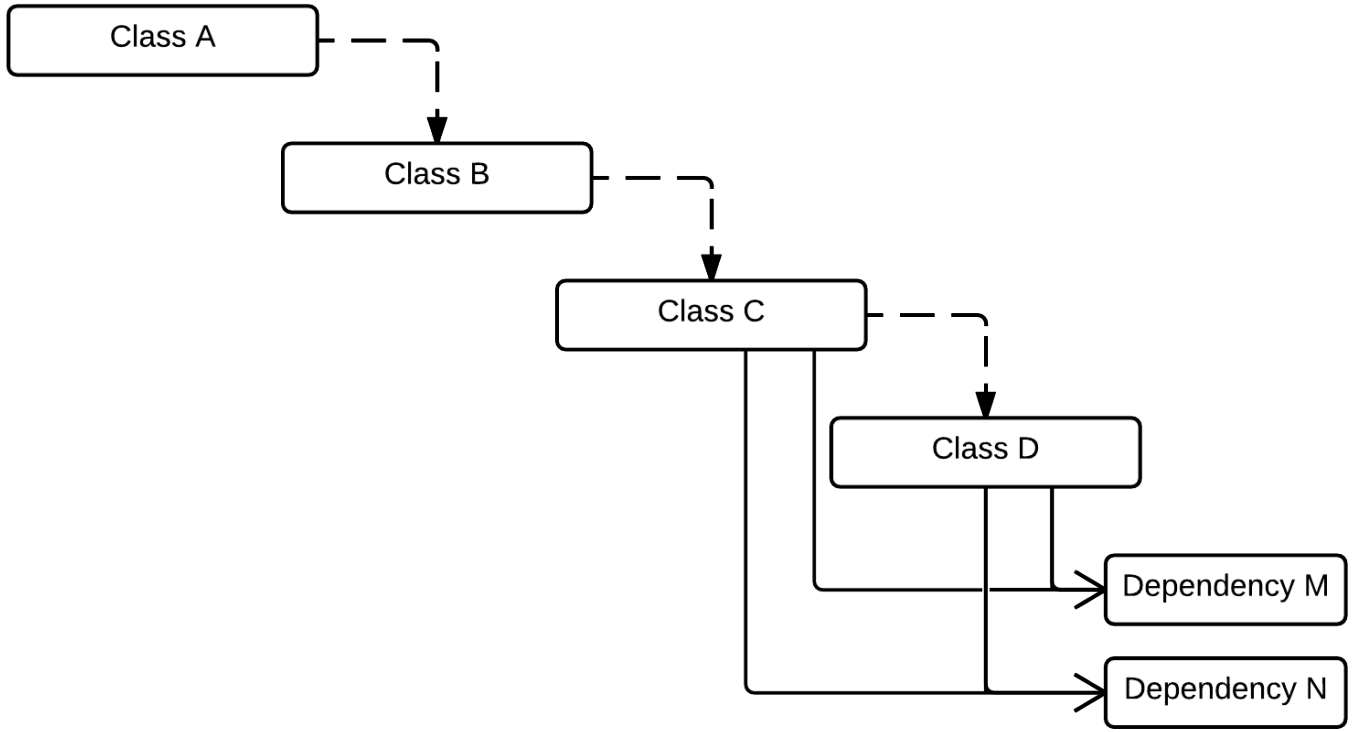
Classes With Dependencies



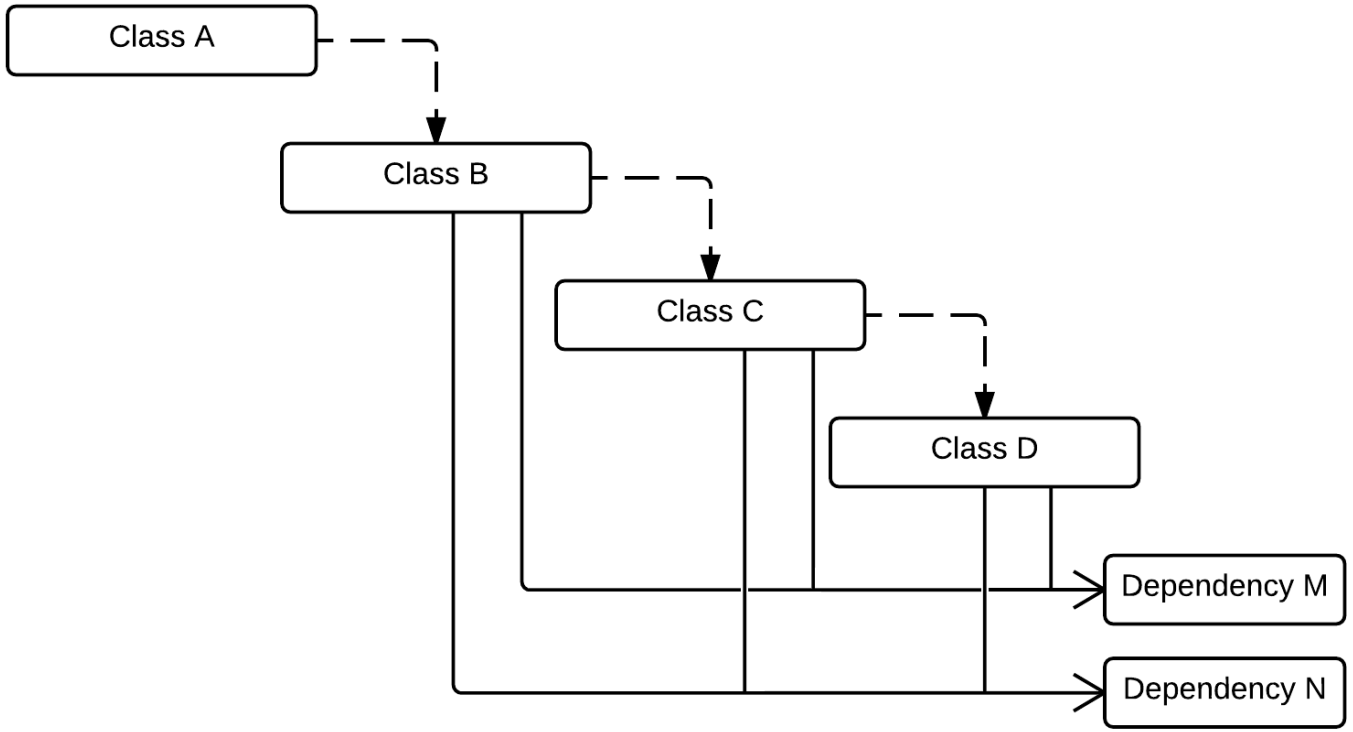
Classes With Dependencies



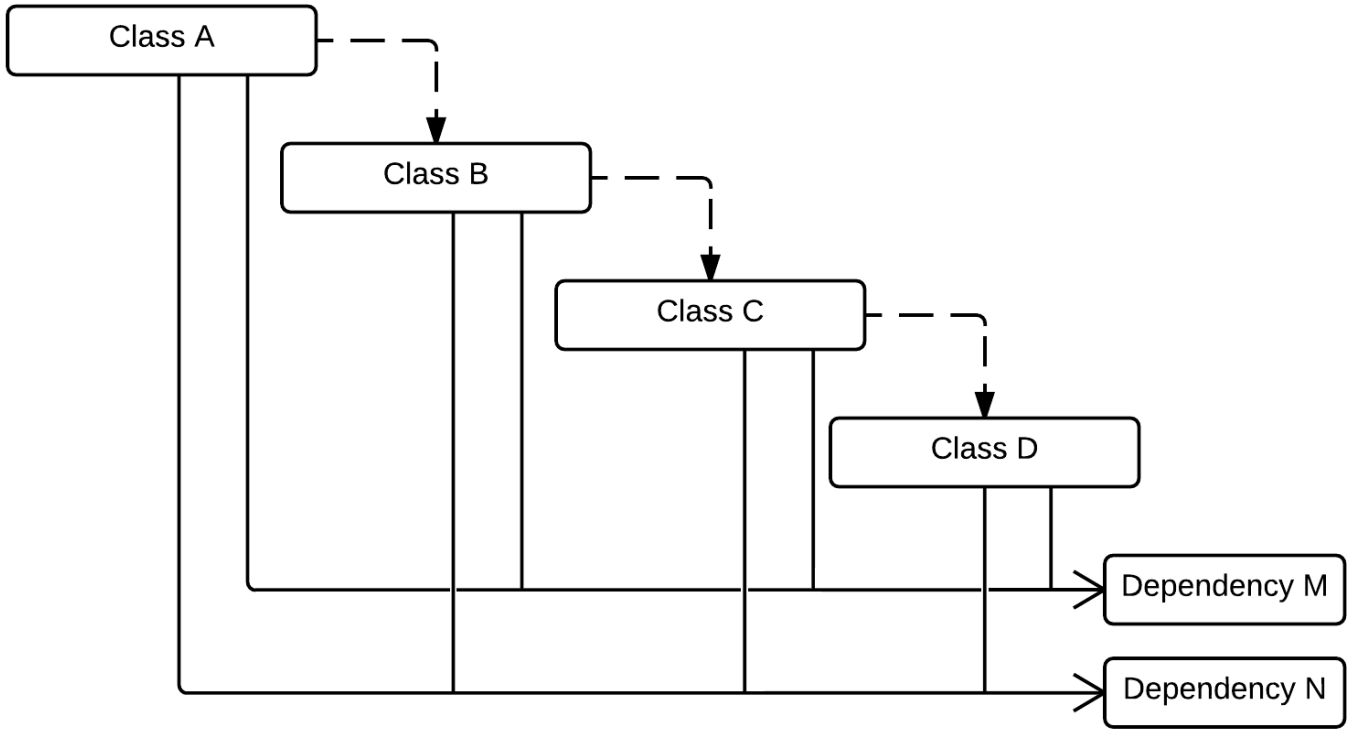
Classes With Dependencies



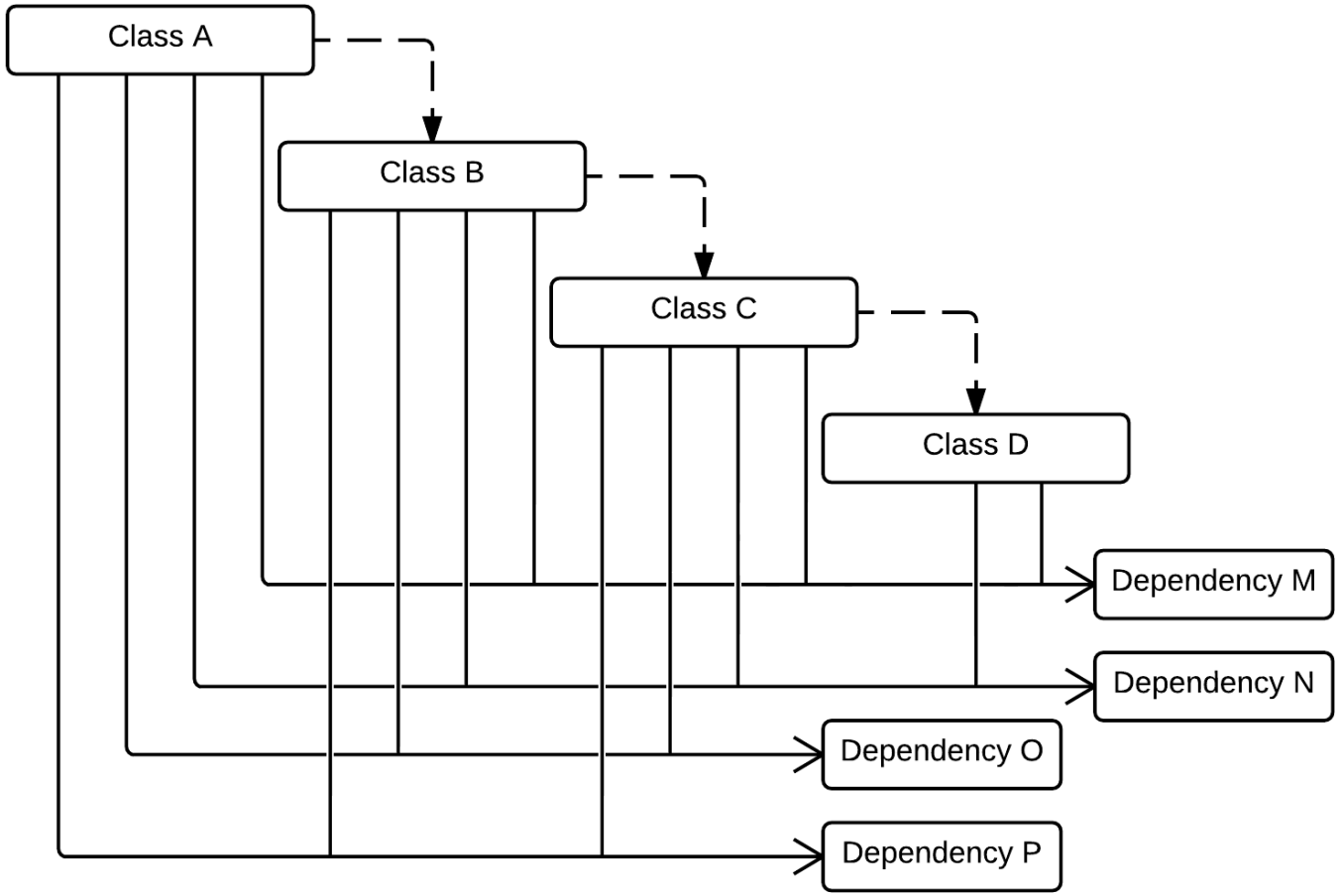
Classes With Dependencies



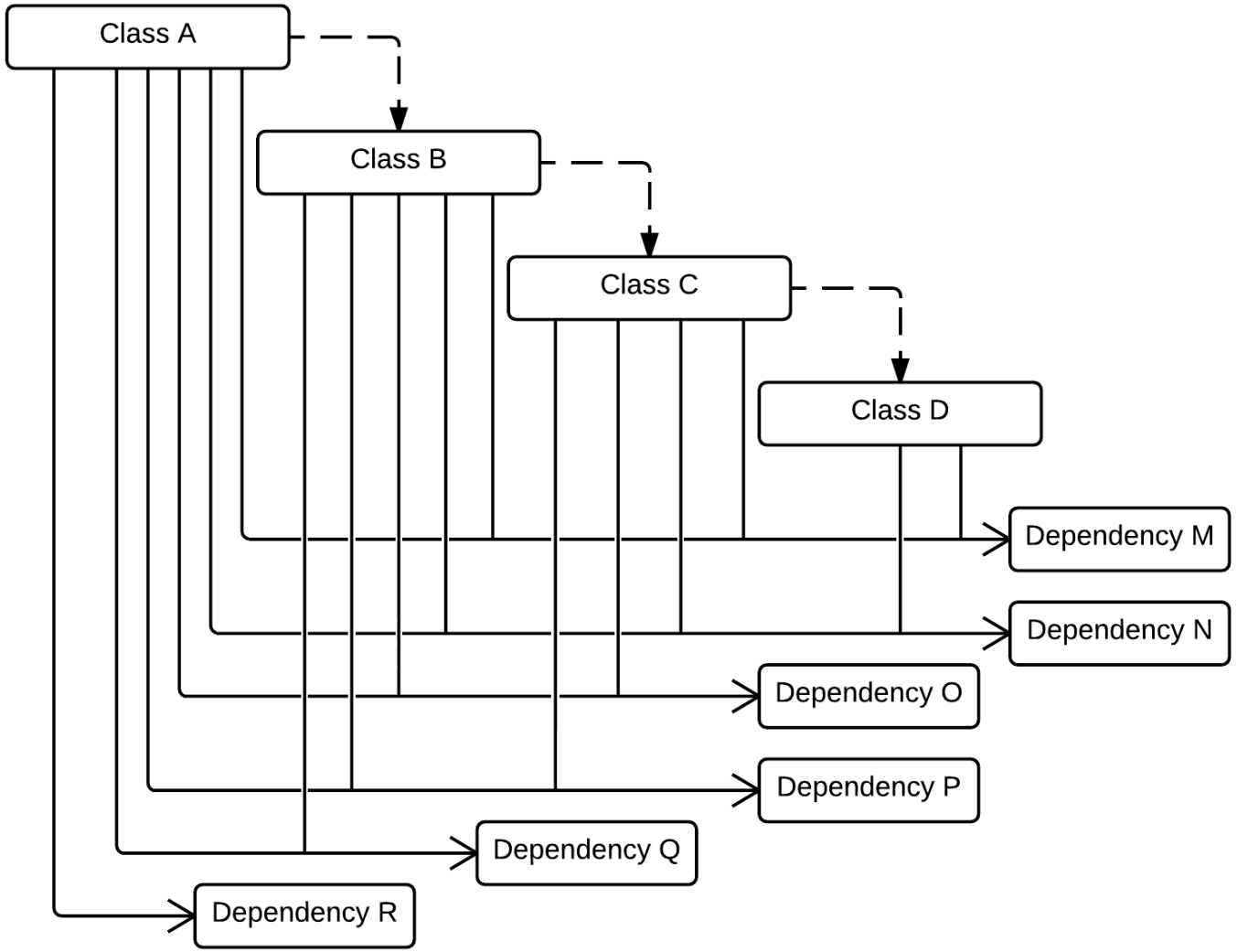
Classes With Dependencies



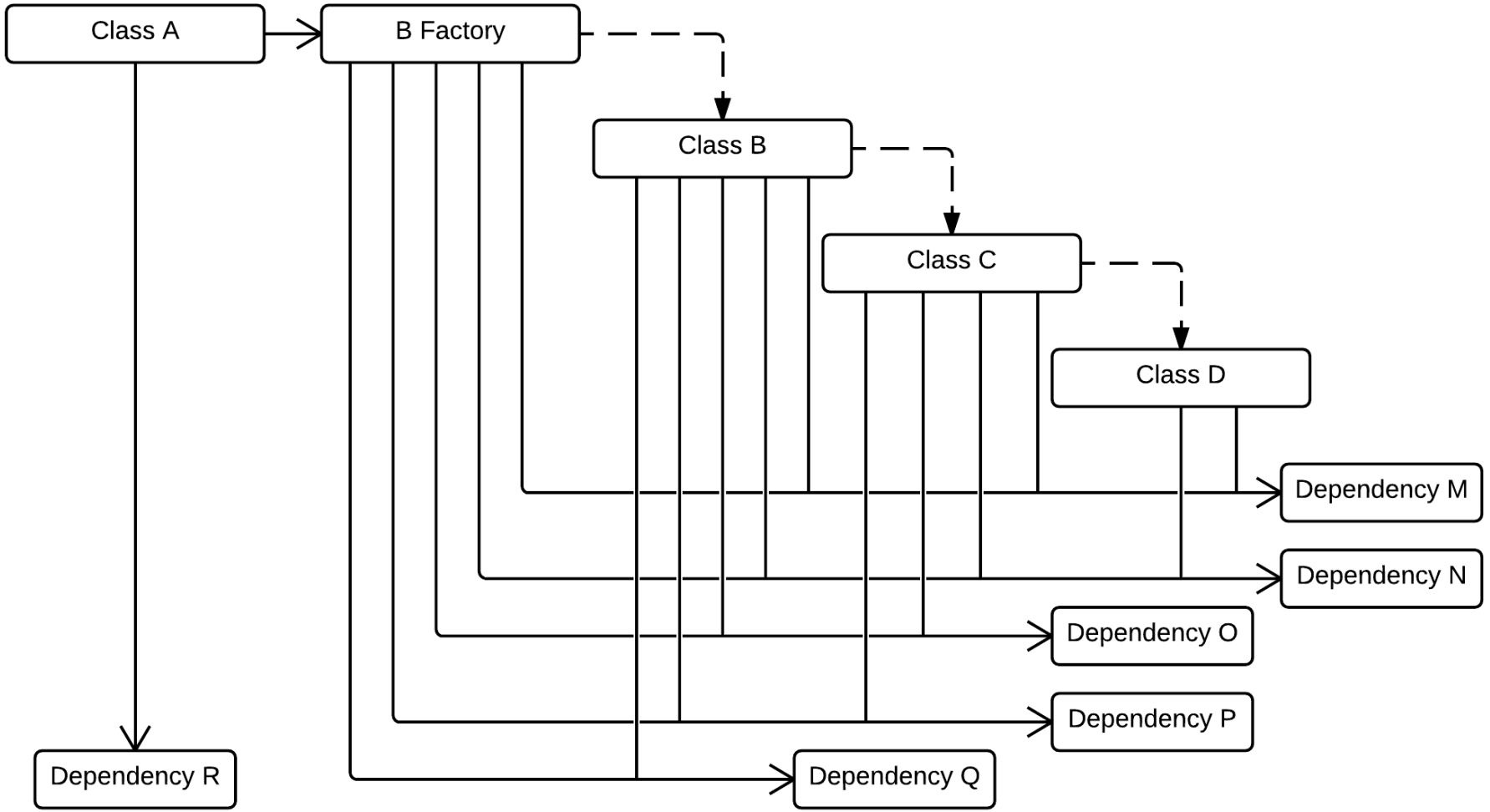
Classes With Dependencies



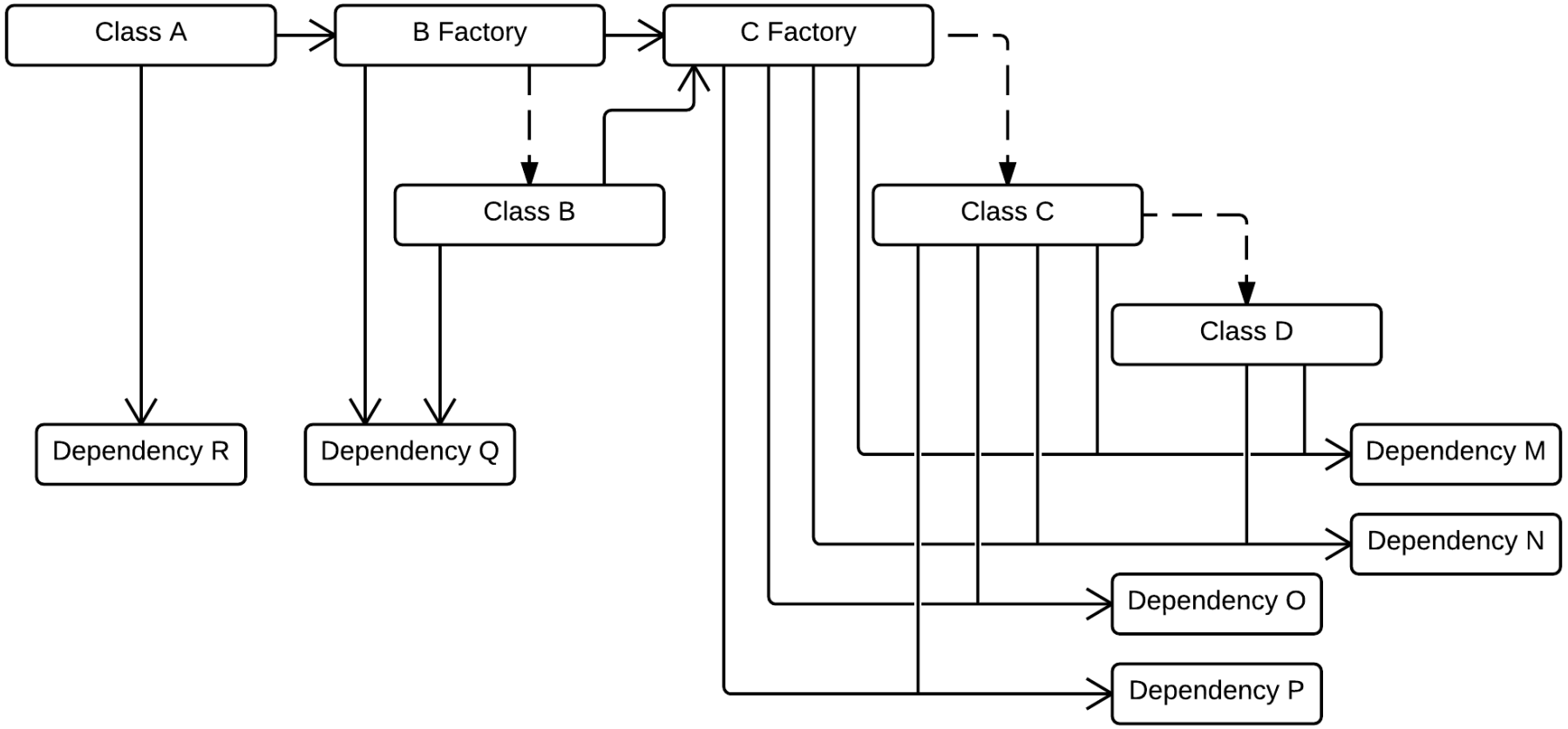
Classes With Dependencies



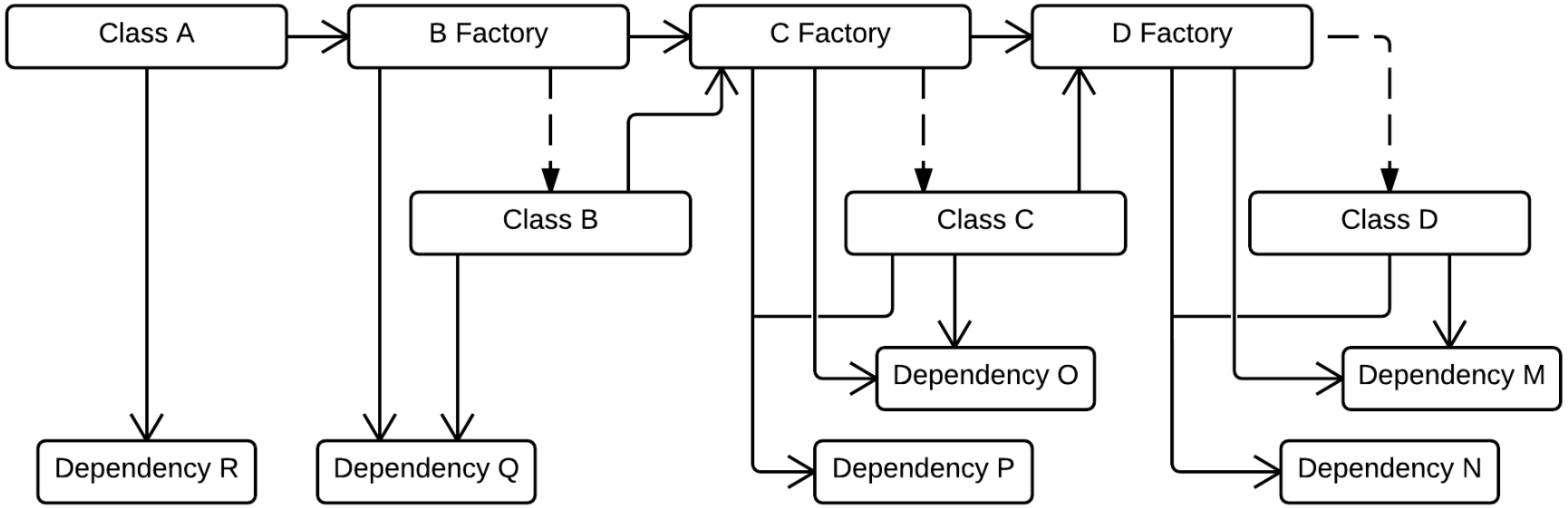
Classes With Dependencies



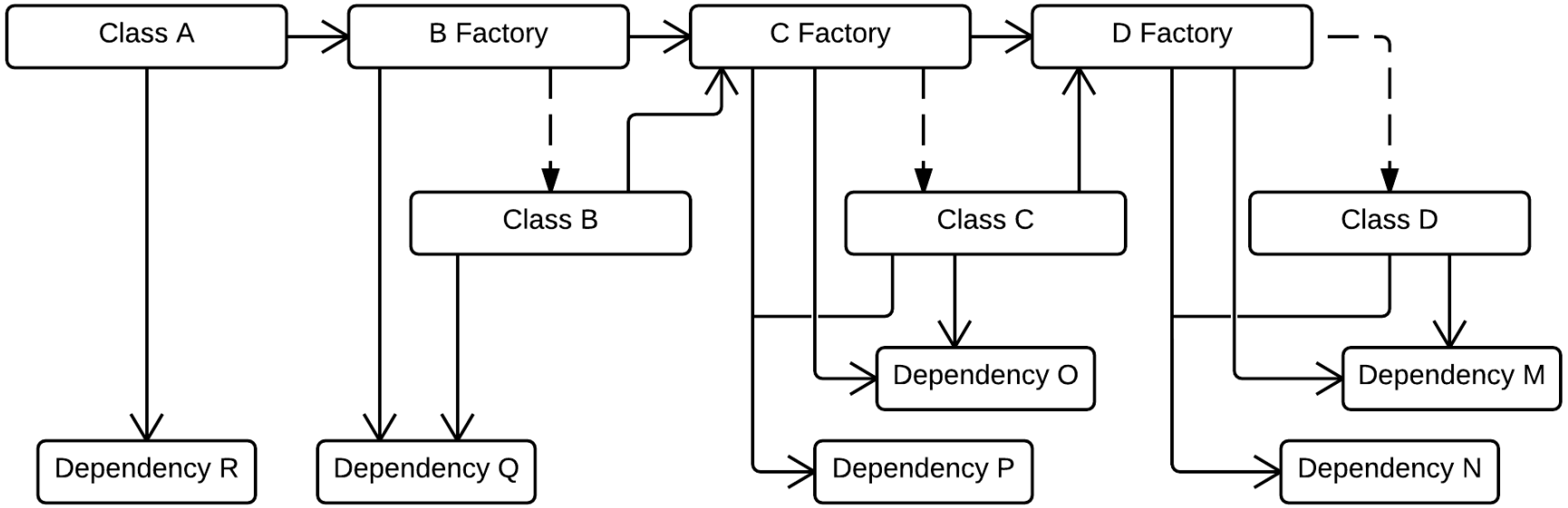
Classes With Dependencies



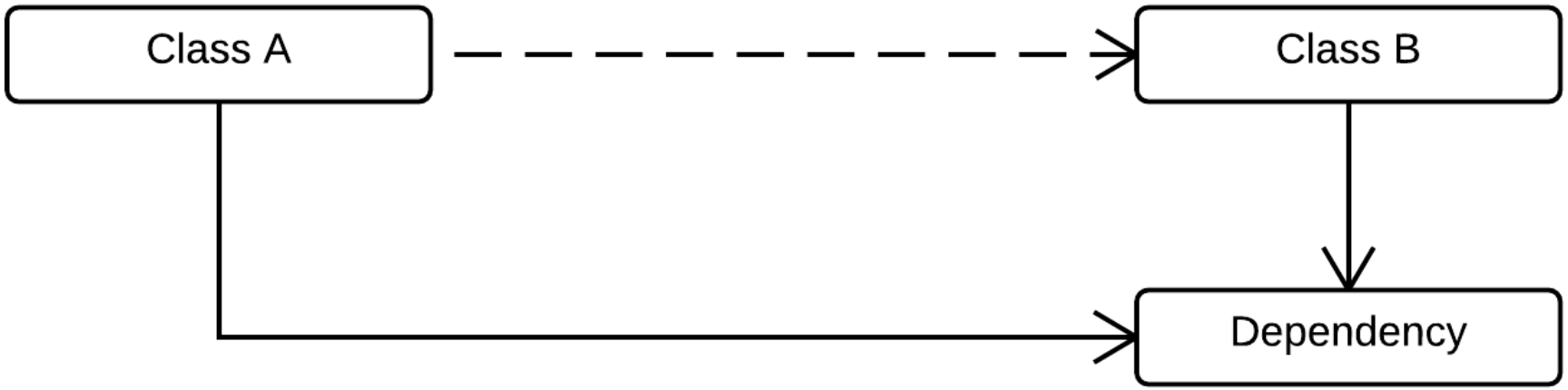
Classes With Dependencies



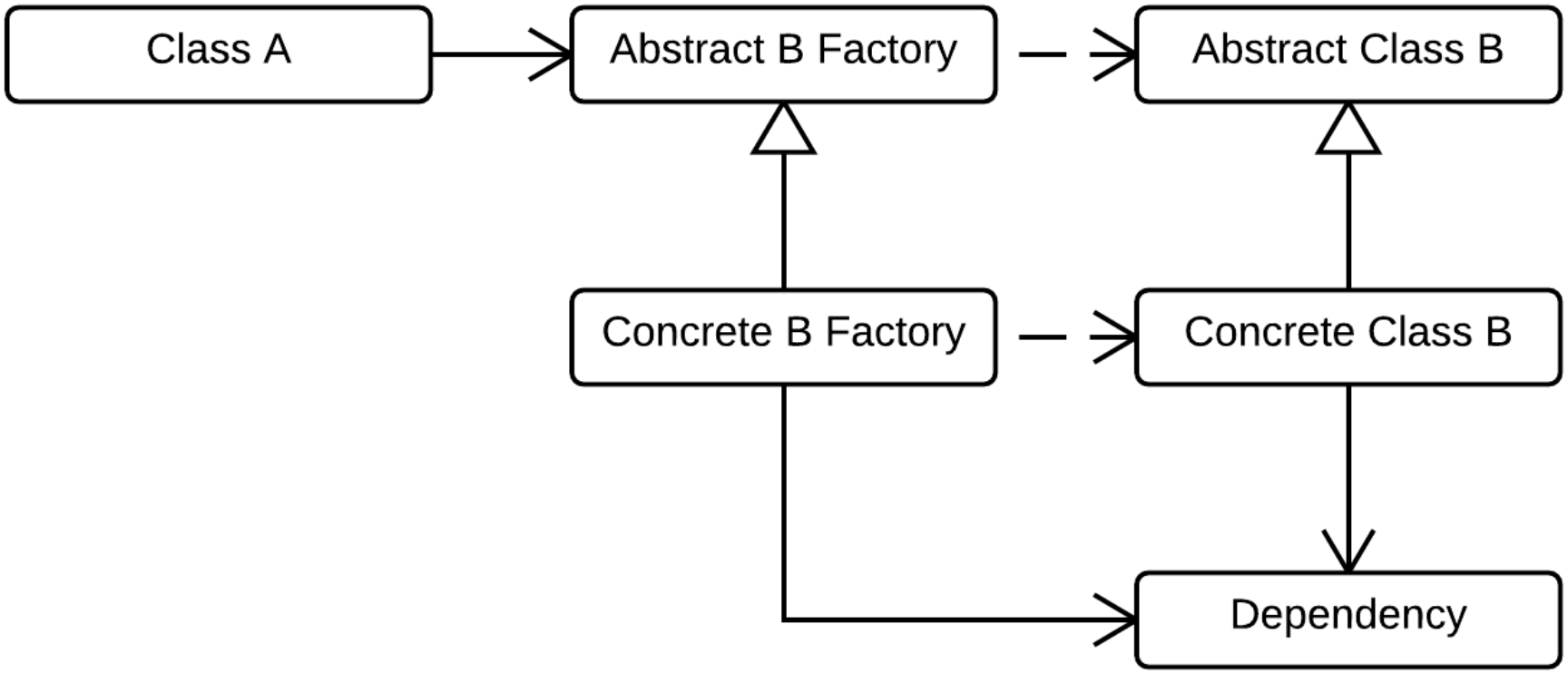
Classes With Dependencies



Removing Dependencies



Removing Dependencies



Summary

- Abstract Factory pattern
  - Decouples the client from concrete product
    - Client is not aware which product will be instantiated by the concrete factory
  - Test double can be the alternative concrete product
    - Abstract Factory may be applied if dependency is going to be faked in testing
- Alternative application of the Abstract Factory pattern
  - Concrete product leaks out dependencies into the client
  - Abstract factory hides dependencies
  - Dependencies are provided by the Inversion of Control (IoC) container