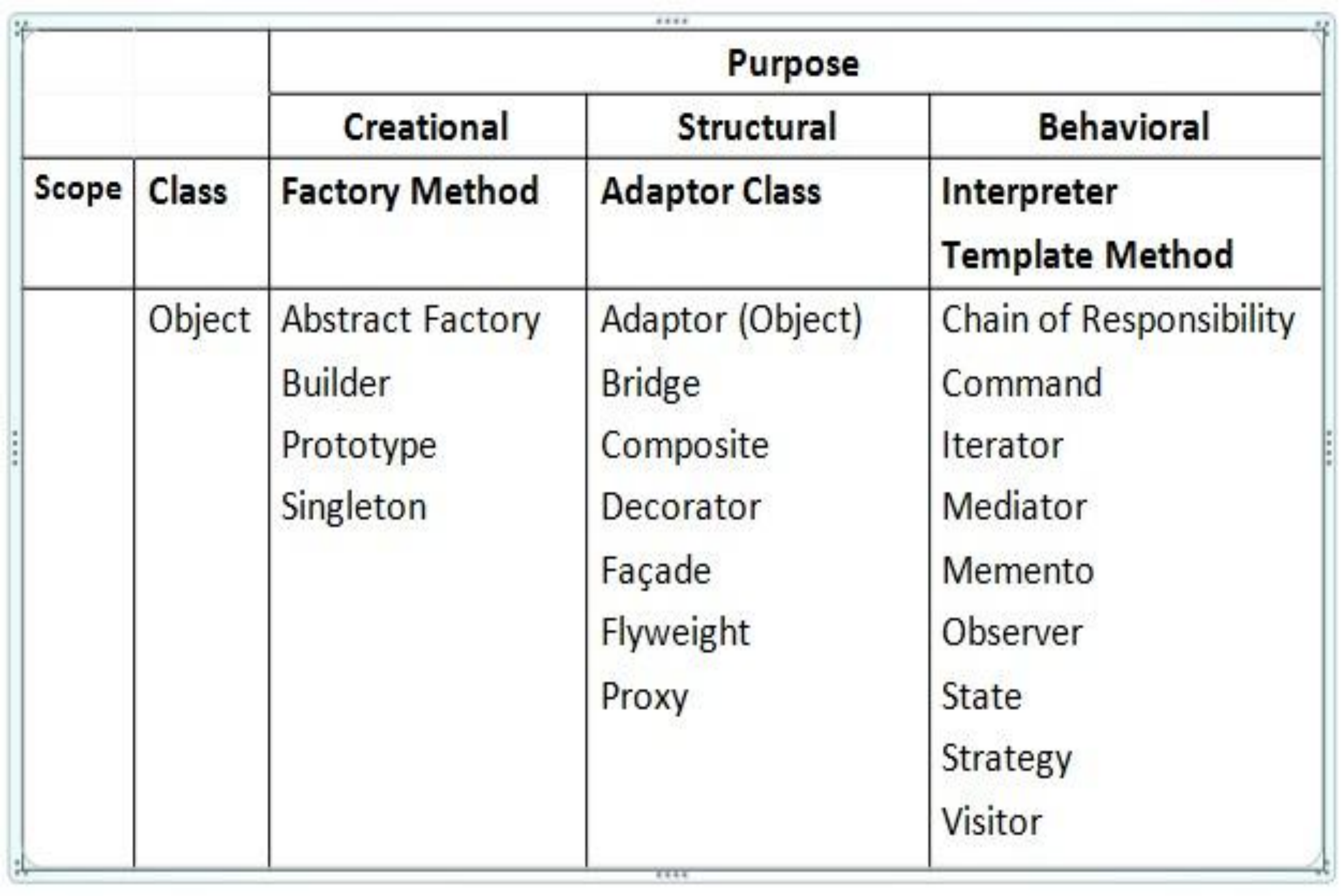
Reuse at Design Level: Design Patterns II

D. K. Yadav

Classification of Patterns

- Creational Patterns
  - concerned about ways to create new objects
- Structural Patterns
  - concerned about the composition of objects and classes
- Behavioral Patterns
  - concerned about ways in which objects interact

Classification of Patterns (Cont..)



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

Creational Patterns (Cont..)

- Singleton
  - To create a sole instance of a class
- Prototype
  - To create objects by cloning existing objects
- Builder
  - Build an object from existing representation

Creational Patterns (Cont..)

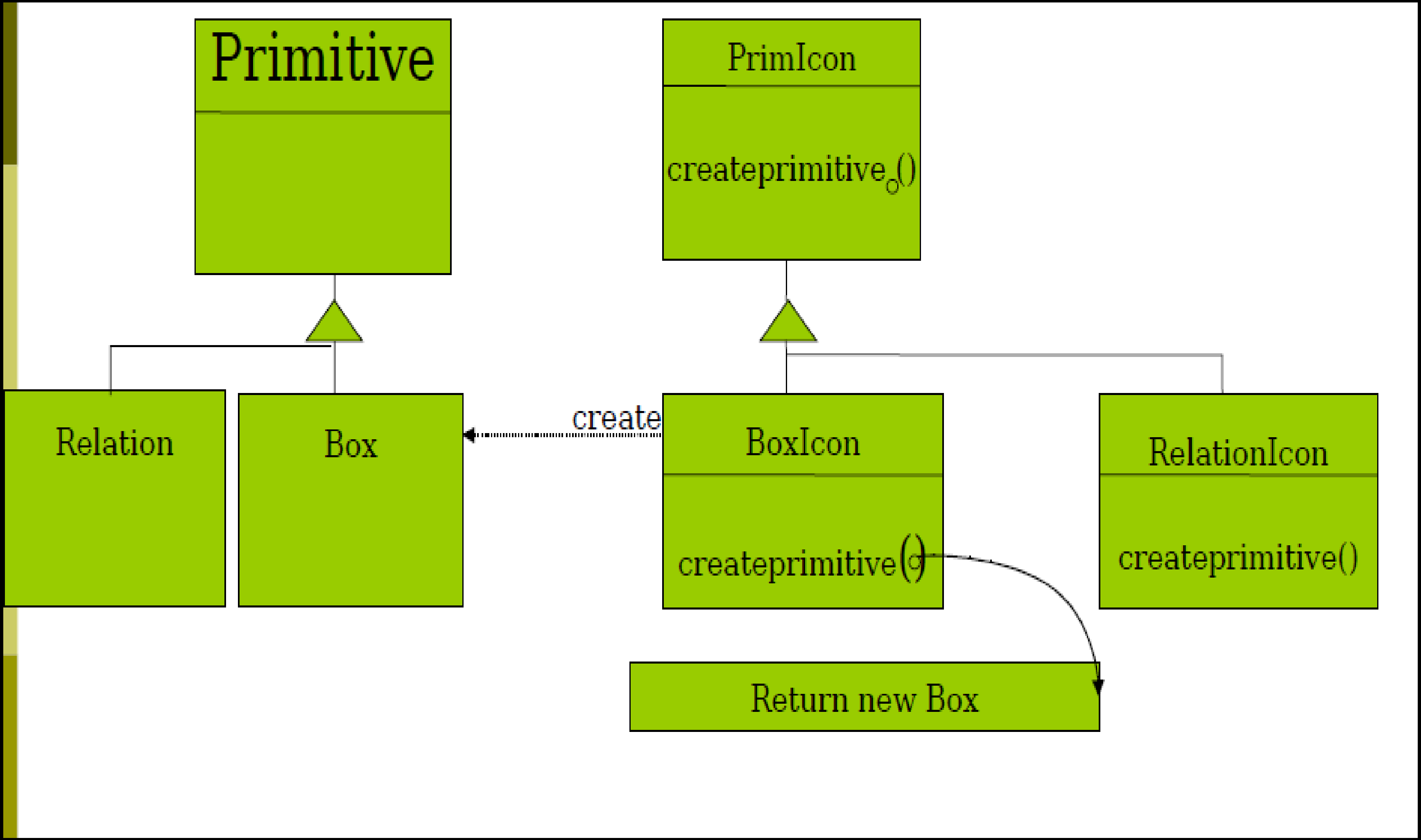
- Factory Method
  - Defer instantiation to subclasses

- Abstract Factory
  - Provides interface to create families of objects without specifying the concrete classes of the objects

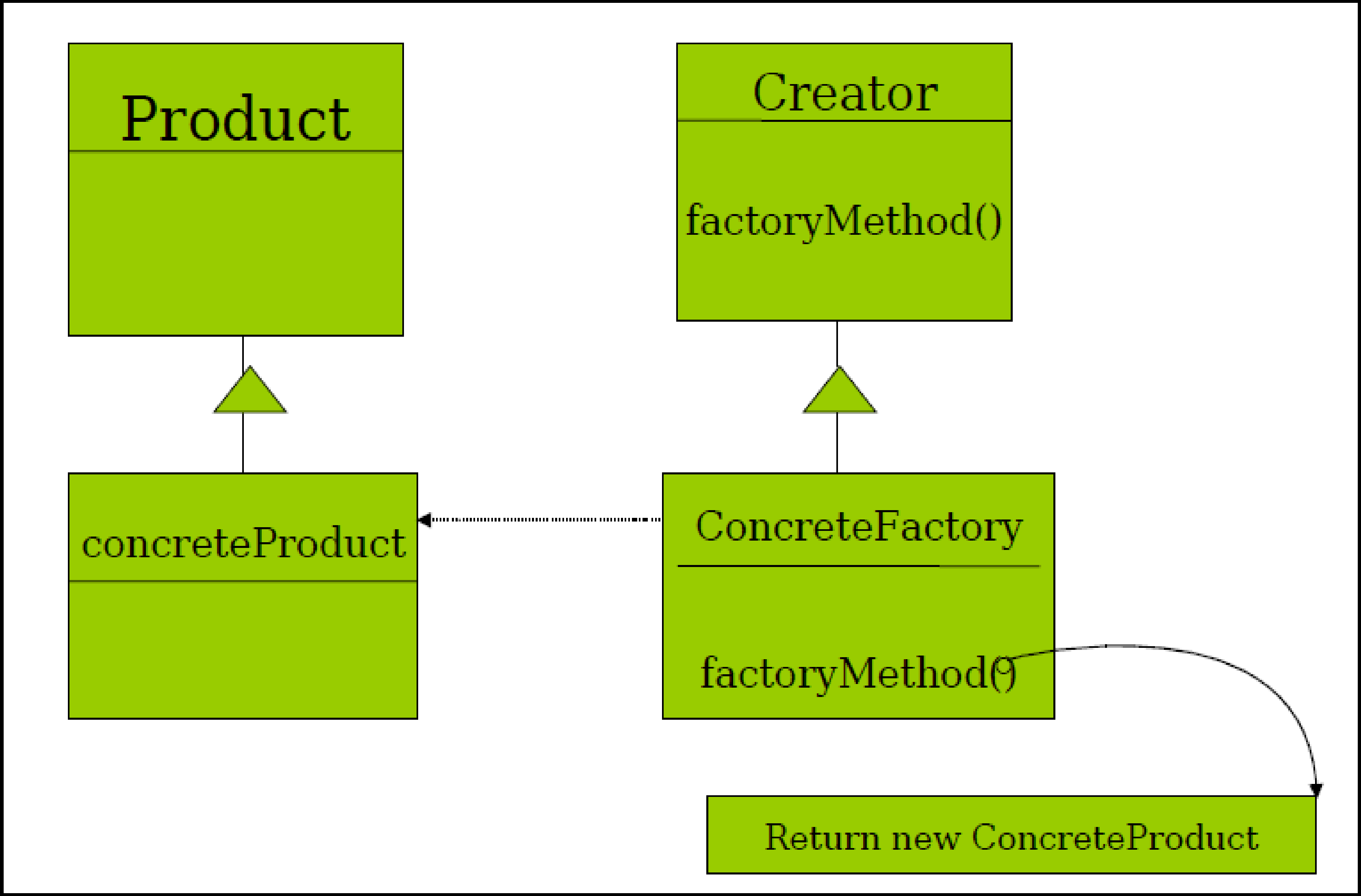
Factory Method

- Example : In a modeling tool, you can select a type of entity or relation and then create an actual entity of the selected type. The selected icon thus acts
- As a factory or creator of some actual type.
- An object is used to create another
- One creator object creates instances of one class Hierarchy of creators, polymorphism on creation

An Example of Factory Method



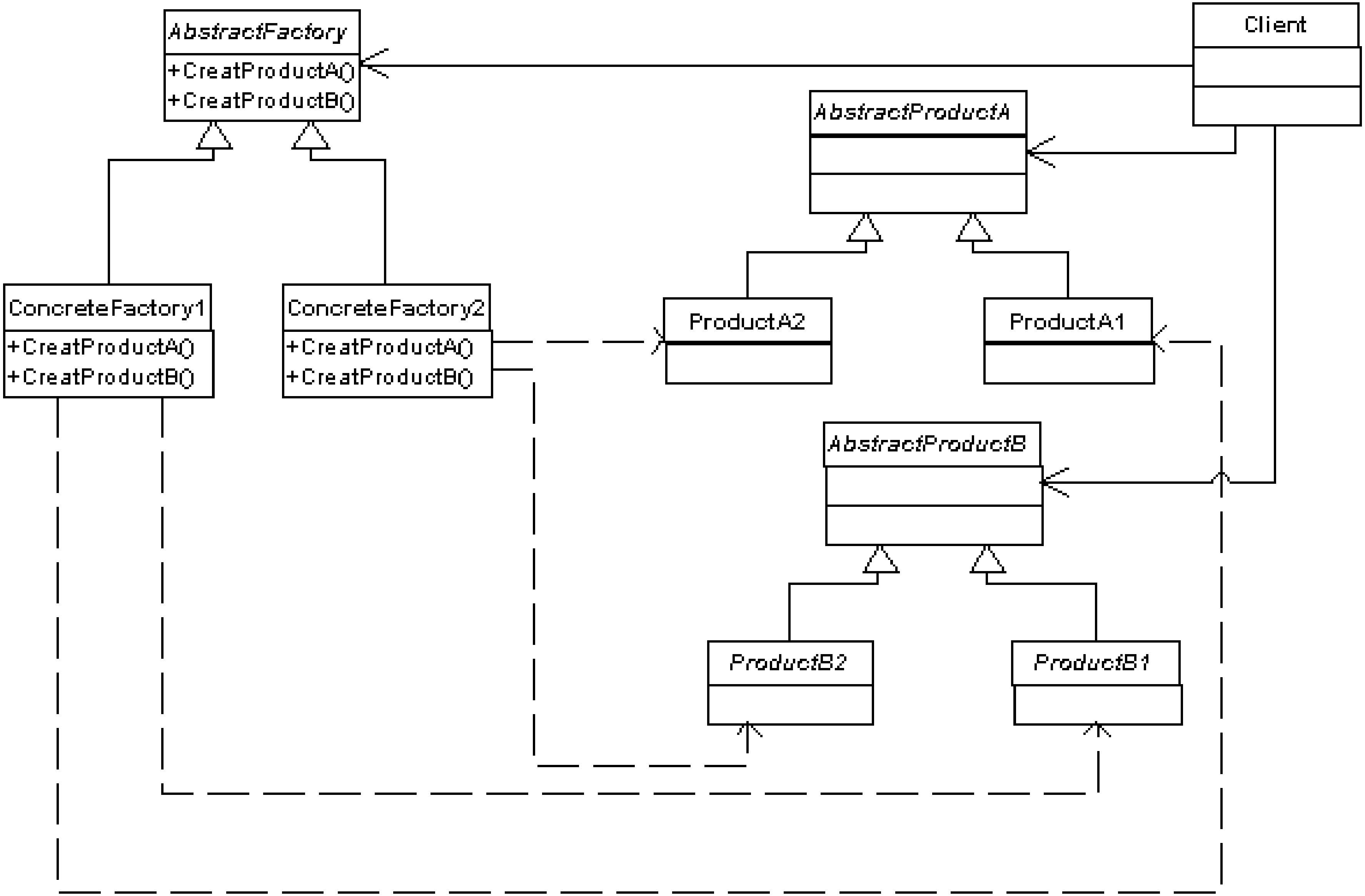
The Factory Method Pattern



The Abstract Factory Pattern

- “Provide an interface for creating families of related or dependent objects without specifying their concrete classes.”
  - provide a simple creational interface for a complex family of classes
    - Client does not have to know any of those details.
  - avoid naming concrete classes
    - Clients use abstract creational interfaces and abstract product interfaces. Concrete classes can be changed without affecting clients.

The Abstract Factory Pattern



AbstractFactory
+CreatProductA()
+CreatProductB()
Client
AbstractProductA
ConcreteFactory1
+CreatProductA()
+CreatProductB()
ConcreteFactory2
+CreatProductA()
+CreatProductB()
ProductA2
ProductA1
AbstractProductB
ProductB2
ProductB1