Abstract Factory Pattern

Shin-Jie Lee (李信杰)
Associate Professor
Computer and Network Center
Department of CSIE
National Cheng Kung University

Design Aspect of Abstract Factory

Families of product objects

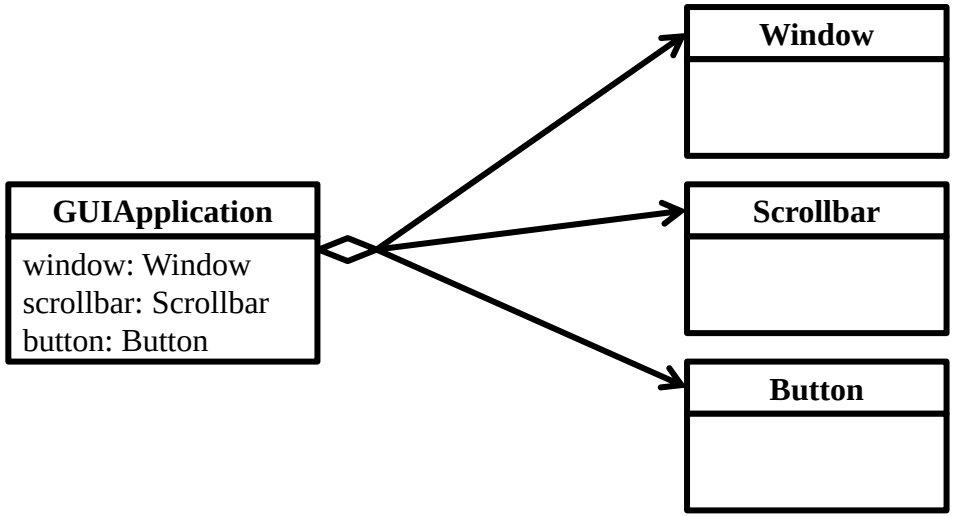
Outline

- Requirements Statement
- Initial Design and Its Problems
- Design Process
- Refactored Design after Design Process
- More Examples
- Recurrent Problems
- Intent
- Abstract Factory Pattern Structure
- Abstract Factory vs. Factory Method

A GUI Application with Multiple Styles (Abstract Factory)

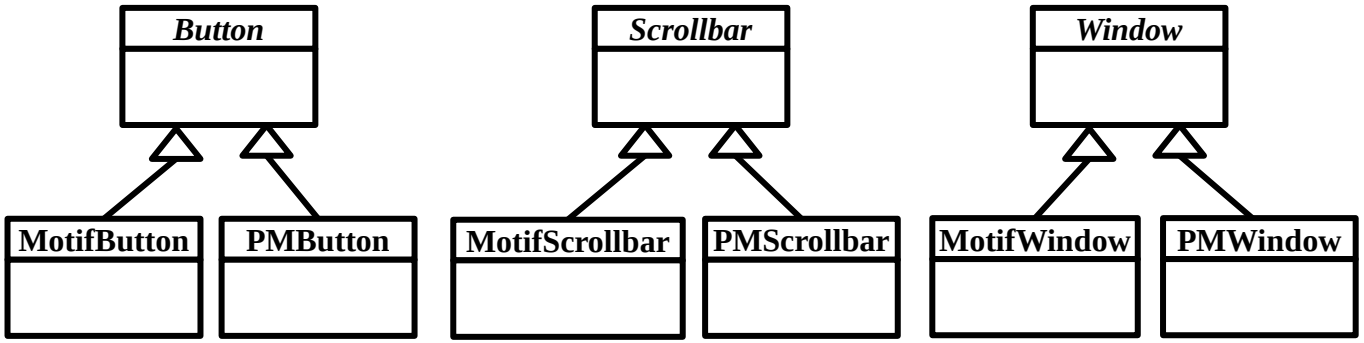
Requirements Statement$_1$

- A GUI Application consists of some kinds of widgets like window, scroll bar, and button.



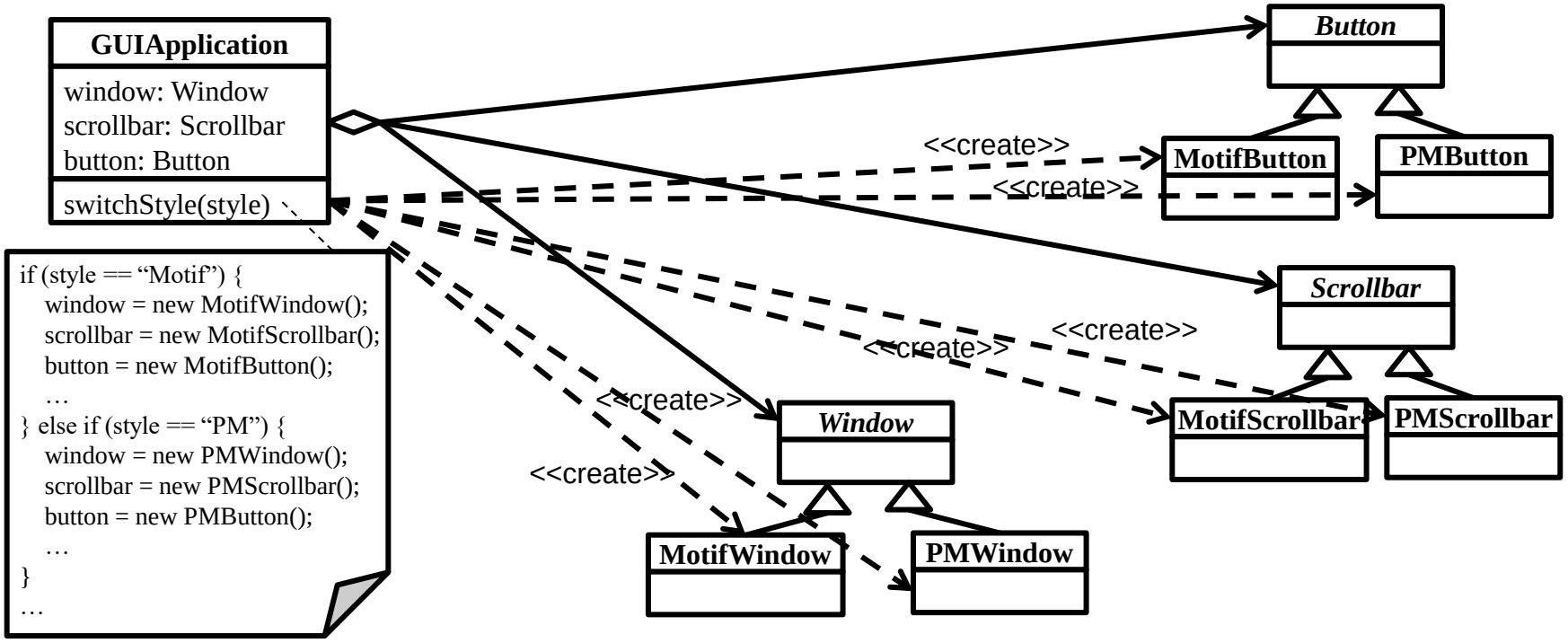
Requirements Statement$_2$

- Each widget in the GUI application has two or more implementations according to different look-and-feel standards, such as Motif and Presentation Manager.



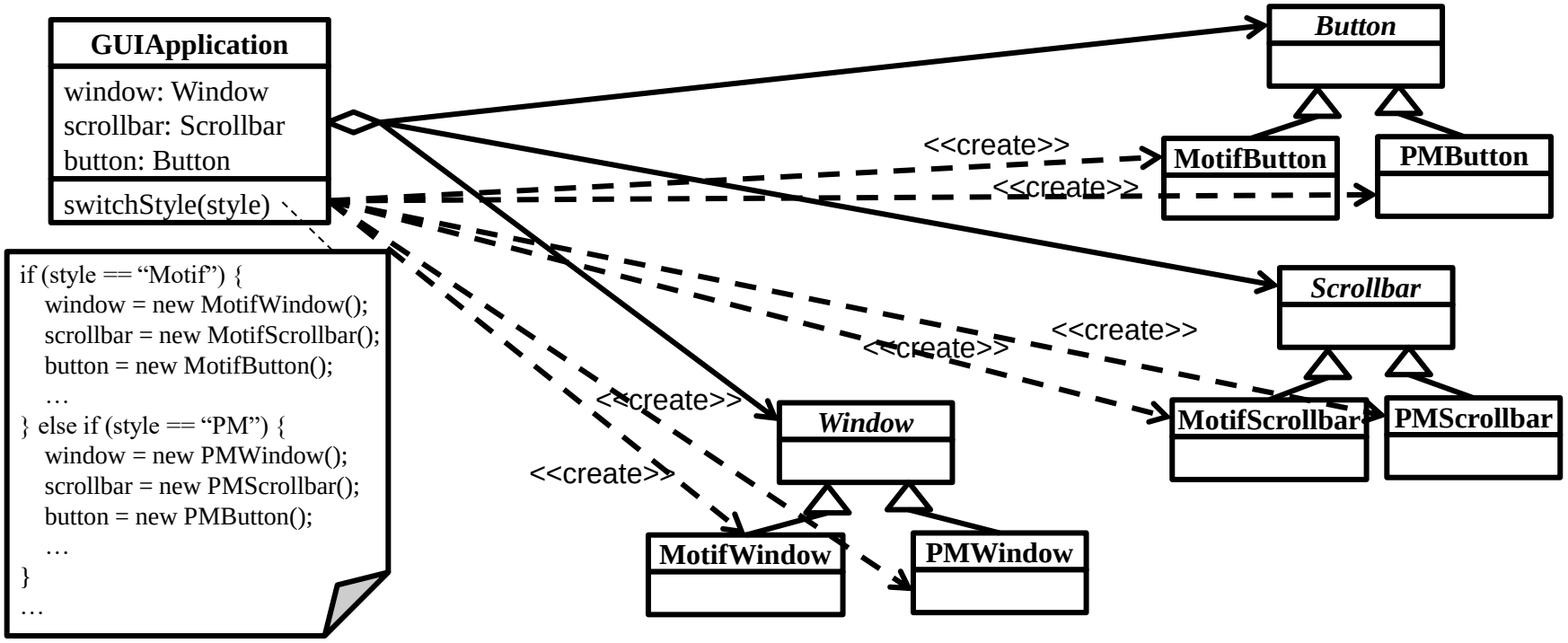
Requirements Statement$_3$

- The GUI application can switch its look-and-feel style from one to another.

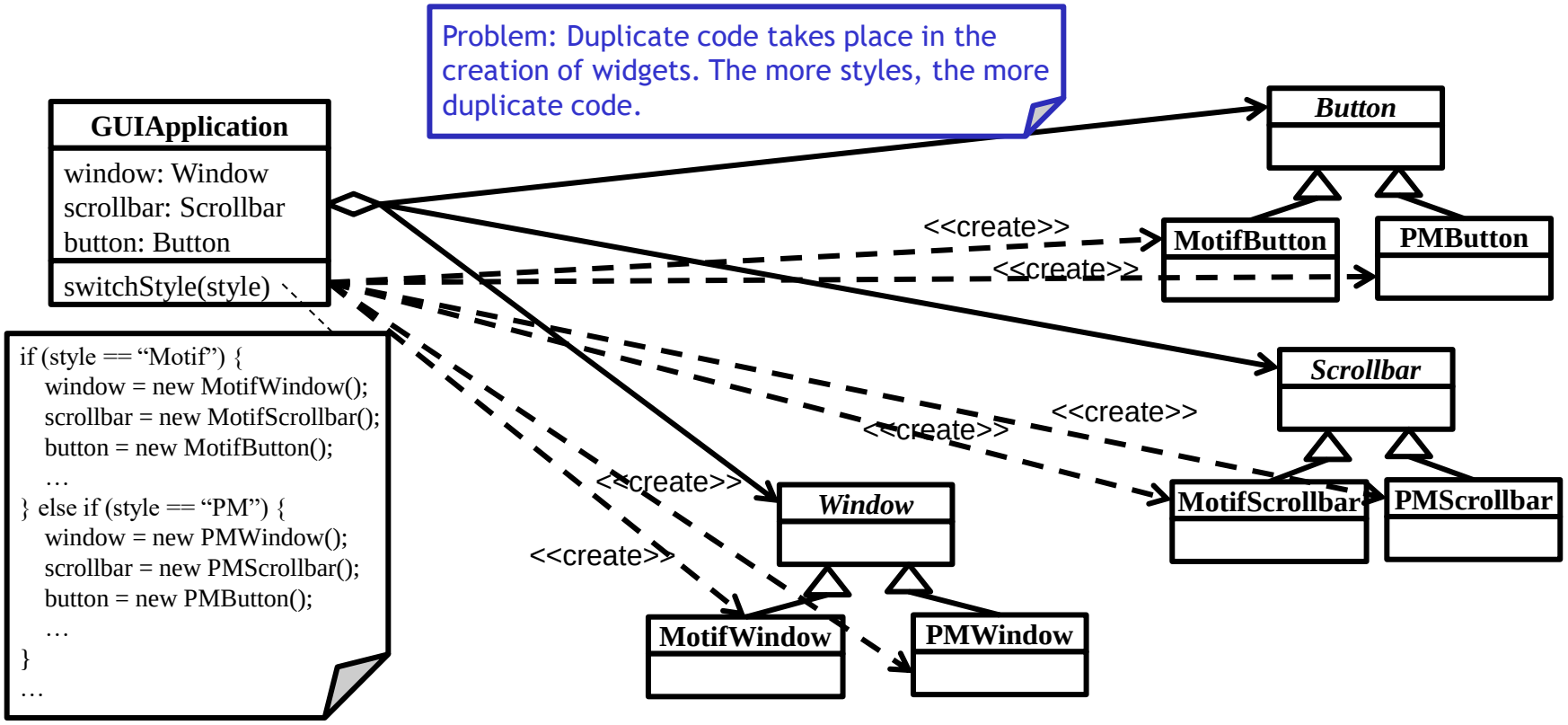


Initial Design

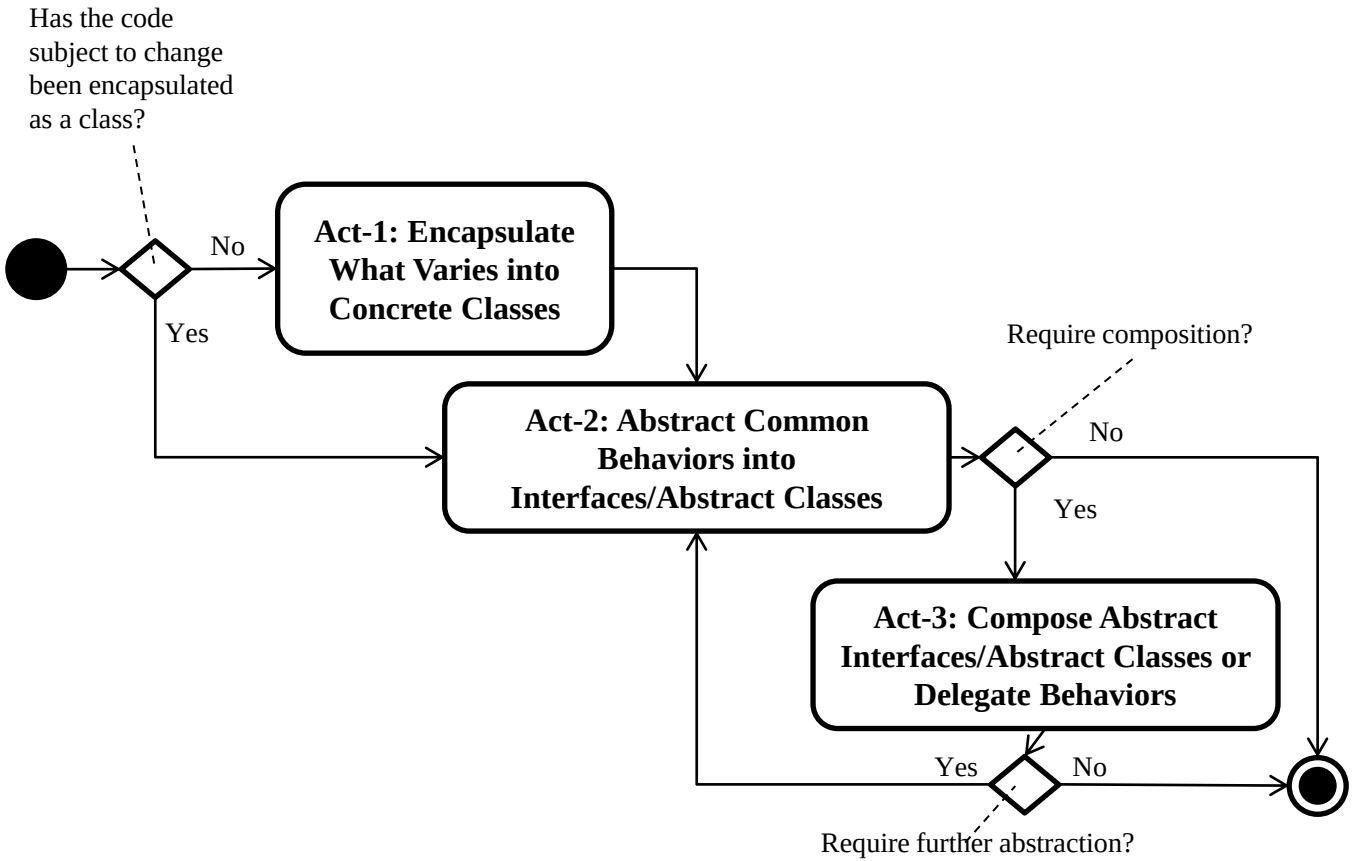
- The GUI application can switch its look-and-feel style from one to another.



Problems with Initial Design

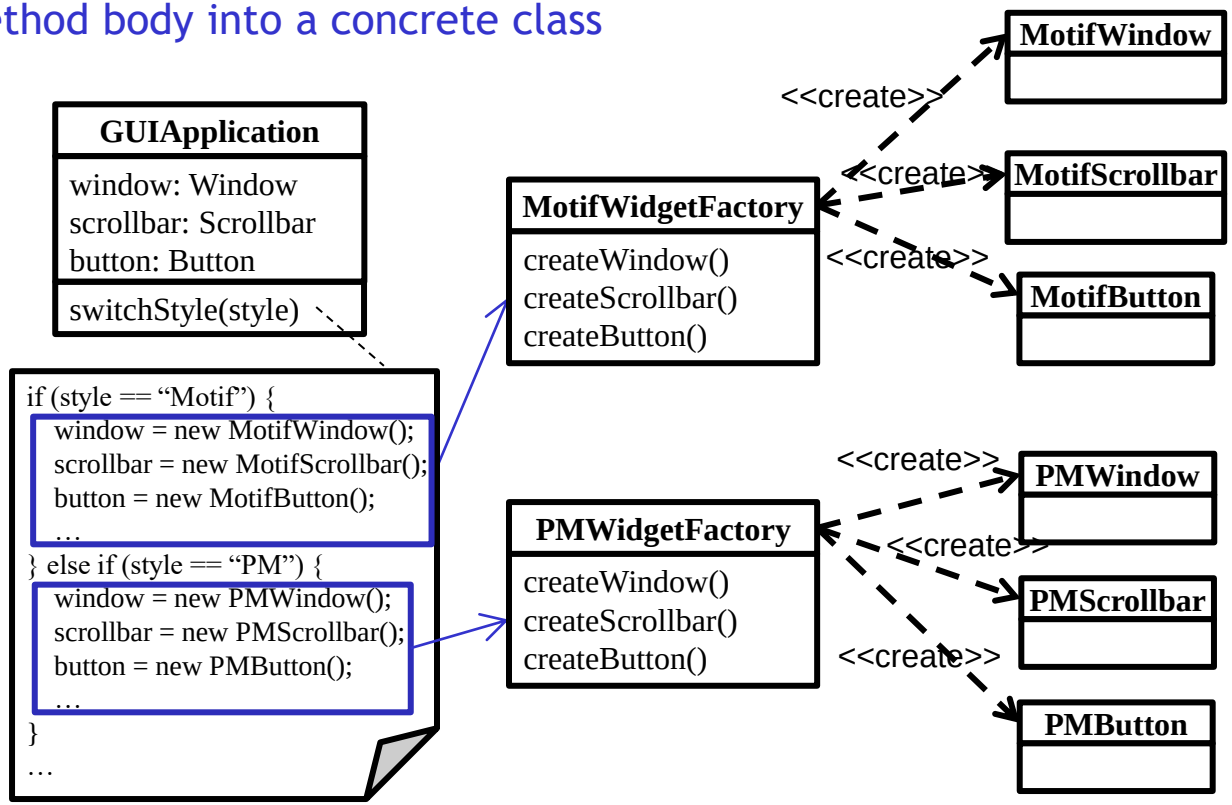


Design Process for Change



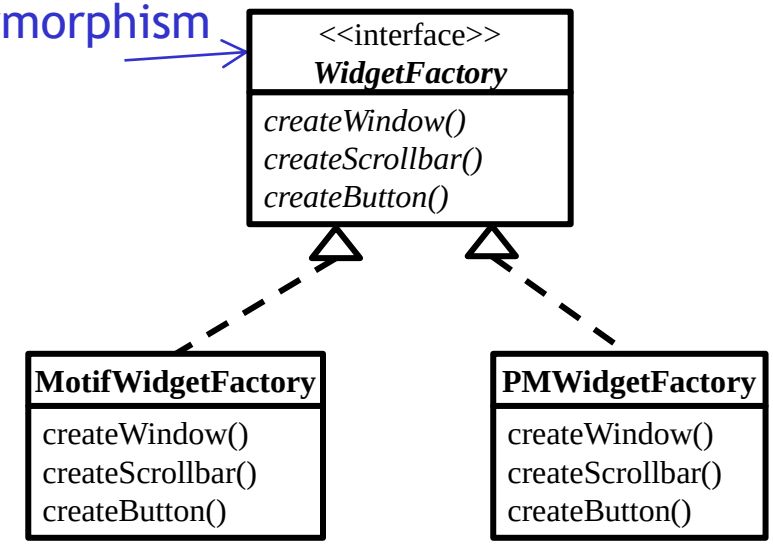
Act-1: Encapsulate What Varies

Act-1.3: Encapsulate a part of a method body into a concrete class

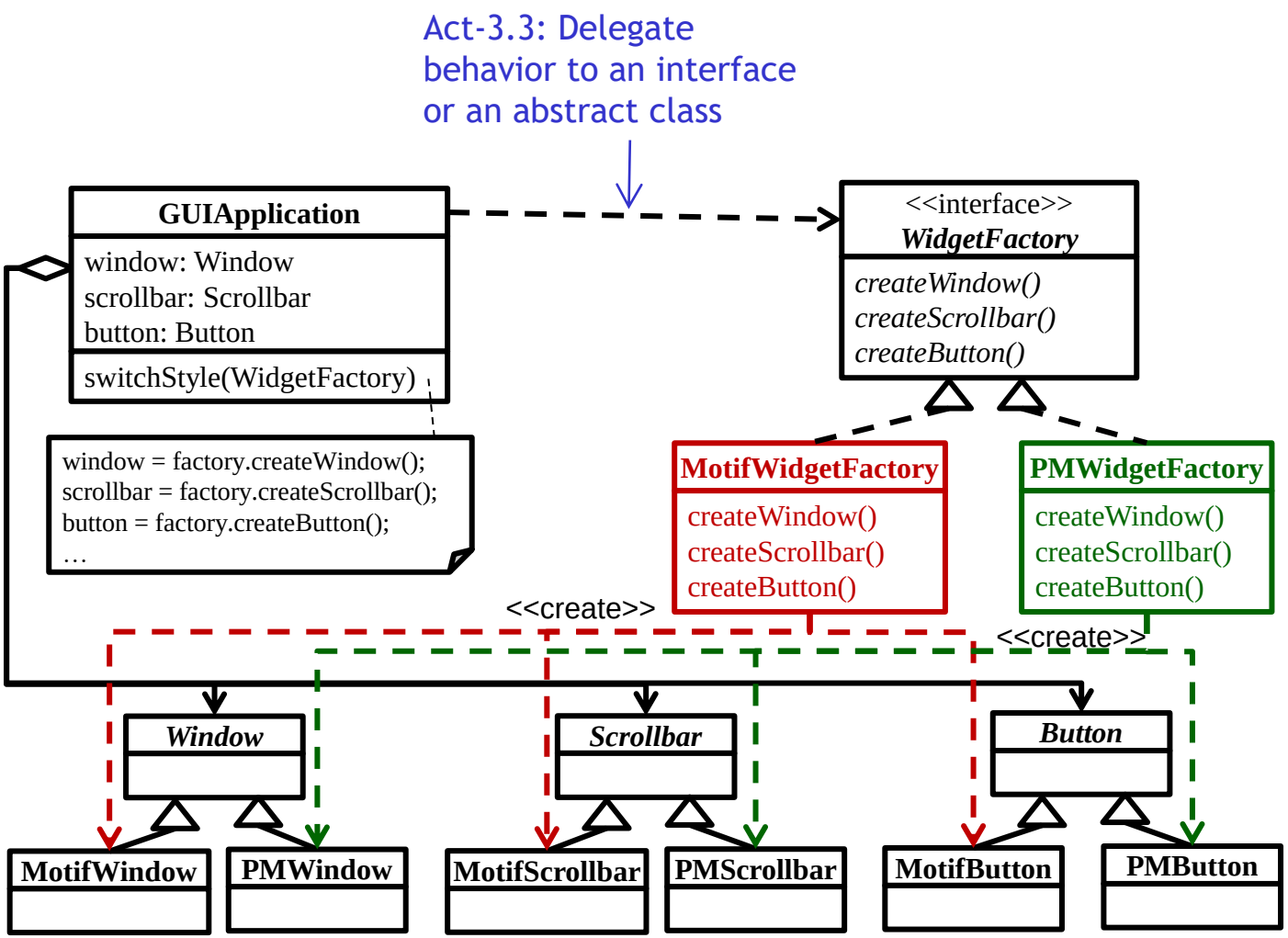


Act-2: Abstract Common Behaviors

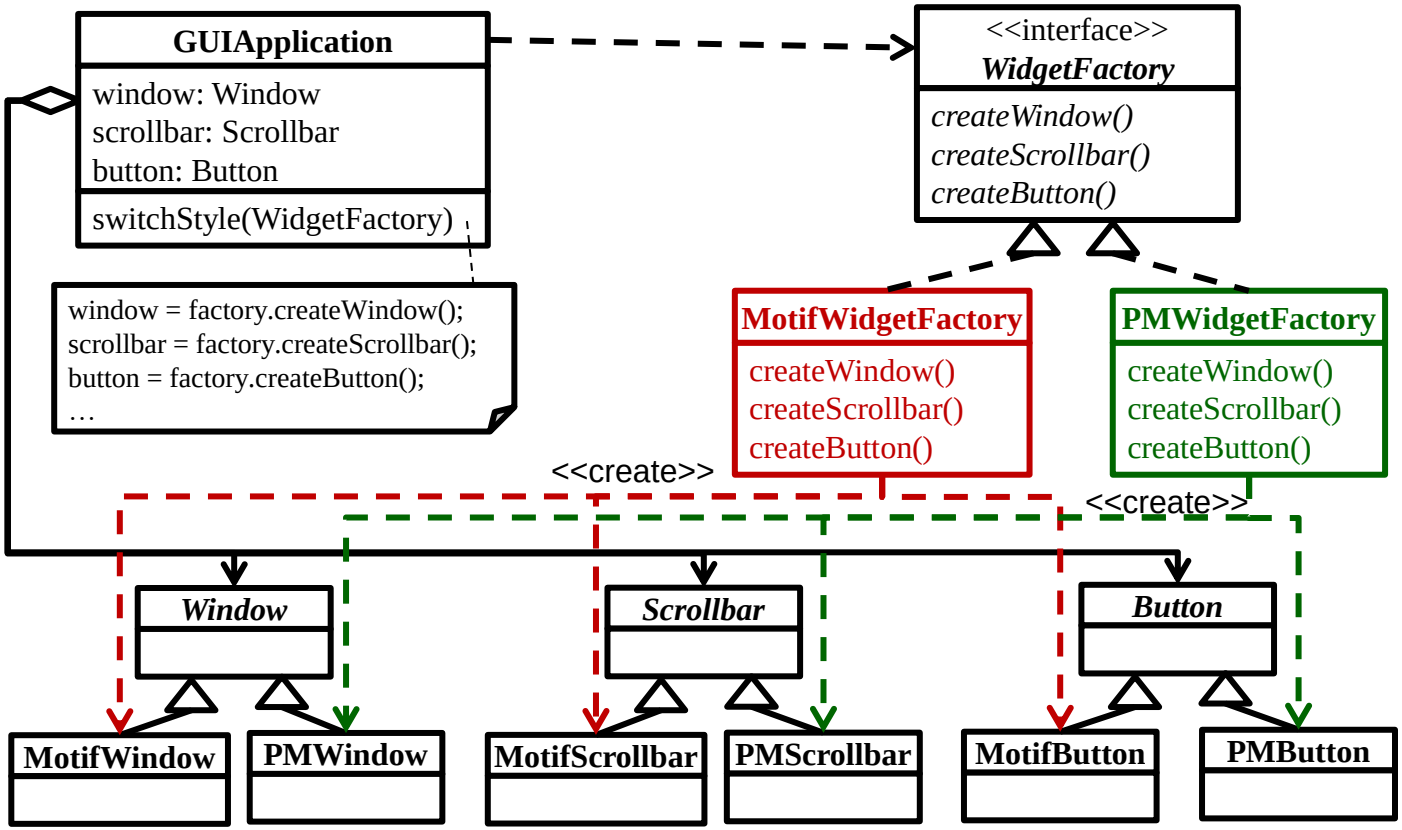
Act-2.1: Abstract common behaviors with a same signature into interface through polymorphism



Act-3: Compose Abstract Behaviors



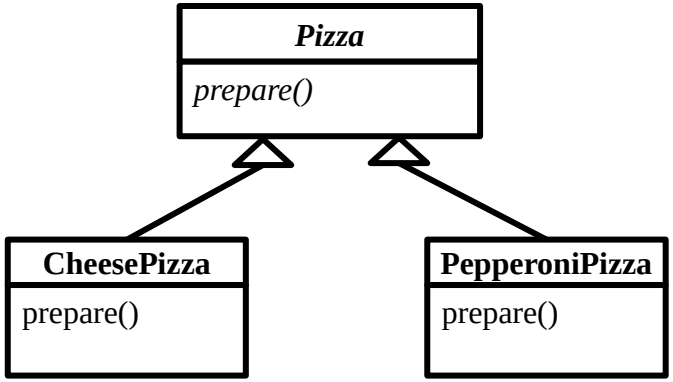
Refactored Design after Design Process



Pizza Store (Extended)

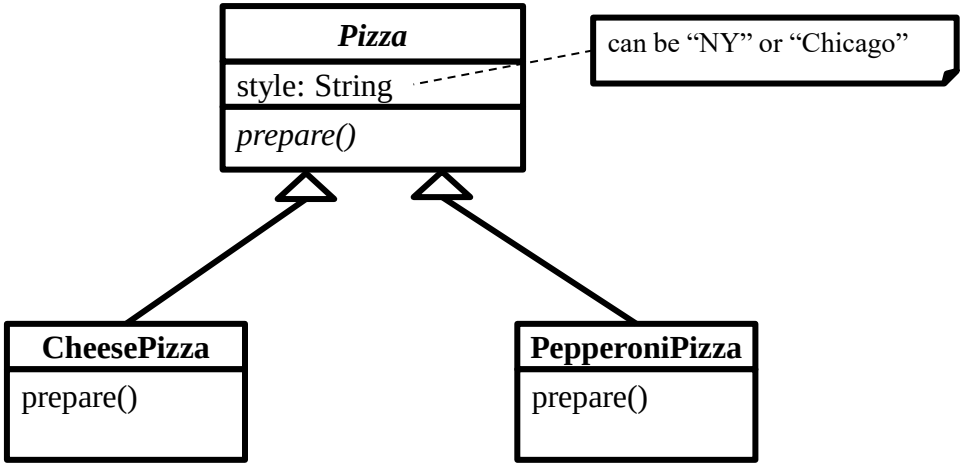
Requirements Statement$_1$

- In a pizza store system, two flavors of pizza are offered: Cheese Pizza and Pepperoni Pizza.



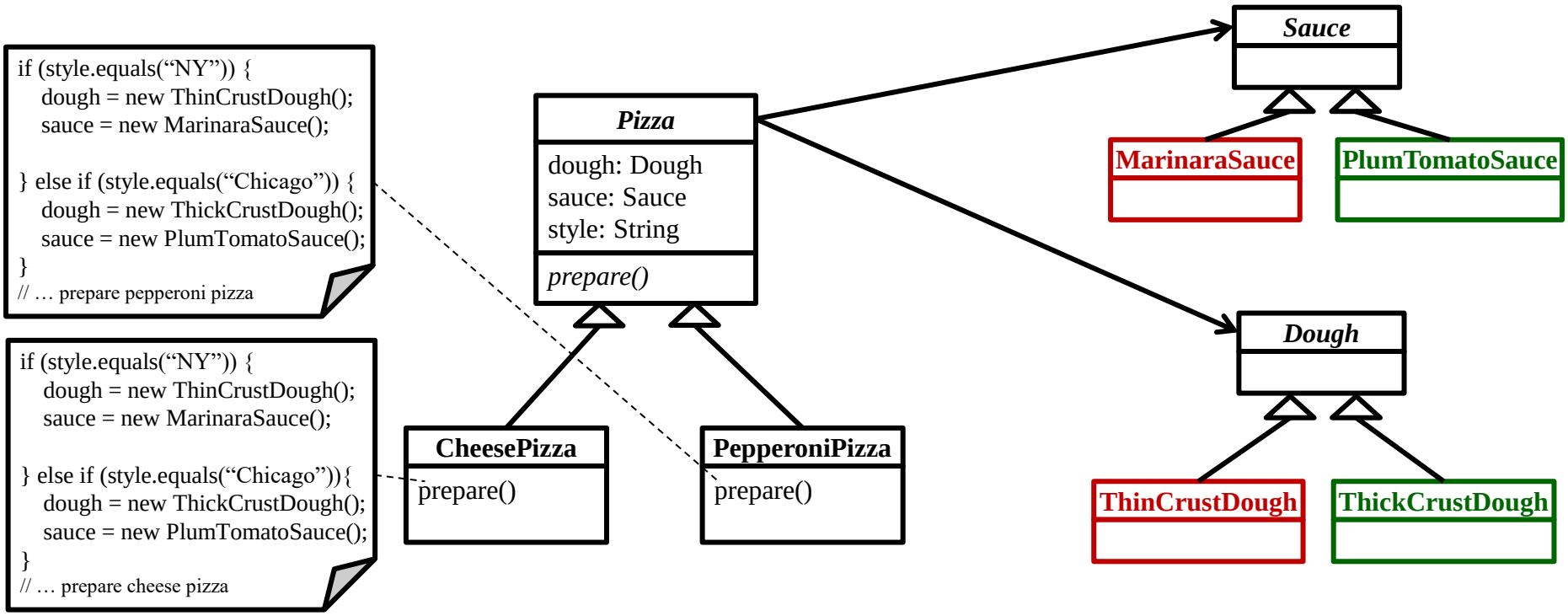
Requirements Statement$_2$

- Each flavor of pizza can be categorized into two styles: New York Style and Chicago Style.

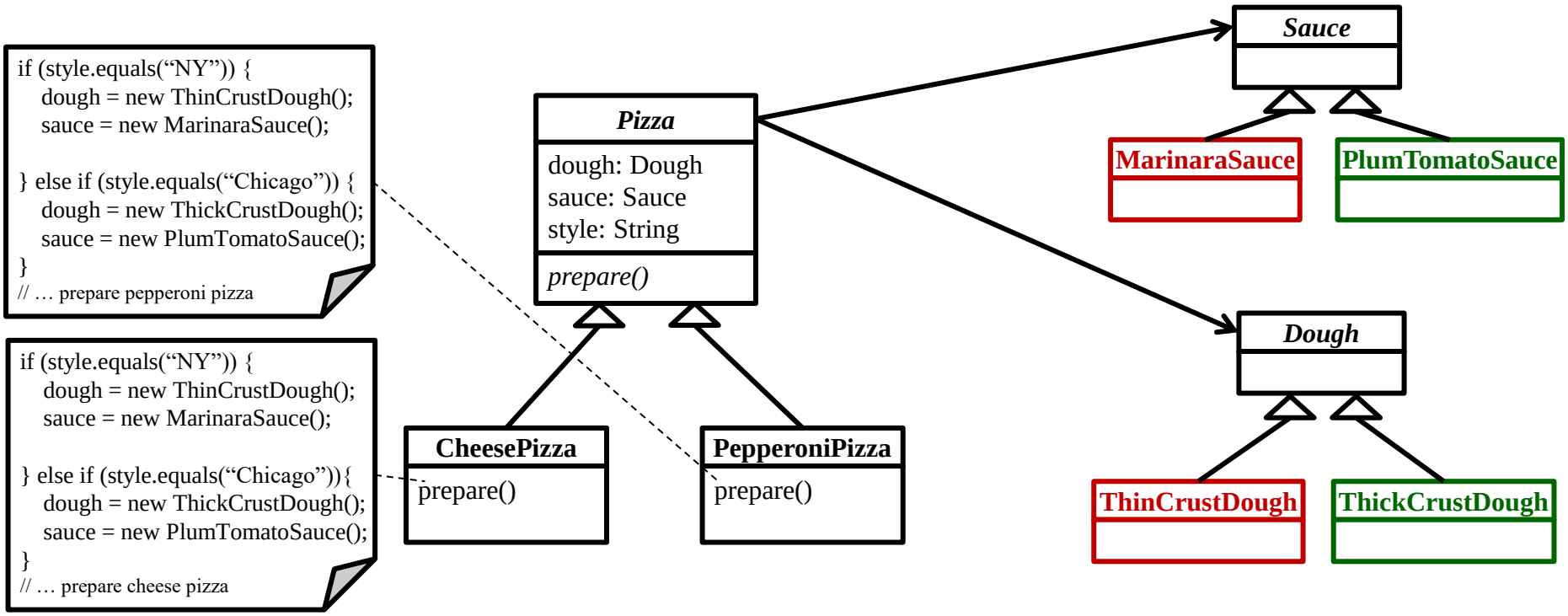


Requirements Statement$_3$

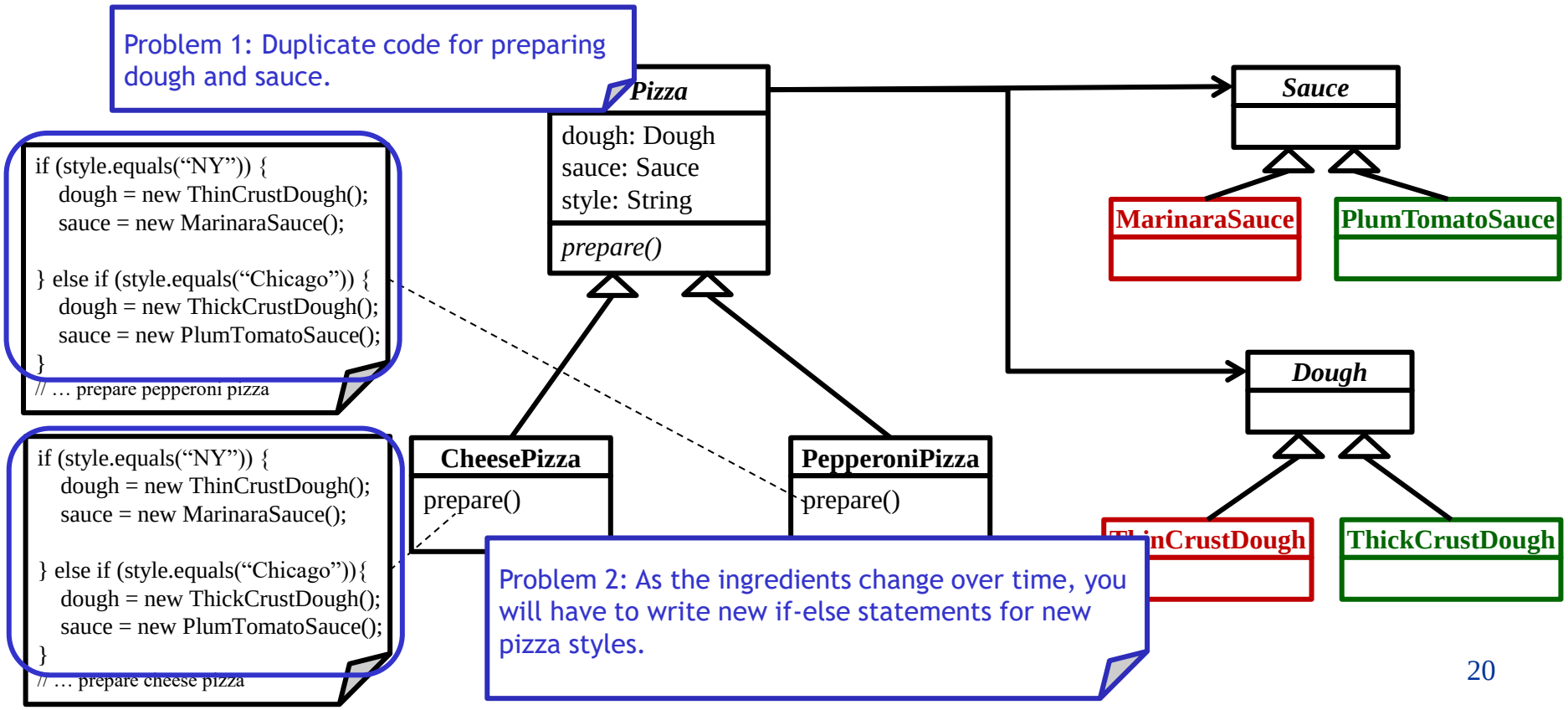
- Different pizza style needs different dough and sauce:
  - NY Style: Thin Crust Dough, Marinara Sauce
  - Chicago Style: Thick Crust Dough, Plum Tomato Sauce



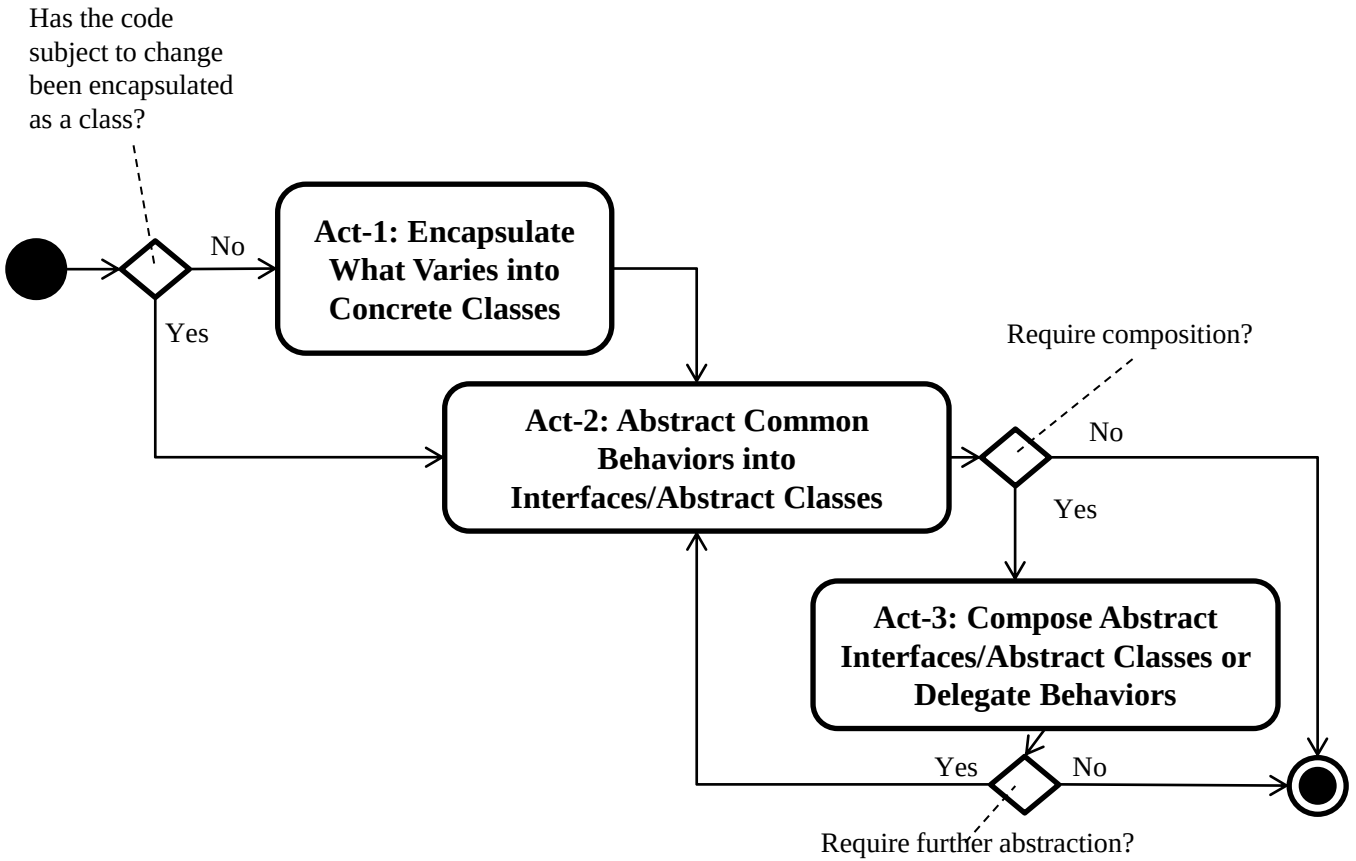
Initial Design - Class Diagram



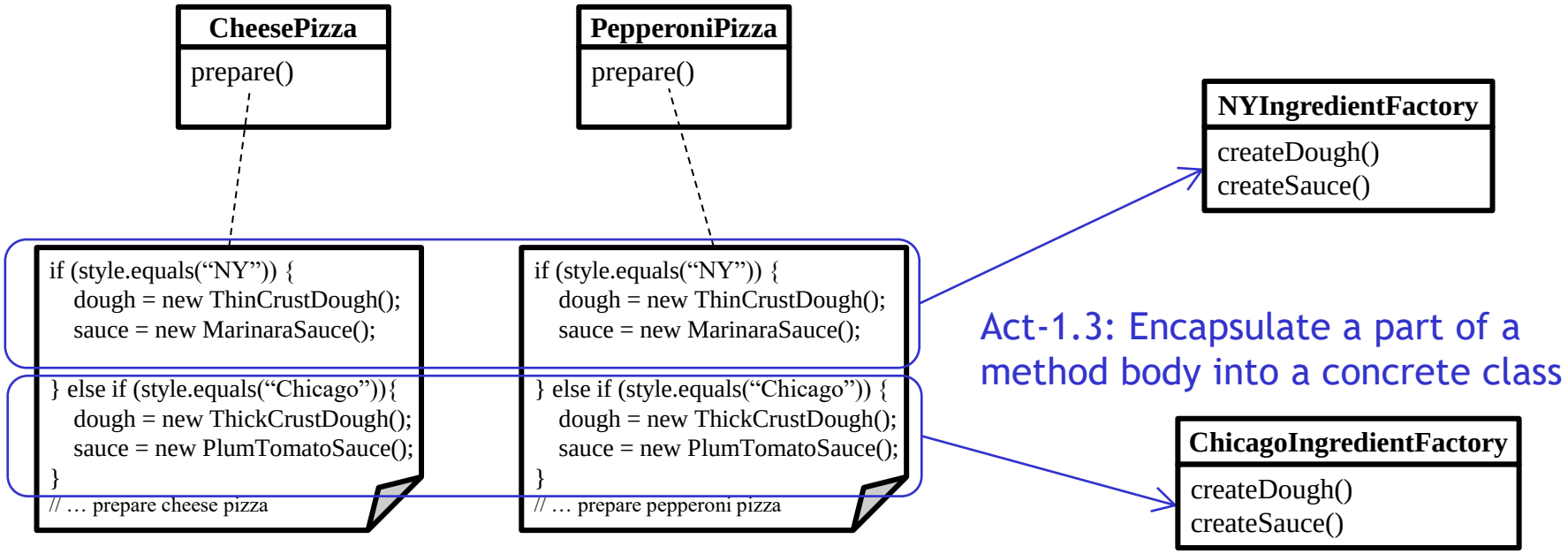
Problems with Initial Design



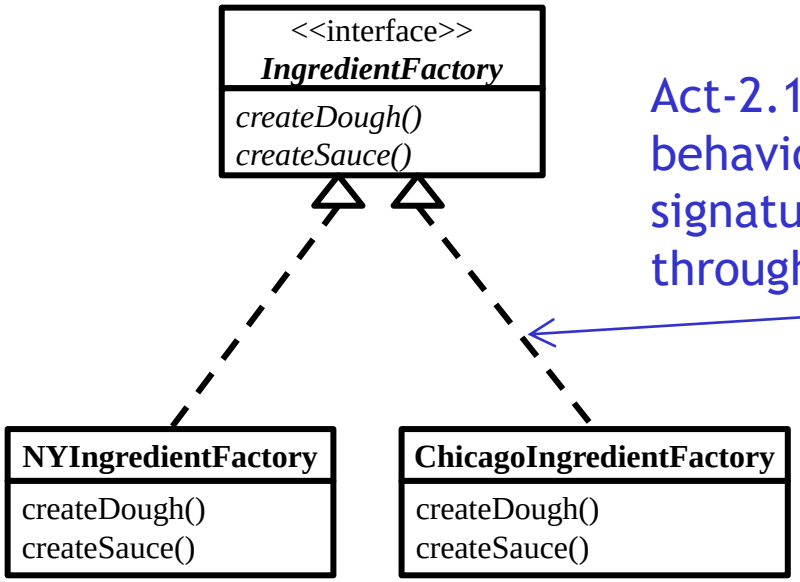
Design Process



Act-1: Encapsulate What Varies

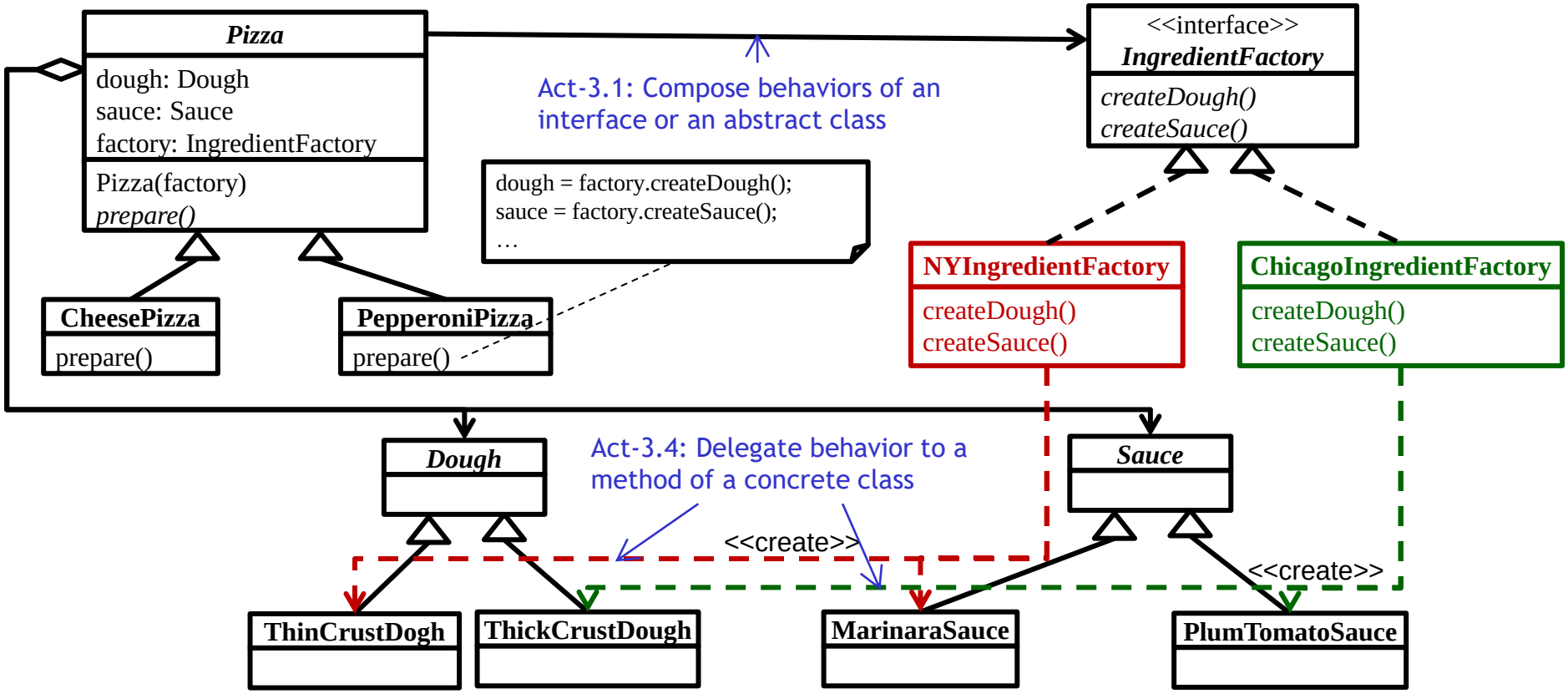


Act-2: Abstract Common Behaviors

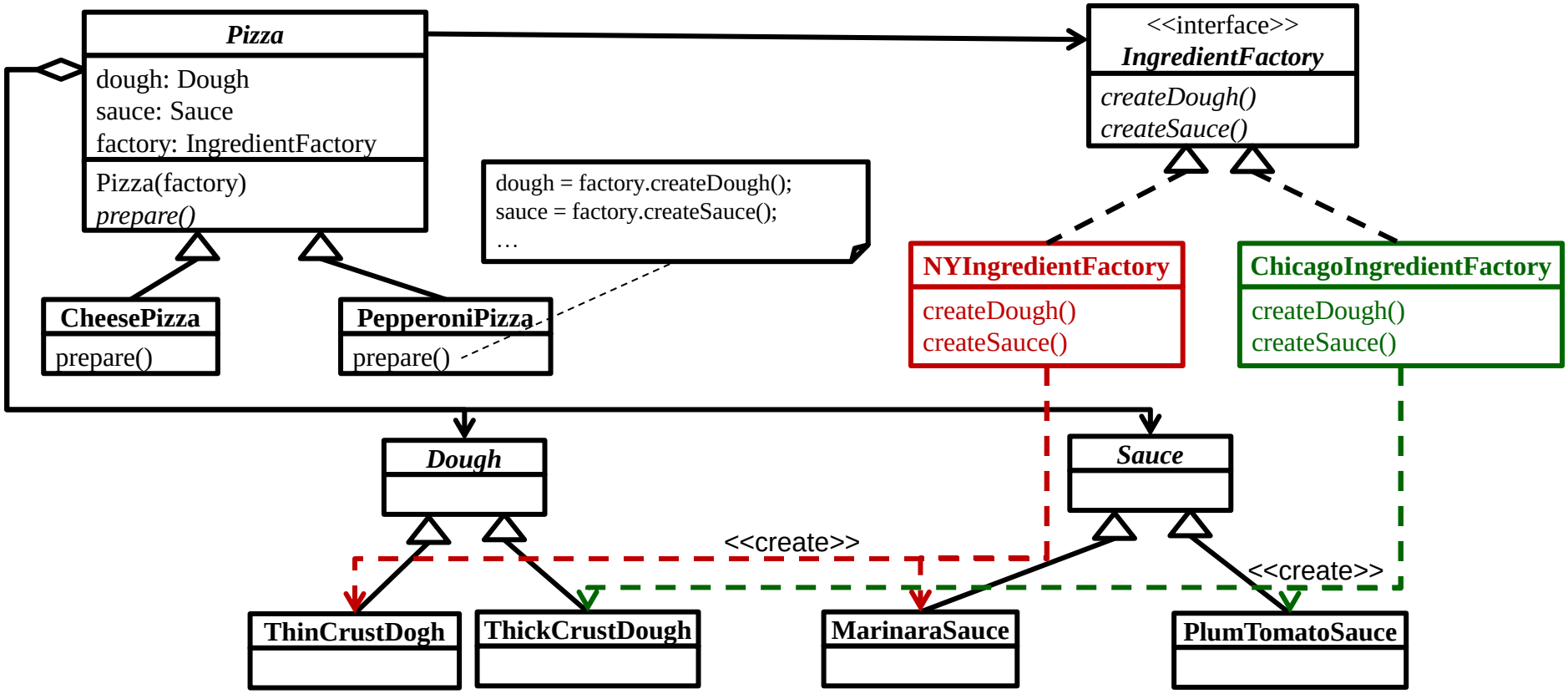


Act-2.1: Abstract common behaviors with a same signature into interface through polymorphism

Act-3: Compose Abstract Behaviors



Refactored Design after Design Process



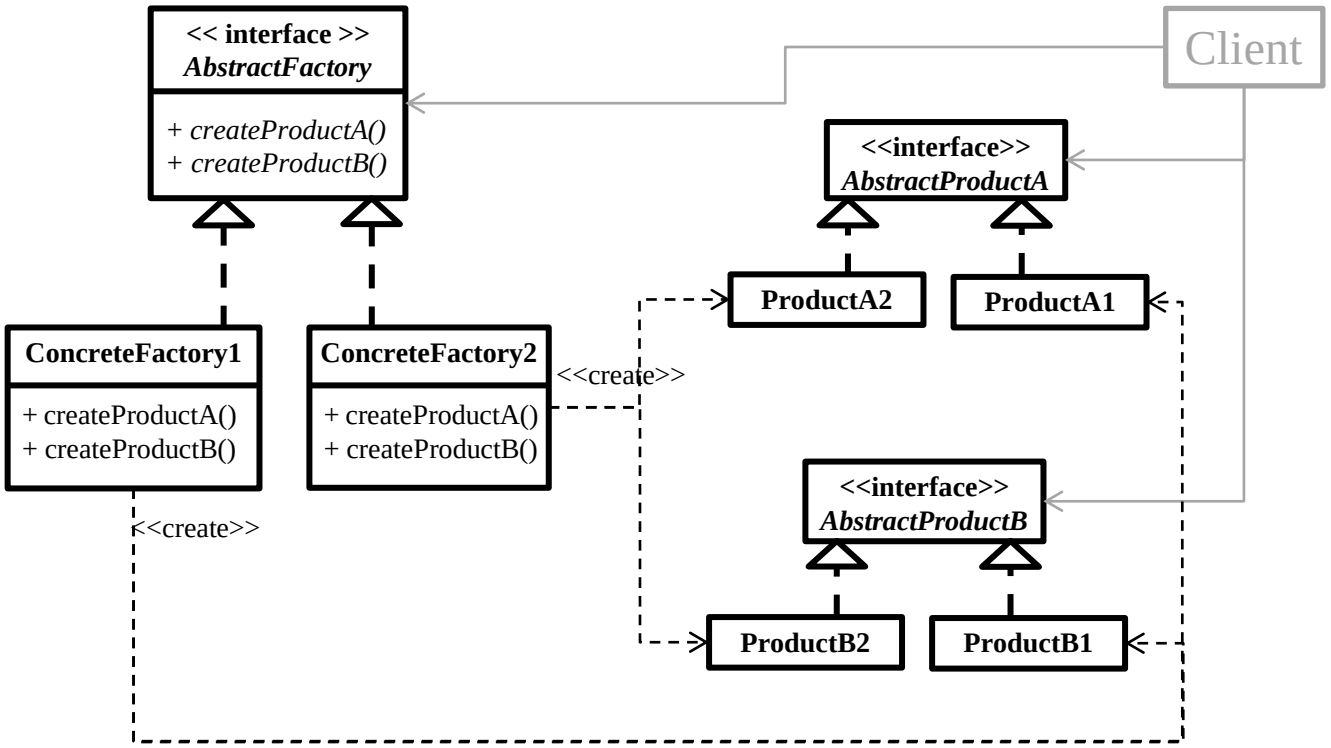
Recurrent Problem

- As the families of related or dependent objects are added, we need to write new object classes for the new families
  - For example, different look-and-feels define different appearances and behaviors for user interface "widgets" like scroll bars, windows, and buttons.

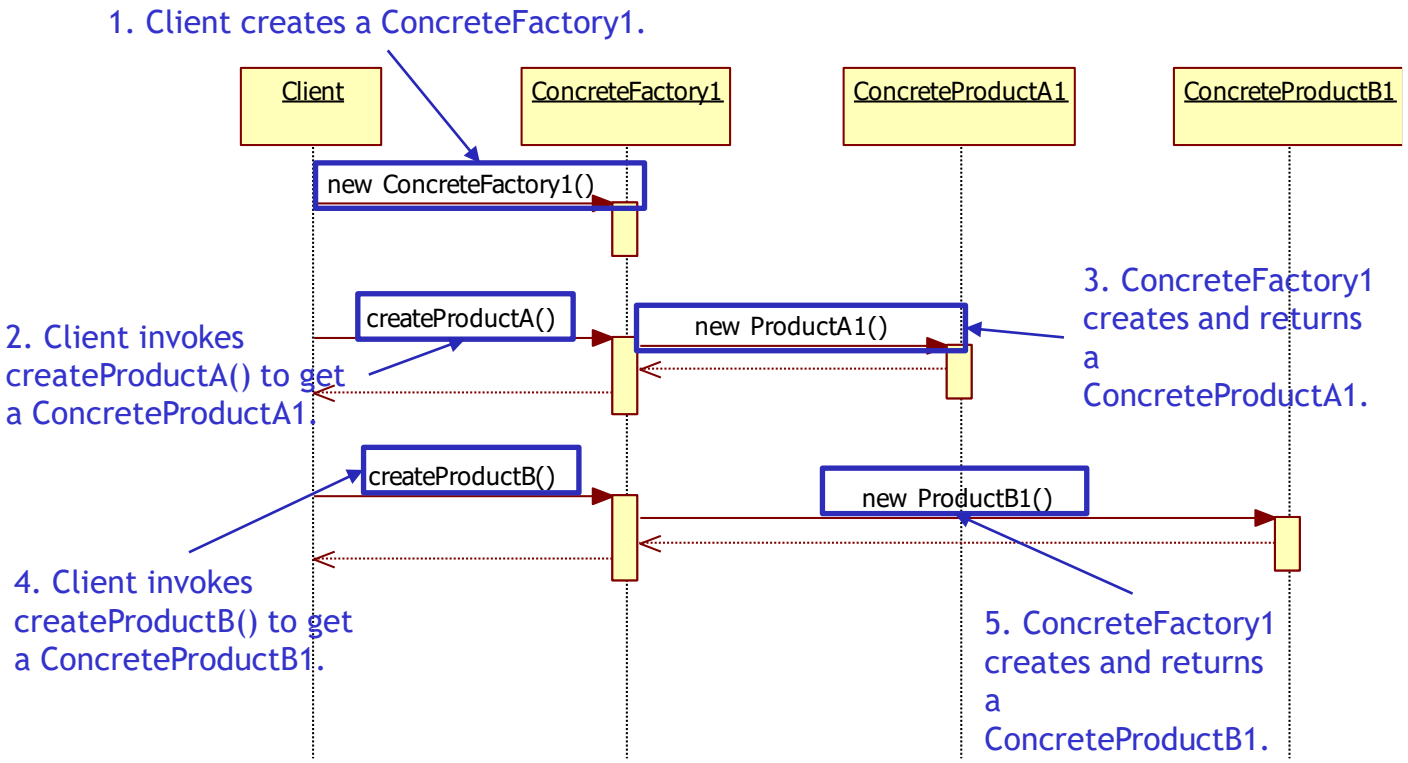
Intent

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

Abstract Factory Structure$_1$



Abstract Factory Structure$_2$



Abstract Factory Structure$_3$

	Instantiation	Use	Termination
Client	Other class except classes in the AbstractFactory	Other class except classes in the Abstract Factory	Other class except classes in the AbstractFactory
Abstract Factory	X	Client class uses this interface to get a product which is produced by ConcreteFactory through polymorphism	X
Concrete Factory	Other class or the client class	Client class uses this class to get a product through AbstractFactory	Other class or the client class
Abstract ProductA	X	Client class uses ConcreteProductA through this interface	X
Concrete ProductA	ConcreteFactory	Client class	Other class or the client class
Abstract ProductB	X	Client class uses ConcreteProductB through this interface	X
Concrete ProductB	ConcreteFactory	Client class	Other class or the client class

Abstract Factory vs. Factory Method

- Factory Method
  - creates single products
- Abstract Factory
  - consists of multiple factory methods
  - each factory method creates a related or dependent product