Design Patterns

宋 杰

Song Jie

东北大学 软件学院

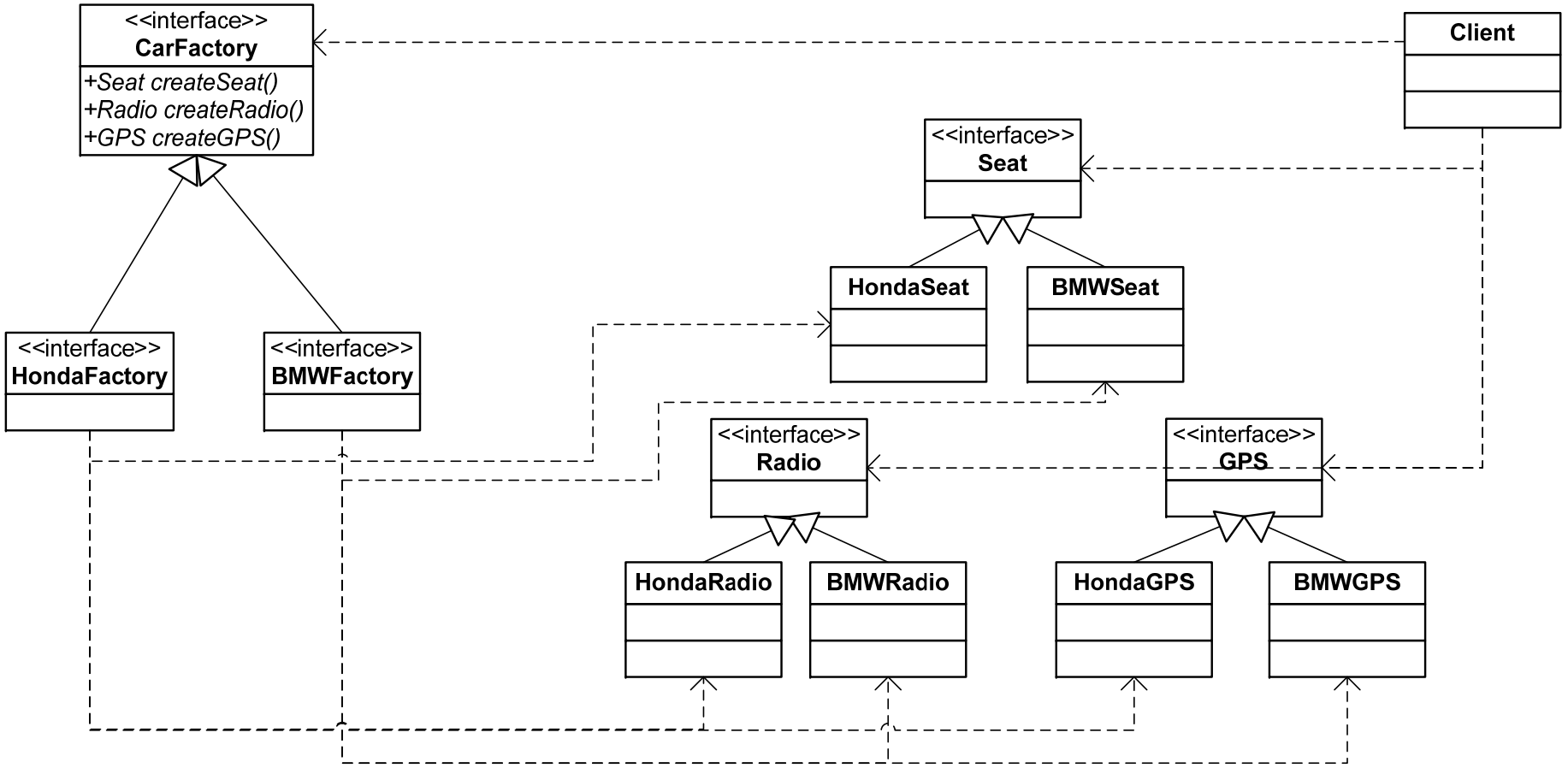
Software College, Northeastern University

2. Abstract Factory Pattern

Intent

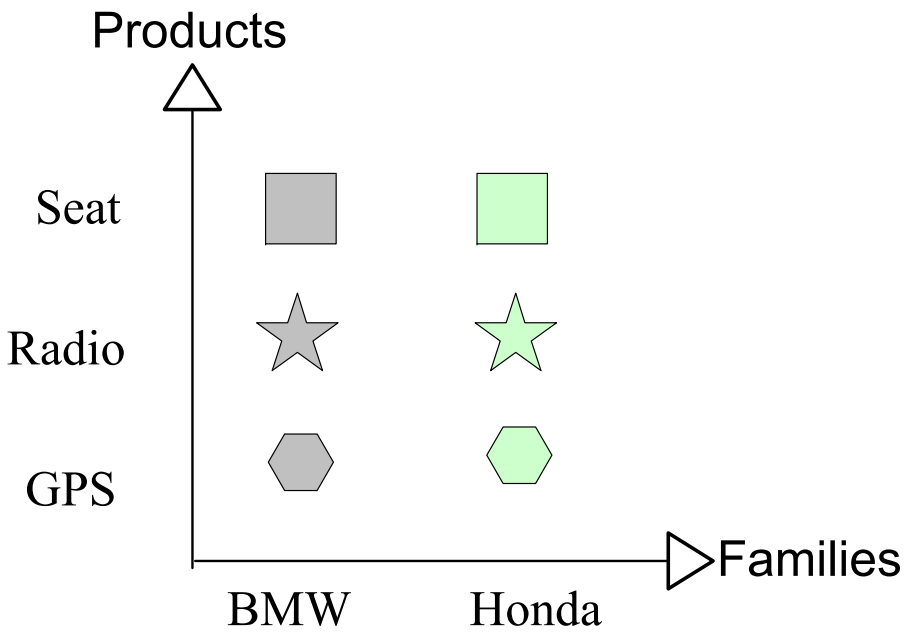
- Provide an interface for creating families of related or dependent objects without specifying their concrete classes.
  - Abstract level: Factory creates products in product-family.
  - Concrete level: Concrete factory create different concrete products in one product family, these products are in same inherited level.
- “Abstract” means both factory and products are abstract.

Product Family



Product Family

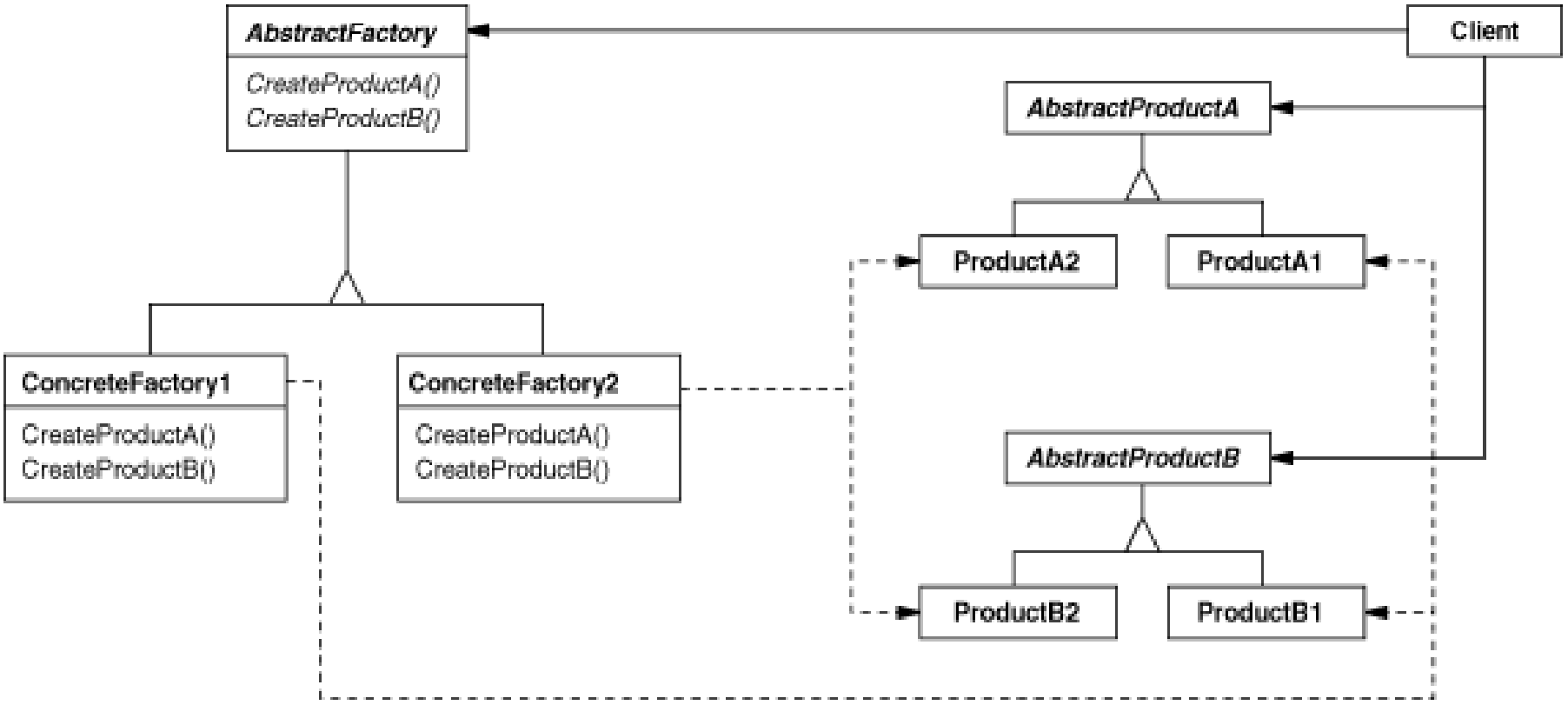
- Product Family have several related Products
- Every Products have same Concrete Products of each Product Family .
- 2-dimensions to classified the related products. The one is product, the other is family.



Factory and Product

- Factory method corresponds to product in the product family;
- Factory class corresponds to all concrete products in product family;
- Generally, the number of factory method match that of product. The number of concrete factory match that of product family.

Structure


AbstractFactory
CreateProductA()
CreateProductB()
Client
AbstractProductA
ProductA2
ProductA1
ConcreteFactory1
CreateProductA()
CreateProductB()
ConcreteFactory2
CreateProductA()
CreateProductB()
AbstractProductB
ProductB2
ProductB1

Participants

- AbstractFactory**:** declares an interface for operations that create abstract product objects.
- ConcreteFactory**:** implements the operations to create concrete product objects.
- AbstractProduct: declares an interface for a type of product object.
- ConcreteProduct**:** defines a product object to be created by the corresponding concrete factory. implements the AbstractProduct interface.
- Client**:** uses only interfaces declared by AbstractFactory and AbstractProduct classes.

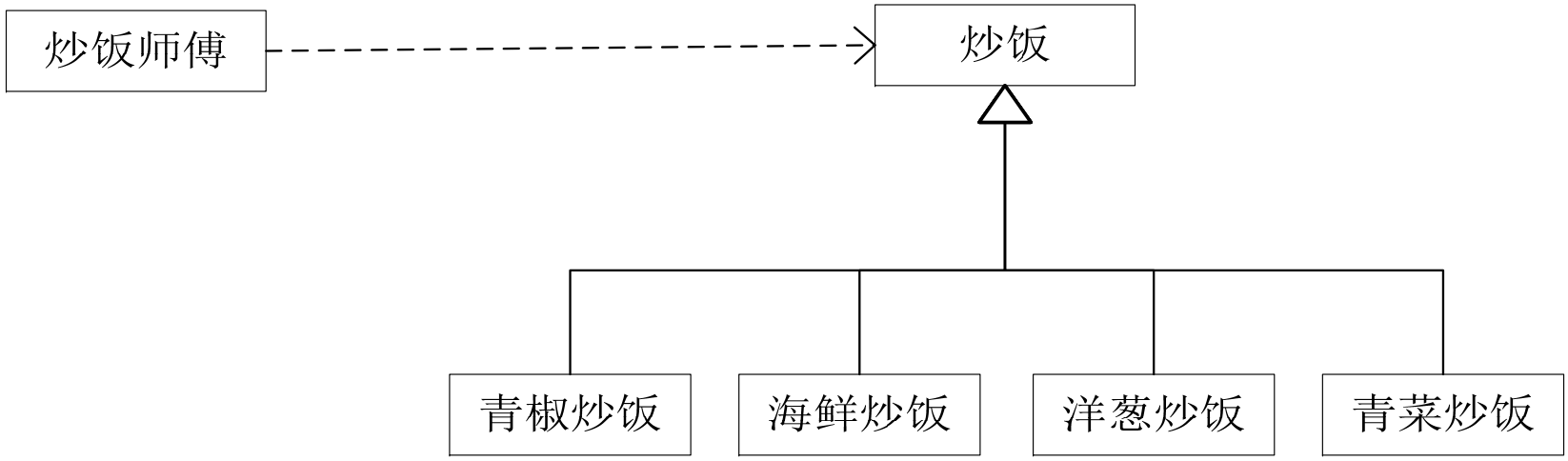
Consequences

- It isolates concrete classes (products);
- It makes exchanging product families easy (OCP).
- It promotes consistency among products. it's important that an application use objects from only one family at a time.
- Supporting new products is difficult.

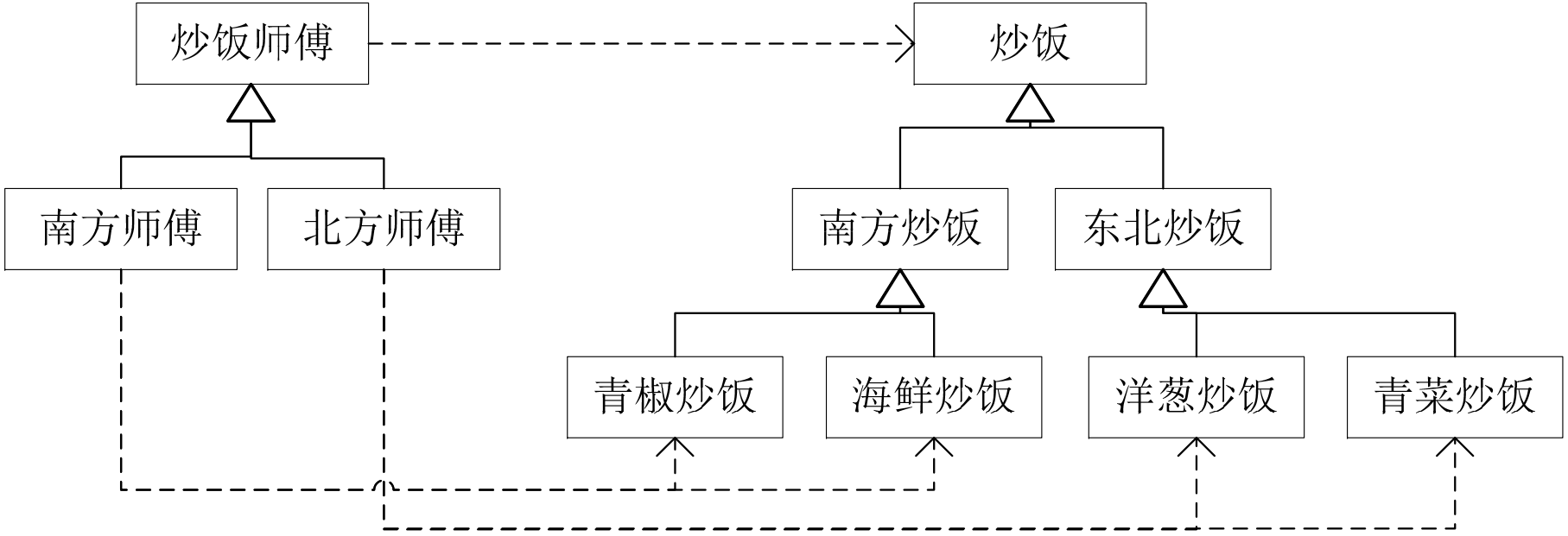
Applicability

- A system should be independent of how its products are created, composed, and represented.
- A system should be configured with one of multiple families of products.
- A family of related product objects is designed to be used together, and you need to enforce this constraint.
- You want to provide a class library of products, and you want to reveal just their interfaces, not their implementations.

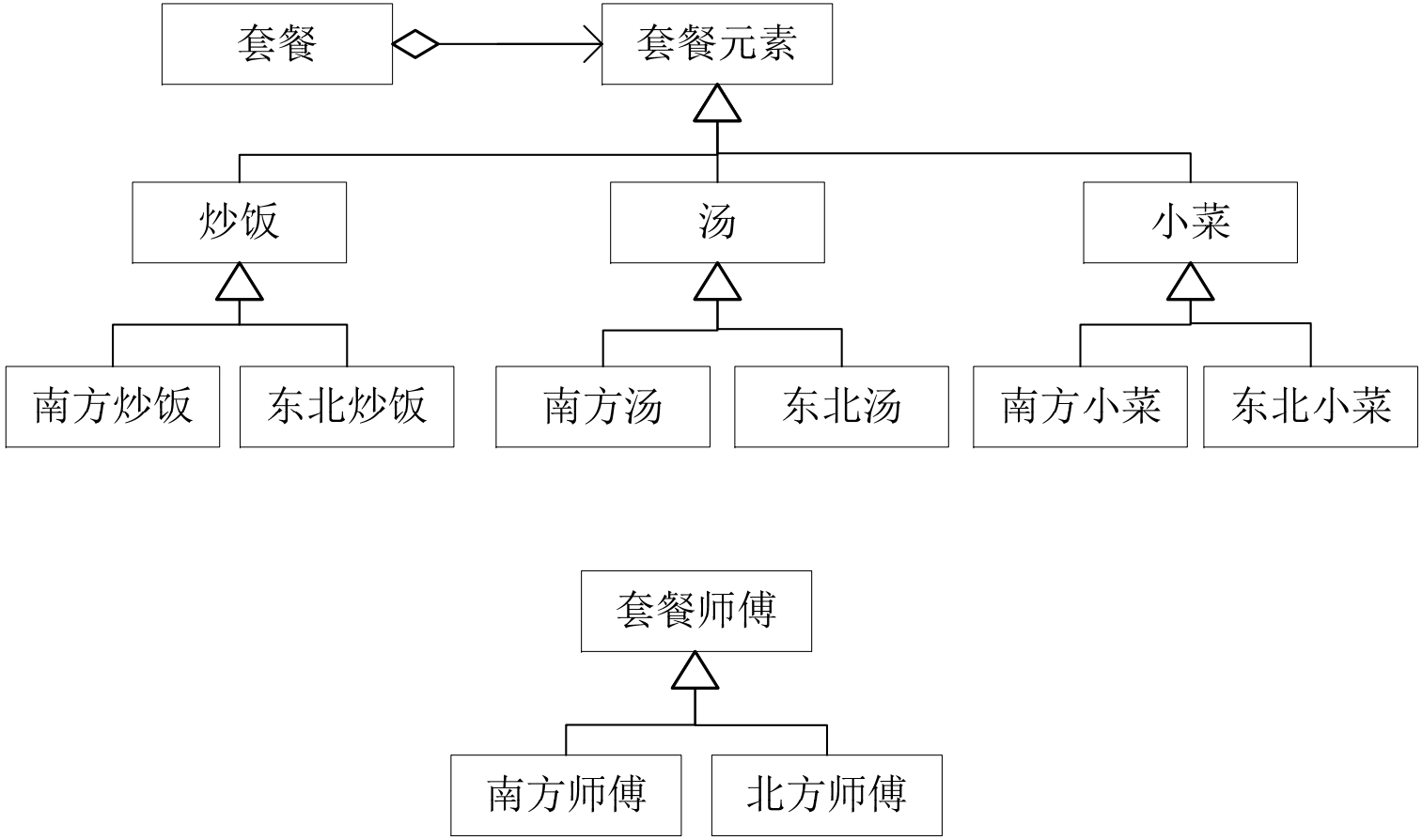
Examples Step 1



Examples Step 2

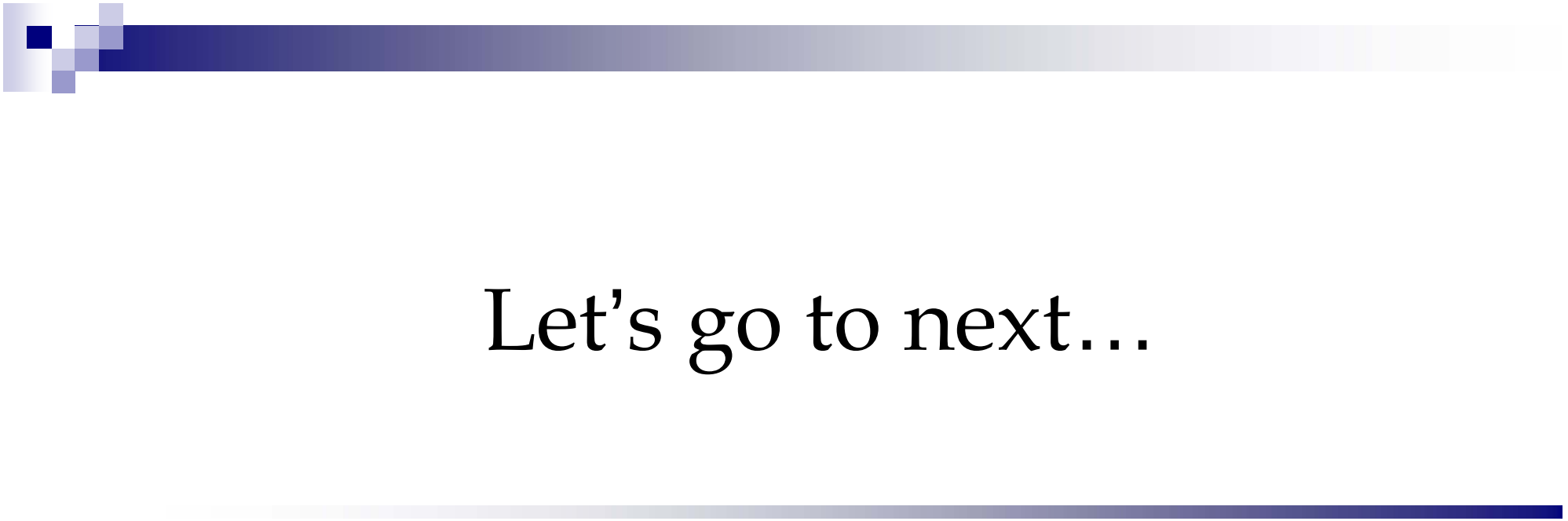


Examples Step 3



Related Patterns

- Factory Method
  - Adding support to product family;
  - Concrete factory use Factory Method to implement the create logic.



Let’s go to next…