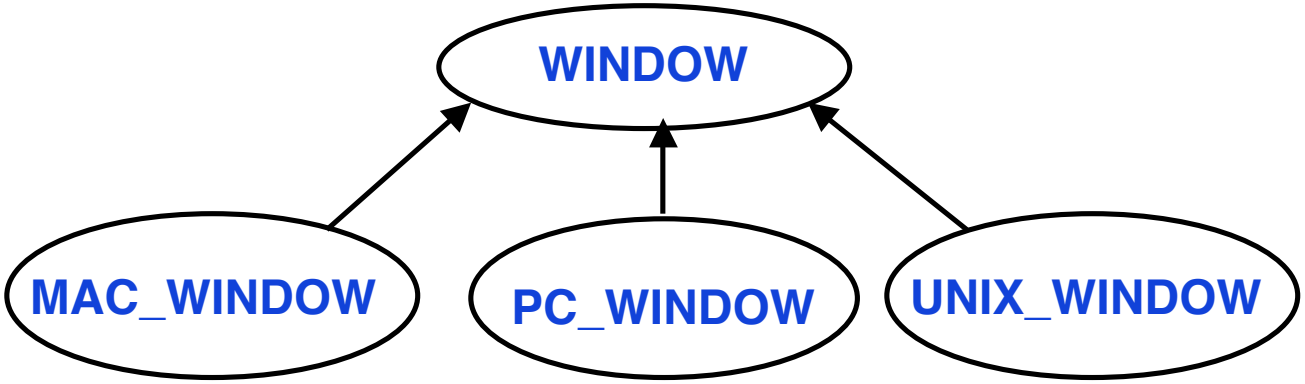
Bridge Pattern – Structural

- Intent
  - » **Decouple an abstract from its implementation so that the two can vary independently**

- Also known as
  - » **Handle / Body**

Class Adapter – Motivation

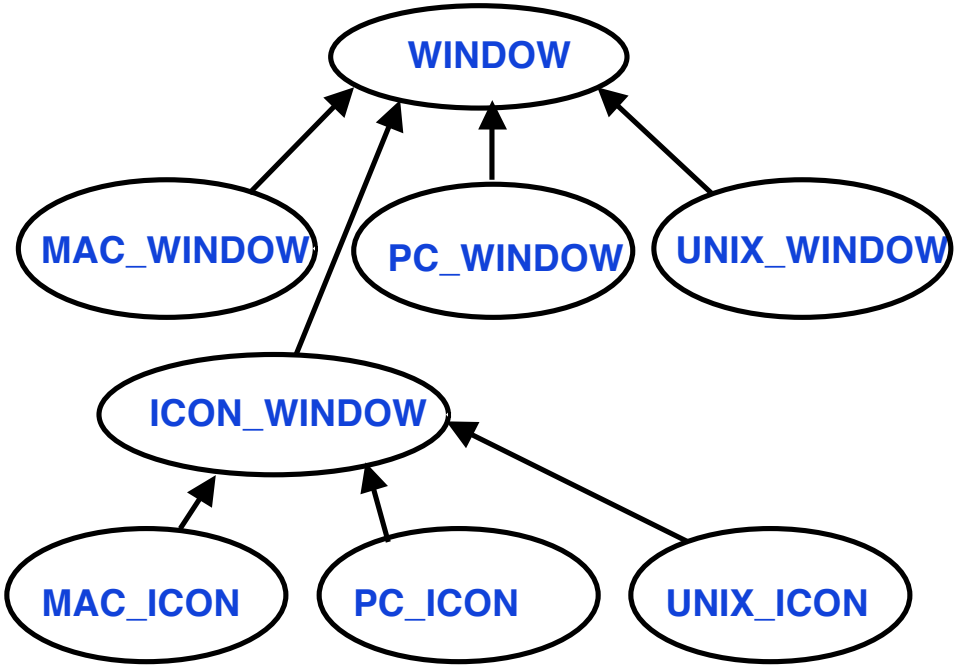
- You want to have applications working on any windowing system – there are many of them



- Introduce ICON_WONDOW a sub-class of WINDOW

Class Adapter – Motivation – 2

- Now need to add 3 more classes
  - » **One for each windowing system**



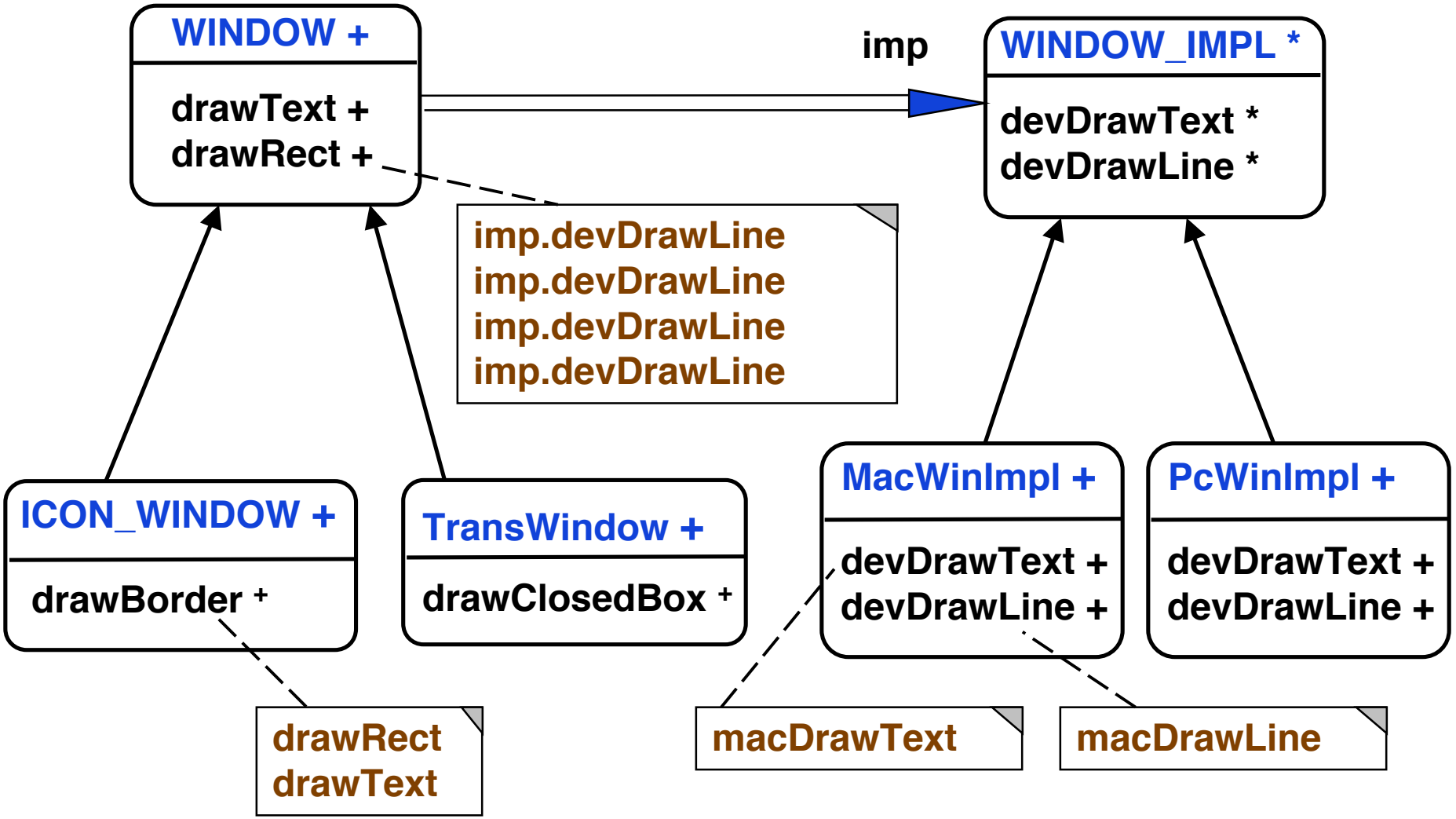
W window systems
T window types

Need ≈W*T new classes

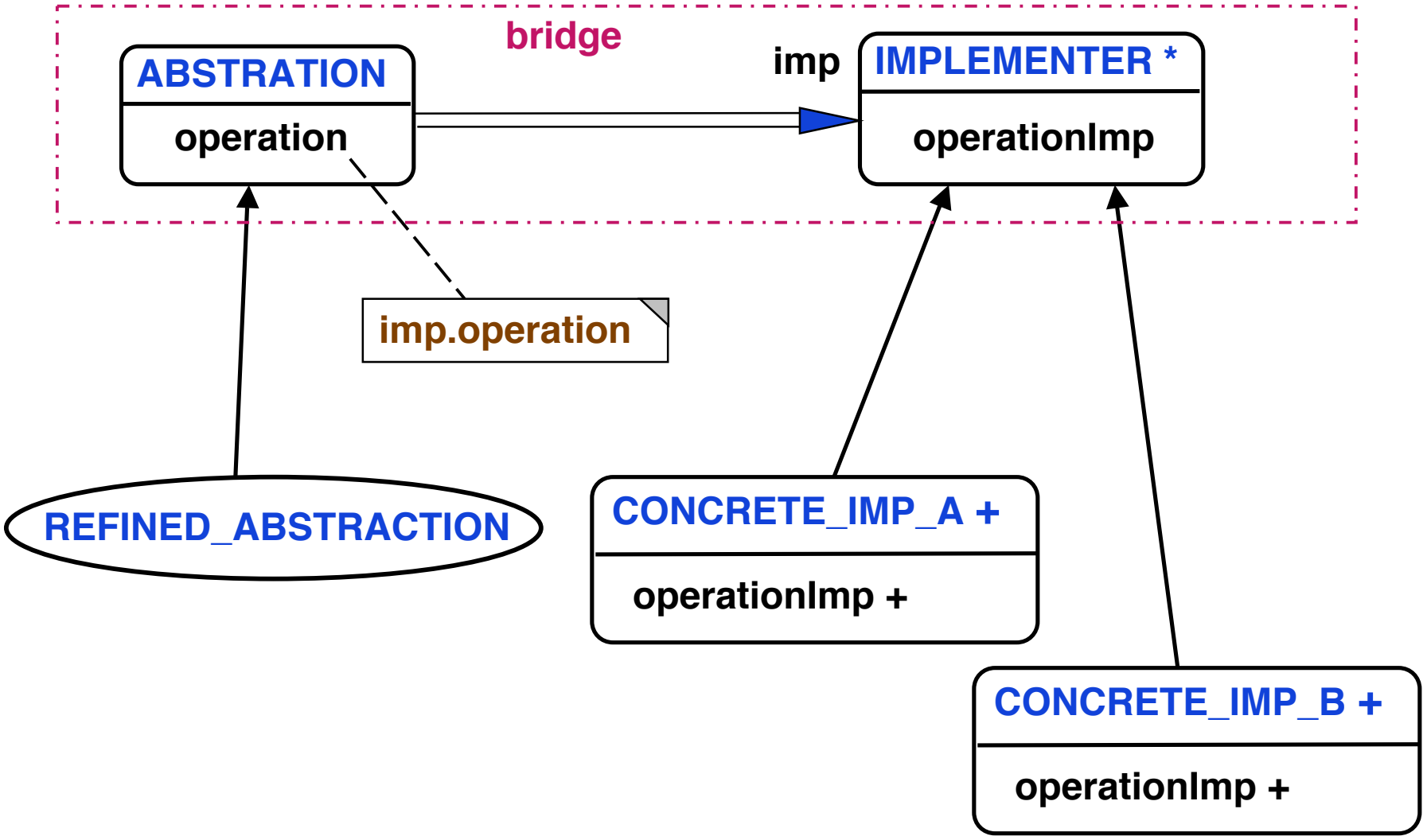
Want ≈W+T new classes

The new classes are hard wired for each type of window system

Bridge – Example Architecture



Bridge – Abstract Architecture



Bridge – Participants

- Abstraction
  - » **Defines the abstractions interface**
  - » **Maintains a reference to an object of type implementor**

- RefinedAbstraction
  - » **Extends the interface defined by Abstraction**

Bridge – Participants – 2

- Implementer
  - » **Defines the interface for implementation classes**
    - > **Can be different from the Abstraction interface**
      - – **Implementer provides primitive operations**
      - – **Abstraction provides higher-level operations**

- ConcreteImplementer
  - » **Implements the Implementer interface**
  - » **Defines its concrete implementation**

Bridge – Applicability

- Avoid permanent binding between an abstraction and its implementation
  - » **Especially if a switch is needed dynamically**
- Both abstractions and implementations should be extensible by subclassing
- Changes in implementation should have no impact on clients

Bridge – Applicability – 2

- Splitting "nested generalizations, as in the window motivation

- Want to share an implementation among multiple objects, and this should be hidden from clients

Bridge – Consequences

- Decouples interface and implementation
  - » **Can configure implementation to use at runtime**
  - » **Encourages better structure through layering**
    - > **The client only has to know about Abstraction and Implementer**
- Improved extensibility
  - » **Extend in Abstraction and Implementer independently**
- Hide implementation details from clients

Bridge – Related Patterns

- Abstract Factory can create and configure a particular Bridge

- Adapter is geared toward making unrelated classes work together
  - » **Usually applied after systems are designed**
  - » **Bridge is used during design**