

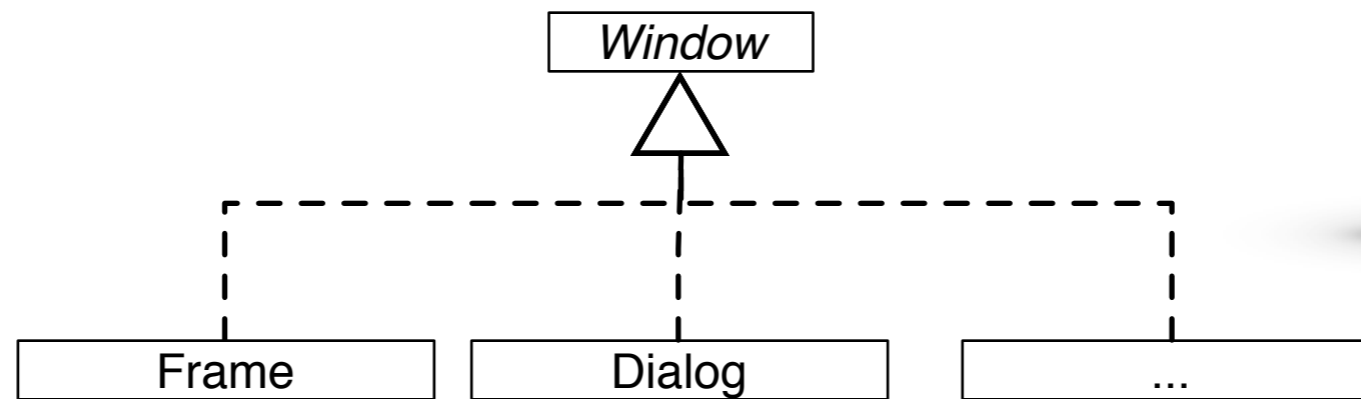
Software Engineering Design & Construction

Dr. Michael Eichberg
Fachgebiet Softwaretechnik
Technische Universität Darmstadt

Bridge Pattern

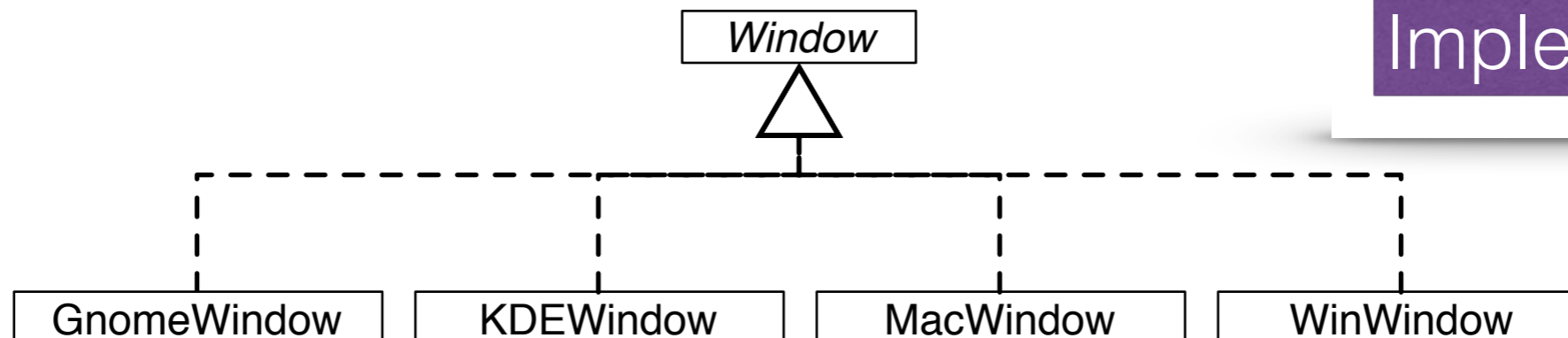
Motivation by Example

We want to provide different types of windows:



Abstraction

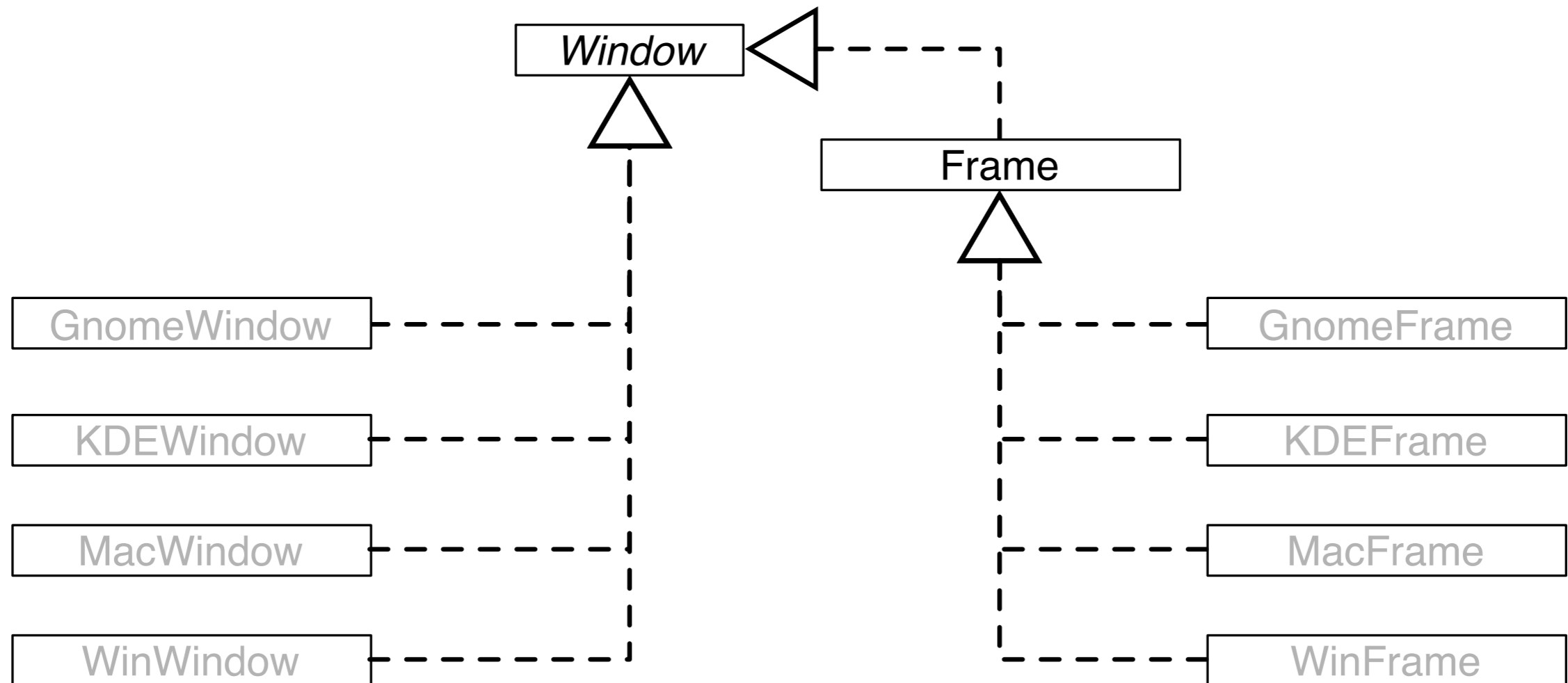
We want to support multiple operating systems:



Implementation

Motivation by Example

Two dimensions of variability!



Can you imagine a better solution?

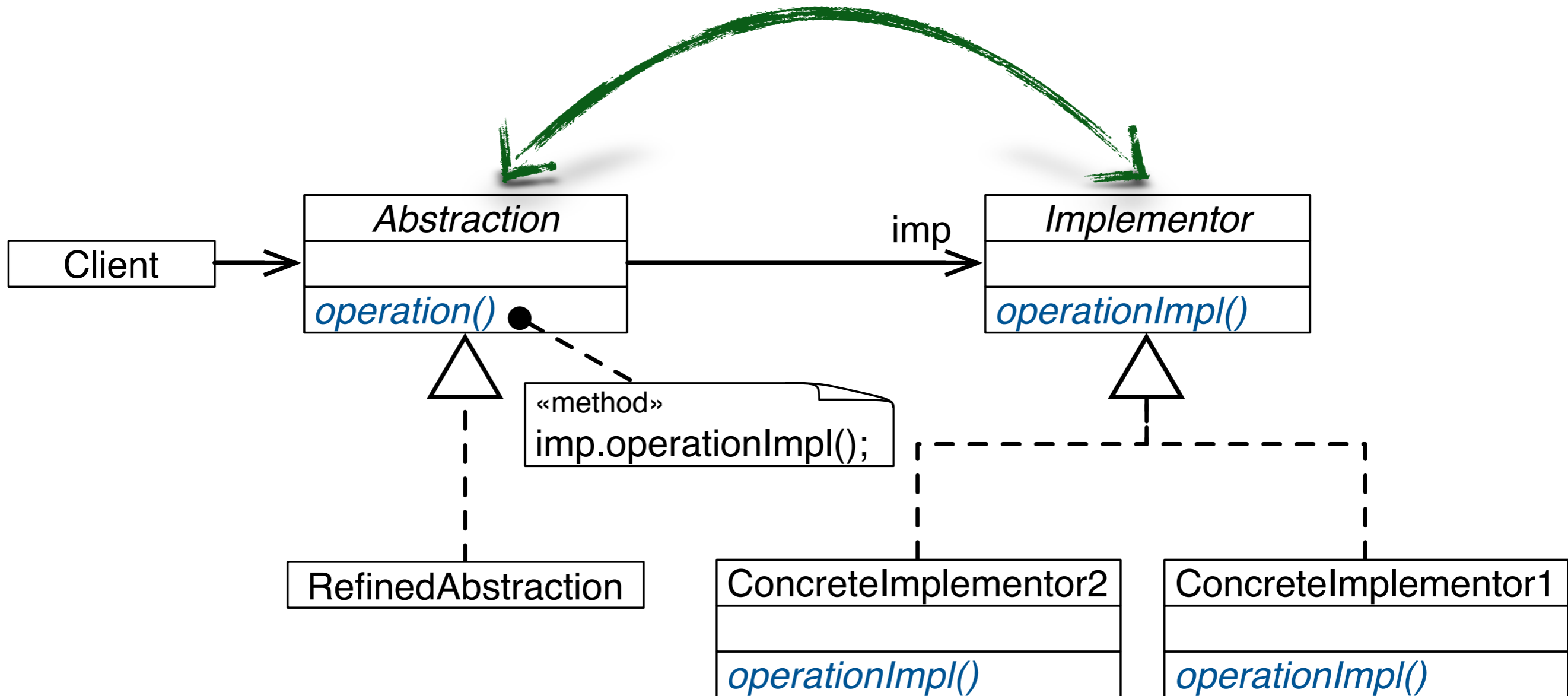
The Bridge Design Pattern

Decouple an **abstraction** from its **implementation**.

So that the two can vary independently.

Bridge Design Pattern - Structure

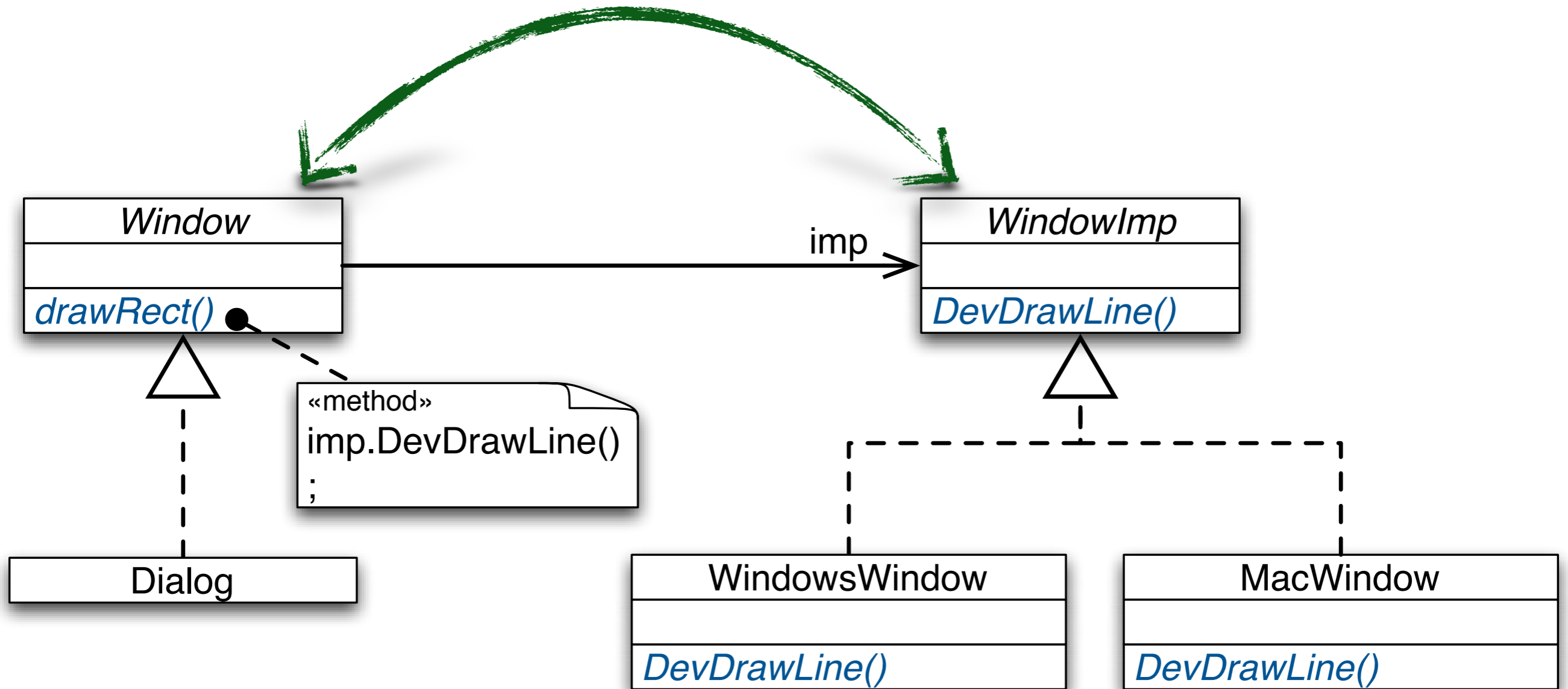
BRIDGE



Combine inheritance and object composition.

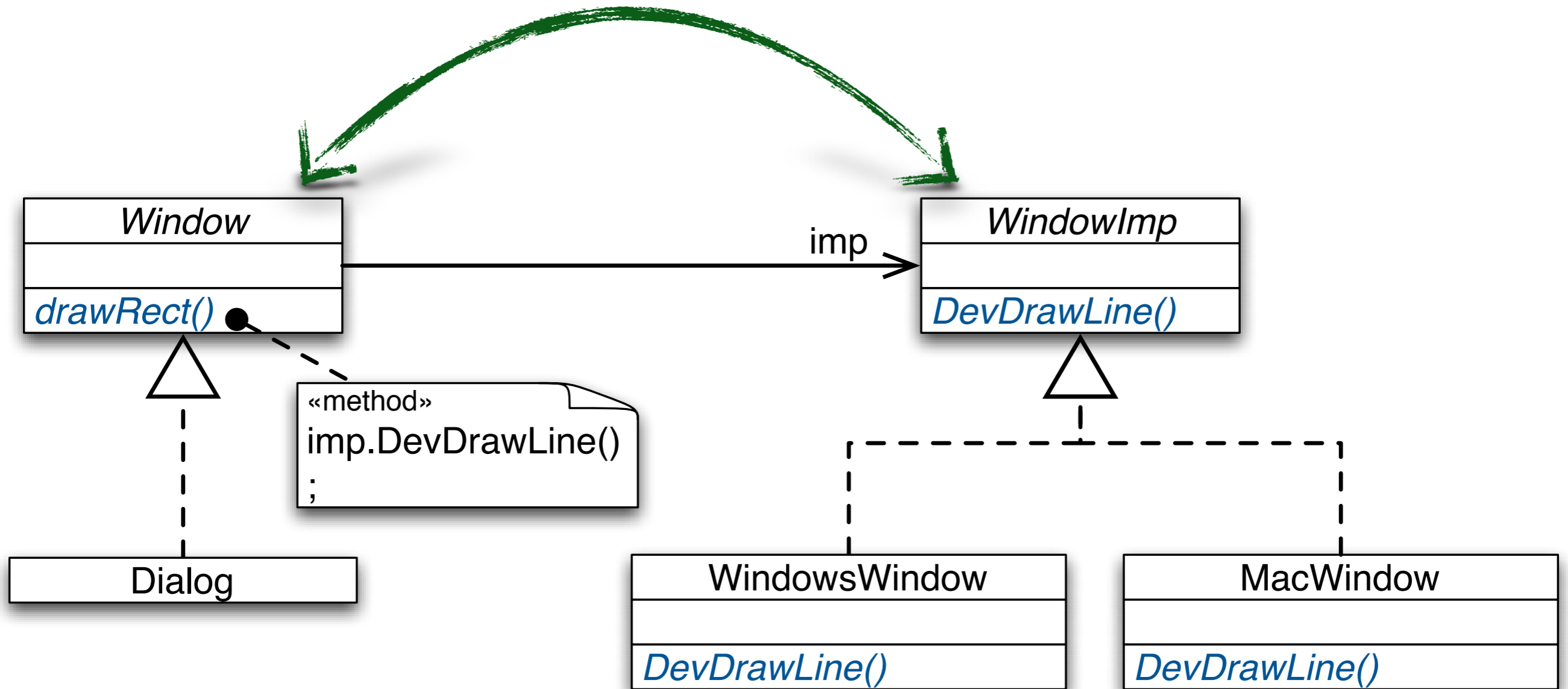
Bridge Design Pattern - Illustrated

BRIDGE



Bridge Design Pattern - Illustrated

BRIDGE



Inheritance allows structural variation: adding of new field and methods.

Composition demands a fixed interface.

Takeaway

- The Bridge Pattern instructs to use object composition to bridge between two inheritance hierarchies when you need to combine two kinds of variations of an object type.
- The Bridge Pattern allows to vary an abstraction and its implementation independently of each other.
- **Works well** as long as there is no dependency between the implementation on abstraction variations, i.e., if they do not vary co-variantly.