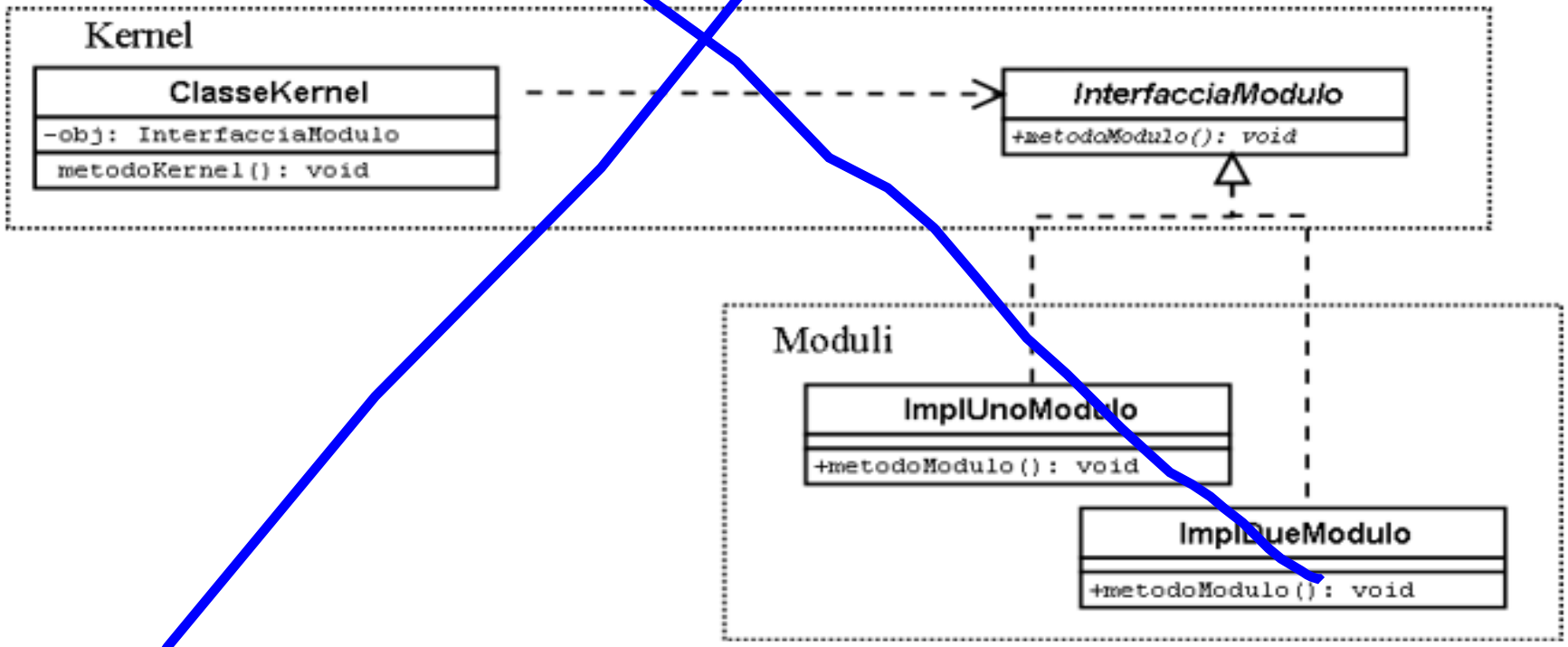
Programmazione III

Prof.ssa Liliana Ardissono
Dipartimento di Informatica
Università di Torino

JAVA – Ereditarietà – parte 3

Verso i pattern architetturali - I

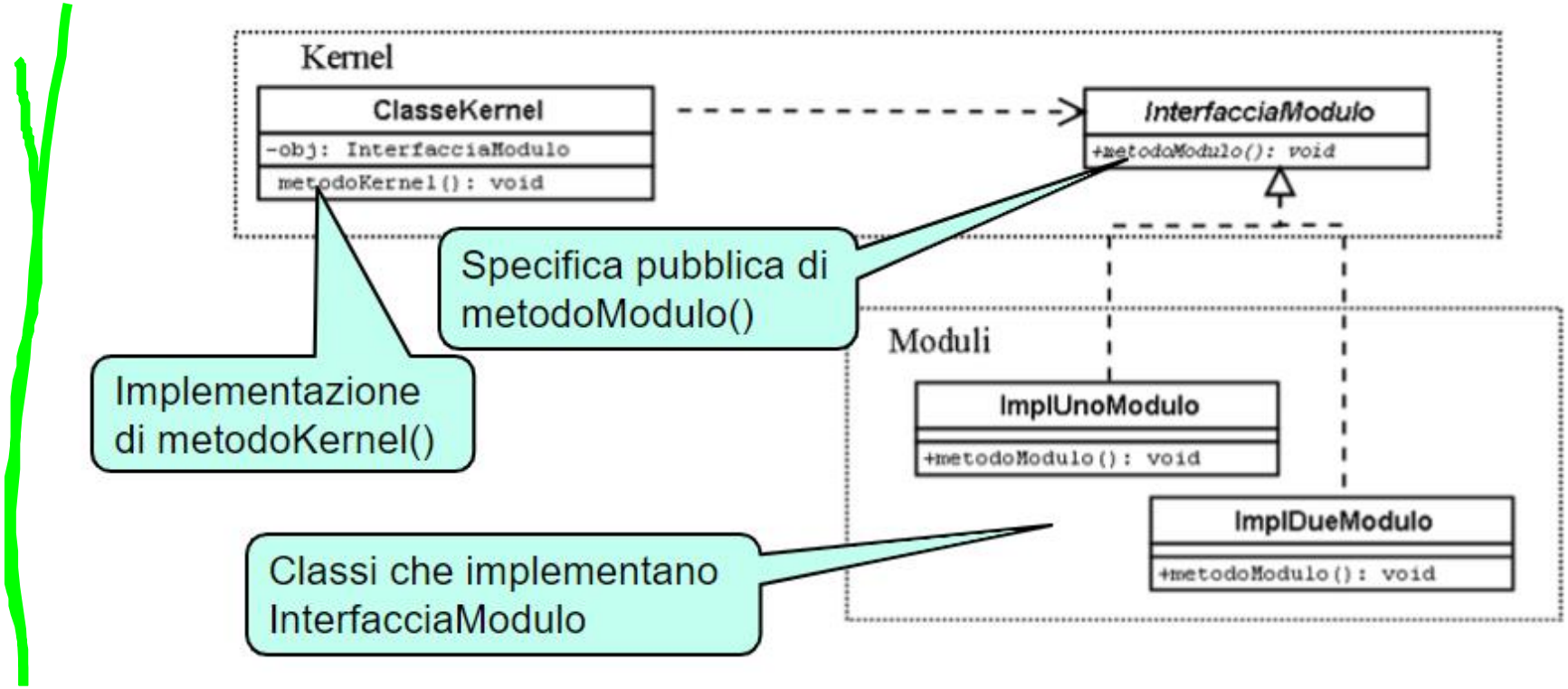
Combinando l'uso di interface/classi che implementano le interface con le relazioni di contenimento si ottengono modelli di programmazione avanzati. In particolare:



Quersta imagine è tratta dai lucidi di presentazione di Programmazione III del Prof. Matteo Baldoni, Università di Torino

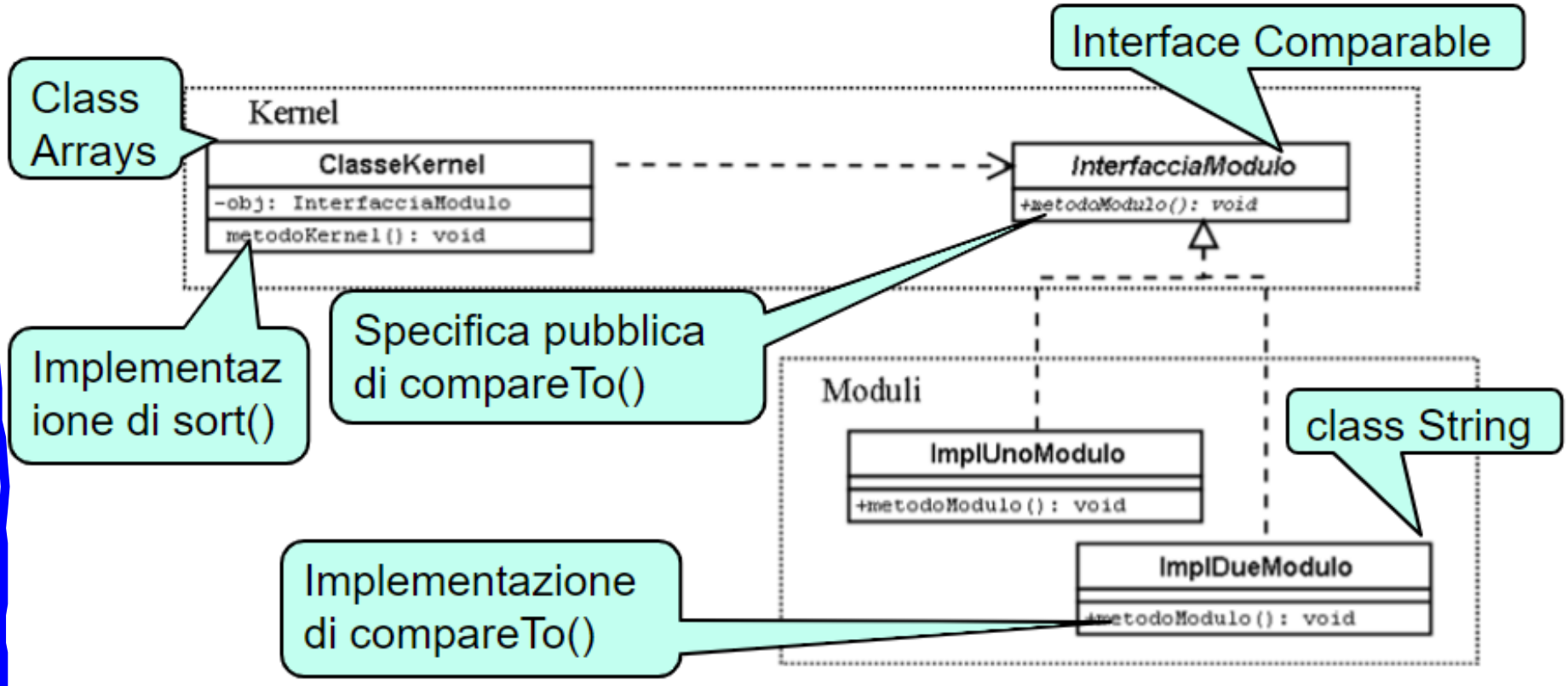
Verso i pattern architetturali - II

La classe ClasseKernel **usa** un componente obj di tipo InterfacciaModulo (in upcasting). Nel body del metodo metodoKernel(), la classe invoca metodoModulo() su obj per operare



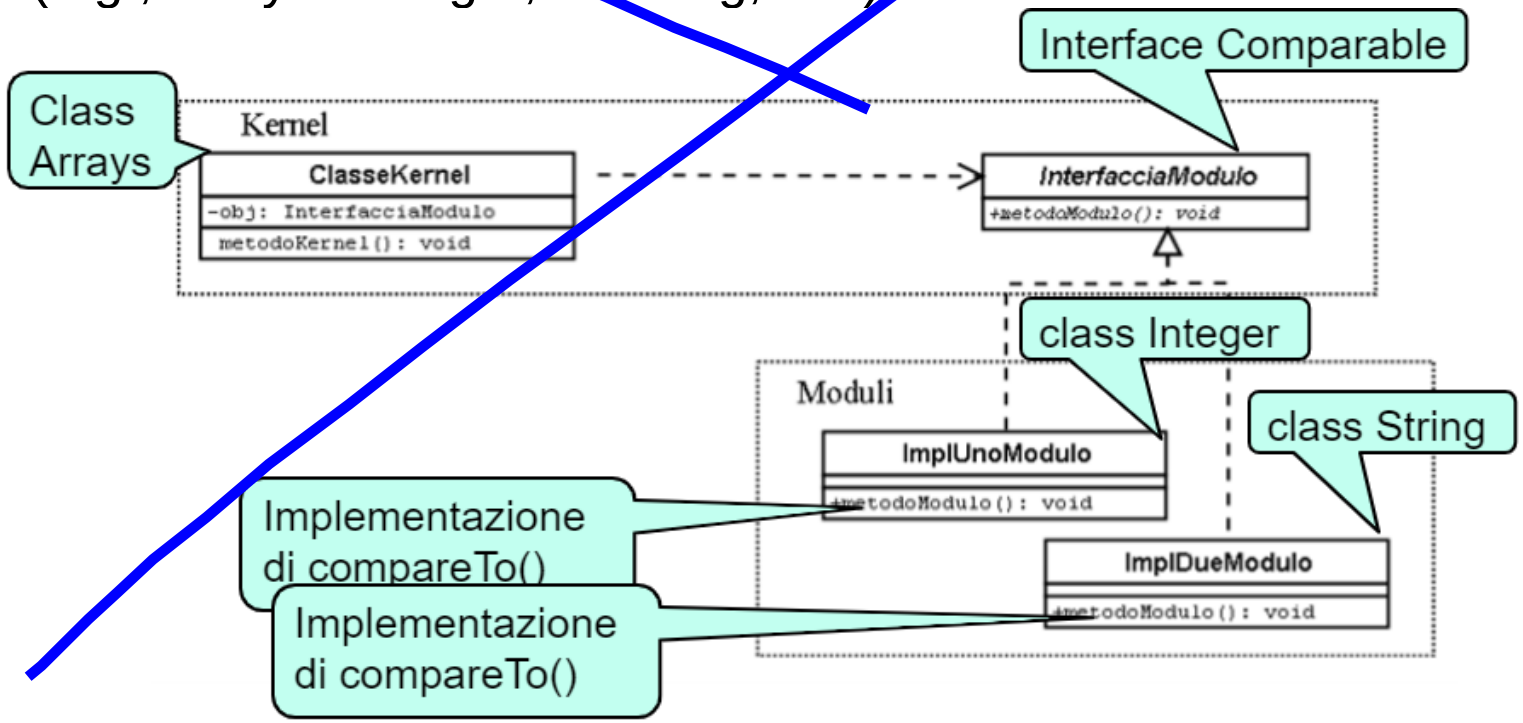
Verso i pattern architetturali - III

Esempio: l'applicazione usa la classe Arrays per ordinare un array di elementi
String[] myArray = new String[3]; *//… inserimento di valori in myArray…*
Arrays.sort(myArray);



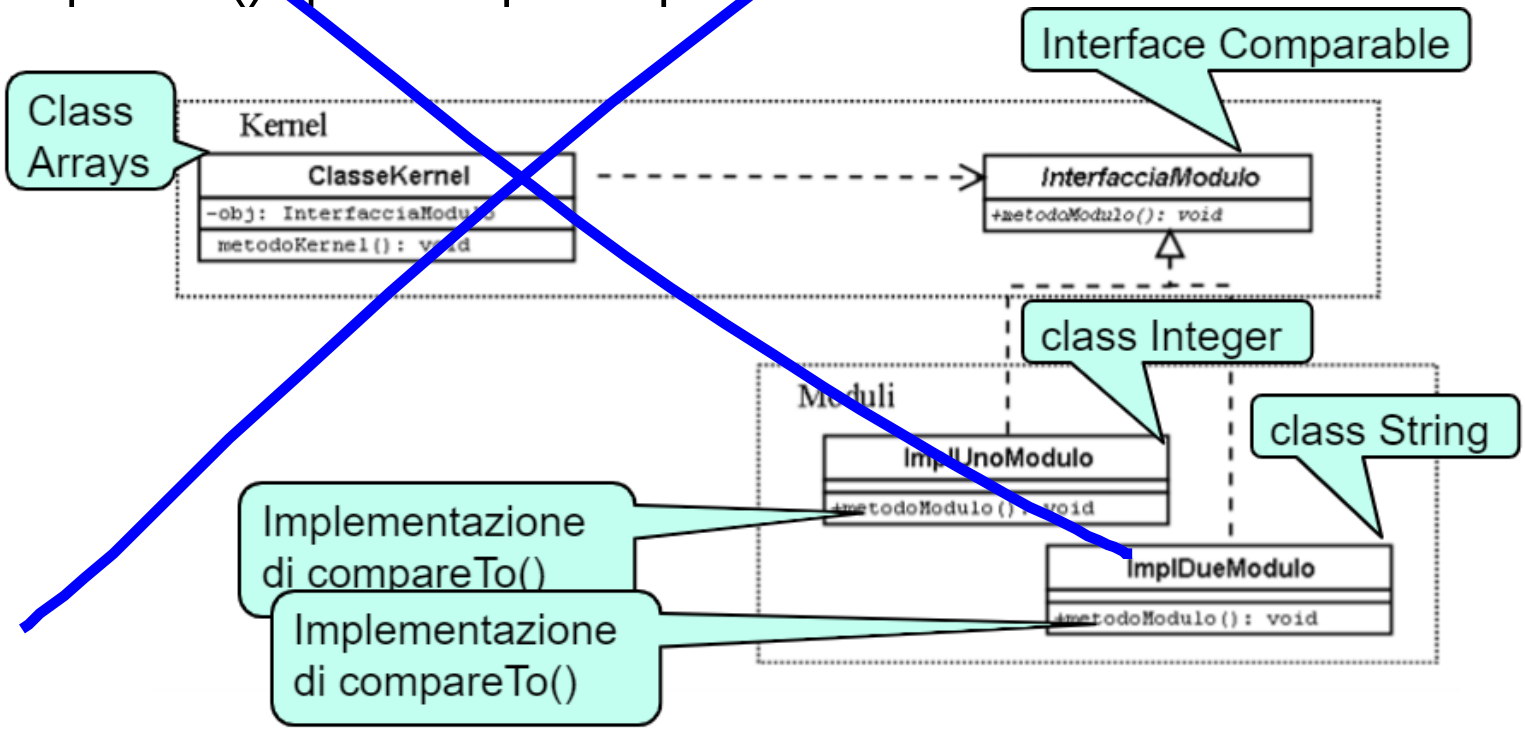
Verso i pattern architetturali - IV

In una specifica applicazione, l'InterfacciaModulo (Comparable) viene implementata da una classe (String nel nostro esempio) che implementa il metodo compareTo(). L'applicazione potrebbe usare implementazioni differenti (e.g., array di Integer, di String, etc.)



Verso i pattern architetturali - V

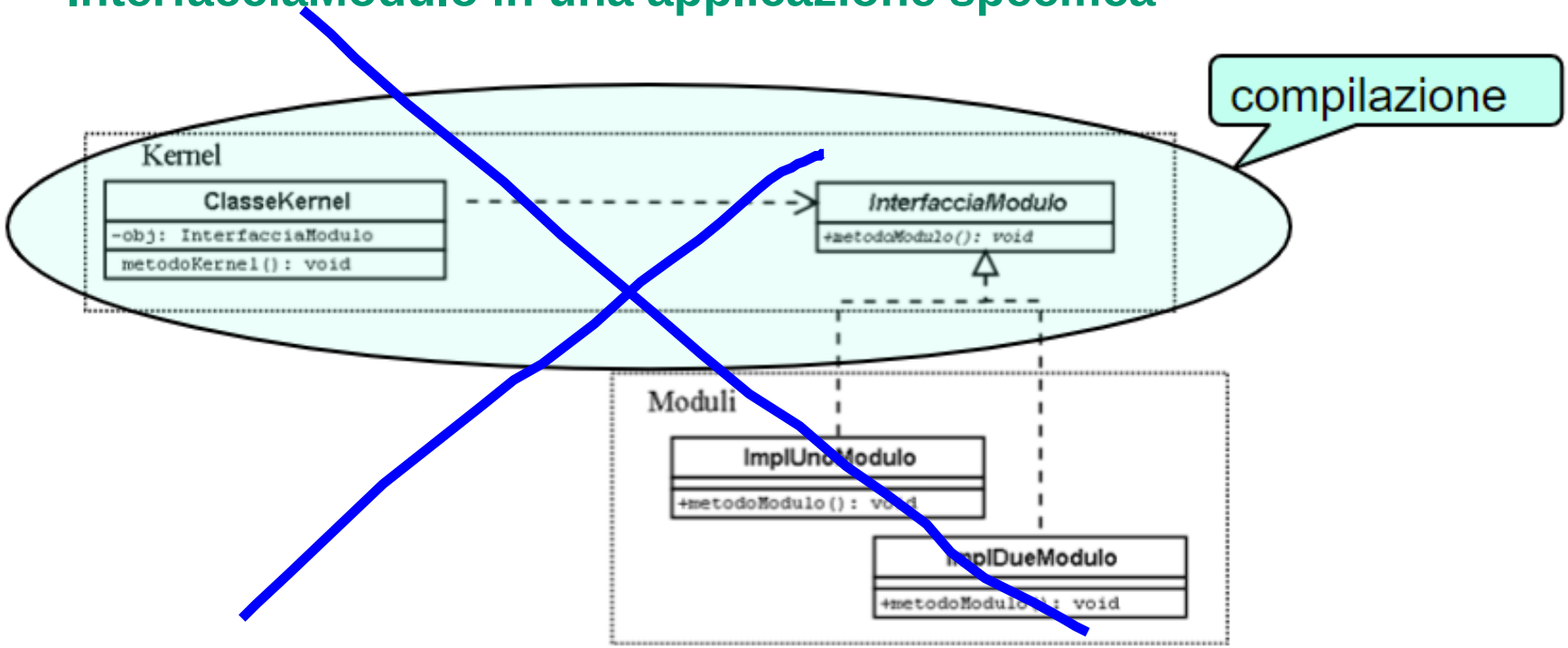
Grazie al polimorfismo e al binding dinamico, a seconda di quale tipo di oggetti viene messo nell'array (Integer, String, …), quando viene invocato Arrays.sort(myArray) verrà eseguita l'implementazione di compareTo() specifica per il tipo di dato



Verso i pattern architetturali - VI

Questa separazione tra interfacce e implementazioni permette la compilazione separata del codice:

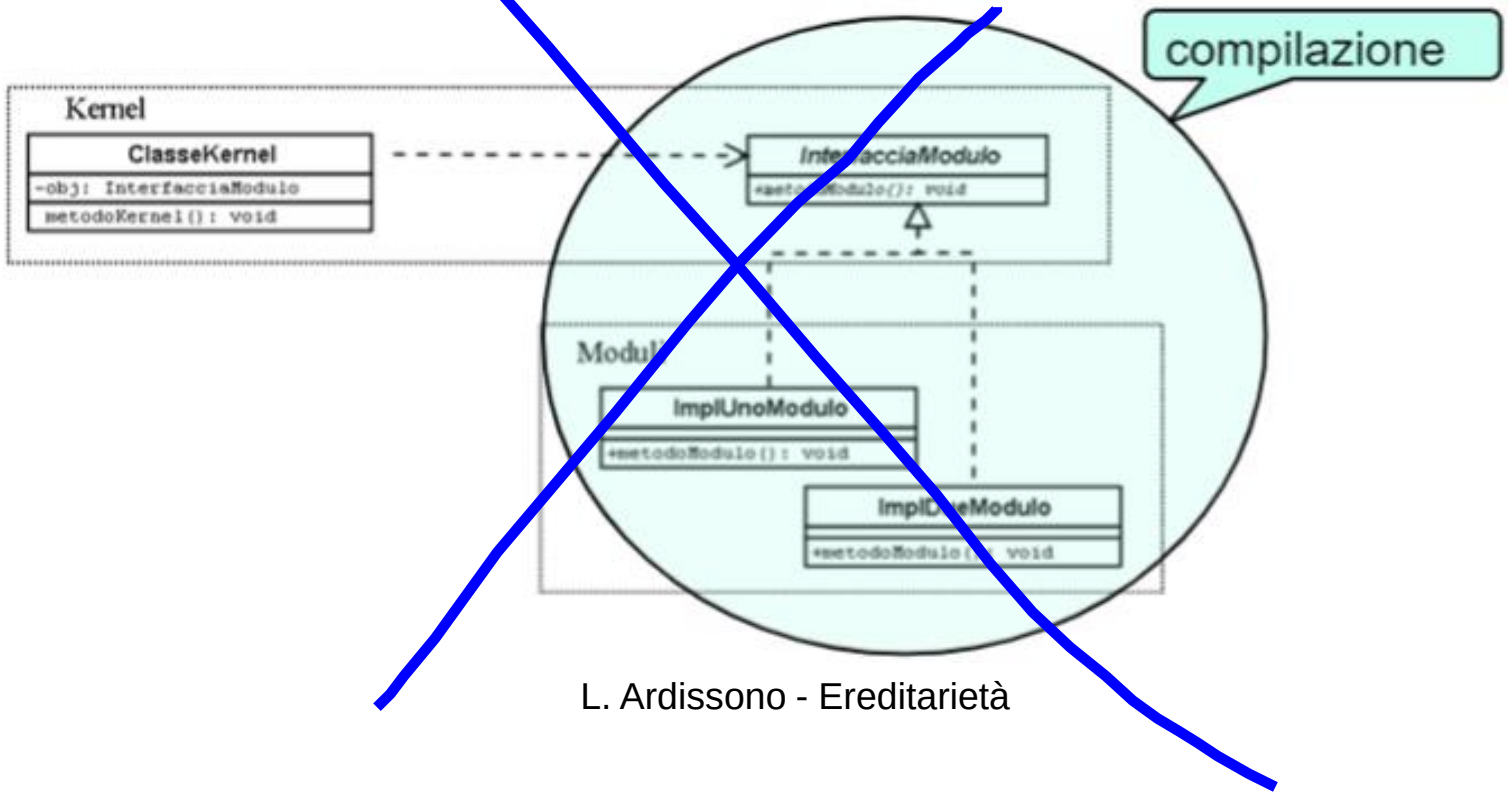
- **ClasseKernel + InterfacciaModulo può essere compilata (come libreria) anche se non si specificano le implementazioni di InterfacciaModulo in una applicazione specifica**



Verso i pattern architetturali - VII

Questa separazione tra interfacce e implementazioni permette la compilazione separata del codice:

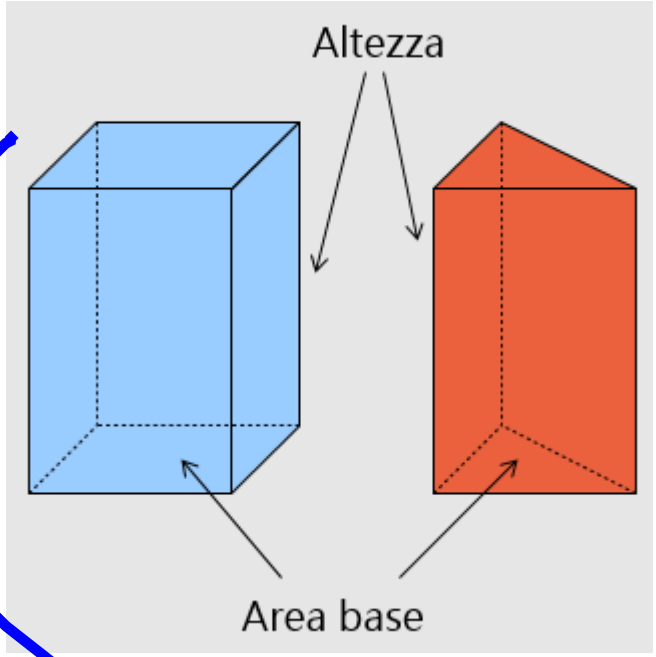
- **ImplUniModulo e ImplDueModulo possono essere compilate, in presenza di InterfacciaModulo, indipendentemente da ClasseKernel**



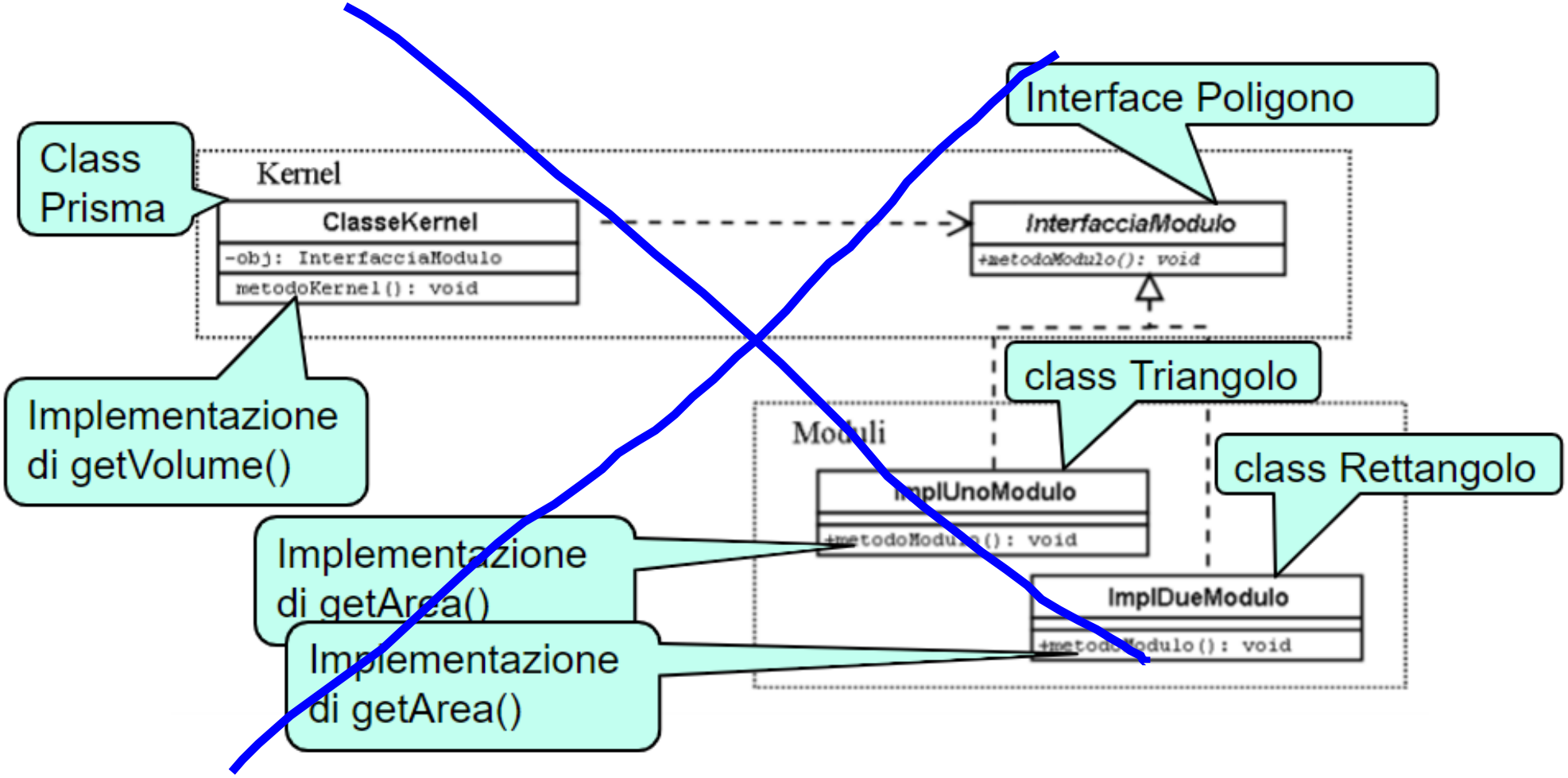
Esempio - I

Definiamo i prismi come oggetti complessi composti da una base che è un poligono:

- ClasseKernel: Prisma
- InterfacciaModulo: Poligono
- ImplementazioneModulo: Triangolo, Rettangolo



Esempio - II


Class Prisma
Kernel
ClasseKernel
-obj: InterfacciaModulo
metodoKernel(): void
Interface Poligono
InterfacciaModulo
+metodoModulo(): void
Implementazione di getVolume()
class Triangolo
Moduli
class Rettangolo
ImplUnoModulo
ImplDueModulo
Implementazione di getArea()
Implementazione di getArea()

Esempio - III

```
interface Poligono {
        public double getArea();
}

class Prisma {
        private Poligono base;
        private double altezza;

        public Prisma(Poligono base, double altezza) {
                this.base = base;
                this.altezza = altezza;
        }

        public double getVolume() {
                return base.getArea()*altezza;
        }
}
```

Esempio - IV

```
class Triangolo implements Poligono {
        private double base;
        private double altezza;
        public Triangolo(double base, double altezza) {
                this.base = base;
                this.altezza = altezza;
        }
        public double getArea() { return base * altezza / 2;}
}

class Rettangolo implements Poligono {
        private double base;
        private double altezza;
        public Rettangolo(double base, double altezza) {
                this.base = base;
                this.altezza = altezza;
        }
        public double getArea() {return base * altezza;}
}
```

Esempio - V

```
public class EreditarietaAdvanced {

   public static void main(String args[]) {

            Prisma p1 = new Prisma(new Triangolo(2, 3), 4);
            System.out.println("volume del prisma: " + p1.getVolume());

            Prisma p2 = new Prisma(new Rettangolo(2, 3), 4);
            System.out.println("volume del prisma: " + p2.getVolume());
   }
}
```