Safe Metaclass Composition Using Mixin-Based Inheritance

Noury Bouraqadi
Computer Science Laboratory (CSL)
Ecole des Mines de Douai
France

Metaclasses

- Classes which instances are also classes
  - Allow defining class properties
    - Abstract, Singleton,
    - Multiple Inheritance, Final,
    - Lazy memory allocation, Persistent instances,…
  - Implicit in Smalltalk
    - Solution for safe usage…
    - But, no reuse/composition

Outline

- Mixin Based Inheritance
- Metaclass Composition Using Mixins
- Conclusion

Need for Mixin-Based Inheritance

- Context
  - Single inheritance
  - Unrelated hierarchies
  - Same Properties
- Goal
  - Reuse shared properties
  - Avoid code duplication
  - Alternative to multiple inheritance

Single Inheritance Behind the scene

Mixin = Subclass Generator
[Bracha & Cook 90]
Example of Inheritance from Different Mixins

- Explicit Linearization on Definition:
  - ColoredBoundedPoint mixins: {Colored, Bounded}
- Lookup list
  - ColoredBoundedPoint, Colored, Bounded, Point

Outline

- Mixin Based Inheritance
- Metaclass Composition Using Mixins
- Conclusion

Main Idea and Issues

- But, we need also:
  - Compatibility (inheritance + inter-level messages)
  - Class specific properties

The Upward Compatibility issue

- foo is NOT understood by B

The Downward Compatibility issue

- bar is NOT understood by aZ

No Class Specific Properties in Smalltalk

- Unwanted property propagation: B becomes abstract!
Compatibility & Class Specific Properties

Compatibility

Class Specific Properties

Class level

Metaclass level

Class level

Metaclass level

Compatibility

Properties

AbstractClass

subclass of

instance of

Mixin

bar

subclass of

bar

self new bar

new

self error: 'Abstract class!'