

1

## *MetaclassTalk*

*a Testbed for Exploring  
Programming Paradigms*

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

2

### *Starting Point*

- **Goal: Experiment Various Paradigms**
  - Multiple-Inheritance
  - Aspect-Oriented Programming
  - Mixin-based Inheritance
  - Types
  - ...
- **Requirements: a "Ball of Mud" (© R. Fateman ;-)**
  - OO Programming Language
  - API + "framework" to ease changes

3

### *Why Extend Smalltalk?*

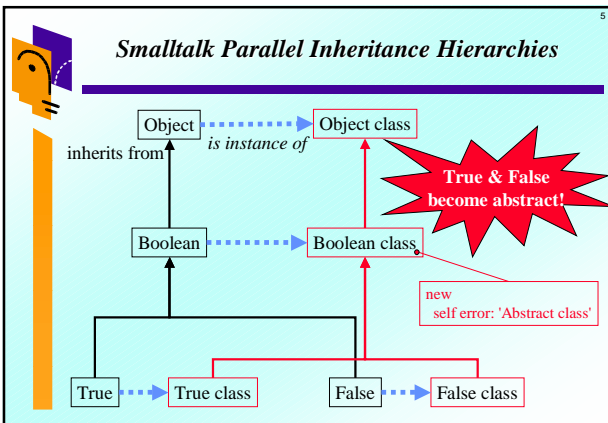
- ✓ **Powerfull (Reflective Facilities)**
- ✓ **Simple & Uniform**
- ✗ **Difficult to define new kinds of classes**
  - ✗ *Metaclasses are Implicit*
  - ✗ *Parallel Inheritance Hierarchies*
- ✗ **Complex to change the evaluation process**
- ✗ **No API or framework easing extensions**

4

### *Smalltalk Metaclasses are Implicit*

**Metaclasses: Classes which instances are also classes**

System Browser			
Kernel-Objects	Boolean	-- all --	
Kernel-Classes	False	instance creation	
Kernel-Methods	Model	documentation	
Kernel-Processes	MorphObjectOut	private	
Kernel-Magnitud	Object	objects from disk	
Kernel-Numbers	ObjectOut	plugin generatio	
Kernel-ST80 Rem	ObjectTracer		
Collections-Abstr			
Collections-Unord	instance ?	class	
Collections-Seque			
Object class			
	instanceVariableNames: ''		



6

### *Outline*

- **What is MetaclassTalk?**
- **Class Properties Reuse Using Mixins**
- **Aspect-Oriented Programming Using MetaclassTalk**
- **Conclusion**

7

## What is MetaclasTalk?

8

## Meta... what?

```

    graph TD
      Metaclass --> MetaclassTalk
      SmallTalk --> MetaclassTalk
  
```

MetaclassTalk

Programming with Explicit Metaclasses

9

## What to do with Metaclasses?

- **Class Properties**
  - Definition of new kinds of classes
- **Control of Program Execution**
  - instance creation
  - IV reads & writes
  - message sends & reception
  - method lookup & evaluation

10

## MetaclassTalk "Conceptual" Kernel

The ObjVLisp Model [Cointe 87]  
= The Smalltalk-76 Kernel + ability to subclass the root metaclass

```

    graph TD
      Object -- inherits from --> StandardClass
      StandardClass -.->|is instance of| Object
  
```

11

## Metaclass Core Protocol

Each class controls the evaluation process for its instances

- instance memory allocation (allocate)
- object creation (new = allocate o initialize)
- reading instance variables (atIV:of:)
- writing variables d'instance (atIV:of:put:)
- sending messages (send:...)
- receiving messages (receive:... = lookupFor:...@apply:...)
- method lookup (lookupFor:...)
- method evaluation (apply:...)

12

## Decomposition of a Message Sending

```

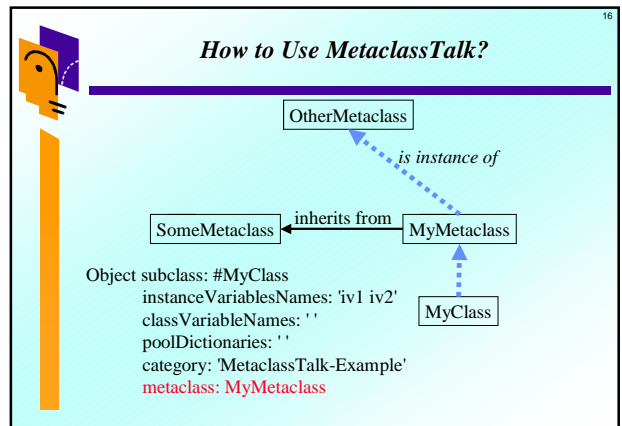
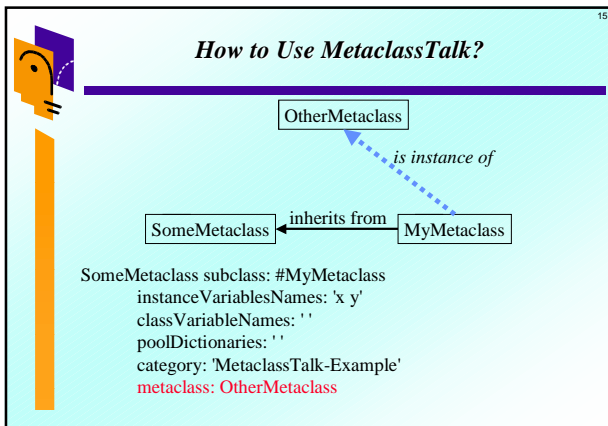
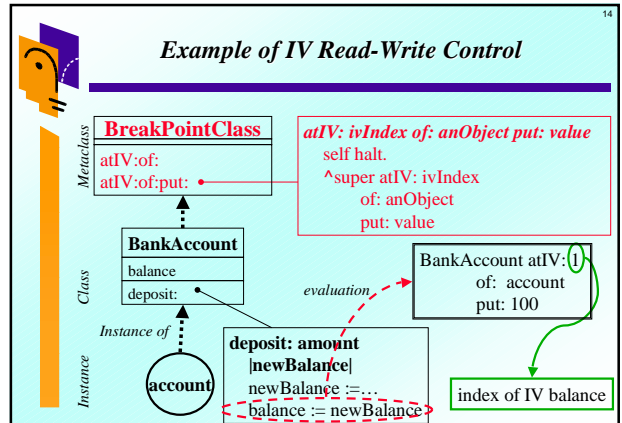
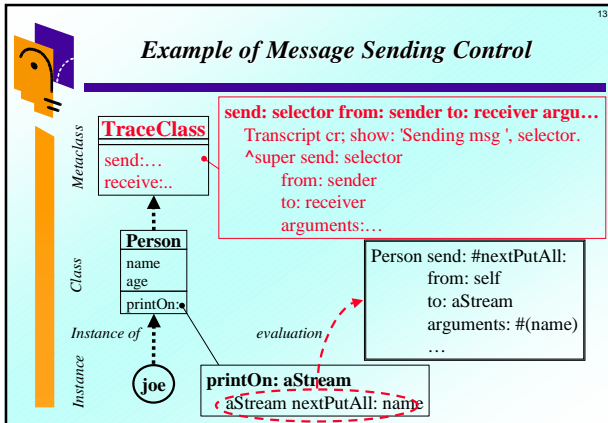
    graph TD
      anA((anA)) -- send: #foo:bar: --> A[A]
      A --> B[B]
      B --> aB((aB))
      subgraph "Method Evaluation"
        aB
      end
  
```

1 send: #foo:bar: from: self to: aB arguments: #(value1 value2)

2 receive: #foo:bar: from: self to: aB arguments: #(value1 value2)

3 lookupFor: #foo:bar: superSend: false originClass: self

4 apply: methodFooBar to: aB arguments: #(value1 value2)



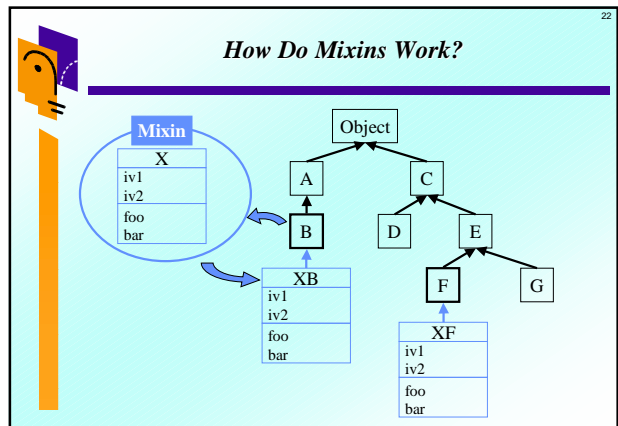
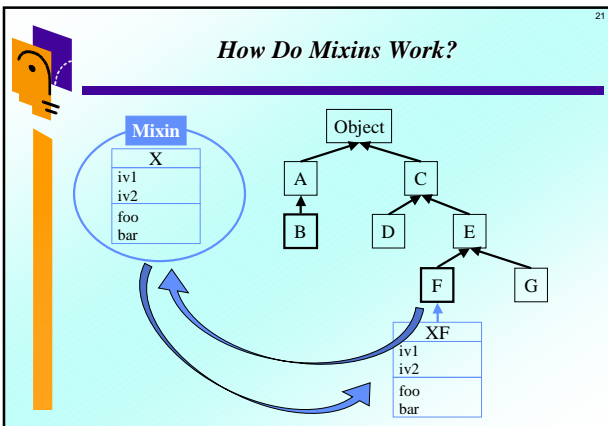
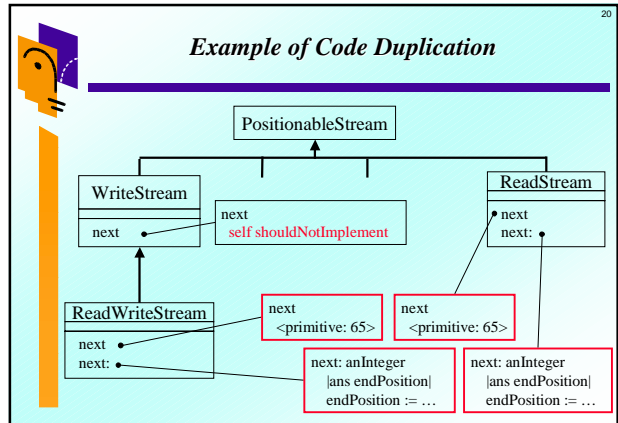
### Some Possible Uses of the MetaclassTalk's MOP

	Allocate	New	Read	Write	Send	Receive	Lookup	Apply
Abstract Class		X						X
Asynchronous Communication					X	X		
Break Point			X	X		X		
Lazy Memory Allocation	X		X	X				
Message Caching						X		
Multiple Inheritance								X
Persistent Objects (Data Base)	X		X	X				
Sole Instance (Pattern)		X						
Subject Class (Observer Pattern)				X				X
Synchronisation			X	X				X
Tracing			X	X	X	X		
Type Checking								X

### Class Properties Reuse Using Mixins

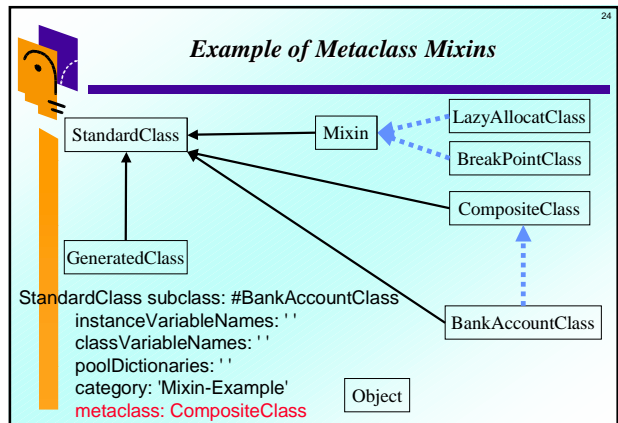
### What are Mixins?

- Context:**
  - Single inheritance
  - Unrelated class hierarchies
- Problem:**
  - Reuse shared properties
  - Avoid code duplication
- Solution:**
  - Mixin = Subclass builder [Bracha 90]

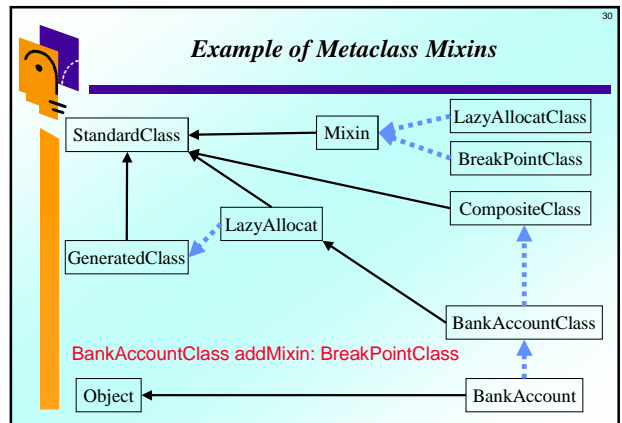
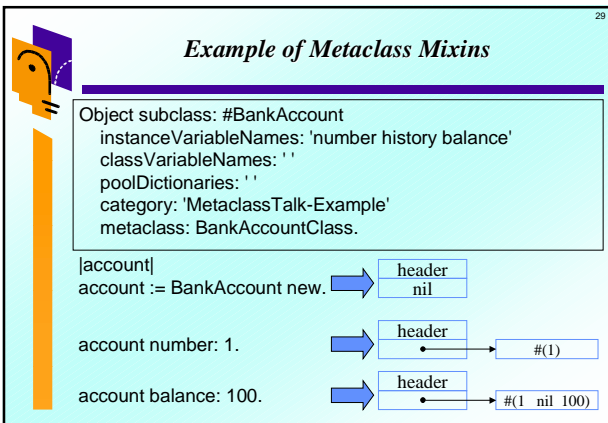
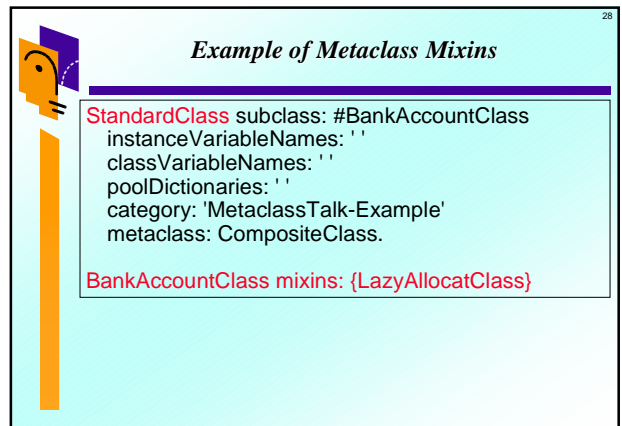
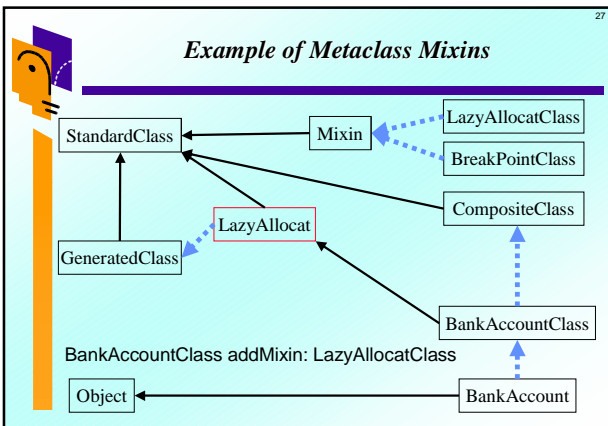
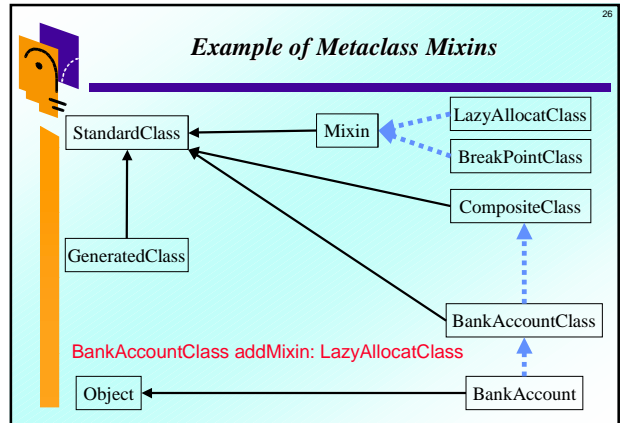
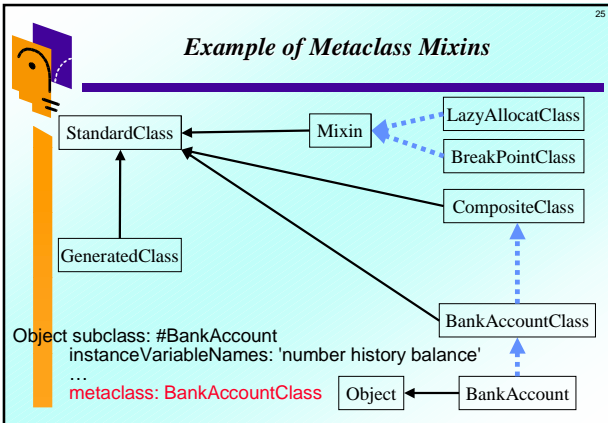


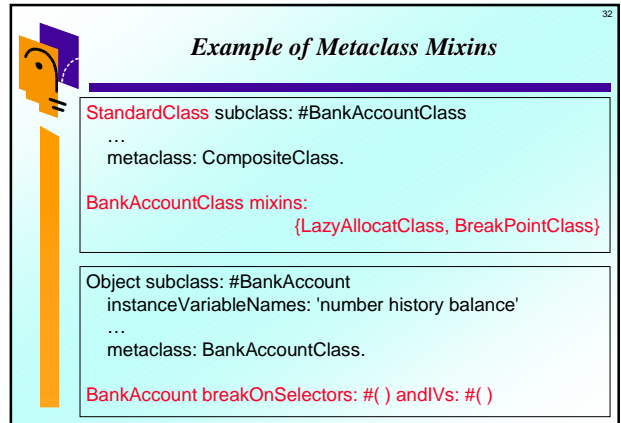
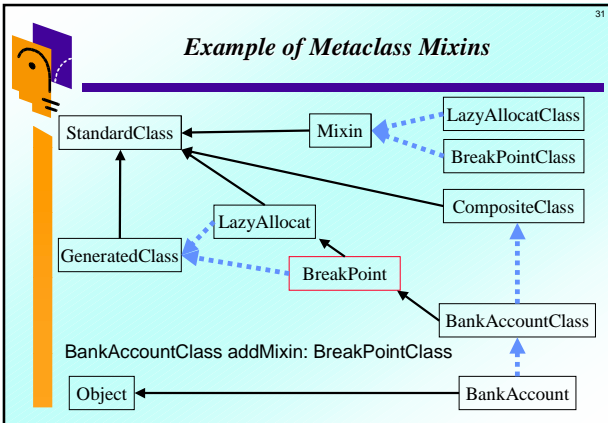
### Mixins Implementation in MetaclassTalk

- 3 metaclasses (i.e. 3 class properties):**
  - Mixins
  - Classes generated by mixins
  - Classes that make use of mixins
- Generated classes are updated on mixin change**
- Use of Smalltalk reflective facilities:**
  - Method Compilation
  - Class Building
  - Adds & Removals of Methods and Instance Variables







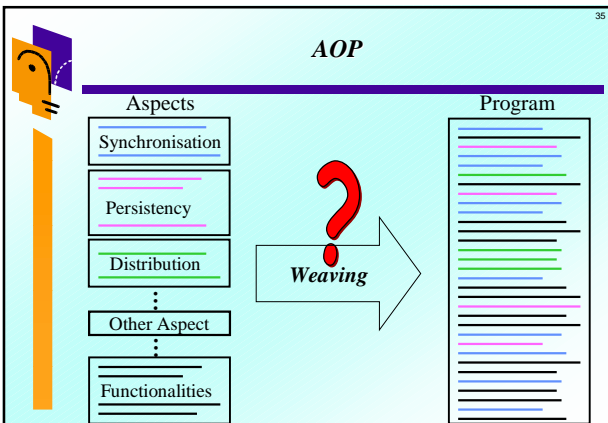


- 33
- ### Mixins: a Usefull Extension
- Experiment of a New Paradigm
  - A Tool for Composing class Properties

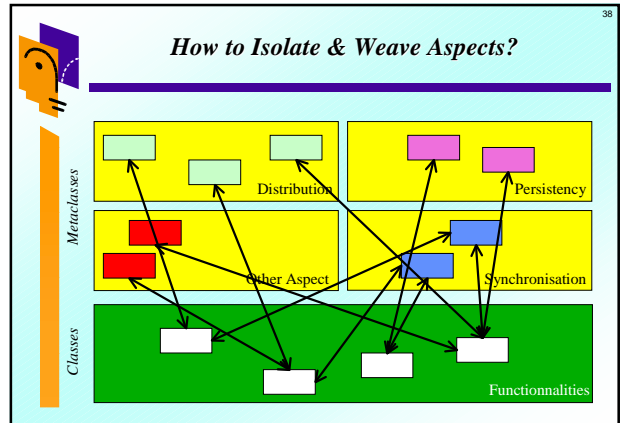
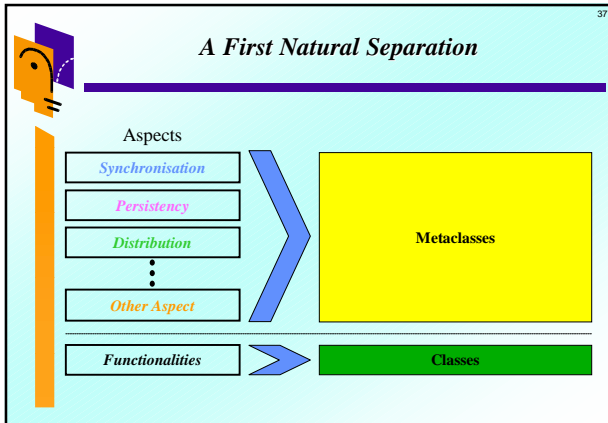
34

### Aspect-Oriented Programming Using MetaclassTalk

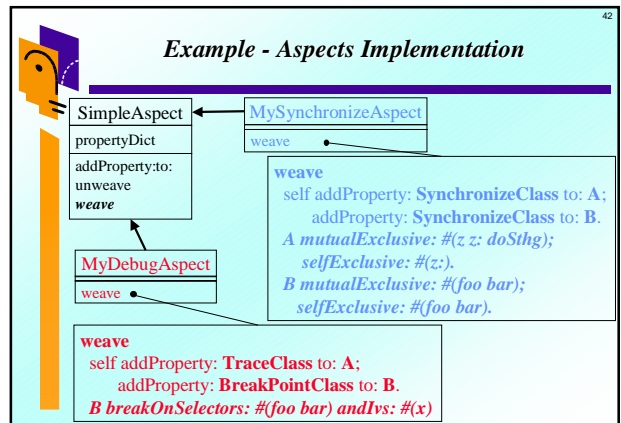
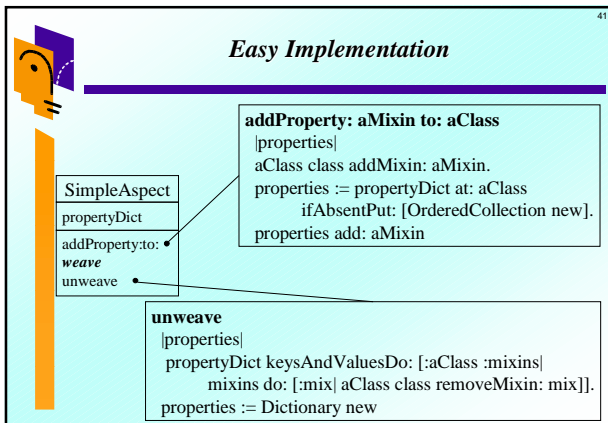
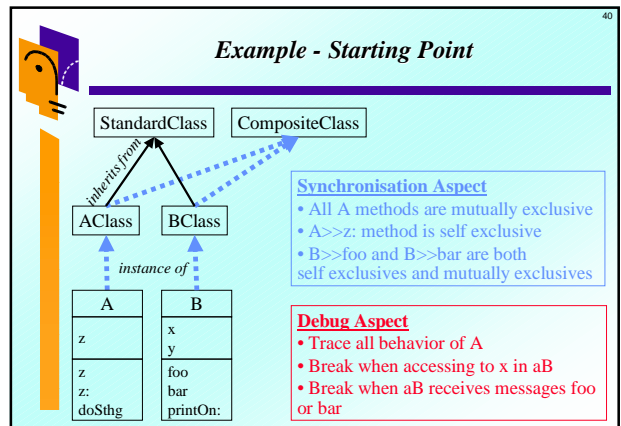
Experimenting  
a New Programming Paradigm

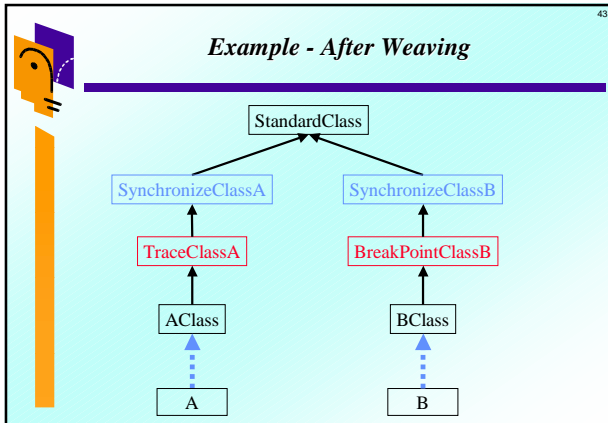


- 36
- ### Issues & Solutions
- How to define isolated aspects?
    - Metaclasses
  - How to weave aspects to build an application?
    - The "instance-of" link
    - Mixin-based inheritance
  - What about reuse?
    - Generic Metaclasses



- ### Aspects in MetaclassTalk
1. Set of Class Properties: Metaclasses
  2. Set of Classes
  3. Class-Metaclass relationships
  4. Class initialization (Properties setup)





- MetaclassTalk for AOP**
- **Easy implementation of AOP**
  - **Reuse**
    - *Metaclasses are generic*
  - **Dynamicity**
    - *Run-time weaving/unweaving*

**Conclusion**


- Summary**
- **MetaclassTalk eases experiments**
    - *Mixins*
    - *AOP*
  - **MetaclassTalk is usefull**
    - *Definition of new kinds of classes*
    - *Change the evaluation process*
      - *creation*
      - *IV access*
      - *Message handling*
  - **An implementation is available for Squeak 3.0**

- Near Future Plans**
- **Extending the library of metaclasses**
  - **Migration to the latest Squeak release**
  - **Improving Performance:**
    - *Code inlining?*
    - *Extending the VM?*
    - *Other?*

- Long Term Plans**
- **Rebuild a complete image with explicit metaclasses**
    - *New Kernel for Smalltalk*
  - **How to compose conflicting class properties?**
    - *e.g. two redefinitions of new*
    - *CLOS like before/after methods*
    - *other?*










49



***Happy Birthday Smalltalk!***

Smalltalk was born  
on September 1972



50

*Questions? Comments?*

**Download**  
<http://csl.ensm-douai.fr/MetaclassTalk>

