

# La Réflexion dans les langages à objets

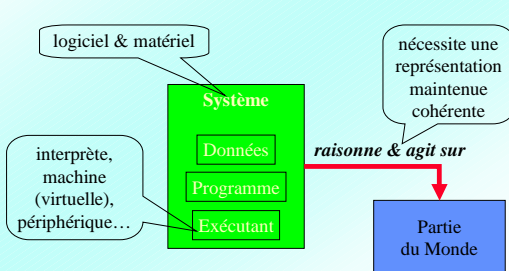
**Noury Bouraqadi**  
<http://csl.ensm-douai.fr/noury>  
 Ecole des Mines de Douai

Caen - 14 Janvier 2003

## Plan

1. **Définition de la réflexion**
  - Dans un contexte objet
2. **MetaclassTalk**
  - Extension réflexive de Smalltalk
3. **Héritage à base de Mixins**
  - Exemple d'extension de langage avec la réflexion
4. **Conclusion**

## Structure d'un système informatique



logiciel & matériel

interprète, machine (virtuelle), périphérique...

**Système**

Données

Programme

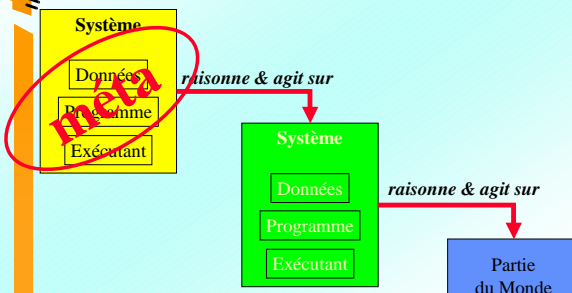
Exécutant

raisonne & agit sur

Partie du Monde

nécessite une représentation maintenue cohérente

## Méta-système



**Système**

Données

Programme

Exécutant

raisonne & agit sur

**Système**

Données

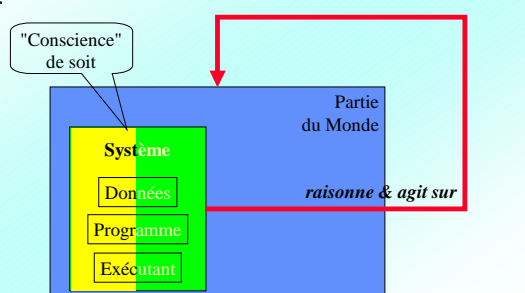
Programme

Exécutant

raisonne & agit sur

Partie du Monde

## Système réflexif



"Conscience" de soit

**Système**

Données

Programme

Exécutant

raisonne & agit sur

Partie du Monde

## Définitions

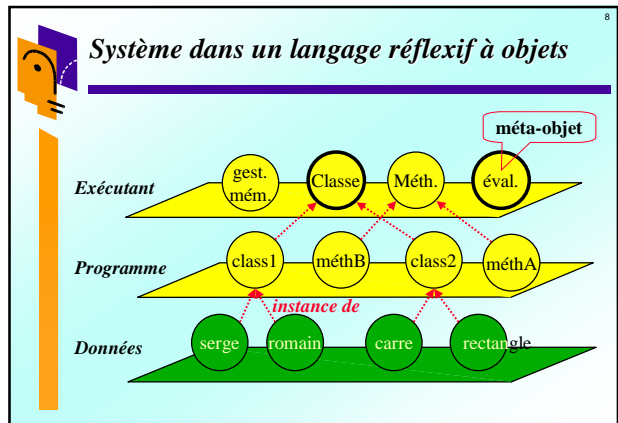
- **Réflexion : Capacité d'un système à**
  - s'auto-observer
    - Raisonner sur soit même
  - s'auto-modifier
    - Altérer sa structure (programme)
    - Altérer son comportement (exécutant)
- **Réifier ("chosifier"): Représenter sous forme explicite**
  - les éléments constituant le programme et l'exécutant
- **Langage réflexif**
  - Langage dont les constructions & "l'interprète" sont réifiés

7

## Système dans un langage à objets

- **Exécutant = machine (virtuelle), interprète, ...**
  - sémantique du langage
  - chargement/compilation de programme
  - gestion mémoire
  - ...
- **Programme**
  - classes, méthodes, champs, messages ...
- **Données**
  - objets : banque, clients, comptes, ...

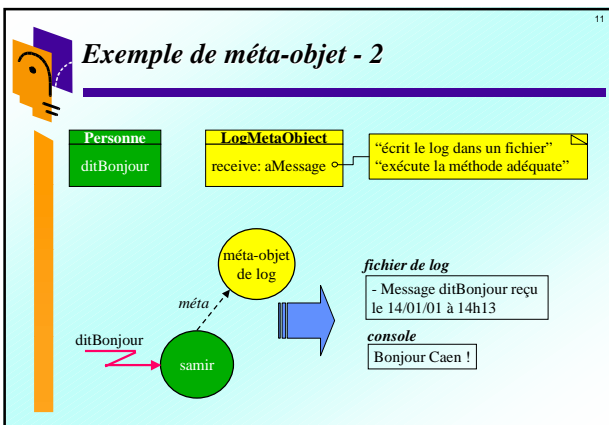
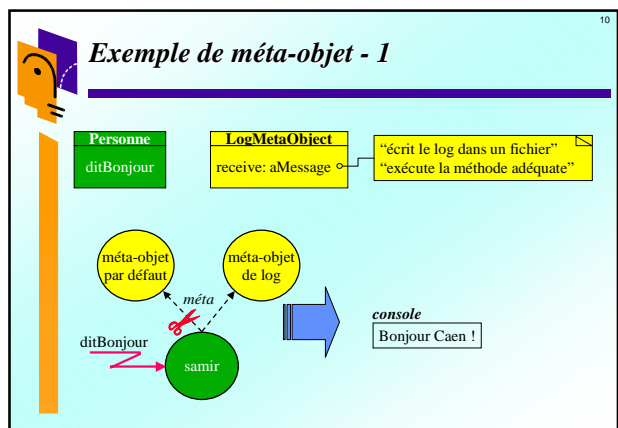
+ ou -  
réifiés  
↓  
Degré de réflexion



9

## Langage réflexif à objets

- **Deux niveaux de programmation**
  - programme
  - exécutant
- **Chaque objet est instance d'une classe**
  - les classes sont aussi des objets
  - donc instances d'autres classes : les **méta-classes**
- **Exécution à la charge d'un méta-objet (évaluateur)**
  - le méta-objet est un objet
  - contrôle les messages, les accès aux champs, ...
  - possibilité de lier un objet à un méta-objet particulier



12

## Intérêts d'un langage réflexif

- **Développement d'outils de programmation**
  - Navigateur, Débogueur, Générateurs de code, ...
- **Flexibilité à l'exécution**
  - Qualité de service, Evolutions non-prévues, ...
- **Adapter et étendre le langage**
  - Héritage multiple, Communication asynchrone, Allocation paresseuse...
- **Simplifier le développement**
  - Code générique (donc plus réutilisable)
  - Séparer les codes fonctionnels et non-fonctionnels

13

## Plan

1. **Définition de la réflexion**
  - Dans un contexte objet
2. **MetaclassTalk**
  - Extension réflexive de Smalltalk
3. **Héritage à base de Mixins**
  - Exemple d'extension de langage avec la réflexion
4. **Conclusion**

14

## Pourquoi Smalltalk ?

- **Simple : permet de focaliser sur les concepts et la conception**
  - tout est objets : y compris les classes
  - 1 action = 1 message (y compris la création d'objet)
  - Syntaxe simple (sur 1 carte postale !)
  - Pas de typage = typage dynamique = pas de "cast"
- **Puissant**
  - Bibliothèque riche copié (collections, streams, ...)
  - Nombreuses capacités réflexives
  - Extensible : écrit en lui-même (y compris outils de dev.)
  - Outils de dev. puissants (Refactoring Browser, SUnit...)

15

## Modèle objet de Smalltalk

- **\*\*\*Tout\*\*\* est objet**
  - Pas de types primitifs !
  - Manipulation de références d'objets
  - Communication par messages
- **Champs = Variables d'instances**
  - "private" à l'objet!
  - visibles dans les sous-classes
- **Méthodes**
  - toutes "public"
  - toutes retournent une valeur
- **Héritage simple**
- **Machine virtuelle**
- **Garbage collector**

16

## Syntaxe Smalltalk

- **Mots réservés**
  - **nil** référence nulle
  - **true, false** objets booléens
  - **self** (l'objet lui-même) **super** (l'objet dans le contexte de la superclasse)
  - **thisContext** partie de la pile d'exécution représentant la méthode courante
- **Caractères réservés**
  - := (ou ←) affectation
  - ^ (ou ↑) = return
  - | varTemp1 varTemp2 varTemp3 |
  - \$ caractère
  - # literal
  - . termine toute expression
  - ; cascade de messages
  - ( ) précedence
  - [ ] bloc of code
  - " commentaire"
  - ' chaîne de caractère '

17

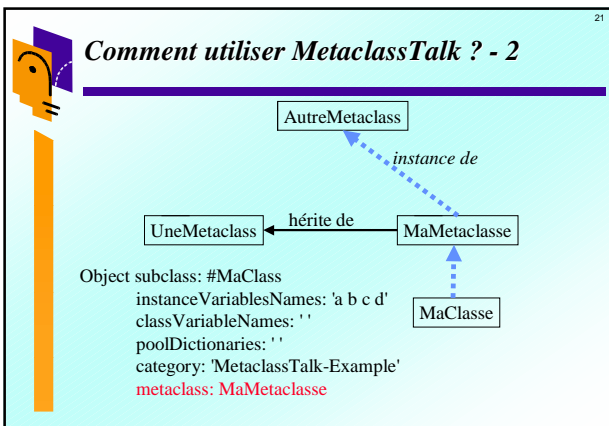
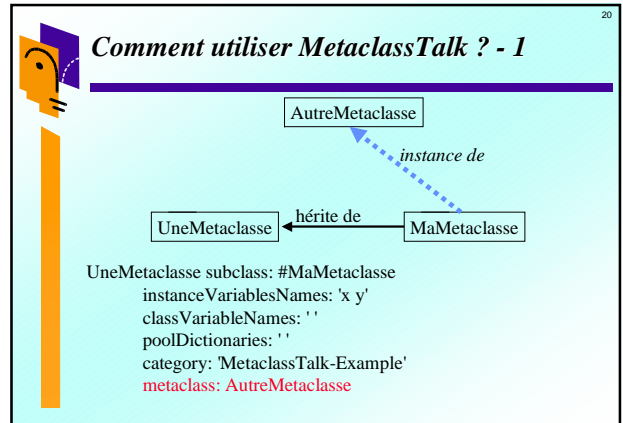
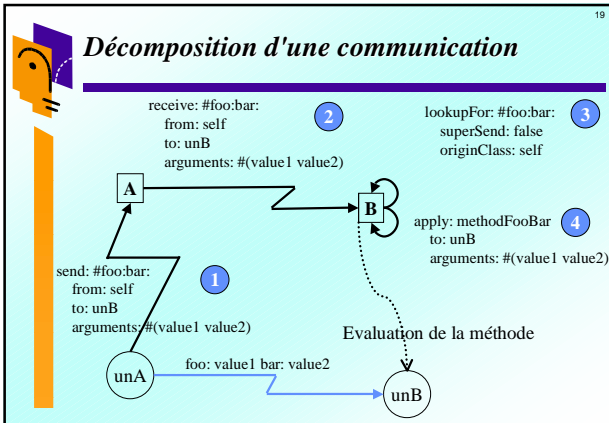
## Pourquoi étendre Smalltalk ?

- × **Difficile de définir de nouvelles sortes de classes**
  - × Métaclasses implicites
- × **Difficile de changer l'évaluateur**
- × **pas d'API ou framework simple pour les extensions**

18

## Qu'est-ce que MetaclassTalk

- **Métaclasses explicites + SmallTalk = MetaclassTalk**
  - nouvelles sortes de classes
  - nouvelles constructions dans le langage
- **Étendre le processus d'évaluation des programmes**
  - création d'objets (allocation mémoire)
  - lecture/écriture des champs
  - envoi/réception de messages
  - choix des méthodes à évaluer (héritage)
  - évaluation des méthodes
- **Par défaut la classe = méta-objet pour ses instances**

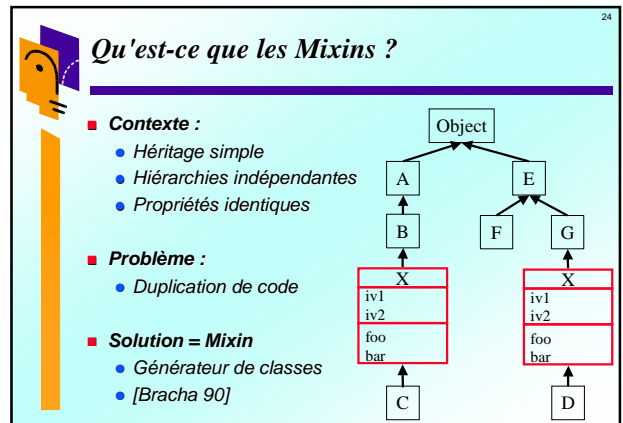


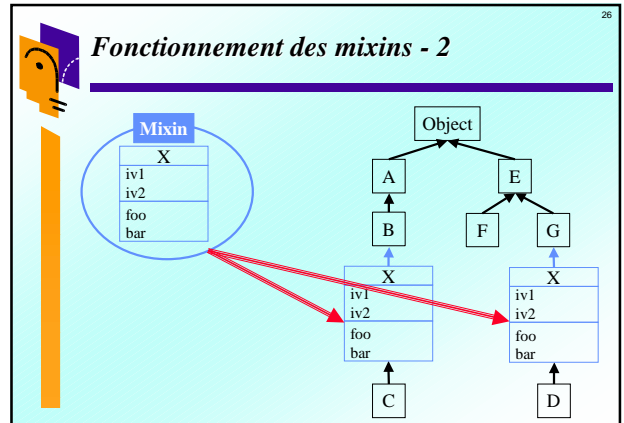
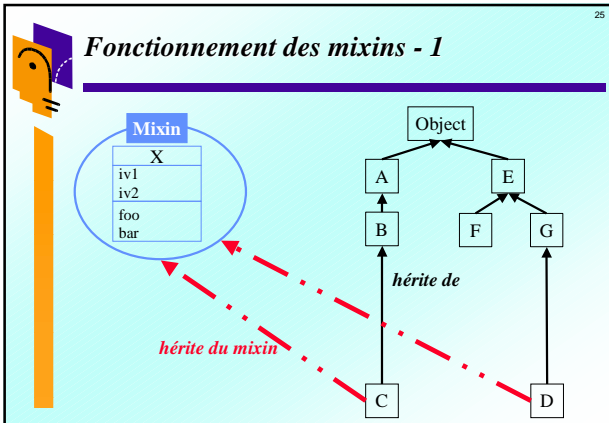
### Quelques applications de MetaclassTalk

	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				X
Synchronisation			X	X				X
Logging			X	X	X	X		
Type Checking							X	

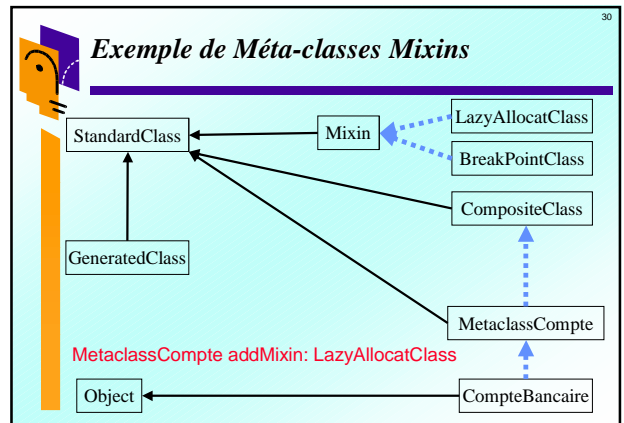
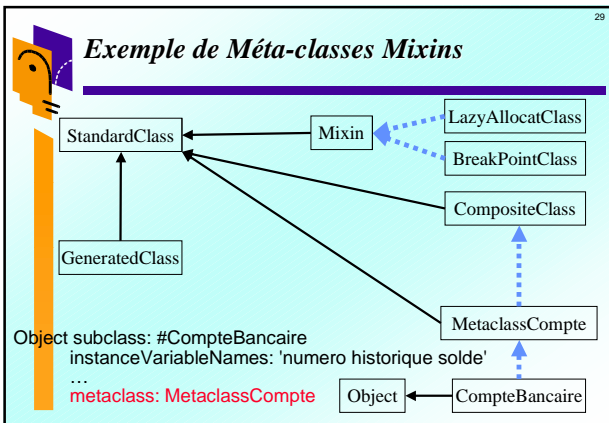
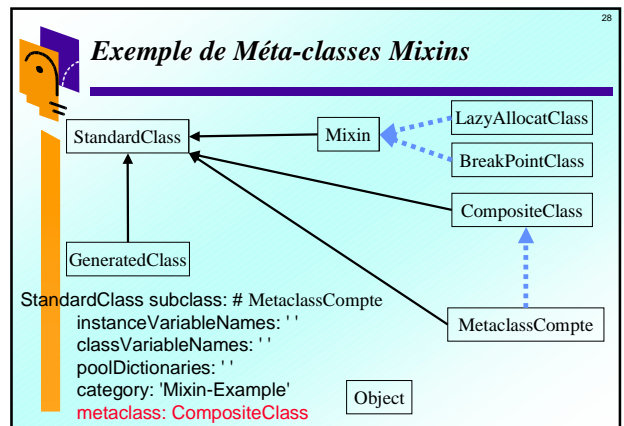
22

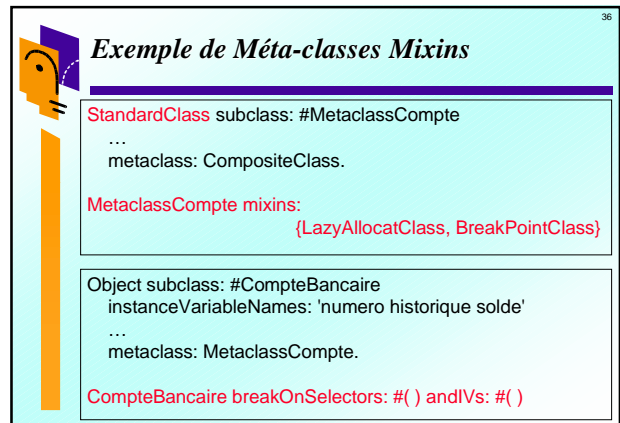
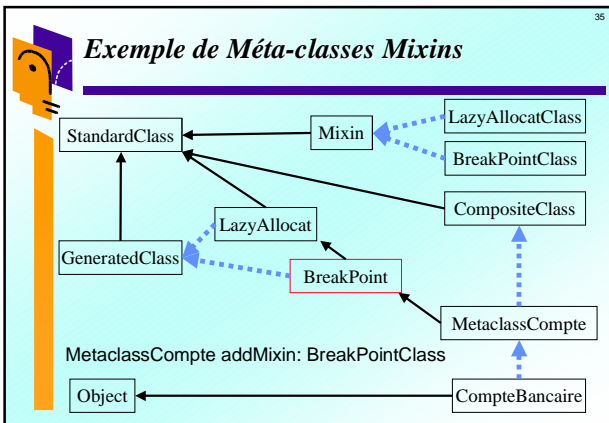
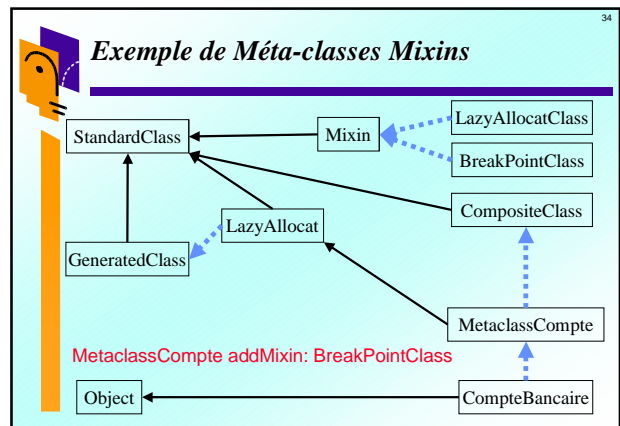
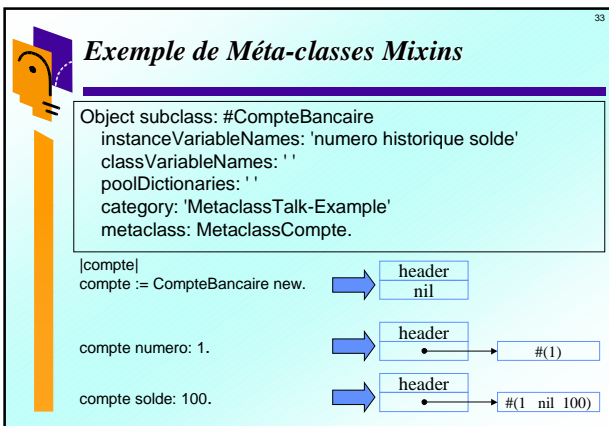
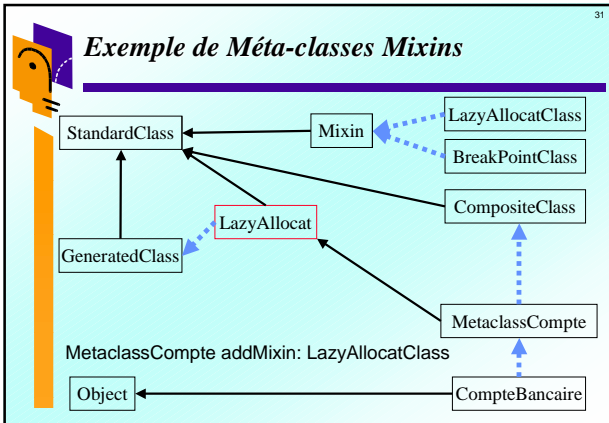
- ### Plan
- Définition de la réflexion**
    - Dans un contexte objet
  - MetaclassTalk**
    - Extension réflexive de Smalltalk
  - Héritage à base de Mixins**
    - Exemple d'extension de langage avec la réflexion
  - Conclusion**
- 23






- ### Implanter des mixins avec MetaclassTalk
- 3 méta-classes :
    - Mixin
    - GeneratedClass : Classes générées par les mixins
    - CompositeClass : Classes qui "héritent" des mixins
  - Mise à jour des classes générées si modif. mixin
  - Recours aux capacités réflexives de Smalltalk:
    - Compilation des méthodes
    - Construction des classes
    - Ajout/suppression de méthodes et champs à l'exécution





37




## Plan

---

1. **Définition de la réflexion**
  - Dans un contexte objet
2. **MetaclassTalk**
  - Extension réflexive de Smalltalk
3. **Héritage à base de Mixins**
  - Exemple d'extension de langage avec la réflexion
4. **Conclusion**

38




## Bilan

---

- **La réflexion est utile**
  - outils, extensions, qualité de service, simplifier le développement
- **Exemple d'extension du langage**
  - Héritage à base Mixin
  - Utilisation au niveau méta
  - D'autres extensions sont possibles
- **Implantation disponible sous Squeak 3.2 (Smalltalk libre)**

39



## Perspectives

---

- **Différentes extensions du langage**
  - Agents
  - Composants logiciels
- **Résoudre les problèmes de performance**
  - code inlining
  - Etendre la machine virtuelle
- **Applications**
  - Systèmes distribués
  - Systèmes embarqués

40



## Merci pour votre attention

Questions? Commentaires?



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