

École Doctorale Sciences et Technologies de l'Information et des Matériaux

Soutenance de thèse de doctorat

9 juillet 2007

Expression et composition des motifs de conception avec les aspects

Simon Denier

Directeur de thèse : Pierre Cointe

Équipe OBASCO
INRIA Rennes – Bretagne Atlantique
Laboratoire d'Informatique de Nantes Atlantique

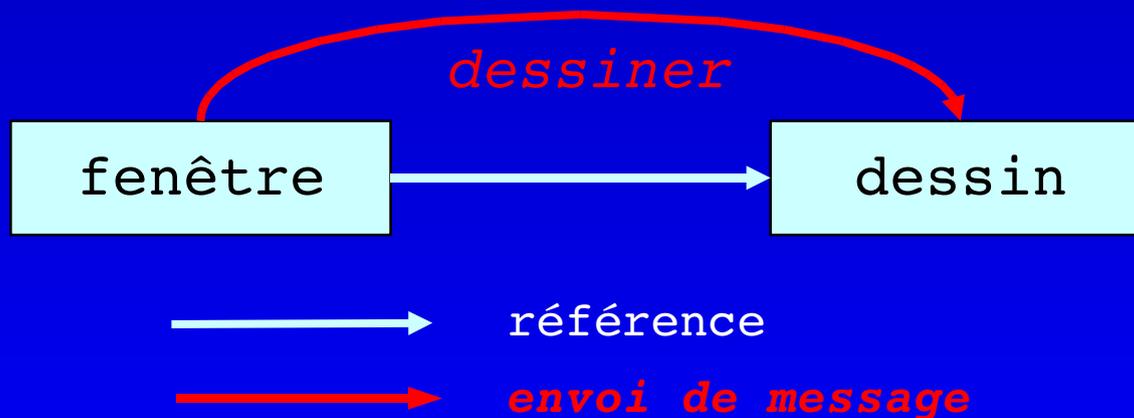
École des Mines de Nantes
Université de Nantes

Plan

- Contexte : implémentation des motifs
- Problématique
- Démarche et contributions
- Conclusions et perspectives

Programmation par objets (PPO)

- Concepts
 - Objet : cellule de données et opérations associées
 - Communication : envoi de messages entre objets
 - Polymorphisme : sélection du traitement du message par l'objet
 - Classe : descripteur d'objets similaires, module
- Paradigme utilisé dans de nombreux projets

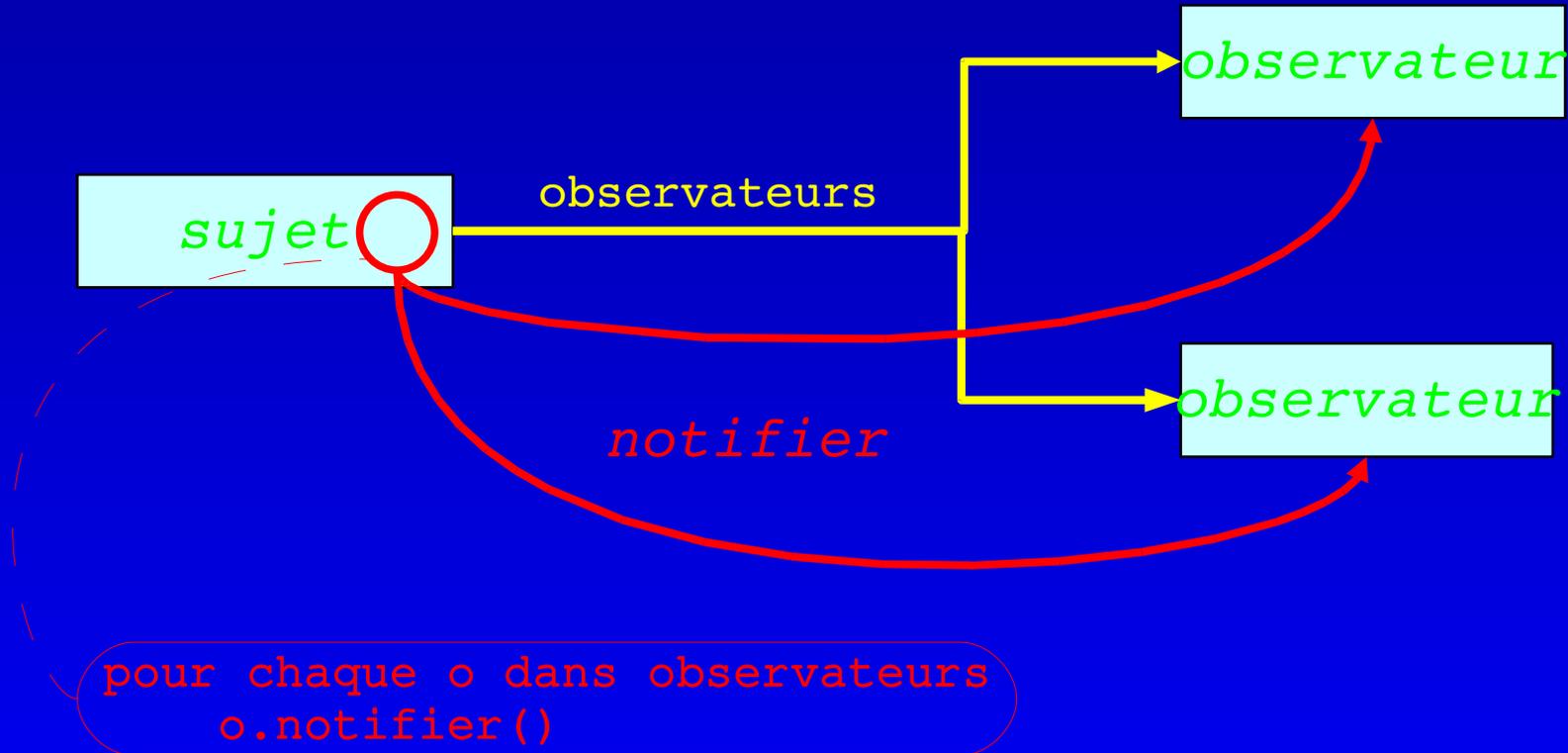


Patrons et motifs de conception

- Capturer l'expérience de la PPO
- Patron de conception
 - Description en langage naturel
 - Problème particulier et récurrent
 - Solution éprouvée mais pas de code réutilisable
 - Contraintes liées au patron
- Motif de conception
 - Partie solution du patron
 - Code source : par imitation des exemples du motif

Motif Observateur (extrait)

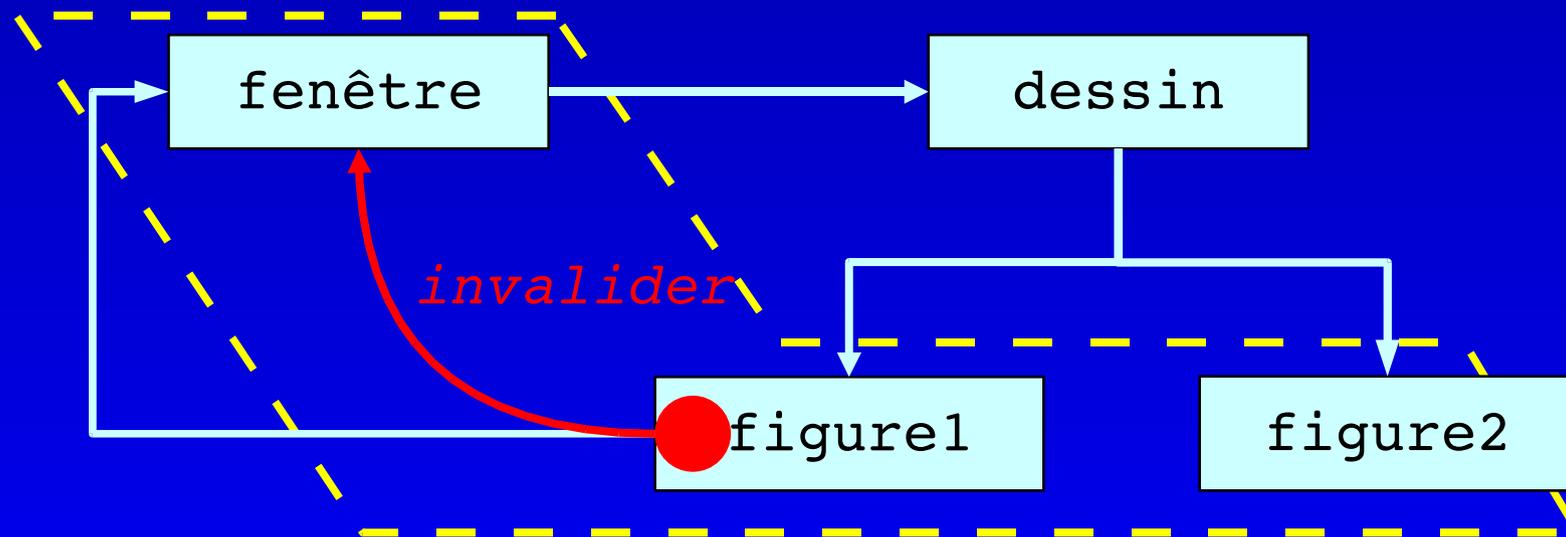
- Notifier des changements d'un sujet à des observateurs
- Constitution d'un motif : **Rôles**, **Structure**, **Collaborations**



Motif Observateur dans JHotDraw

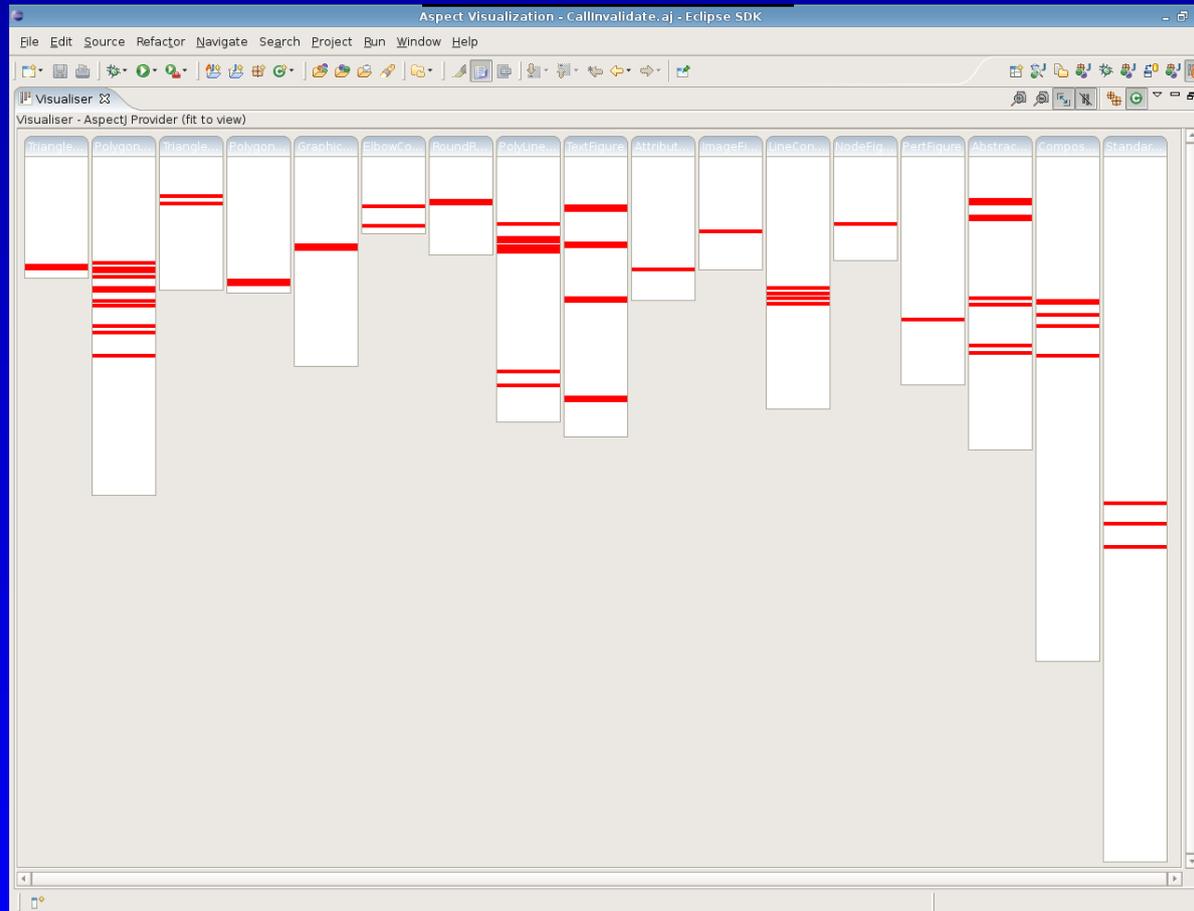
- JHotDraw : cadre (squelette réutilisable d'application) en Java pour dessin vectoriel
- Objectif : rafraîchissement de la fenêtre

motif Observateur



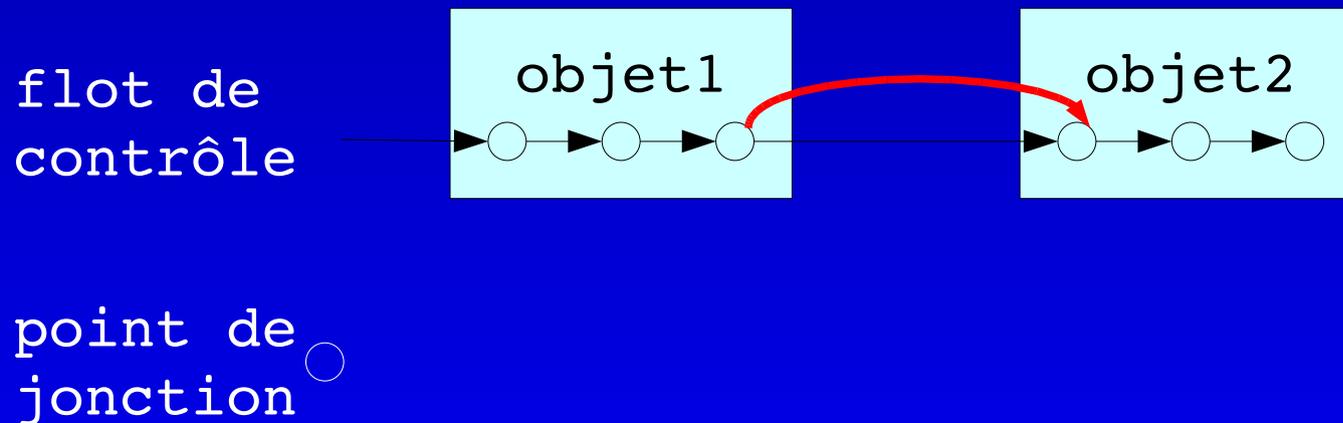
Problème : dispersion du motif Observateur

- JHotDraw : notification par les sujets du motif Observateur



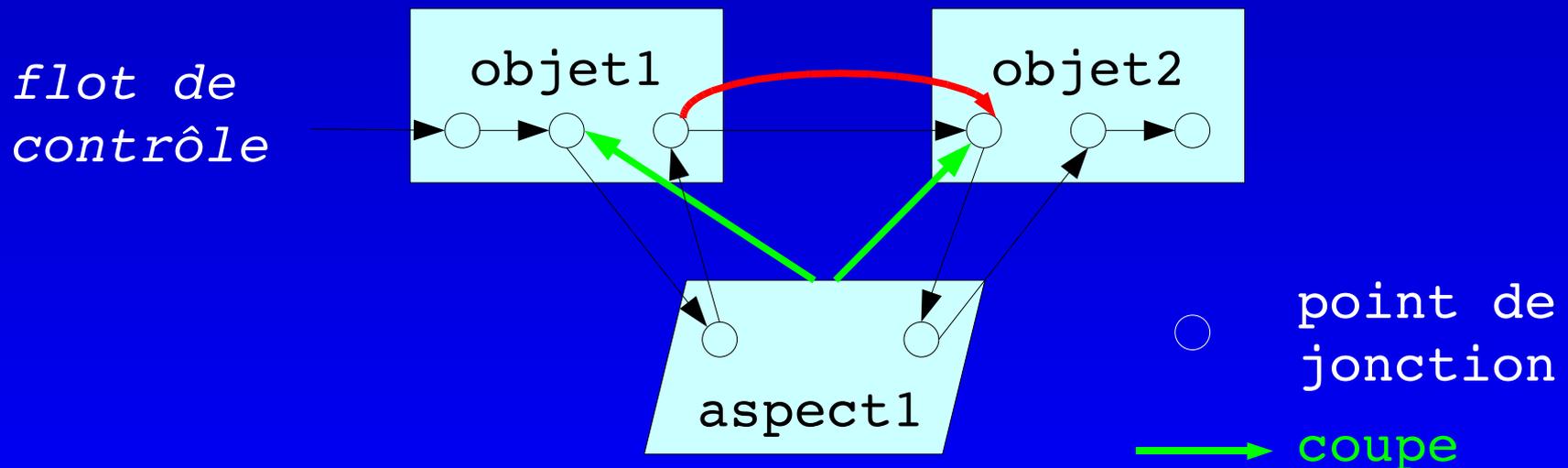
Programmation par aspects

- Modulariser le code dispersé et mélangé dans les classes
- Concepts (aspect comportemental)
 - Exécution = trace d'événements : points de jonction

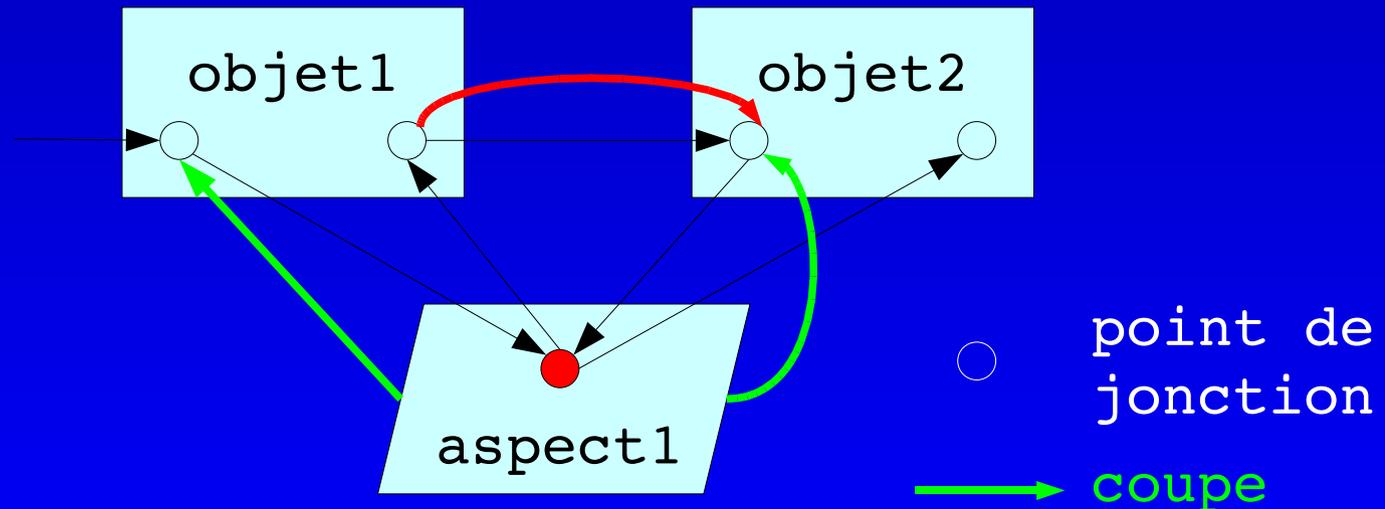
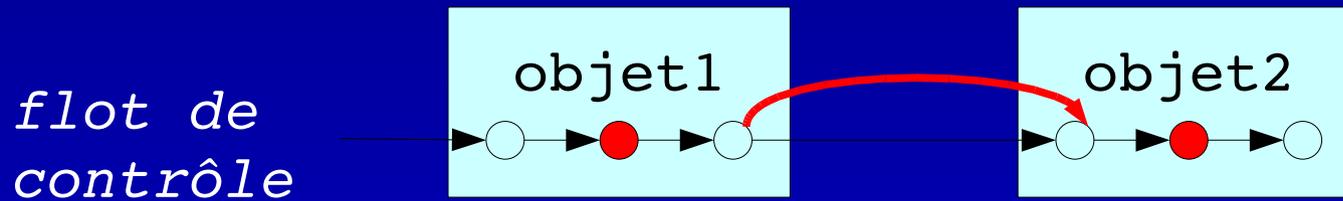


Aspect, coupe et action

- Concepts (aspect comportemental)
 - Déclaration des points de jonction à intercepter : coupes
 - Réaction à la capture d'un point de jonction : action
- Exemple d'aspect incrémental



Factorisation du code dispersé



Implémentation des motifs avec les aspects

- Hannemann et Kiczales, OOPSLA'02
 - Étude extensive avec AspectJ sur le catalogue des patrons
- AspectJ
 - Extension de Java pour les aspects
 - Robuste pour l'industrie
- Bilan sur les 23 motifs individuels du GoF
 - 17 motifs modularisés (~ 70%)
 - dont 12 motifs réutilisables (50%)

Observateur AspectJ (Hannemann)

```
aspect Observateur {
    List<Fenetre> listeObservateurs(Figure f) { ... }

    pointcut invalider(Figure fig) :
        call( void Figure.deplacer(..) ) && target(fig);

    after(Figure figure): invalider(figure) {
        Iterator<Fenetre> iter = listeObservateurs(figure).iterator();
        while ( iter.hasNext() ) {
            rafraichir(figure, iter.next()); }
    }

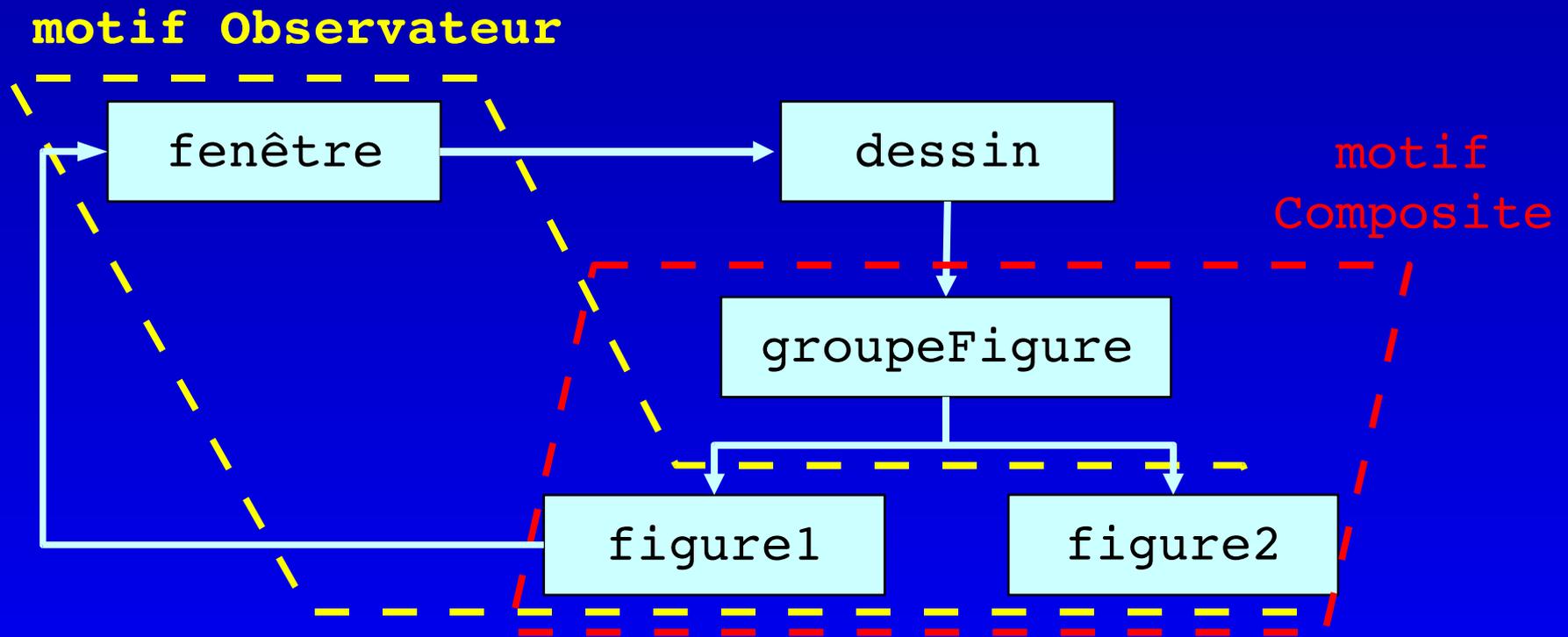
    void rafraichir(Figure figure, Fenetre fenetre) { ... }
}
```

Plan

- Contexte
- **Problématique : composition des motifs**
- Démarche et contributions
- Conclusions et perspectives

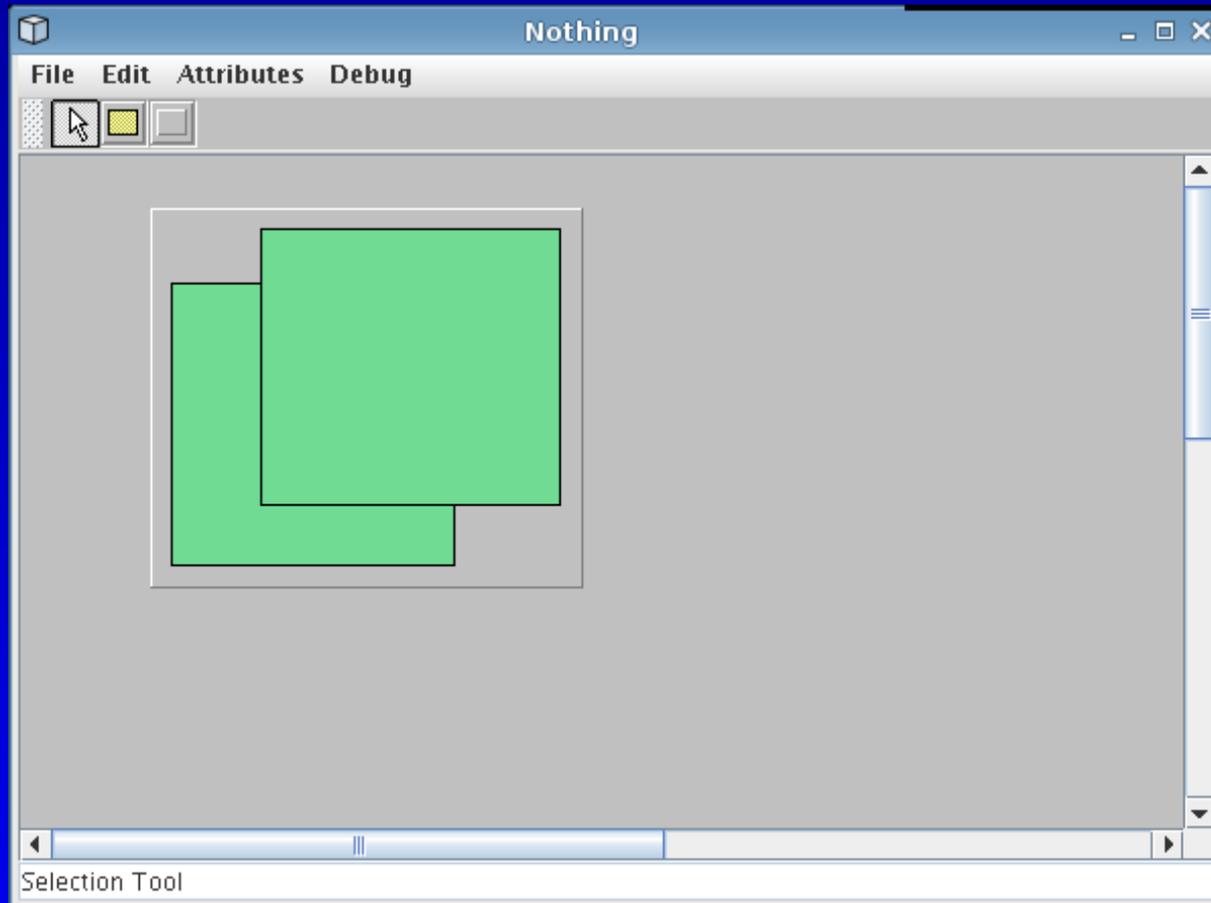
Composition de motifs

- Superposition de deux rôles sur une classe
- Exemple : ajout classe GroupeFigure (motif Composite)



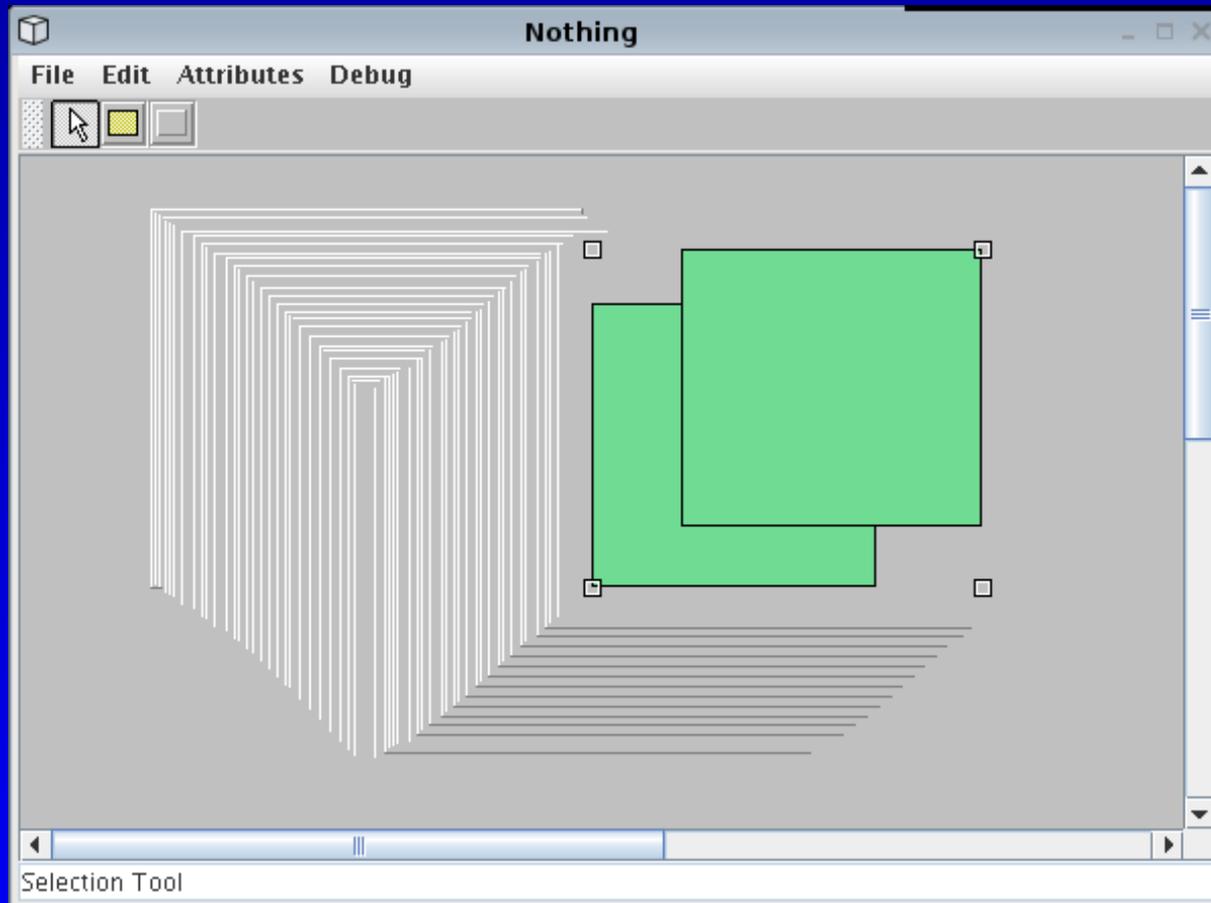
Exemple GroupeFigure

- GroupeFigure décore les figures avec une bordure



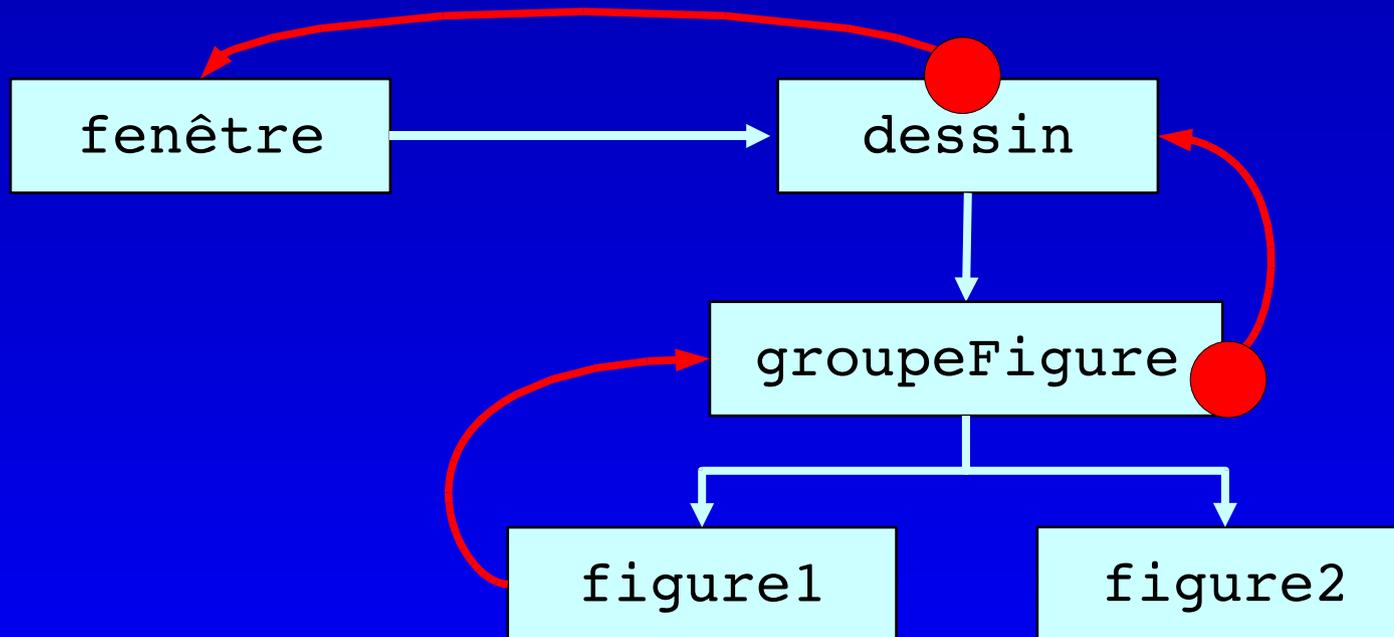
Interaction Composite et Observateur

- Observateur doit prendre en compte la bordure de GroupeFigure



Résolution et impact sur les objets

- Restructuration du motif Observateur
 - Notification récursive par Figure de son parent
 - Modification de la notification dans GroupeFigure
- Problème : impact sur plusieurs classes



Plan

- Contexte
- Problématique
- **Démarche et contributions**
- Conclusions et perspectives

Démarche de la thèse

- Base d'exemples pour la composition
 - Cas général (catalogue GoF) : faible nombre, pas de description
 - Cas particuliers : à partir de la restructuration des motifs de JHotDraw avec des aspects
- Modularisation des implémentations des motifs
 - Application des solutions de Hannemann
 - Adaptation, recherche de nouvelles solutions adéquates
- Composition de motifs
 - Catégorisation des compositions de motifs
 - Implémentation de la composition de motifs modularisés

Guide de restructuration d'un motif

Rôles

```
aspect ObservateurRelais {
```

```
    interface Sujet { }
```

```
    interface Observateur { }
```

- Interfaces Java

Structure

```
List<Observateur> listeObservateurs(Sujet s) {...}
```

```
public void ajout(Sujet s, Observateur o) {...}
```

- Cardinalité ?
- Portée ?
- Interface publique ?

Collaborations

```
pointcut changement(Sujet s) : ... ;
```

```
after(Sujet s): changement(s) {
```

```
    while ( ... ) { notifier(s, o); }
```

```
}
```

```
void notifier(Sujet s, Observateur o) {...} }
```

- Action relais ou centrale ?
- Polymorphisme objet ou ad hoc ?

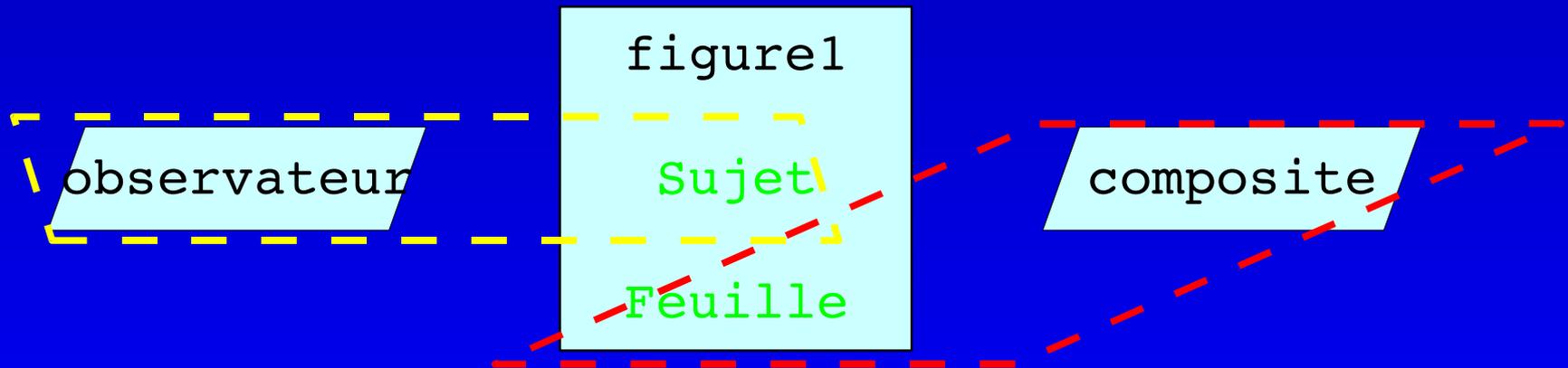
Idiomes d'implémentation des motifs

- Idioms : règle d'usage particulière à un langage
- Relation entre objets
 - Suivant la cardinalité des objets et de la relation
 - Suivant la portée des coupes
- Propriétés particulières à certains motifs :
 - Décorateur : dynamicité, mais pas la récursivité !
 - Visiteur : généricité, portée, état concurrent
- Polymorphisme ad hoc avec les coupes (Visiteur)
- [Denier et Cointe, L'Objet 2006]

Composition de motifs

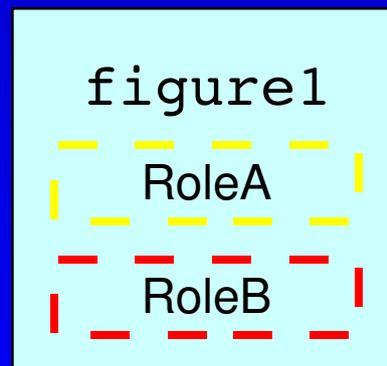
- Superposition de deux rôles sur une classe
- Base de la composition avec les aspects
 - Composition des rôles dans les classes
 - Rôle = interface marqueur, pas d'implémentation

```
class Figure implements Observateur.Sujet, Composite.Feuille
```



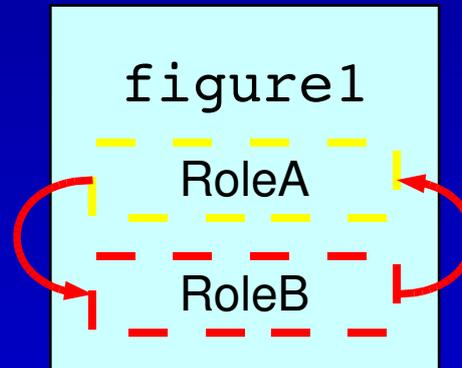
Catégoriser les compositions

- Modalité de composition
 - Degré d'indépendance entre les motifs composés
 - Indépendant du langage (objet, aspect)
- 5 modalités
 - Cohabitation, collaboration, utilisation, partage, interaction
- Exemple : cohabitation
 - Composition sans effet l'un sur l'autre

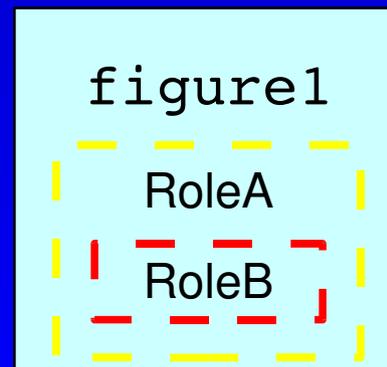


Collaboration, utilisation

- Collaboration : composition par envoi de messages
 - Aspects Composite+Visiteur [Denier et Cointe, L'Objet'06]

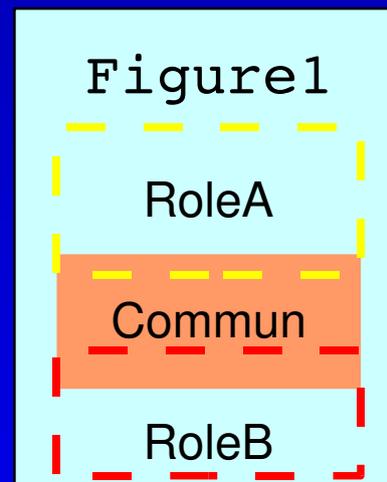


- Utilisation : spécialisation d'un motif pour un autre
 - Objets Composite+Visiteur



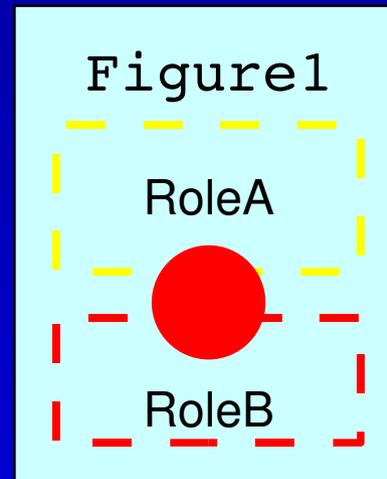
Partage

- Réutilisation d'implémentation
 - Cas objet : opportunité de réutiliser un motif déjà implémenté
 - Risque de mélange des spécifications (interaction)
 - Restructuration par aspect structurel pour réutilisation modulaire



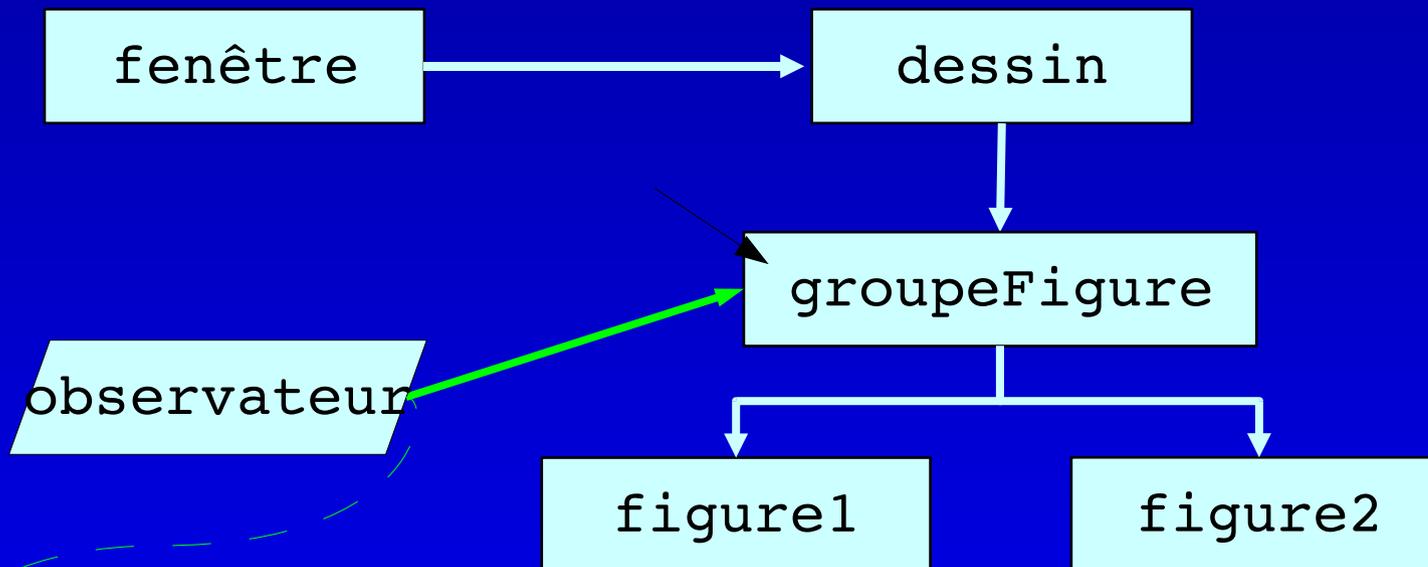
Interaction

- Motifs incompatibles en l'état – modification invasive de leur implémentation



Interaction par les coupes (1)

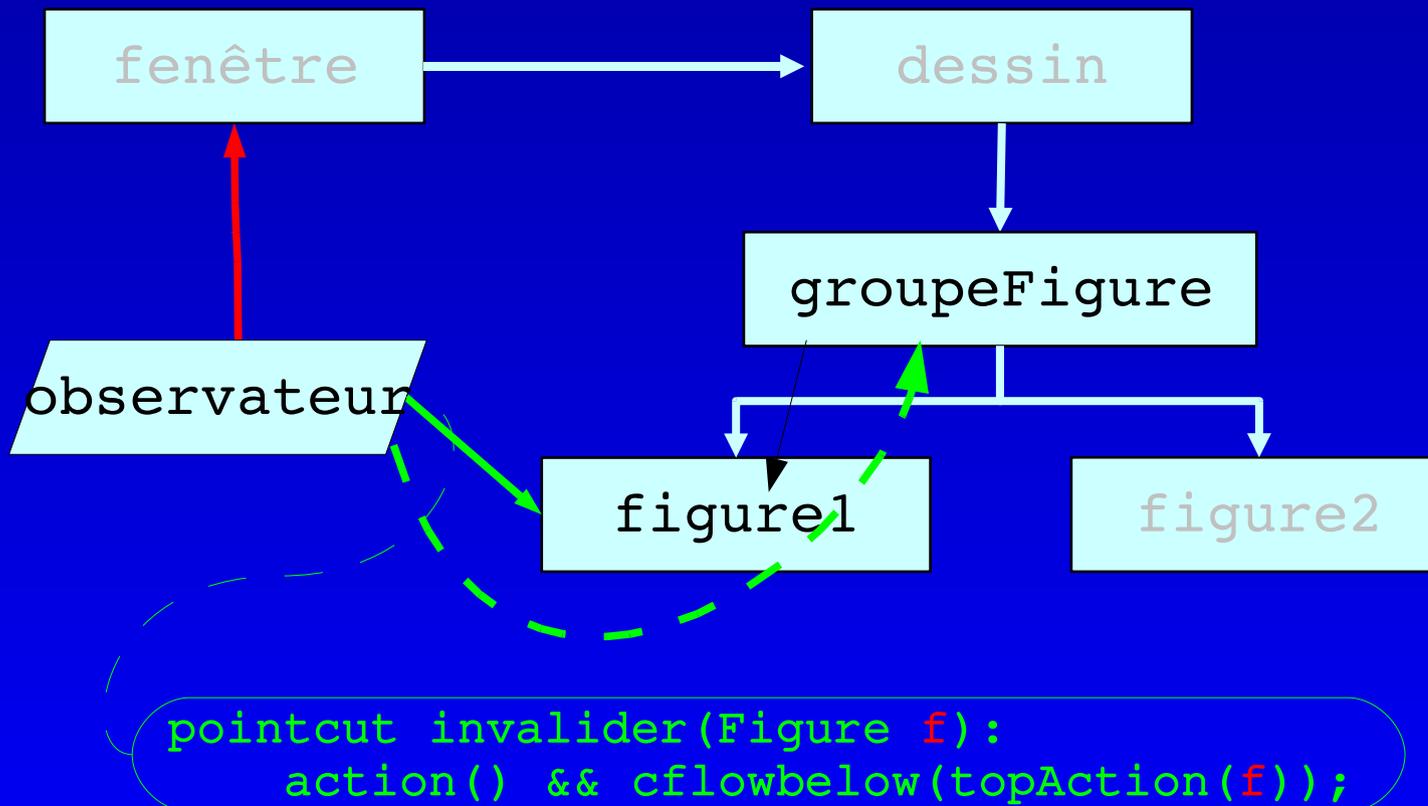
- Hypothèse : flot de contrôle récursif dans l'arbre
- 1ère étape
 - Coupe sur la figure en tête du flot : capture de groupeFigure



```
pointcut action(): execution(void Figure.deplacer(..));
pointcut topAction(Figure f):
    this(f) && action() && !cflowbelow(action());
```

Interaction par les coupes (2)

- 2ème étape : dans le flot de contrôle de groupeFigure
 - Coupe sur les changements dans figure1
 - Notification passant groupeFigure en paramètre



Impact de la résolution

- Aucun impact sur Dessin et GroupeFigure
- Impact sur Observateur
 - Modification des coupes pour le flot de contrôle récursif
- [Denier et Cointe, Software Composition 2006]

Plan

- Contexte
- Problématique
- Démarche et contributions
- Conclusion et perspectives

Conclusion

- Modularisation des motifs par les aspects
 - Utilisation de divers idiomes pour émuler relations et propriétés
 - Idiomes : prédominants pour la restructuration avec AspectJ
- Composition des motifs par les aspects
 - Composition simple, plus modulaire
 - Résolution d'interactions avec faible impact
- Application des aspects
 - Non systématique
 - Résolution des problèmes dépassant l'échelle de la classe

Autres contributions de la thèse

- Exposition des problèmes de la composition des motifs en programmation par objets
 - Basée sur des exemples concrets de JHotDraw
- Évaluation des mécanismes structurels d'AspectJ
 - Composition et interactions structurelles
- Méthode de programmation des aspects
 - Critères et principes pour la restructuration comportementale et structurelle avec AspectJ

Perspectives

- AspectJ
 - Amélioration du langage (interactions structurelles, coupes)
- Implémentation des motifs
 - Étude avec d'autres langages à aspects
 - Comparaison avec techniques de transformation
- Composition des motifs
 - Étude extensive : outils de détection des motifs pour explorer les cas particuliers – constitution d'un catalogue
 - Implémentation comparée des objets et des aspects

Publications

- [Denier et Cointe, L'Objet 2006]
 - Modularisation : présentation des idiomes AspectJ
 - Composition : résolution d'interactions comportementales
- [Denier et Cointe, Software Composition 2006]
 - Cas d'étude JHotDraw
 - Impact de la composition des motifs sur les objets
 - Restructuration par les aspects
- [Denier, L'Objet 2005]
 - Composition et interactions structurelles avec AspectJ