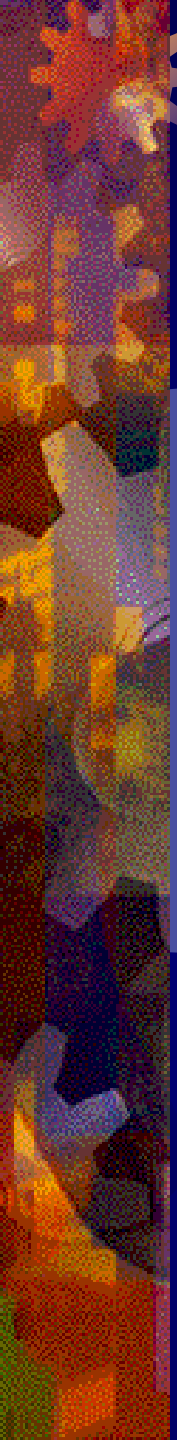


Design Patterns

Cours IUT 7 mars 2001

Arnaud Nauwynck & Nédra Mellouli
arnaud.nauwynck@socgen.com

Introduction

- Thèse de Erich Gamma
- Edité en un livre
- Auteurs: E. Gamma, R. Helm, R. Johnson, J. Vlissides
- Livre devenu best-seller informatique
- Vision nouvelle(?) et incontournable

Plan

- Orienté-Objet & Design Patterns
- Généralités sur les Design Patterns
- Étude de Cas
- Utilisation & méthode d’apprentissage

Conclusion

Mots - clefs

- Titre : Design Patterns
- Catalogue de Modèles de conception réutilisables
- **Elements of Reusable Object-Oriented Software**
- Mots-clefs = architecture, organisation, rôles, simple, intelligible, éprouvé, flexible, concepts OO, modulaire, (ré)utilisable ...

Objectifs / Positionnement

- Pré requis
  - connaissance Orientée-Objet
  - Langage OO : C++ / Java..
  - Concepts de Librairies
- Buts
  - Concepts abstraits
  - Vocabulaire des concepts (complémentaire d'UML)
  - Nouvelle vision du monde du logiciel
- Non – Buts
  - Pas liés à un langage précis
  - Pas un livre d'apprentissage, pas de recettes !

L'Héritage en Orienté-Objets

- 3 Façons de réutiliser les Objets
  - Héritages (d'interface / de code)
  - Composition
  - Templates (généricité de types..)
- **Héritage de code** : souvent utilisé **à tords**
- L'**héritage d'interface** : Light Motifs des Design Patterns

Limitation d'une approche naïve de l'Orienté-Objets

- Recensement « Merisien » des objets
  - Données, pas Interfaces !
  - Objets fonctionnels seulement, Pas informatiques!
- Héritage de code
  - forte corrélation classes / sous-classes..
- Traitements mélangés entre classes
  - Grande difficulté de compréhension
  - insuffisance des diagrammes de classes de UML

Buts : Rôles des objets

- Limitation des dépendances / connaissances entre objets
- Introduction de **dépendances dynamiques tardives** (« late binding »)
- Par opposition : suppression des **dépendances à la compilation**..
- Rôles des objets systématiquement épurés, et définis par des interfaces

1 rôle => 1 interface + délégation à 1 objet
Possibilité de changement ouverte

23 Patterns / 3 classifications

Des objets où, comment, pourquoi faire ?..

⇒ Identification et rôles des objets et des relations

1) Modèles **Créateurs**

Créer un objet / Accéder à un objet

2) Modèles **Structuraux**

Combiner les objets en structures

3) Modèles de **Comportement**

Utiliser les objets pour implanter des fonctionnalités

Etude de cas : 5 Problèmes

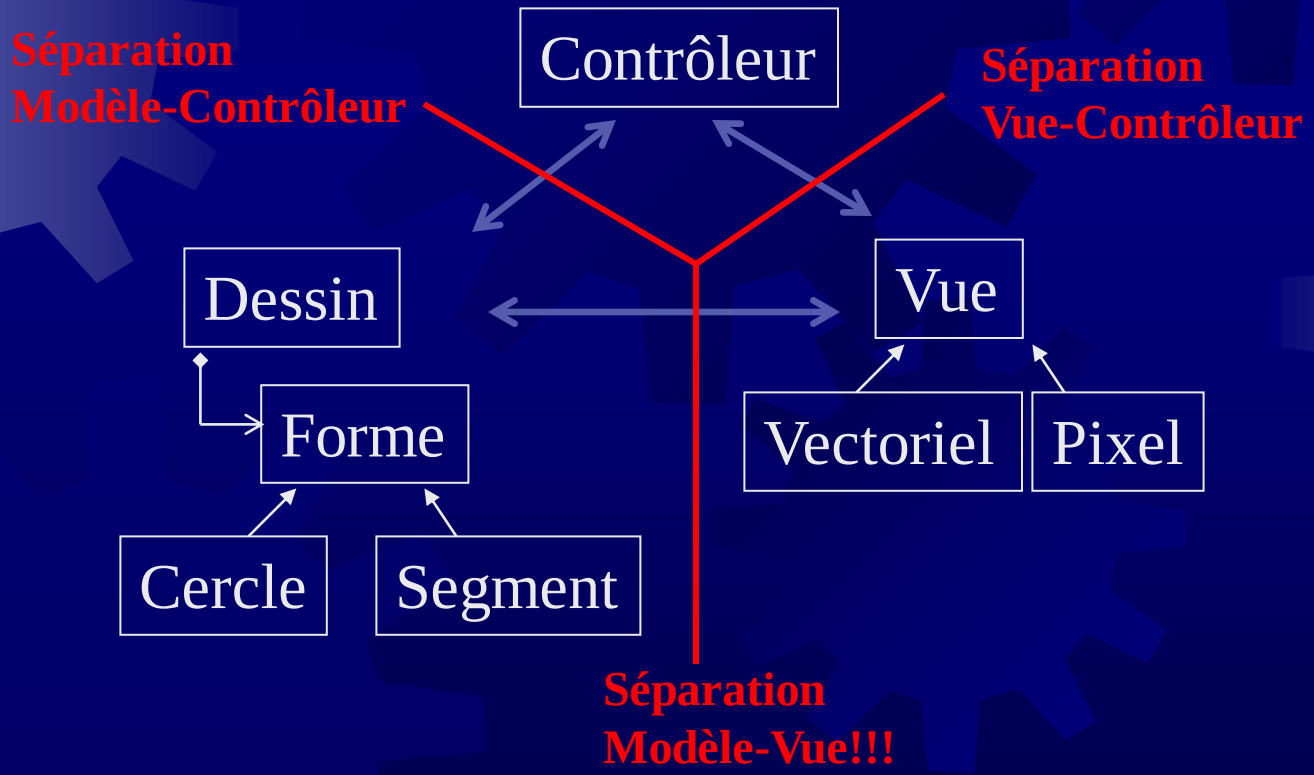
Concevoir l'architecture (classes en UML) d'un logiciel de dessin géométrique supportant les cercles, segments, groupes…

Parties à clarifier :

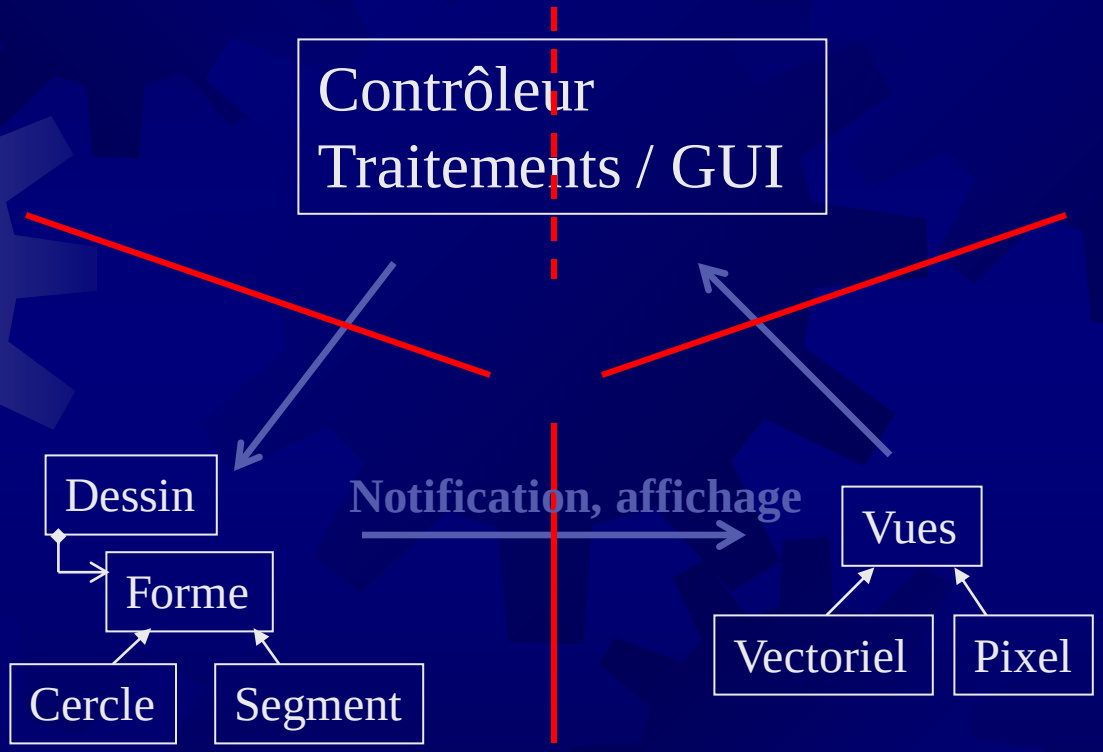
1. Structure interne / Dessins des formes
2. Changements synchronisés
3. Groupes d'objets (Group / Ungroup)
4. Comportements de la souris, des menus contextuels
5. Conversions en multiples formats…

Pb 1/5 : MVC
Modèle - Vue - Contrôleur

- Fichiers / Représentations Internes / Vues / Interactions utilisateurs

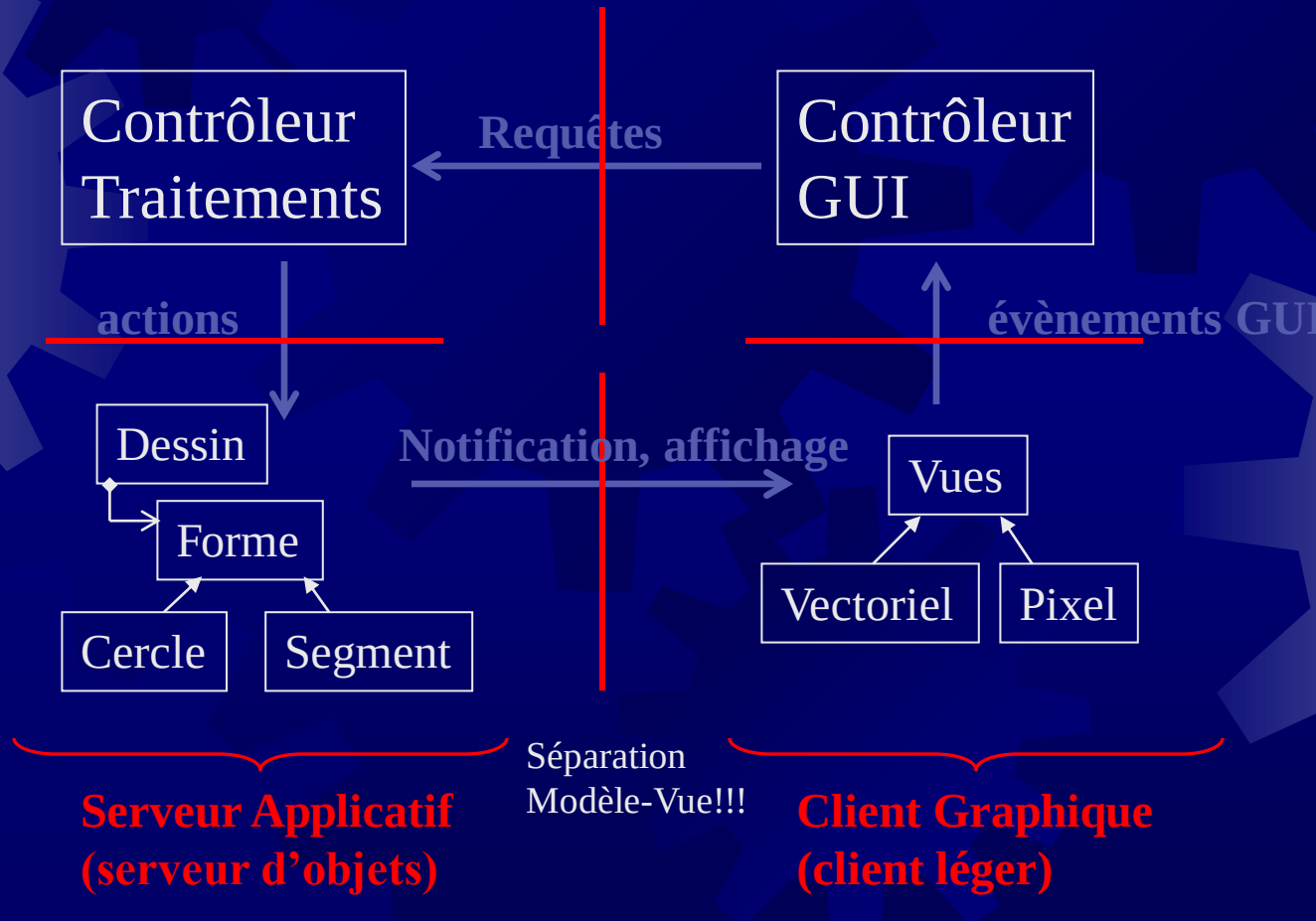


Pb1/5 : MVC (Suite) : Contrôleur Traitements / GUI

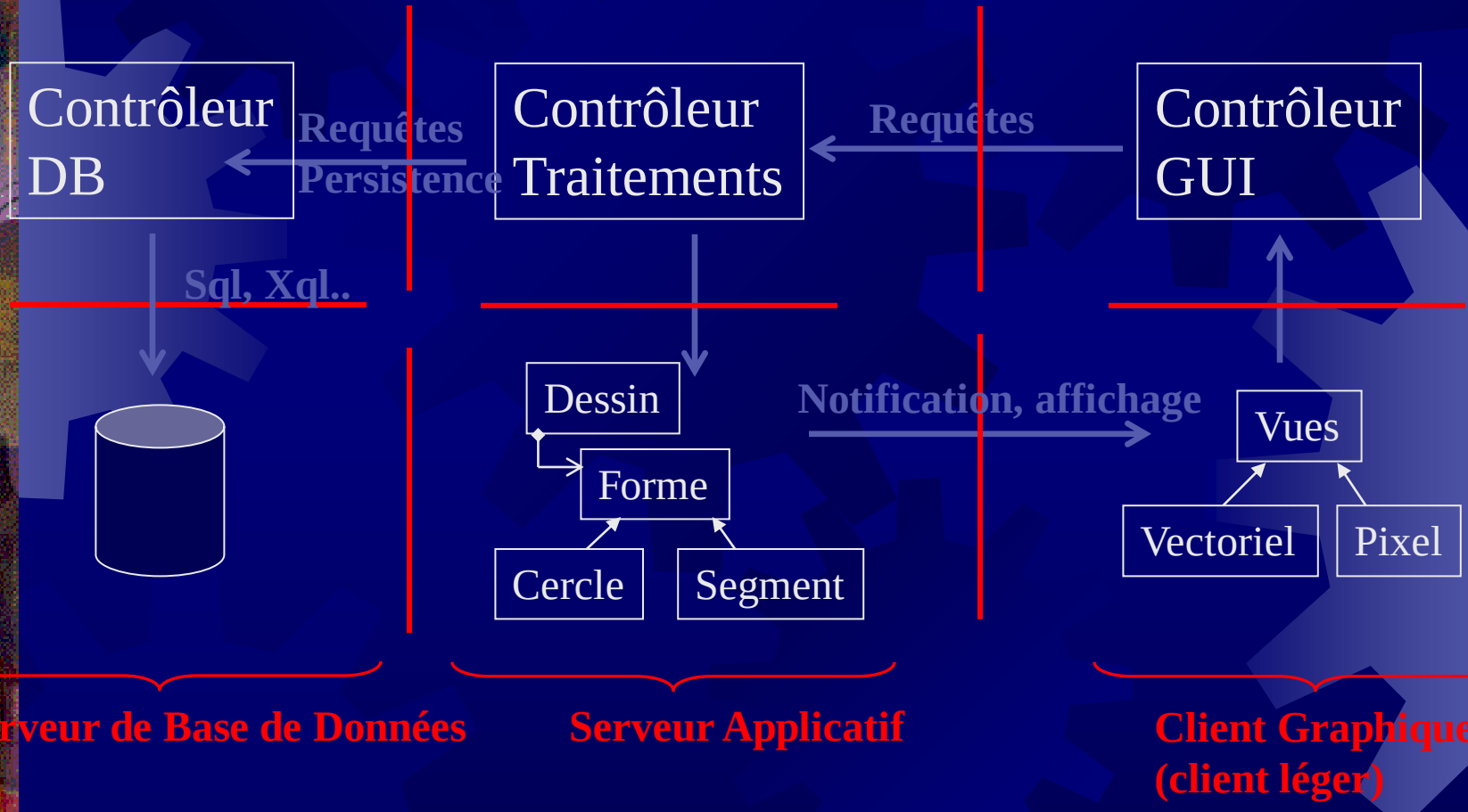


Séparation
Modèle-Vue!!!

Pb1/5 : MVC (Suite)
Architecture 2 tiers

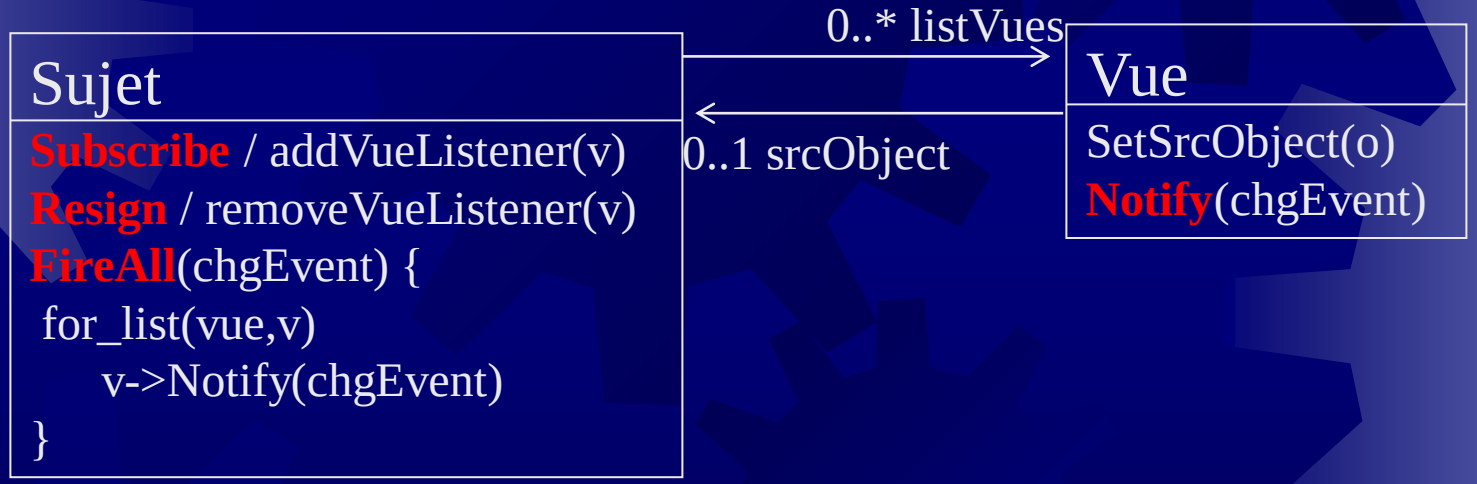


Pb1/5 : MVC (Suite)
Architecture 3 tiers



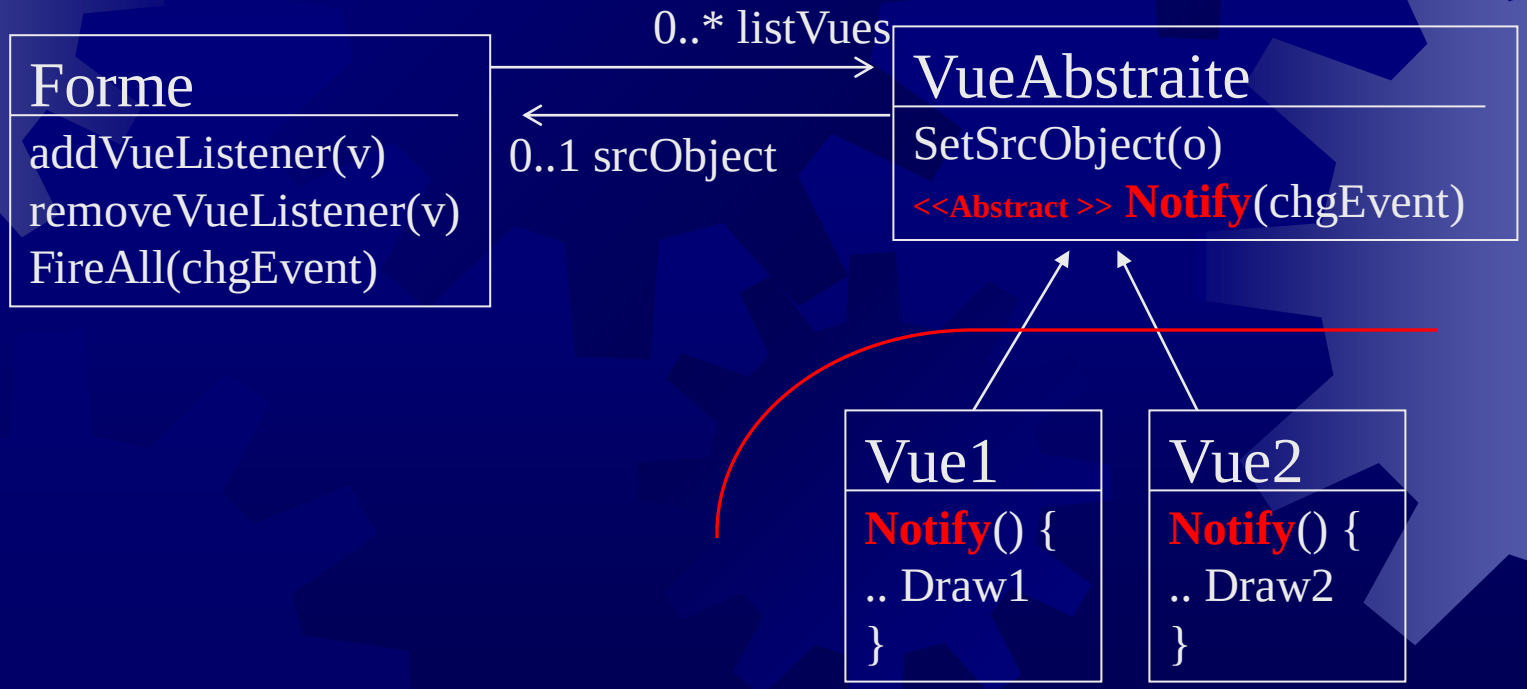
Pb2/5 : Publish & Subscribe

- Notifications de changement



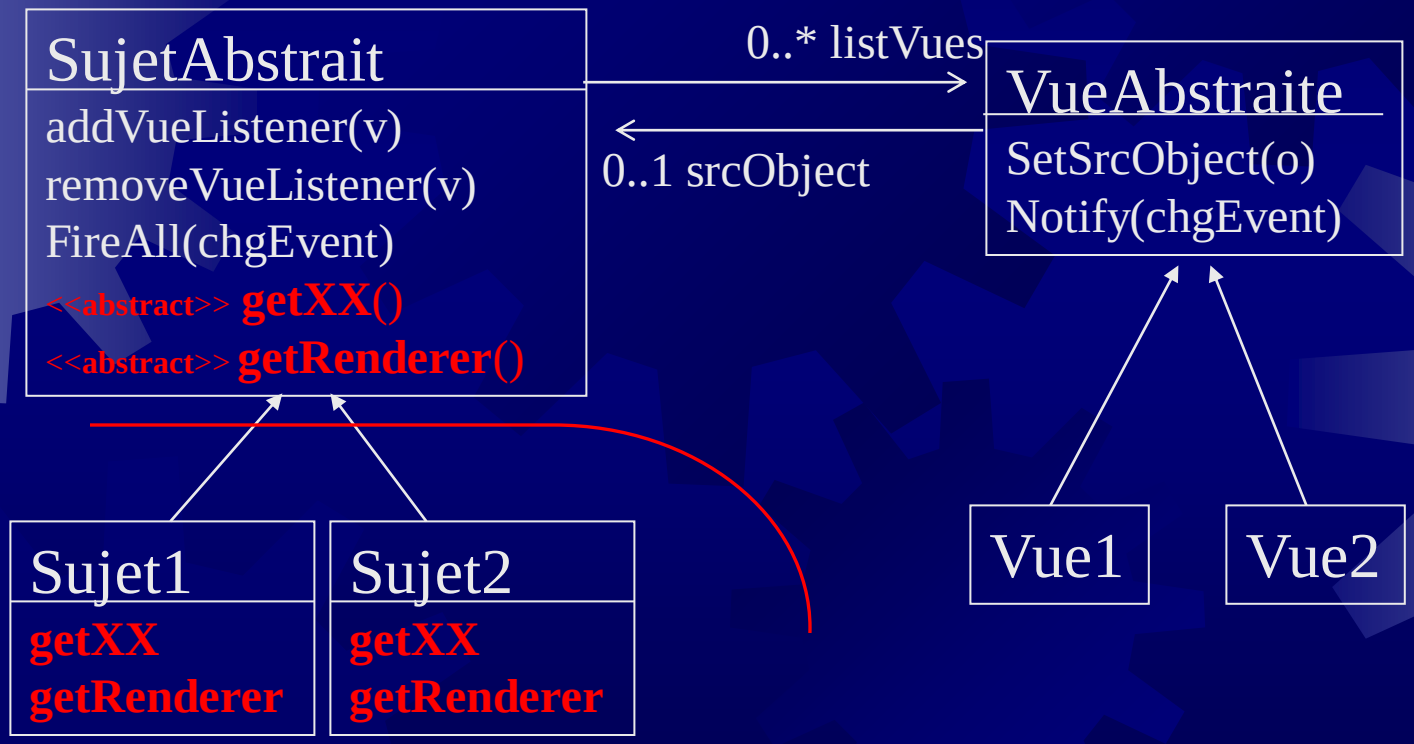
Pb 2/5: Publish & Subscribe (Bis)

- Indépendance des Vues pour l'Objet



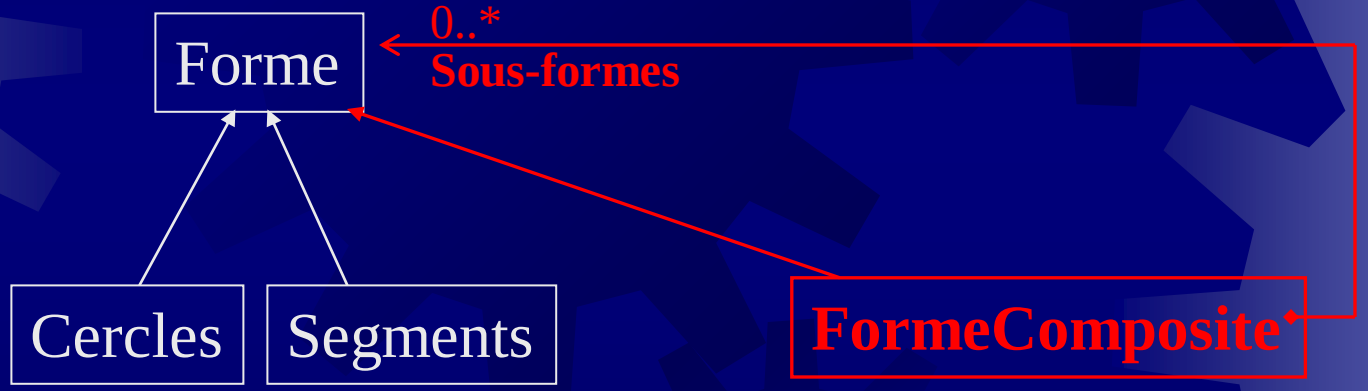
Pb2/5: Publish & Subscribe (Ter)

Indépendance des Objets pour les Vues (cf. MVC)



Pb 3/5 : Composite..

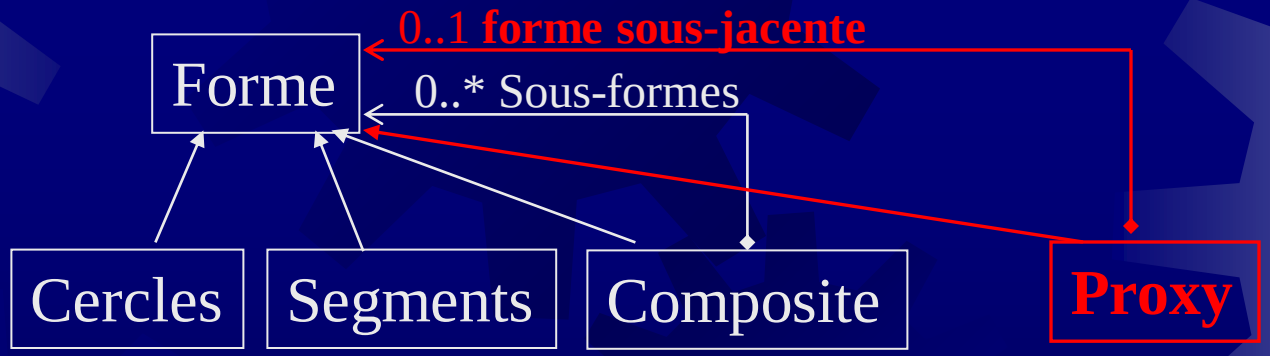
- Group / Ungroup



Pb3/5 : Composite, Proxy..

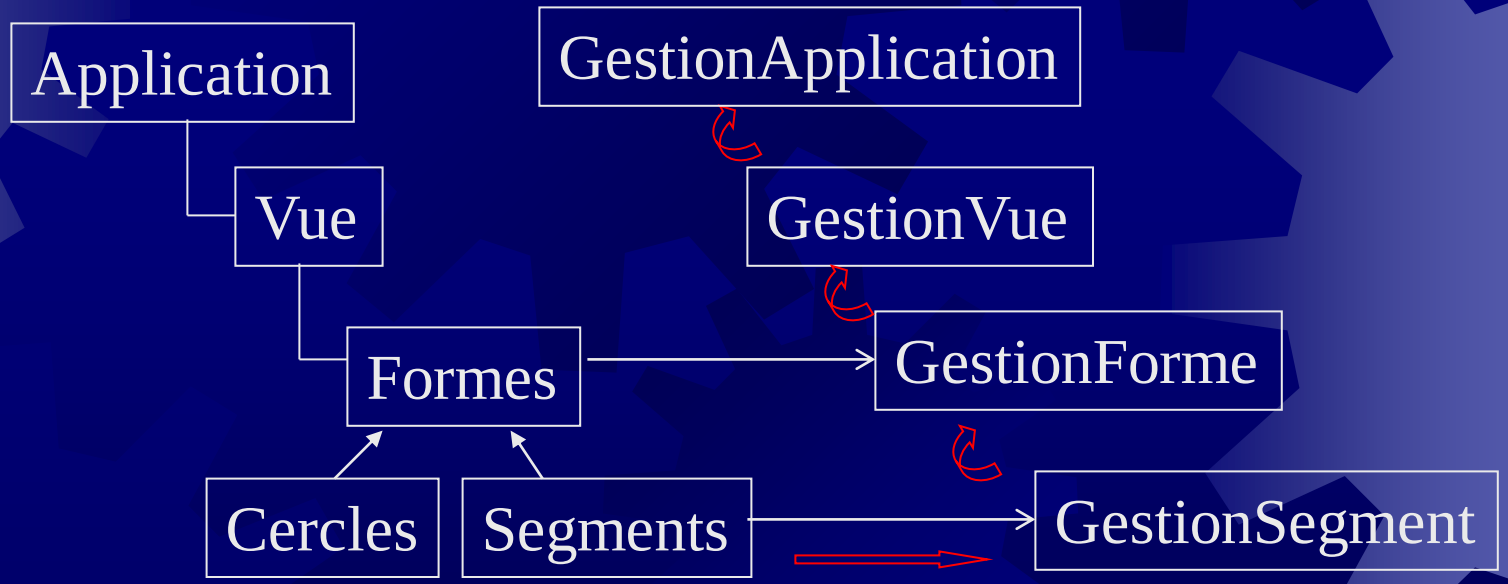
- Formes par procuration

(Rotation, Iconifiée, En cours de chargement, etc..)



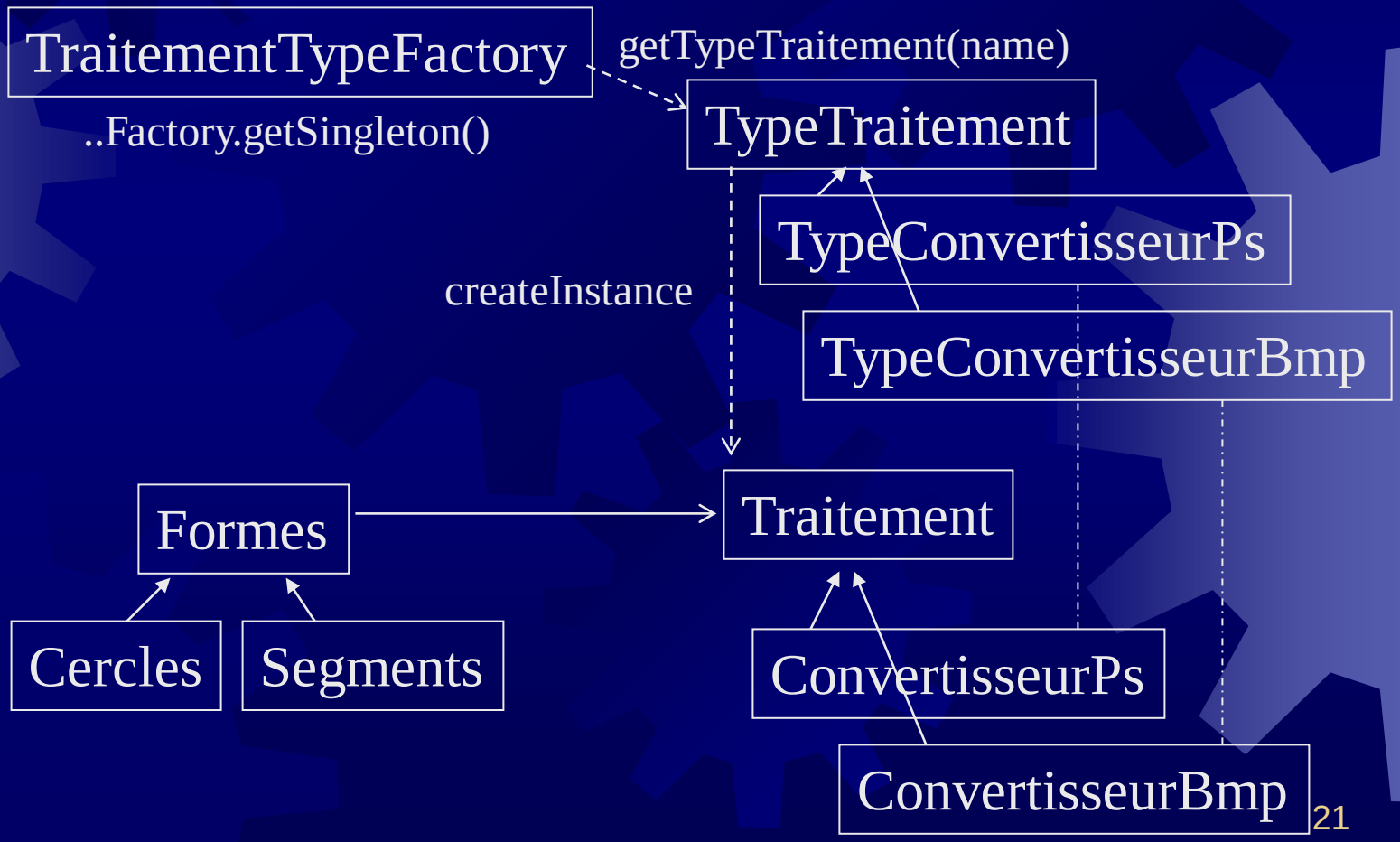
Pb4/5 : Délégation, Chaîne de Responsabilité..

- Gestion de la souris, des évènements graphiques…



Pb 5/5 : Stratégie, Visiteur, Factory, Singleton…

- Conversions Multiples, etc..



Retour sur les 23 Patterns

Les 23 Patterns se trouvent **partout**

Sous formes réduites, déguisées, renommées…

⇒ **Lire des programmes … Savoir les reconnaître et comprendre l'architecture**

⇒ **Ecrire : savoir en mettre partout (!!), en respectant les concepts**

Description des 23 Patterns ? / Réflexion de chacun !!

- Découverte
  - Bon sens, mais c'est bien sûr..
- 1ère Lecture
  - Catalogue Universitaire ?
- 1ère pratique
  - Je connais!.. Je vais réessayer pareil…
  - Oups.. Je dois relire quelques détails..
- 2ème lecture
  - C'est très fort
- 2ème pratique
  - On les vois partout ! On en met partout !

Liste des Patterns : Modèles créateurs (1/3)

- Fabrique Abstraite (Abstract Factory, Kit)
- Monteur (Builder)
- Fabrication (Factory method)
- Prototype
- Singleton

Liste des Patterns : Modèles Structuraux (2/3)

- **Adaptateur**
- **Pont**
- **Composite**
- **Décorateur**
- **Façade**
- **Poids Mouche**
- **Procuration (Proxy)**

Liste des Patterns : Modèles Comportementaux (3/3)

- **Chaîne de responsabilité**
- **Commande**
- **Interpréteur**
- **Itérateur**
- **Médiateur**
- **Mémento**
- **Observateur**
- **État**
- **Stratégie**
- **Patron de méthode**
- **Visiteur**

Conclusion

Un livre à lire 2 fois

1 rôle => 1 interface + délégation à 1 objet

Possibilité de changement ouverte

La programmation devient tellement plus simple !