



GL2 : Patrons de conception

Lydie du Bousquet
& Philippe Lalande



Conception

- La conception est un défi
- Il existe des processus définissant les activités et leur organisation
 - Mais rien sur le comment faire
- Les méthodes définissent/utilisent des notations, des diagrammes...
 - Mais rien sur comment les construire
- Sans espoirs ?
 - Non : capitalisation d'expérience et patrons



Patron de conception - définition

- C'est une description d'un problème et une solution bien connue
 - <problème, solution> validée par l'expérience
- La solution doit avoir été validée sur plusieurs projets
- Un patron est décrit en utilisant des langages de spécification connus
 - description OO par exemple



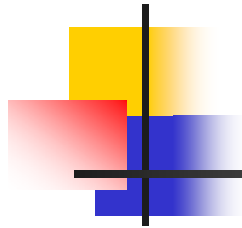
Patron de conception - intérêt

- Patron de conception = réutilisation
 - Il est plus facile de réutiliser une solution de conception plutôt que du code
 - Solution pour
 - décrire les bonnes pratiques de conception
 - Transmettre les connaissances acquises par l'expérience
- Les patrons sont connus par les concepteurs
 - Un langage est créé et partagé



Patron de conception - description

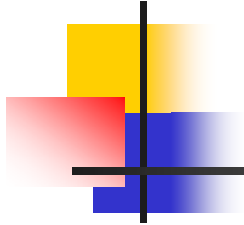
- Nom
- Description globale
- Problème
- Solution
 - Utilisation de diagrammes
- Suggestion d'implantation
- Avantages et limites
- Exemple



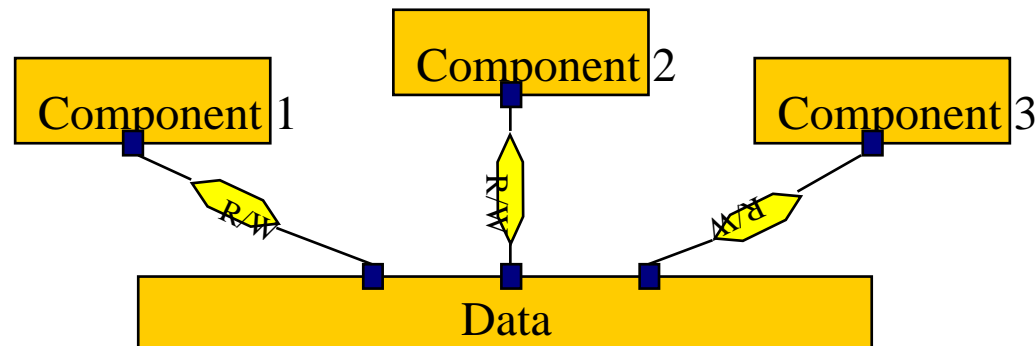
Patron de conception - portée

- Différents niveaux d'abstraction
 - Patron architecturaux (style)
 - Patron de conception
 - Language patterns (idioms)
- Beaucoup de domaines
 - Enterprise Integration Patterns (EIP)
 - Patrons pour les systèmes distribués
 - Patrons orienté service (SOC)

Exemple: style mémoire partagée



Nom	Mémoire partagée
Description	Logiciel temps-réel avec ressources limitées
Problème	Les communications ont un coup
Solution	
Conséquences	+ : Performance - : moins clair, difficile à maintenir



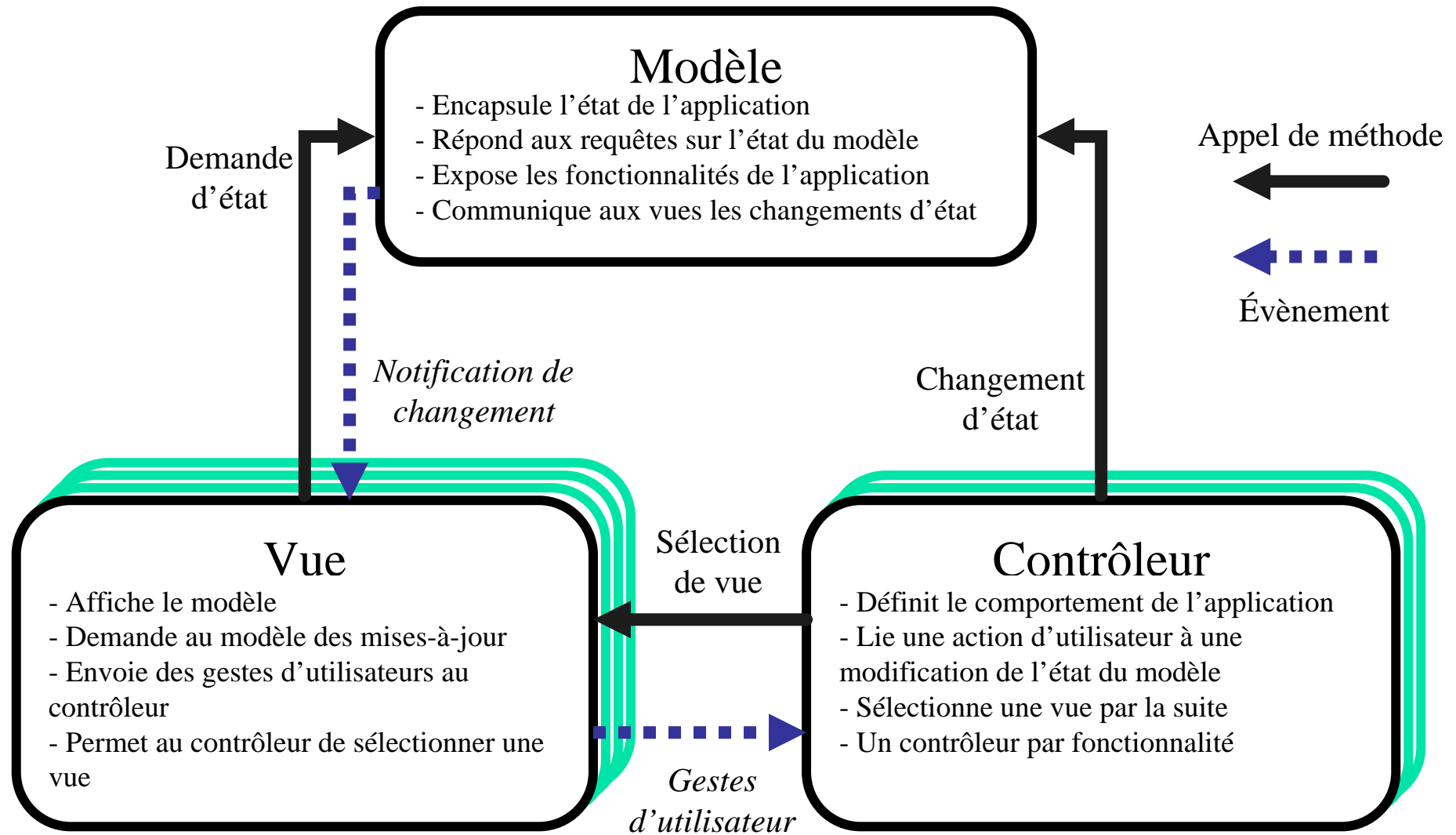


Exemple 2 : style MVC

- Nom : Modèle-Vue-Contrôleur
- Description :

Plusieurs objets représentent (*vue*) des données communes (*modèle*) et agissent (*contrôleur*) sur ces données
- Problème :
 - Organisation globale d'une IHM

Exemple 2 : MVC - solution





Exemple 2 : MVC

- Solution (suite)
 - En général le modèle n'est pas une seule classe
 - Modèle composé de plusieurs instances de plusieurs classes
 - Ces objets peuvent être statiques (créés n'importe quand) ou dynamiques (déjà créés)
 - Ils peuvent disparaître aussi

- Conséquence
 - permet d'avoir plusieurs vues sur le même modèle
 - facilite le support pour de nouveaux clients
 - Il suffit d'implémenter une nouvelle vue, un nouveau contrôleur et les connecter au modèle existant



Exemple 2 : MVC

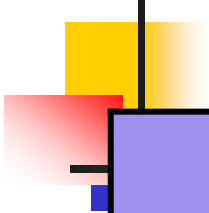
- Le patron MVC est composé en plusieurs patrons
 - S'il y a plusieurs vues, elles peuvent être organisées dans un arbre en utilisant le patron « composite »
 - La relation entre les vues et les modèles est le patron « observer »
 - Les contrôleurs sont des « strategy » des vues
- Mais encore
 - Le modèle est souvent un « mediator »
 - L'arbre des vues contient des « decorators » pour ajouter des propriétés aux vues
 - Une vue utilise un « adaptor » pour convertir le modèle en une interface qu'elle peut utiliser
 - Des vues créent des contrôleurs avec un « factory »

Patron de conception - références



- Elements of Reusable Object-Oriented Software"
E. Gamma, R. Helm, R. Johnson, J. Vlissides
Addison-Wesley, 1995
- Pattern-oriented software architecture –
Volume 2
D. Schmidt, M. Stal, H. Rohnert, F. Buschman
Wiley, 2000
- Enterprise Integration patterns

Patron de conception – Gamma



	Créateur	Structurel	Comportement
Classe	Factory	Adaptor	Interpreter Template
Objet	Abstract Factory Builder Prototype Singleton	Adaptor Bridge Composite Decorator Façade Flyweight Proxy	Chain of responsibility Command Iterator Mediator Memento Observer State Strategy Visitor