# Une infrastructure pour (ré-)ingénierer des configurateurs web

## **Une approche IDM**

Mathieu Acher

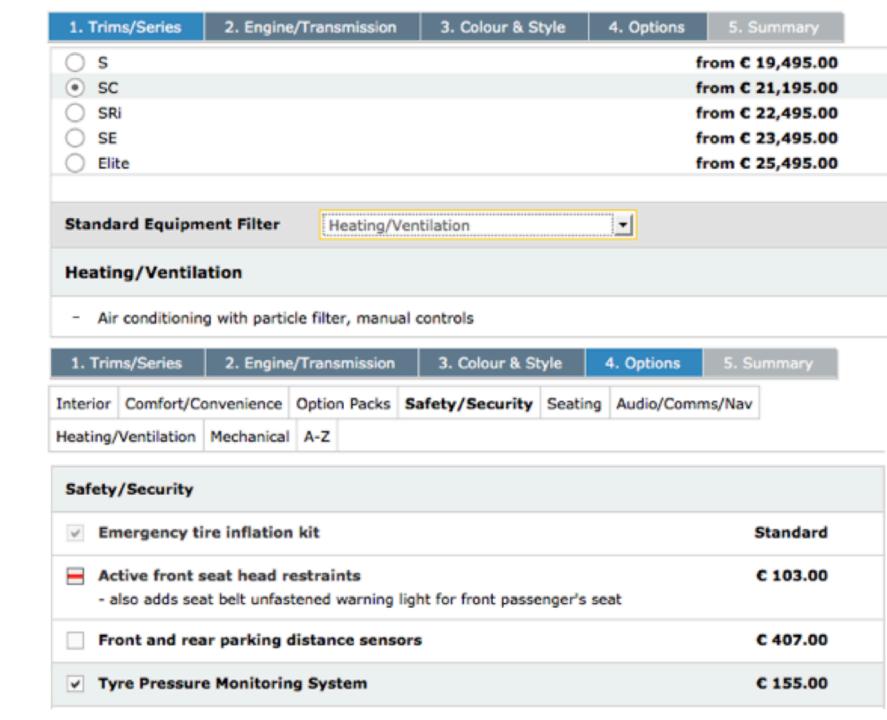
Maître de Conférences mathieu.acher@irisa.fr

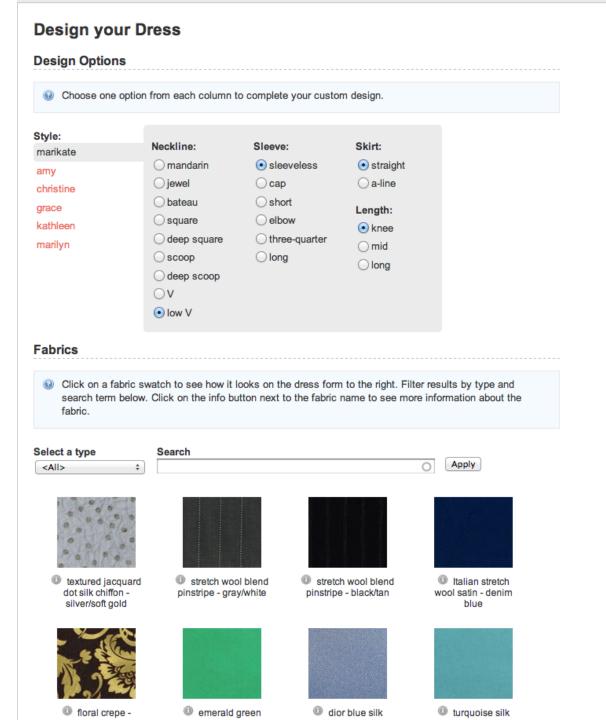




### **Material**

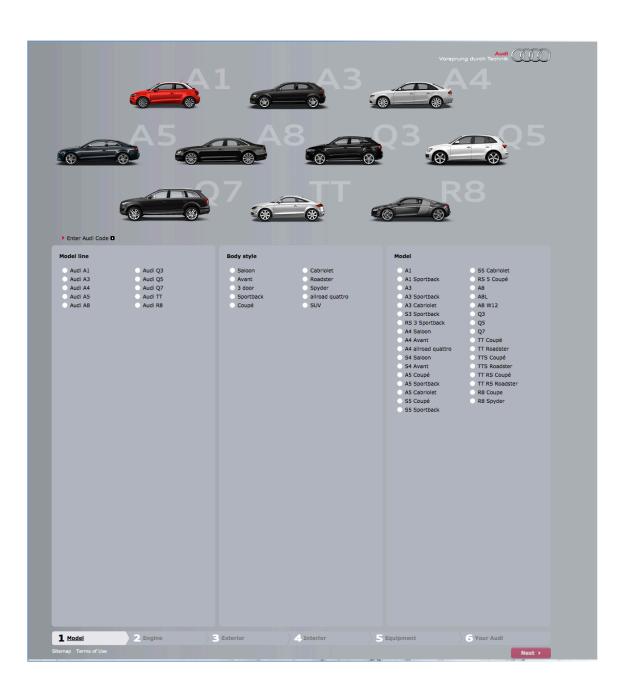
http://mathieuacher.com/teaching/SIMPLE-MIAGE2/projet/





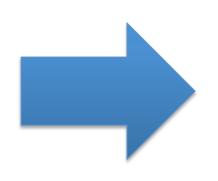
#### 1. FABRIC 2. STYLE 3. PERSONALIZE 4. SIZE 5. SUMMARY

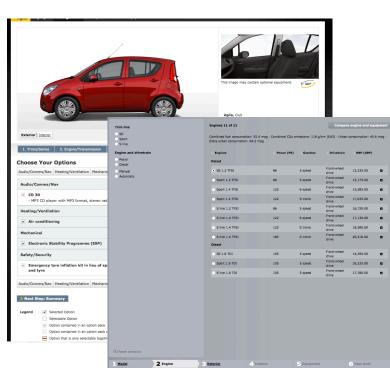




### Une infrastructure basée sur l'IDM

Models
And
Languages

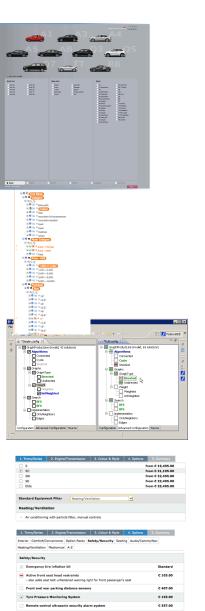




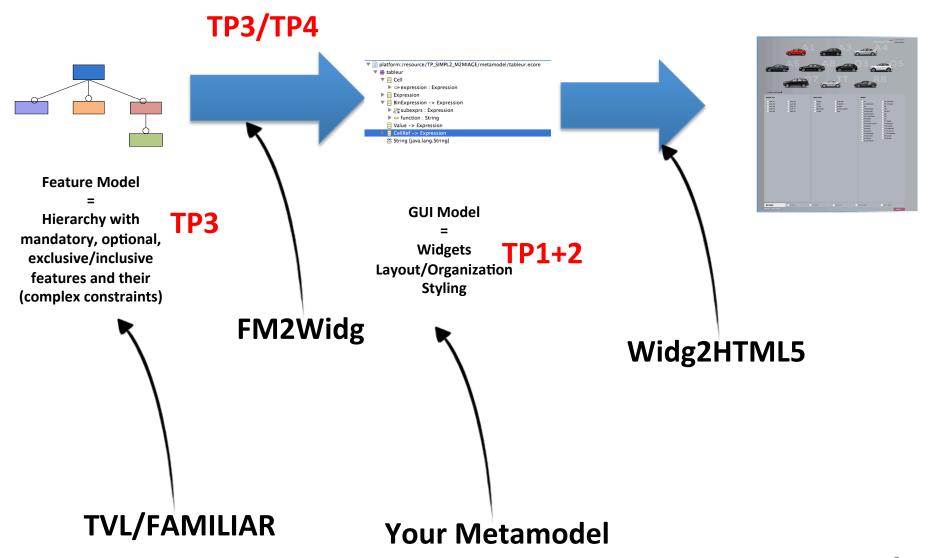
# Solution flexible et générique

Models
And
Languages

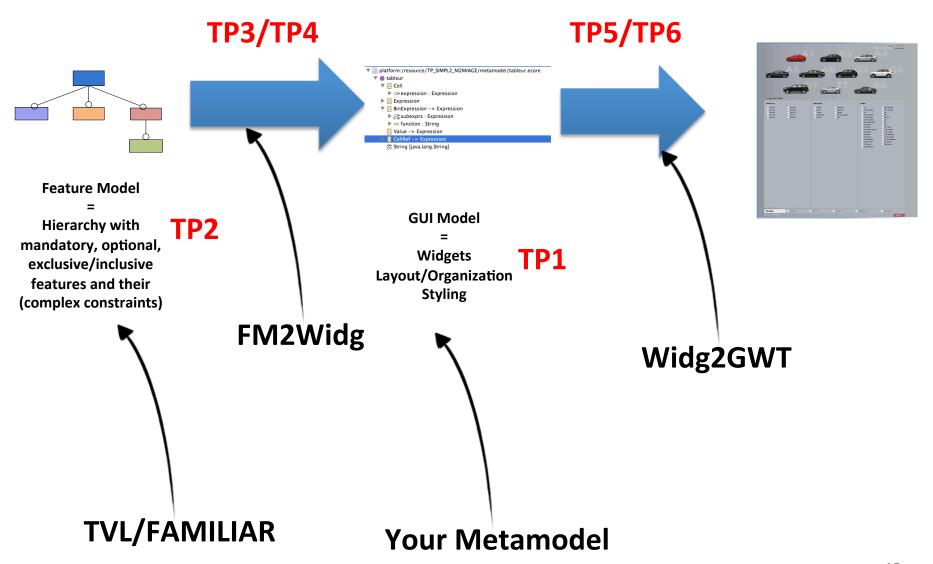


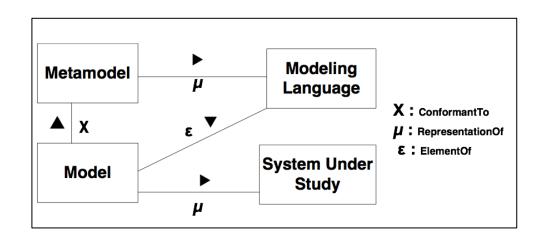


# Vue d'ensemble de l'approche

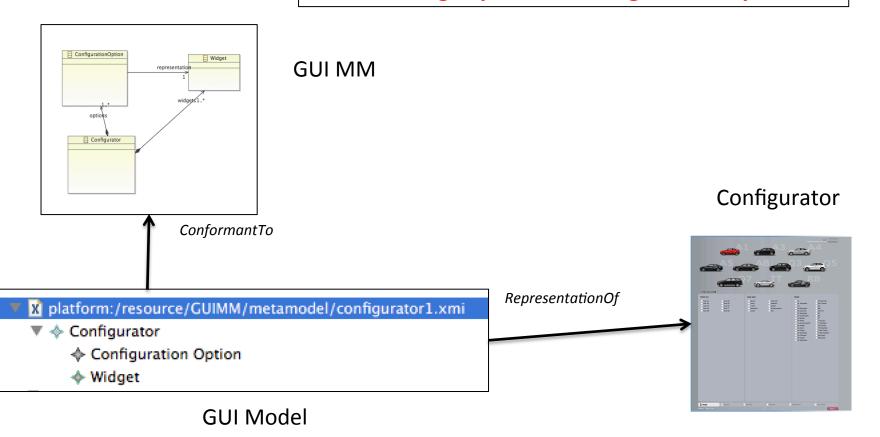


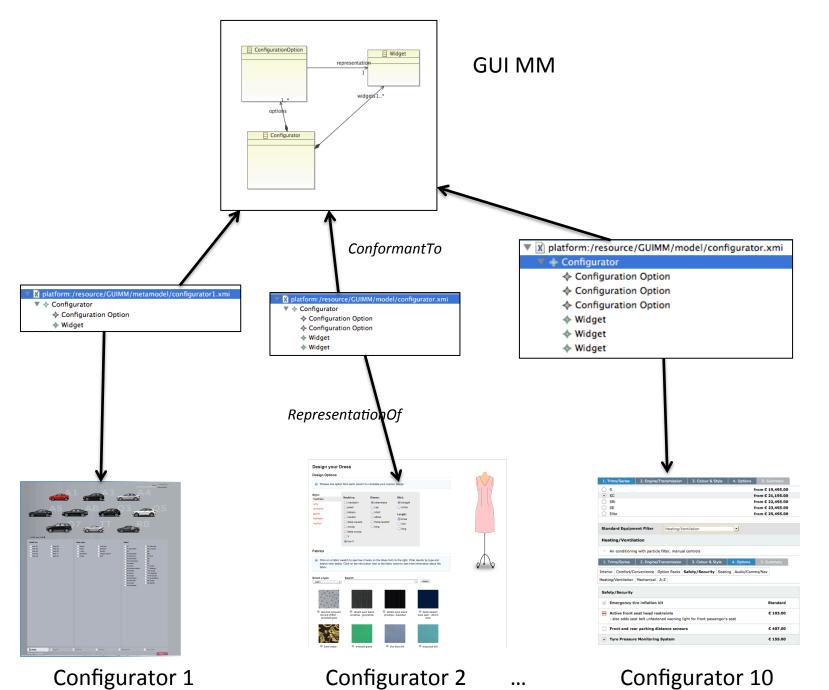
# Vue d'ensemble de l'approche



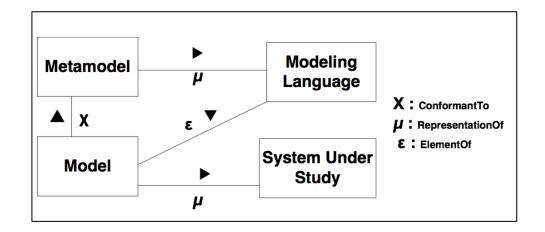


### **Rendering aspects of configuration options**

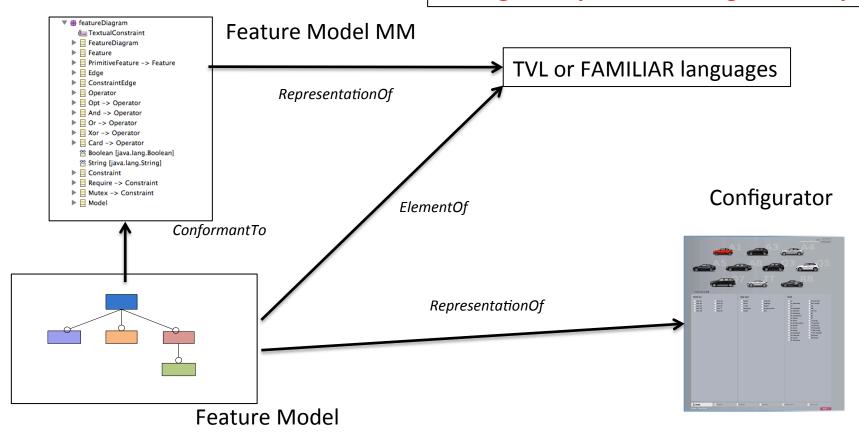




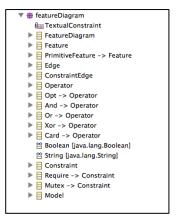
**GUI Models** 



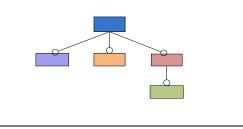
### **Logical aspects of configuration options**



#### Feature Model MM

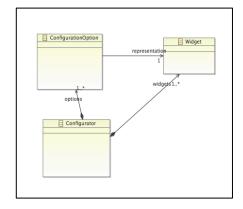








#### **GUI MM**

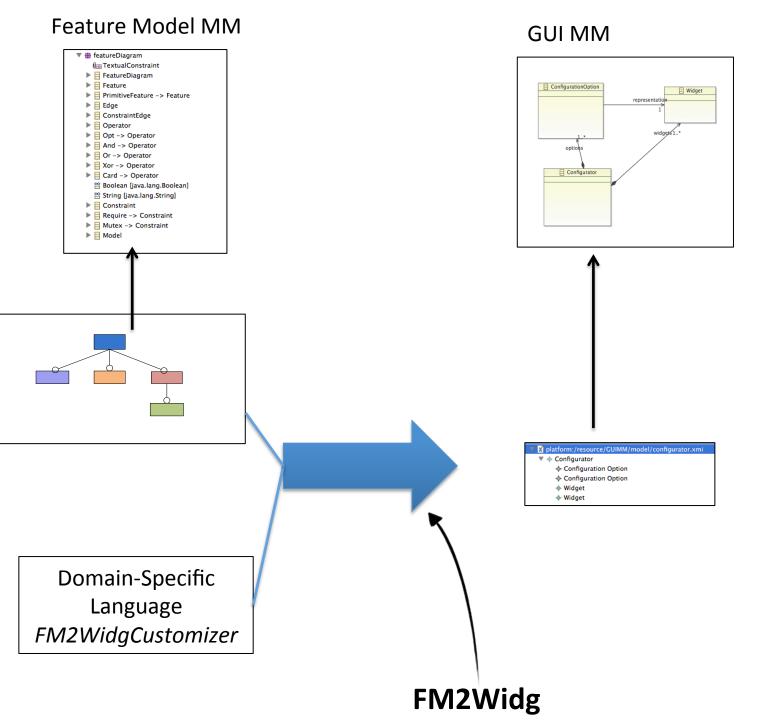


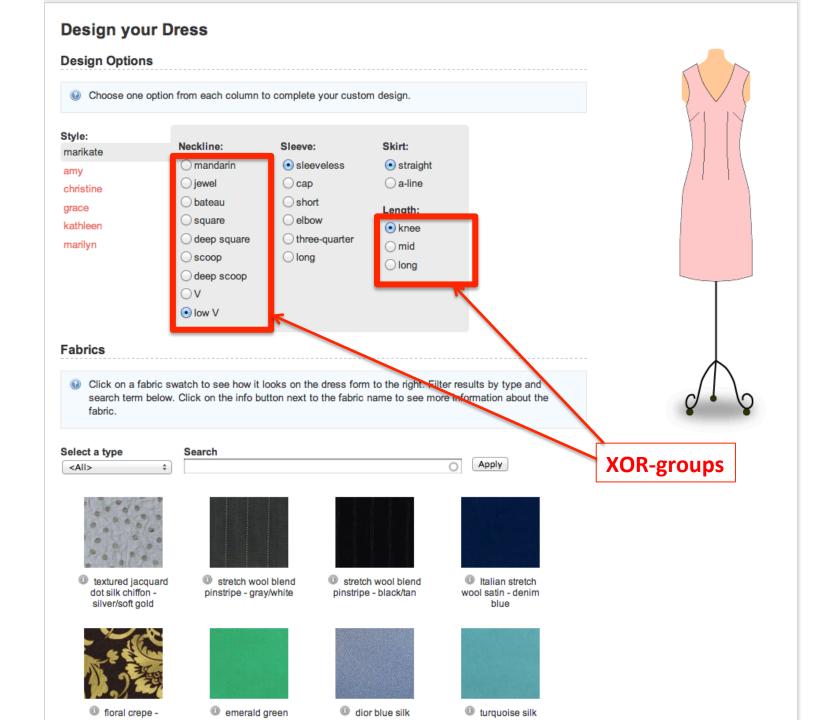


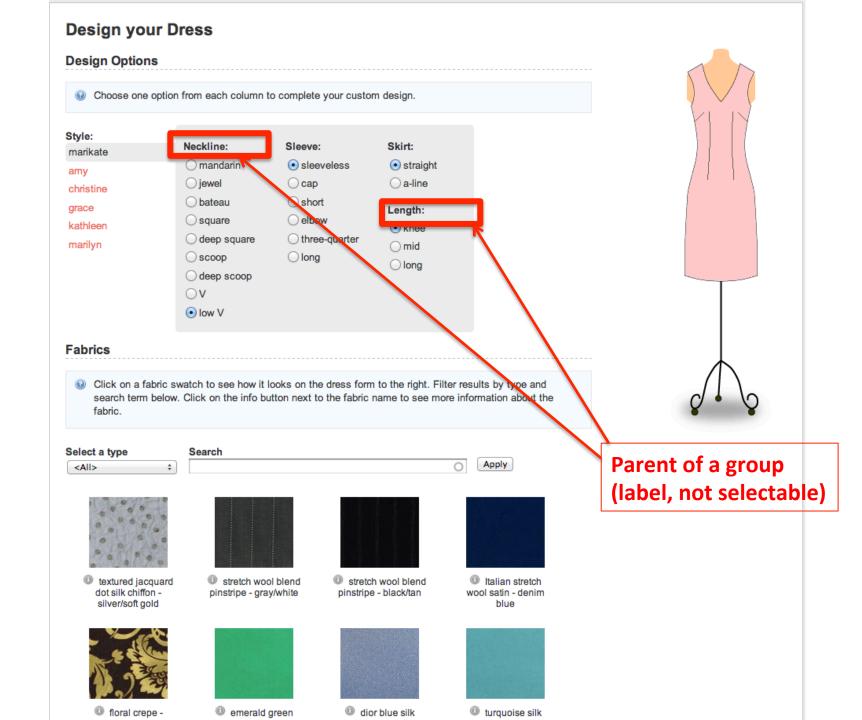


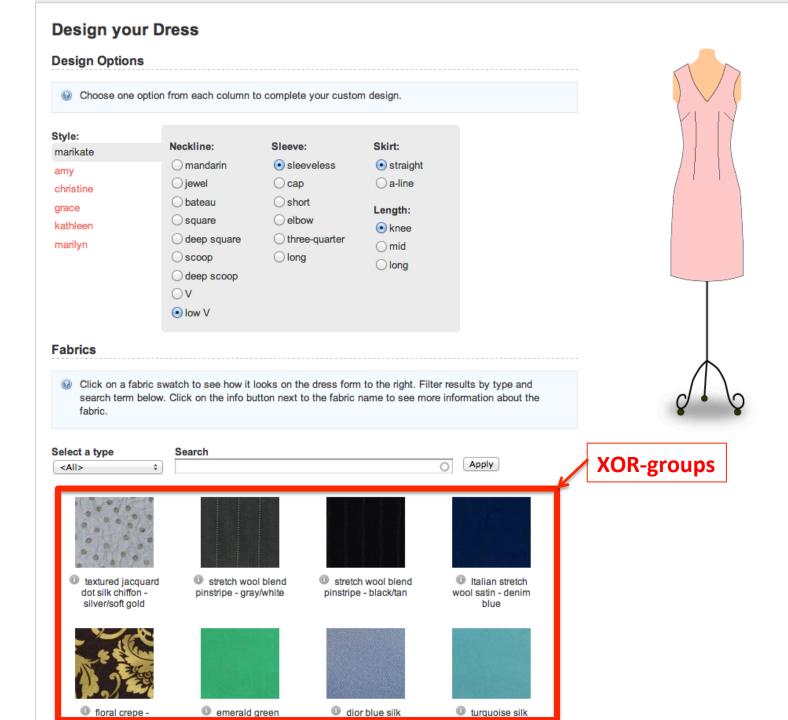
#### Feature Model MM **GUI MM** ▼ # featureDiagram **(**■ TextualConstraint ► FeatureDiagram ▶ 目 Feature ConfigurationOption ▶ ☐ PrimitiveFeature -> Feature representation ConstraintEdge ▶ ■ Operator Dot -> Operator ▶ 🗏 And -> Operator options ▶ ☐ Or -> Operator Xor -> Operator Configurator ▶ ☐ Card -> Operator 🖀 Boolean [java.lang.Boolean] String [java.lang.String] ▶ ☐ Constraint Require -> Constraint ▶ ■ Mutex -> Constraint ▶ ■ Model ▼ x platform:/resource/GUIMM/model/configurator.xmi Configuration Option ♦ Configuration Option Configuration Option Widget Widget Widget ♦ Configuration Option ♦ Configuration Option Widget ♦ Widget 🛮 🔀 platform:/resource/GUIMM/model/configurator.xmi ▼ ♦ Configurator

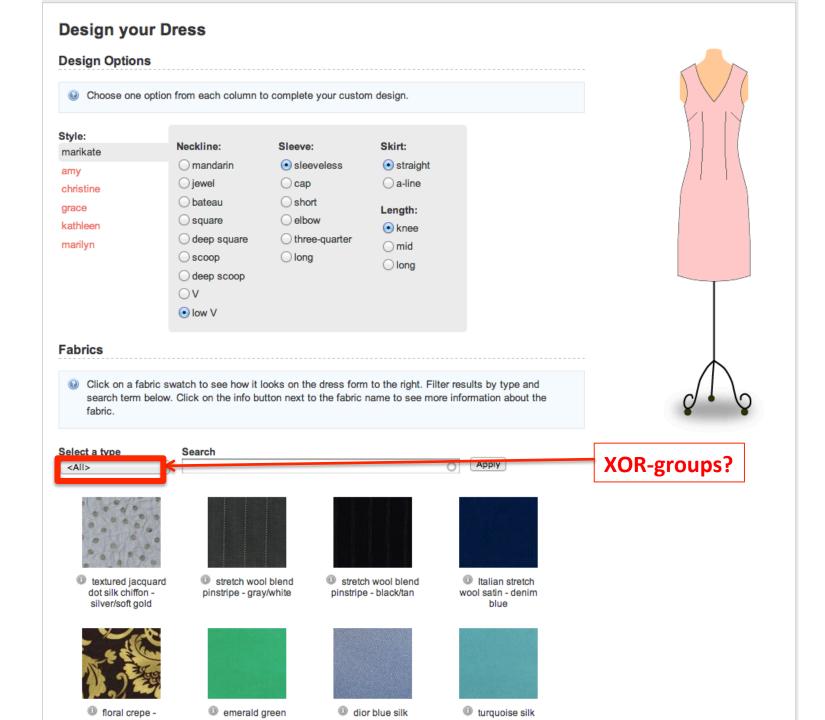
◆ Configuration Option
 ◆ Configuration Option
 ◆ Widget
 ◆ Widget

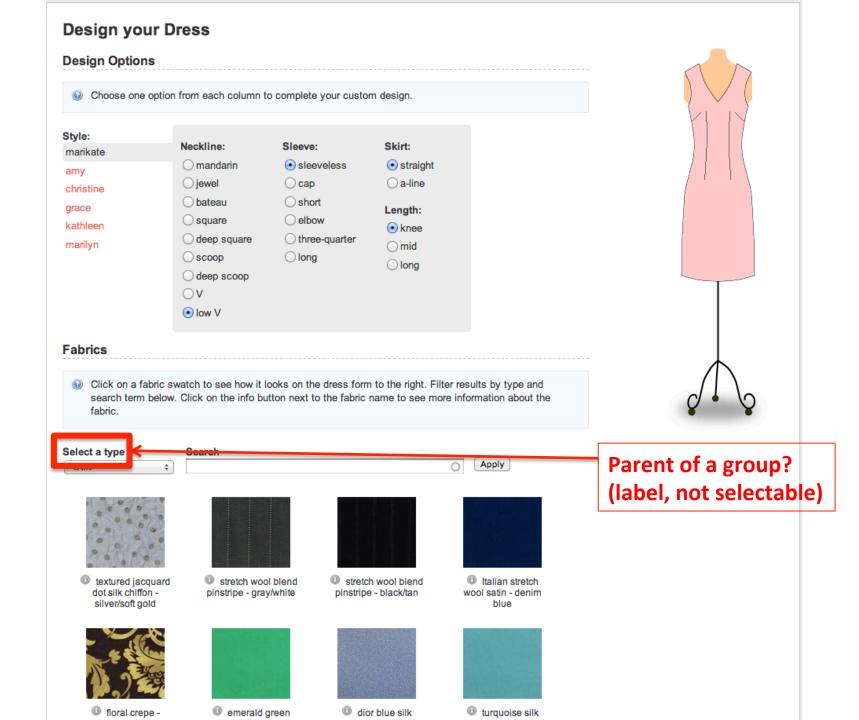


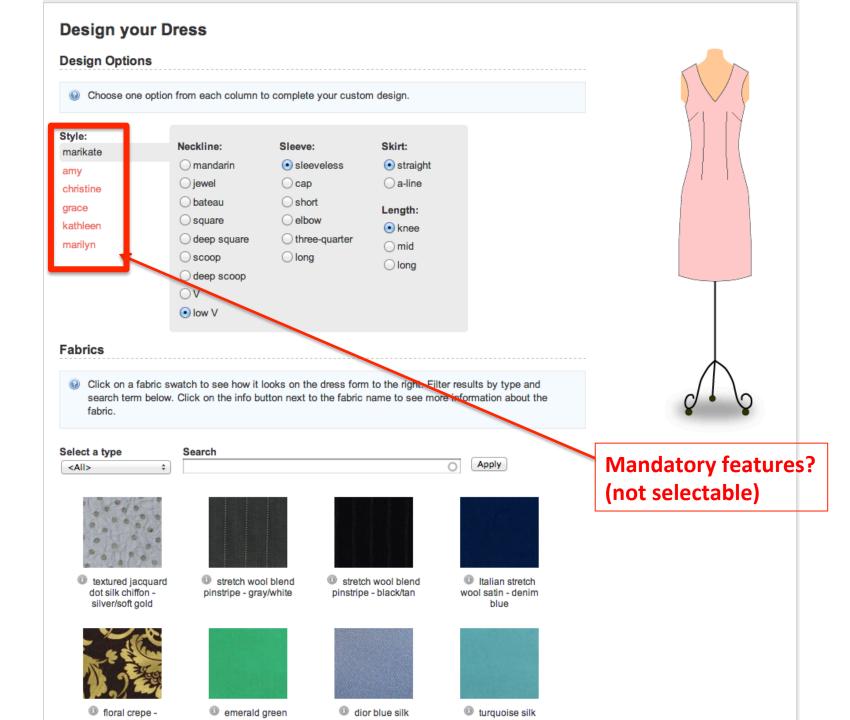


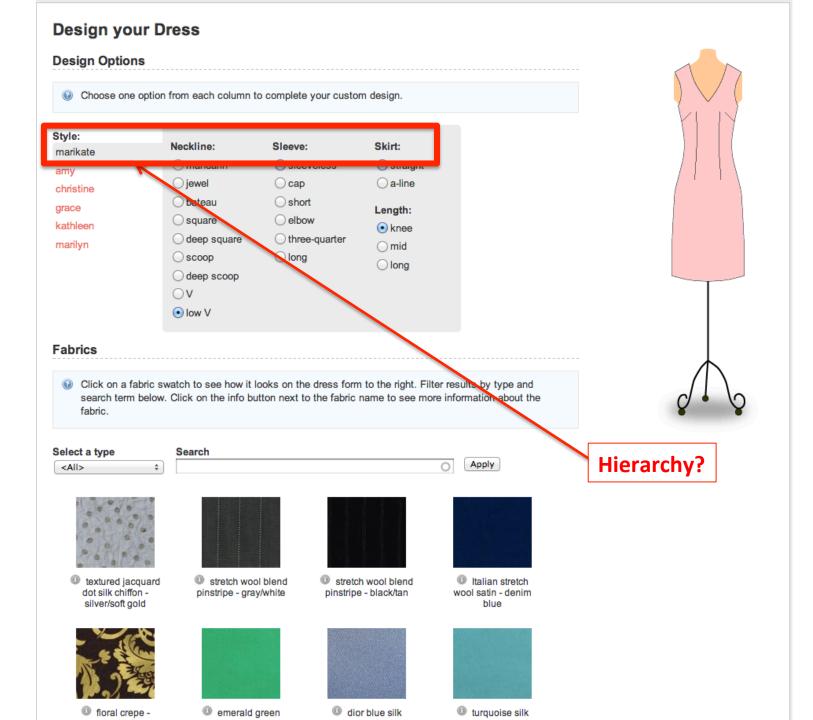


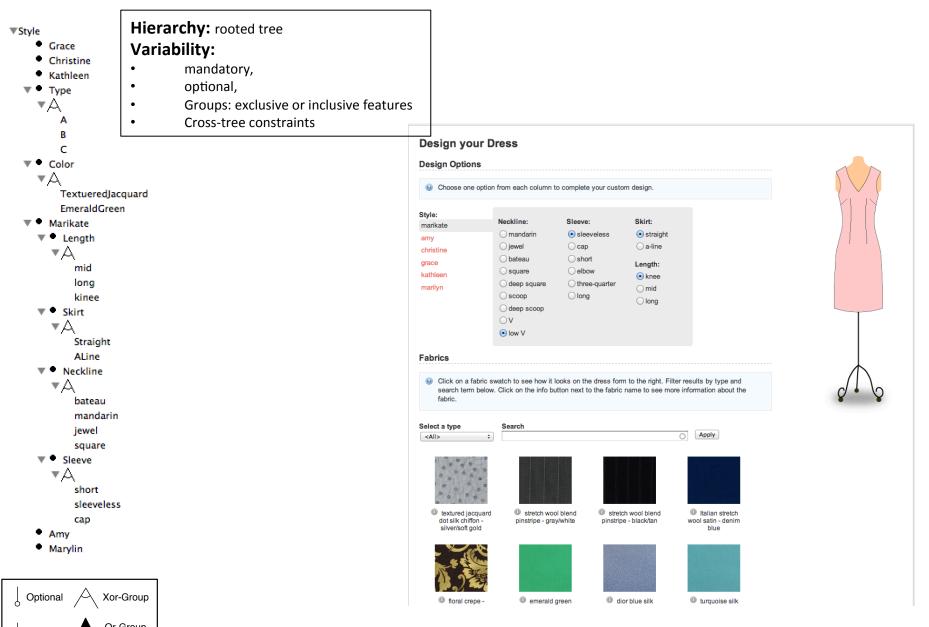


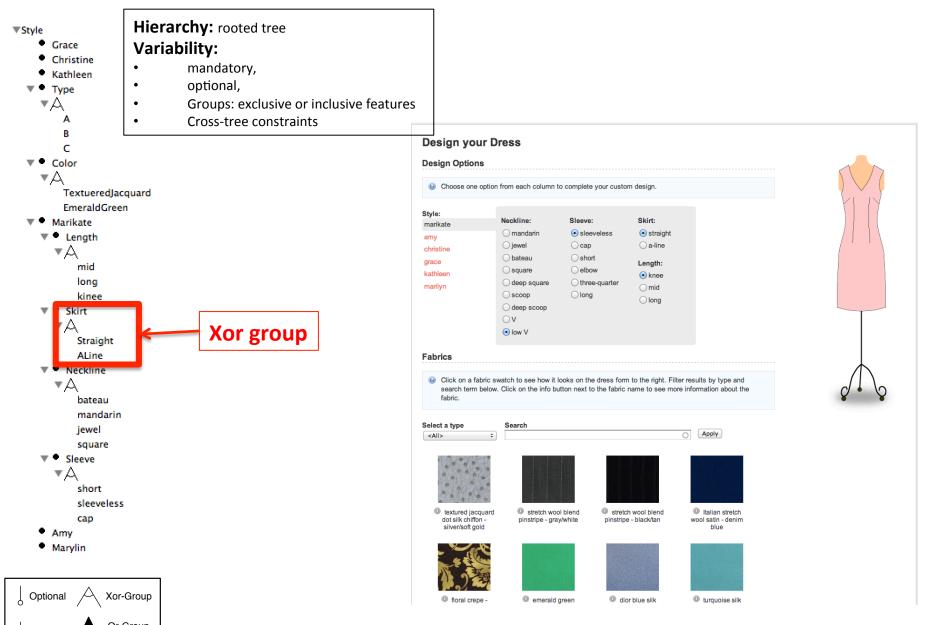


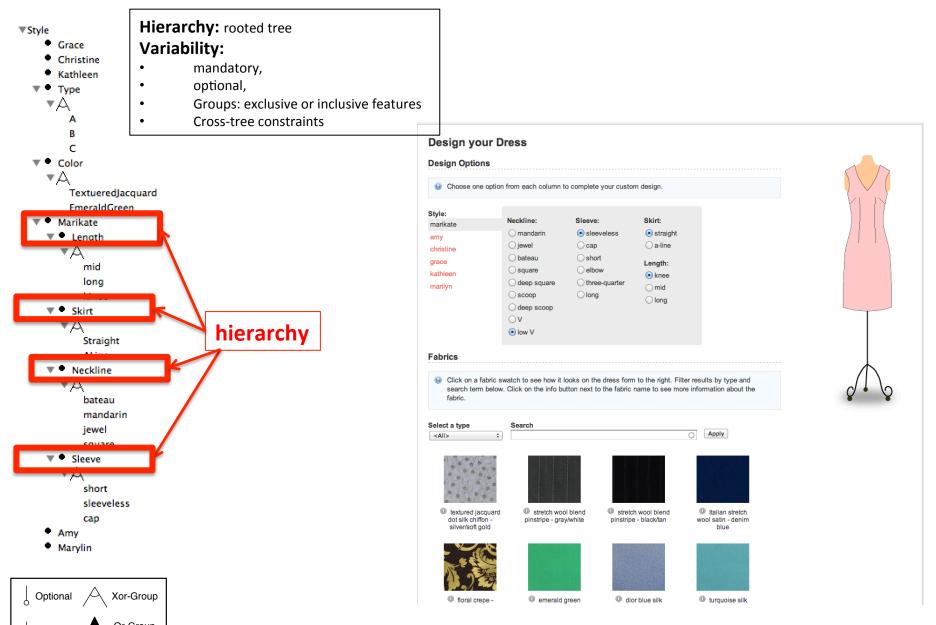


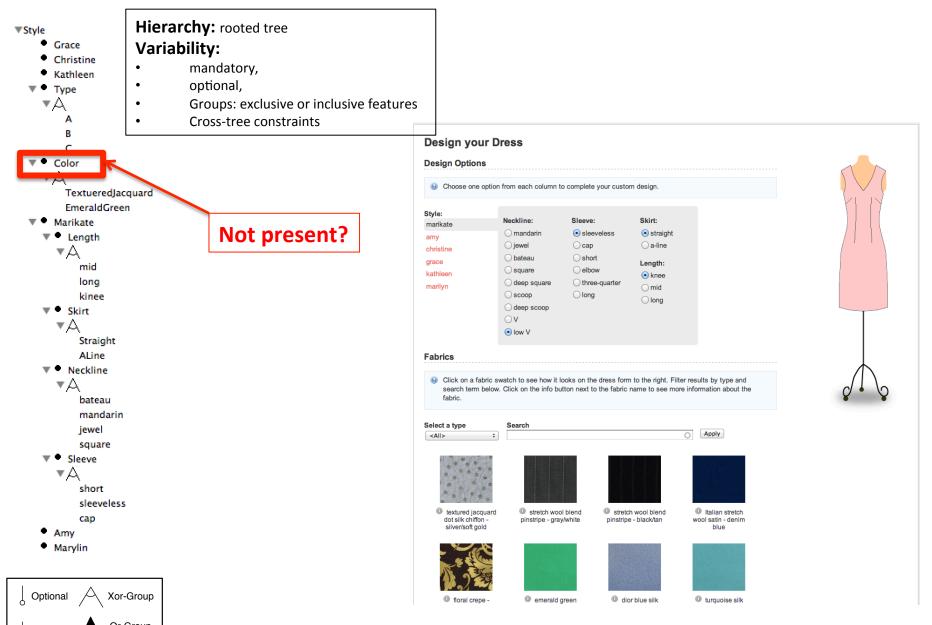












### TP1+2

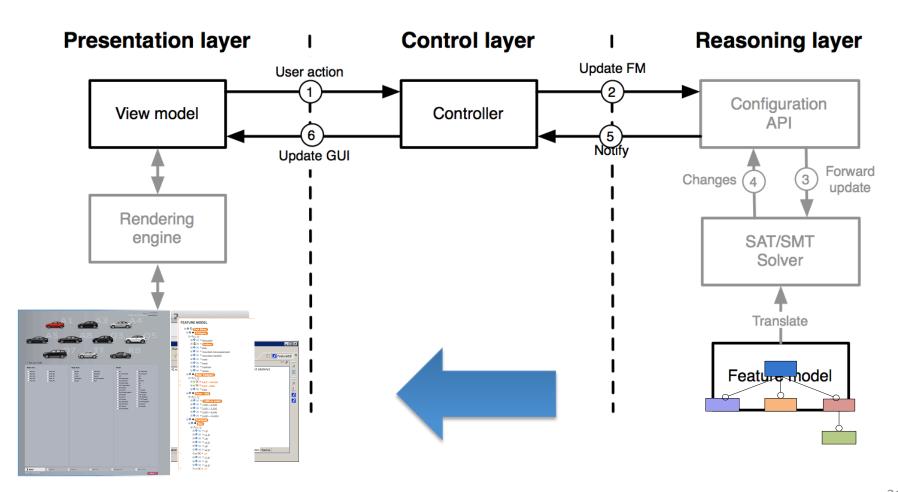
- Groupe de 2
  - Un groupe de 3
- Analyse du domaine des configurateurs
  - Sélection de 10 configurateurs (cf fichier excel)
  - Points communs et variations
  - "Features"
- Métamodélisation (avec Ecore et Eclipse)
  - des "widgets" représentant les options de configurations
  - de la manière dont ils sont organisés (e.g., "tabs")
- Modèles de widgets (avec Kermeta)
  - Montrer des "patterns" représentatifs
  - Conforme au métamodèle

### TP 3

- Sélectionner un configurateur et élaborer le feature model correspondant
  - Télécharger un Eclipse avec Xtext: http://www.eclipse.org/Xtext/
  - Installer le plugin TVL update site dans Eclipse avec l'url suivante: <a href="http://www.info.fundp.ac.be/~qbo/TVL/">http://www.info.fundp.ac.be/~qbo/TVL/</a>
- Identifier des "patterns" de transformation
  - identifier comment les constructions offertes par le formalisme des feature models (e.g., hiérarchie, groupe Xor) peuvent être restituées graphiquement
- Implémenter ces transformations en Kermeta
  - transformation en considérant les métamodèles de TVL et votre métamodèle (GUI), le métamodèle TVL est disponible ici: http://mathieuacher.com/ teaching/SIMPLE-MIAGE2/projet/
  - Pour convertir un modèle TVL (textuel) en XMI (exploitable par Kermeta),
     utilisez le converteur TVLExporter (cf adresse)

# TP3

### **MVC** Architecture



# Model-based approach

