La Modélisation au service de l'Agilité

Philippe Desfray – Journées Neptune –Mai 2013

SOFTEAM - Modeliosoft

Pourquoi poser la question Modélisation et Agilité?

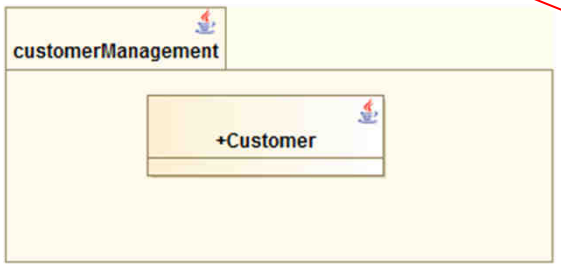
- Outils de productivité bienvenus avec développements agiles (IDE, construction d'IHM, …)
- L'Agilité est une affaire de processus, pas de technologie.

Modéliser n'a pas bonne presse chez les développeurs, encore moins chez les « Agiles »

- Ca n'a jamais été le cas
- Les processus agiles redonnent l'initiative aux développeurs
- Modéliser est perçu comme une activité bureaucratique, éloignée du concret de la réalisation
- Les modèles ne sont pas perçus comme des outils de productivité (la faute aux outils)
- Les cycles de vie (Waterfall, V, …) honnis ont magnifié les modèles en phase amont (*les amis de mes ennemis sont mes ennemis*)

Fournir une valeur ajoutée « pragmatique » - contre exemples

- Le schéma « modèle immuable produisant un code non synchronisé » ne marche pas
  - Les changements et évolutions de code sont permanents en phase de développement
- Remplacer un artéfact syntaxique par un graphique n'a aucun intérêt a priori




```
package customerManagement;

public class Customer {

}
```

Fournir une valeur ajoutée « pragmatique » - un code utile et maintenu lors d'évolutions

- Une coopération modèle/code « fine »
  - Changer règle génération
  - Possibilité accesseur totalement manuel
  - Cohérence permanente code/modèle
- Un support aux changements
  - Changement nom « Order »
  - Changement Cible association
  - …

```
package customerManagement;

import java.util.ArrayList;
import java.util.List;
import com.modeliosoft.modelio.javadesigner.annotations.objid;

public class Customer {
      public List<Order> order = new ArrayList<Order> ();


   public List<Order> getOrder() {
      // programmers code to be entered
      return this.order;
   }

   public void setOrder(final List<Order> value) {
      // programmers code to be entered
      this.order = value;
   }

}
```

Les valeur ajoutées d'un modèle

- Abstraction :
  - Réduit la complexité
  - Facilite la communication (ex: Développeur/Architecte)
- Génération
  - Systématise des manières de programmer, et d'organiser le code
  - Apporte plus en maintenabilité qu'en productivité pure

- Plusieurs utilisations possibles de modèles
  - Comprendre, poser le problème
  - Définir la solution
  - Produire/générer
- Conception et Analyse restent deux activités indispensables parfois malmenées en projets Agiles

Apporter une approche souple, non dogmatique

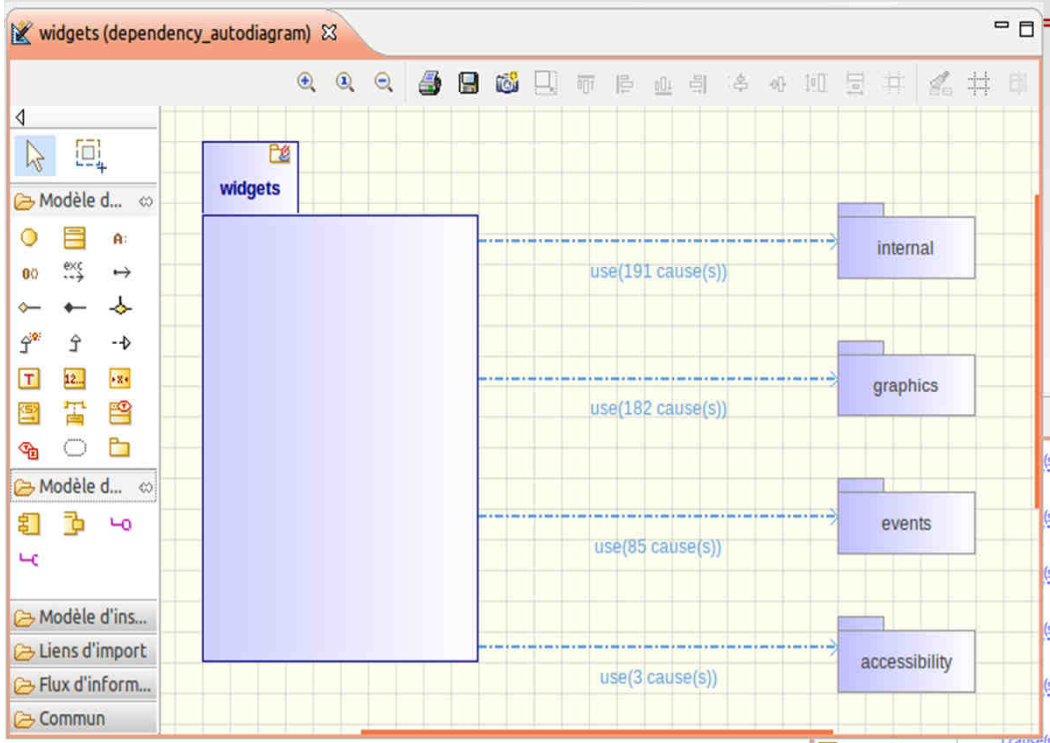
- Le « modèle d'abord » ne doit pas être imposé
- « Modéliser tout le temps » non plus
- Autoriser un référentiel souple : « exemples », « tentatives », « idées », « vision » sont des modèles utiles

- Par exemple : Use Cases ≠ User Stories. L'un explicite les exigences en amont, l'autre fournit une courte phrase déterminant ce que le système doit faire
  - *Log in to my diversified portfolio portal*
  - *See my daily loss or profit.*
  - *Check my current long term interest rates*.

Valeur ajoutée des modèles sur le code existant

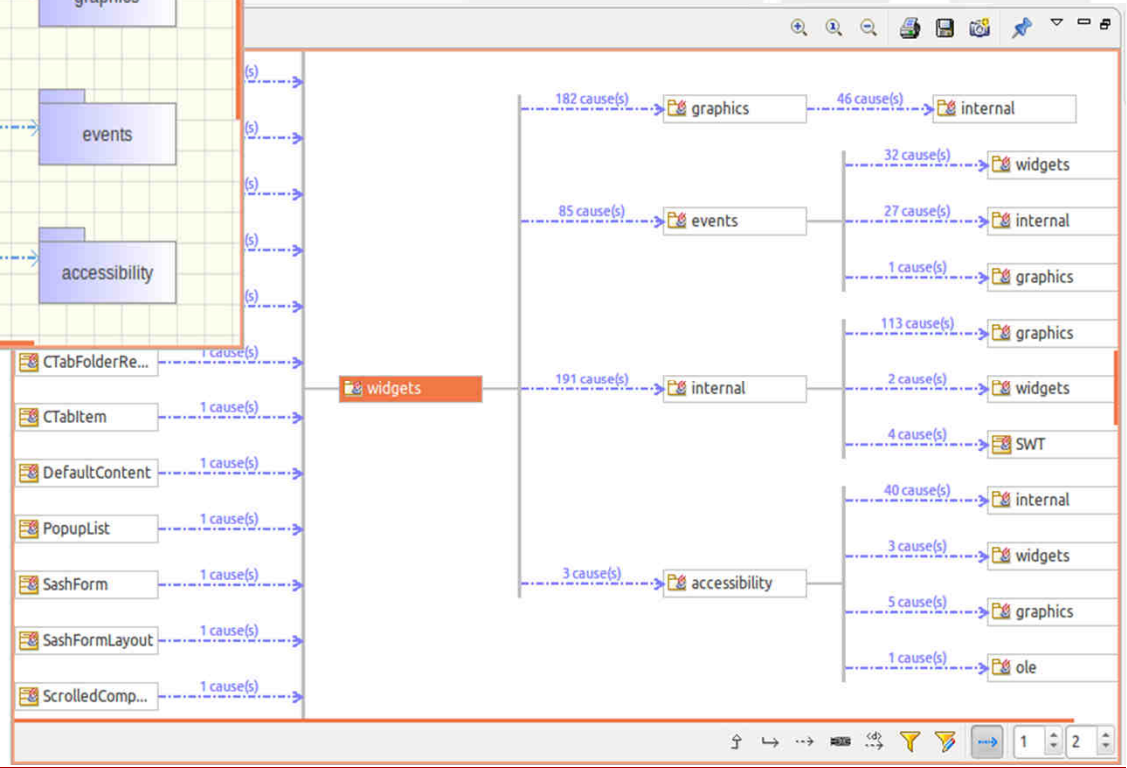
- L'existant en code est énorme : applications, frameworks, librairies …
- Le programmeur doit pouvoir bénéficier des modèles à tout moment (en cours de développement, en maintenance, en mode « pompier » sur existant en crise, …)
- Un support puissant du « reverse engineering » doit permettre
  - De basculer « code pure » vers code guidé par modèle
  - De faire un audit « niveau modèle » d'un code complexe
  - De rétro-documenter
  - de restructurer et moderniser le code

Synthèse d’un code existant



Diagramme

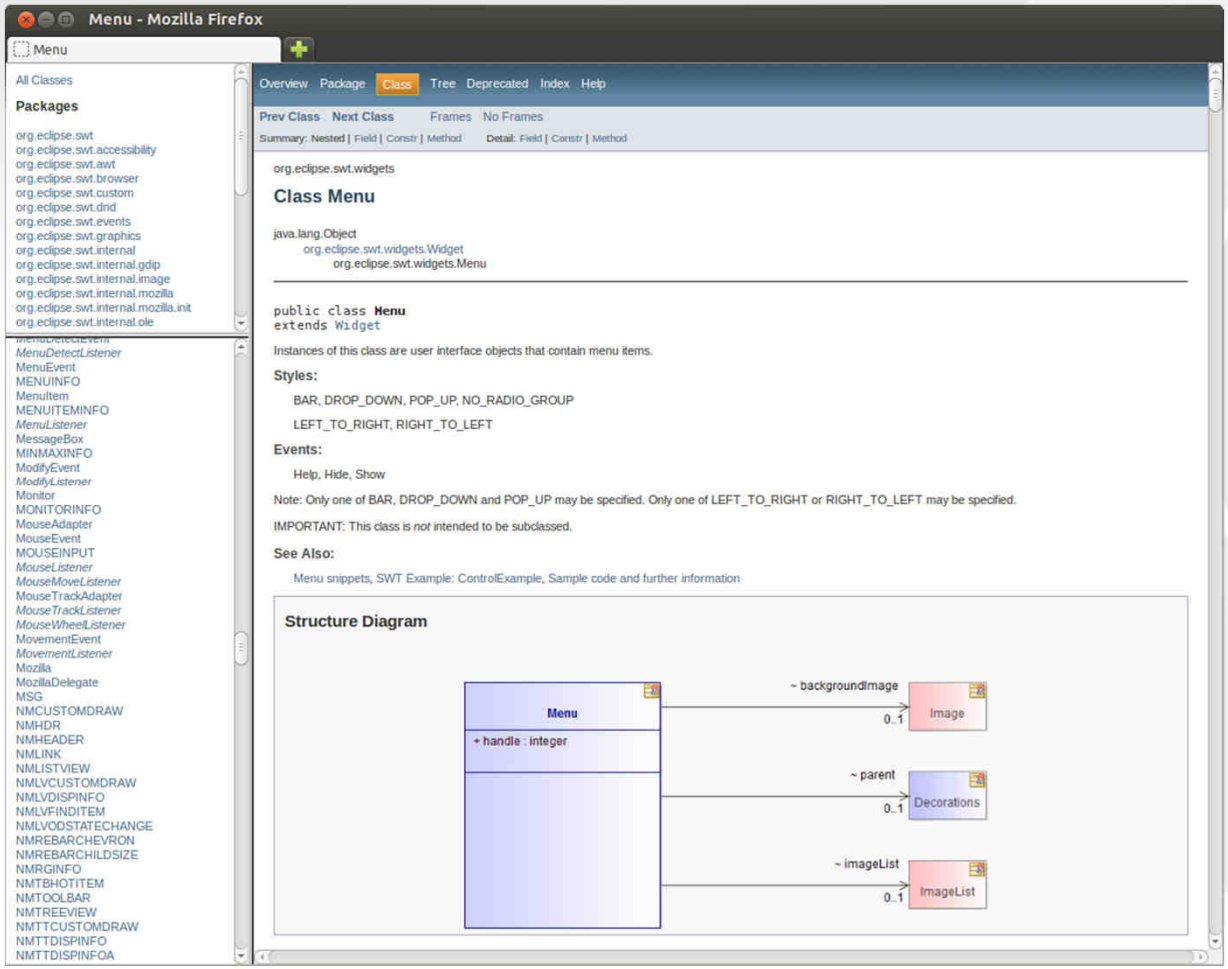
Explorateur Graphique



Synthèse des dépendances

Rétrodocumentation

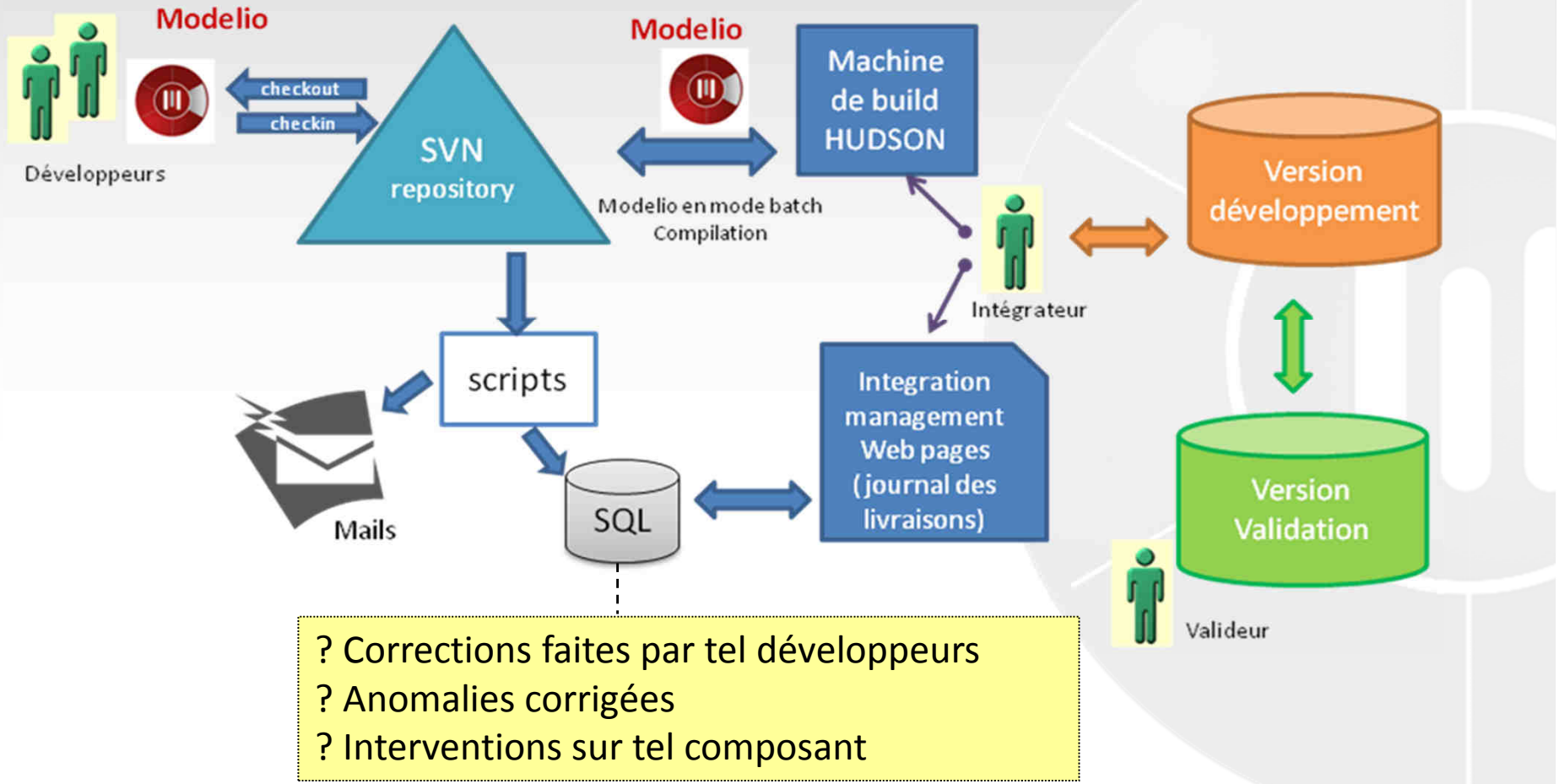
Javadoc augmentée



MDA Agile :
L'outillage doit être intégré dans la chaîne de production

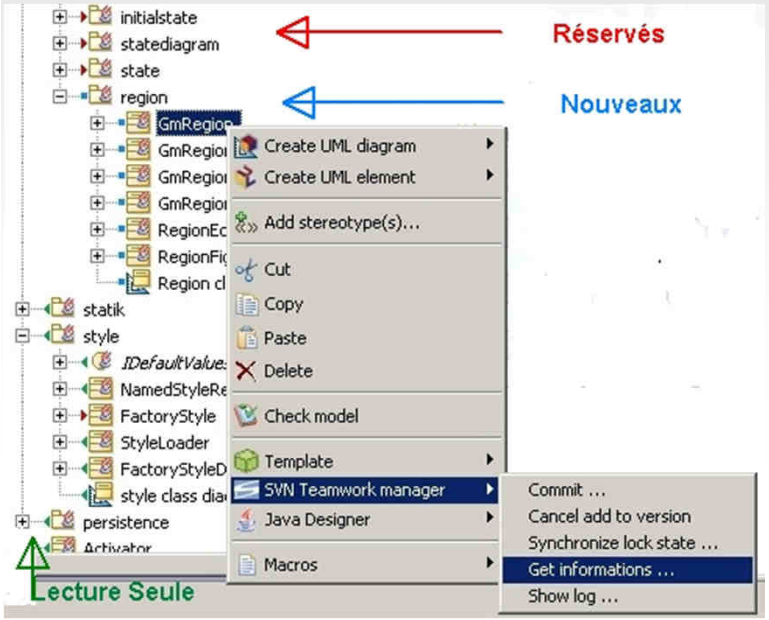
- MDA nécessite une organisation d'équipe dédiée, des outils adaptés pour passer à l'échelle de grandes équipes
- Sa mise en place, en phase initiale doit aboutir à une chaine de production intégrée pour un développement agile
- Domaines essentiels souvent rébarbatifs pour les développeurs peuvent être facilités par une approche guidée par le modèle
  - Gestion de version et configuration,
  - Intégration,
  - Validation,
  - Organisation du travail de groupe

Intégrer l'atelier de modélisation dans la chaîne de production (exemple)

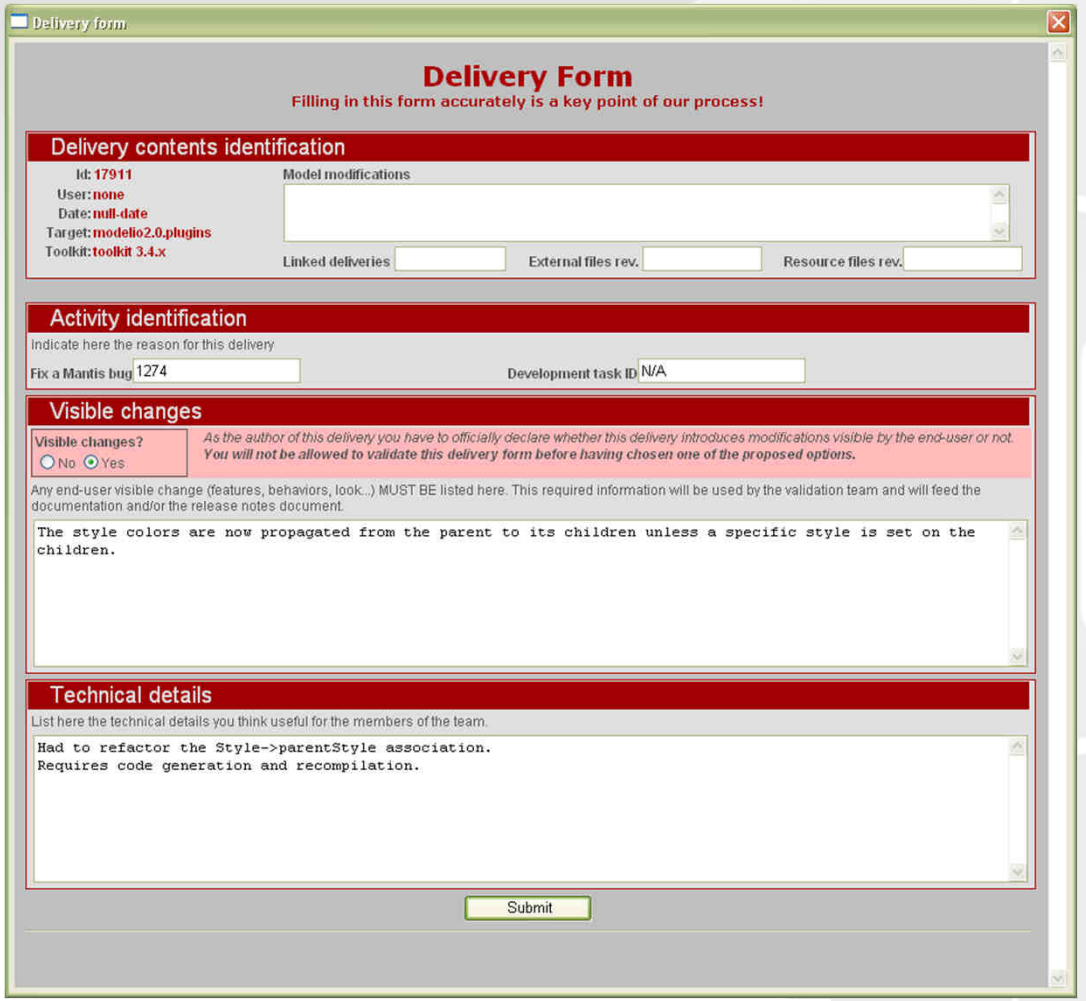


Model Driven, MDA, Intégration continue

Customisation Modelio: Outillage Dédié



Modelio
Teamwork manager



Paramétrage Modelio

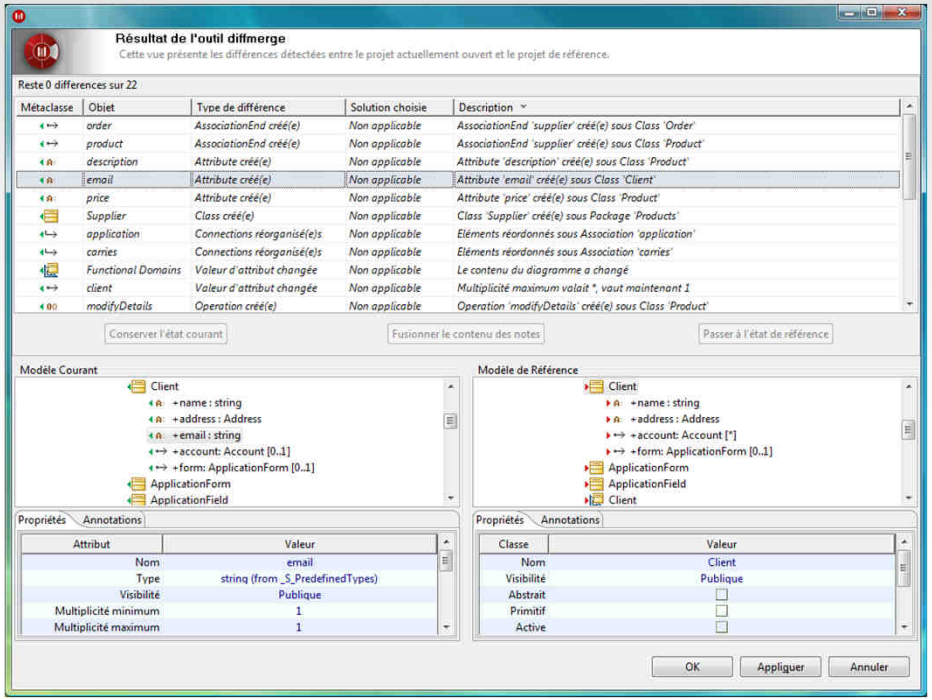
MDA: plusieurs pratiques au sein du même projet

- Génération complète (Metamodèle ➔ framework): Travail exclusif au niveau du modèle
- Génération partielle (conception technique): Génération en mode « Model Driven ». Le modèle est maître. La modification de code est encadrée (balisée). Architecture globale, classes essentielles et structurantes.
- Génération « roundtrip » (codage): Le code est maître. génération depuis modèle, modification/synchronisation à partir de code. Classes techniques, complémentant l'architecture (ex: IHM, « ActionListener », …)

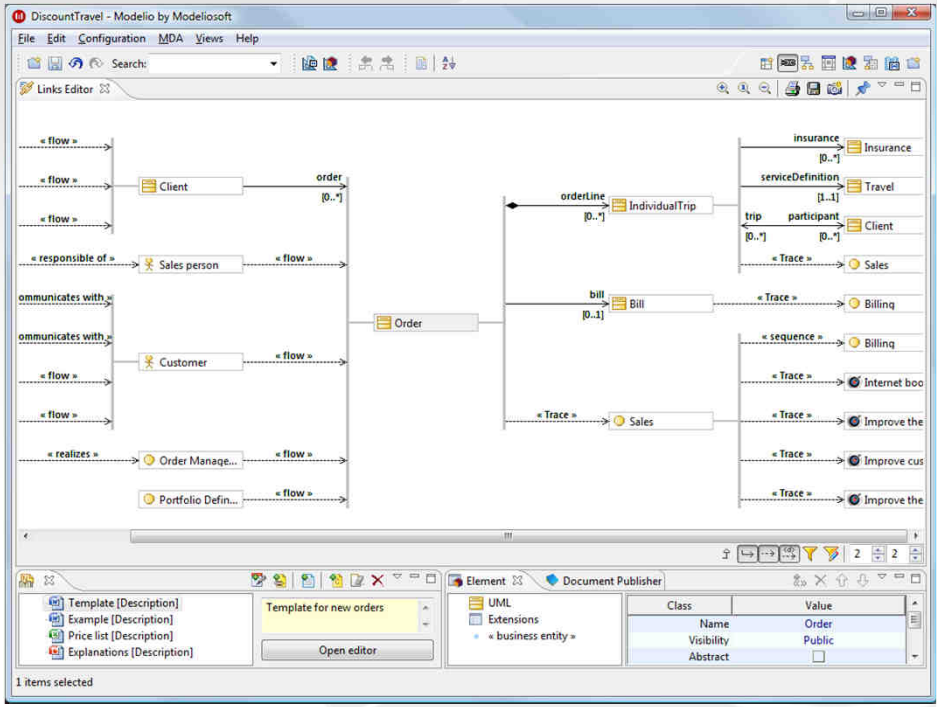
Passage à l'échelle MDA: on évite

- Cascade de transformations de modèles/mises à jour manuelles
  - Mise à jour des modifs d'un modèle vers un modèle transformé: OK
  - ~~Mise à jour des modifs d'un modèle vers un modèle transformé modifié manuellement~~
- Plusieurs modèles/artefacts partiellement déduits les uns des autres, chacun gérés en configuration (ex: modèle et code séparés)
- Un modèle de haut niveau sur lequel interviennent un nombre limité de personnes génère/met à jour des modifications disséminées dans un modèle de plus bas niveau sur lesquelles interviennent plusieurs personnes

Modelio : Des outils pour assister les développeurs sur les grands projets



Diff/Merge de modèles



Analyse d'impact sur modèles

Gains de productivité: ne pas oublier l'essentiel

- Coût codage = 30% coût de réalisation
- Coût de réalisation = 30% coût de possession d'un logiciel

  ➔ Maintenabilité constitue un enjeu très important

- Une approche guidée par le modèle bien gérée
  - Améliore l'analyse et la conception, l'architecture
  - Automatise (partiellement) le codage
  - Assiste la gestion de configuration, de version, test et validation donc améliore la maintenabilité
- Gains obtenus par des procédures et de l'outillage adaptés.
- Pour être compatibles avec l'agilité, les outils de modélisation doivent apporter des bénéfices tangibles aux développeurs