

Sodifrance

Le MDA Mises en œuvre industrielles

IFSIC – Janvier 2006



Sodifrance

- Société de **prestations de services** de Proximité
- **Spécialisation** en modernisation des patrimoines applicatifs
- Filiale d'**édition de logiciels** pour le développement d'applications


En bref

- **600 collaborateurs**
- Une clientèle de **grands comptes**
- **Proximité des sites clients** : Savoir-faire en prestations de services réalisées depuis nos 12 agences
- **Expertise Technologique** reconnue sur le marché de la modernisation industrielle des systèmes d'information
- Capitalisation du savoir-faire dans une **offre logicielle** à forte valeur ajoutée

Références clients


- CNP
- MMA
- CREDIT MUTUEL
- MAIF
- AXA Belgique
- MAAF
- MEDERIC
- CREDIT AGRICOLE
- ING SUISSE ET BELGIQUE
- BERGERAT MONNOYEUR
- LA POSTE
- BANQUE POPULAIRE...





Sodifrance

- Siège basé à **Rennes**
- Centre de R&D basé à **Nantes** (env. 20 personnes)
- Plate-formes projet à **Nantes, Rennes, Orléans et Toulouse**



SODIFRANCE
L'inspiration technologique



SOMMAIRE

- Introduction
- Des outils MDA
- Processus de développement
- La migration
- Au-delà du génie logiciel ...
- Conclusion

SODIFRANCE
L'inspiration technologique



SOMMAIRE

- **Introduction**
- Des outils MDA
- Processus de développement
- La migration
- Au-delà du génie logiciel ...
- Conclusion

 **SODIFRANCE**
L'inspiration technologique



LE CONSTAT

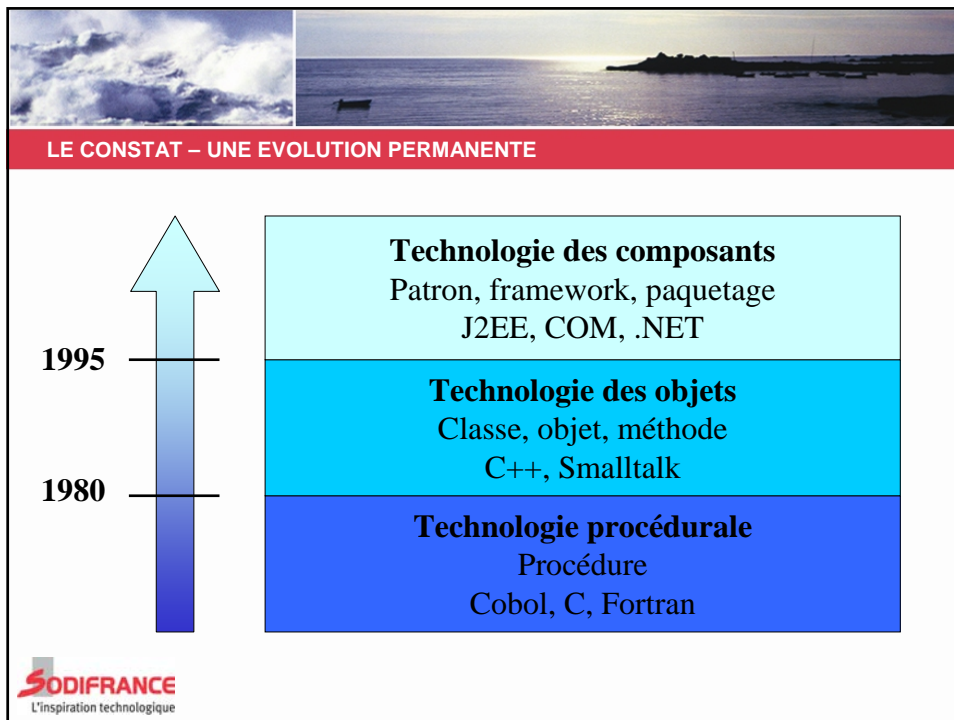
Quels sont les objectifs d'une DI ?

- Intégrer au plus vite les nouveaux besoins métier,
- Maîtriser les dernières innovations technologiques,
- ... pour livrer des applications adaptées et performantes !

Quelles sont ses contraintes ?

- Maîtriser les Coûts et les Délais,
- Assurer la Qualité des applications,
- Garantir un niveau de Risque minimal,
- Valoriser le Patrimoine applicatif

 **SODIFRANCE**
L'inspiration technologique



LE CONSTAT – UNE EVOLUTION PERMANENTE

A chaque rupture technologique, migration des applications

- Processus coûteux sans valeur ajoutée pour l'utilisateur

Cette évolution est sans fin

- Des migrations perpétuelles
- Un poids financier permanent pour les entreprises

Les entreprises ne veulent plus payer le prix fort pour passer d'une plate-forme à une autre alors que le modèle métier n'évolue pas !!!

SODIFRANCE
L'inspiration technologique



LE CONSTAT – DES SYSTEMES HETEROGENES

Raisons

- Développements à des périodes différentes
- Rachat / fusion d'entreprises
- Intégration de progiciels

Le système d'information d'une entreprise devient un vrai plat de spaghetti

- Des liens inter-applicatifs dans tous les sens
- Multiplication des couches pour l'interopérabilité

Problèmes

- Avoir une vue globale du système
- Assurer son évolutivité

**SODIFRANCE**
L'inspiration technologique



LE CONSTAT – UNE COMPLEXIFICATION

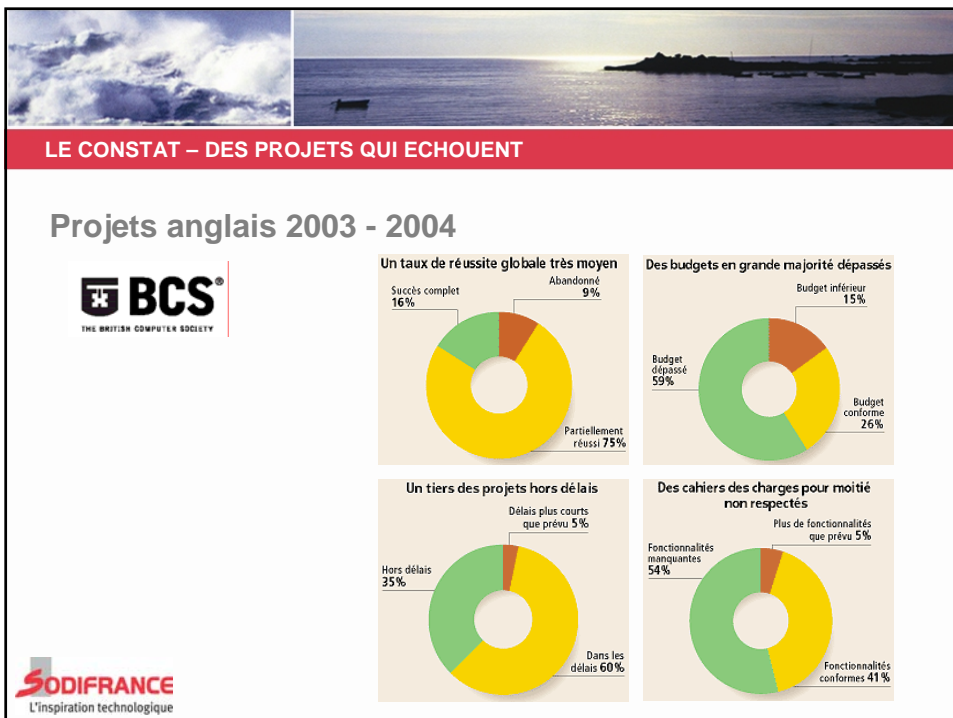
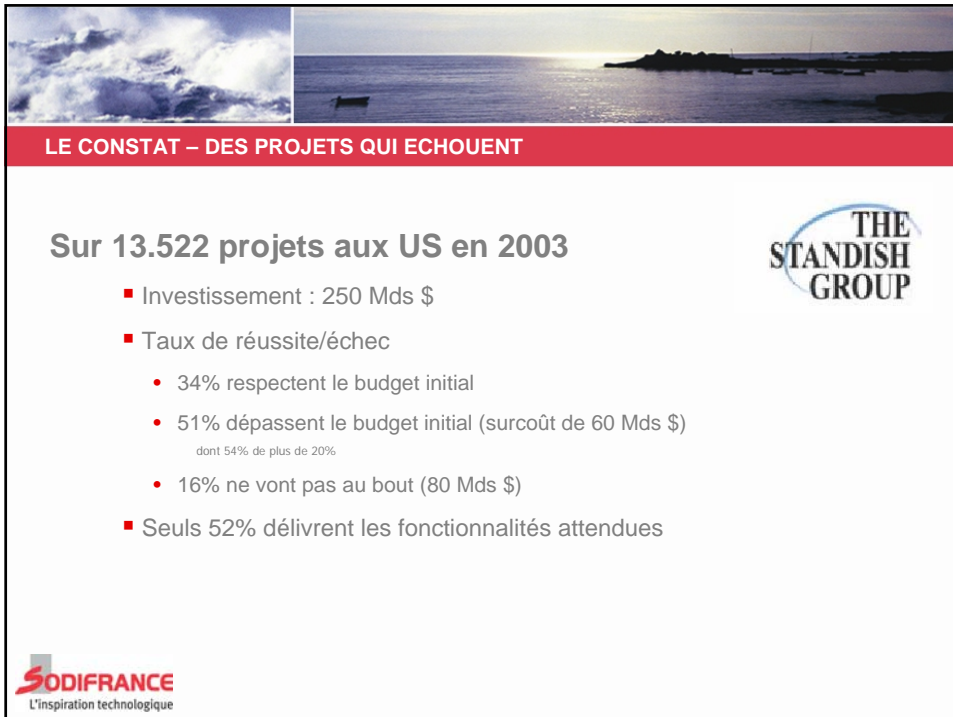
Au temps préhistorique :

- 1 langage : Cobol
- 1 concept : la procédure

Aujourd'hui

- N langages : Java, HTML, JSP, Javascript, XML, SQL ...
- N concepts : classe, composant, framework, patron, aspect ...

**SODIFRANCE**
L'inspiration technologique






RESULTAT

Les décideurs disent stop


- Investissement informatique trop important
- Mal maîtrisé
- ROI peu évident



Emergence de solutions alternatives


- Offshore, nearshore ... (réduction du coût de la main d'œuvre)
- Engouement pour l'open-source (réduction du coût du logiciel)
- MDA (réduction du besoin en main d'oeuvre)

SODIFRANCE
L'inspiration technologique



LE MDA

MDA : Model Driven Architecture




Initiative de l'OMG (septembre 2001)

Objectif :

- **Pérenniser** les investissements informatiques en déduisant **automatiquement** un SI à partir de sa description dans un formalisme **indépendant** des technologies cibles.

SODIFRANCE
L'inspiration technologique



LES PRINCIPES DU MDA

Principe :

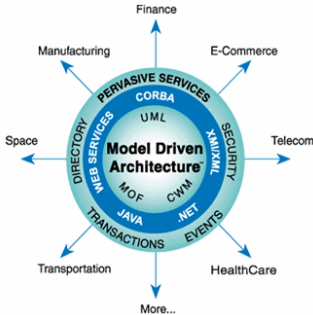
- Spécifier un système sous la forme de modèles sur lesquels on applique des transformations

Deux natures de modèles :

- PIM : Platform Independent Model
- PSM : Platform Specific Model

Bases du MDA

- Standards de modélisation (UML)
- Standards de méta-modélisation (MOF)
- Techniques de transformation de modèles et de génération de code



SODIFRANCE
L'inspiration technologique



UTILISATION SUR DES PROJETS DE DEVELOPPPEMENT

Modélisation

- Données, cinématique, processus, ...

Génération de code

- Vers l'architecture cible du projet
- Exemple : Java/J2EE, Struts, JSP, Spring, Hibernate, Log4J

Nouveau processus de développement

- Cycle en Y
- Mise en œuvre de la séparation des préoccupations

SODIFRANCE
L'inspiration technologique



ET LES APPLICATIONS EXISTANTES ?

Problème

- Comment les intégrer dans le MDA ?


Solution 1

- En les encapsulant (par des services Web, ...)
- On ne les maîtrise toujours pas

Solution 2

- En utilisant les principes du MDA (transformation de modèles, modélisation)

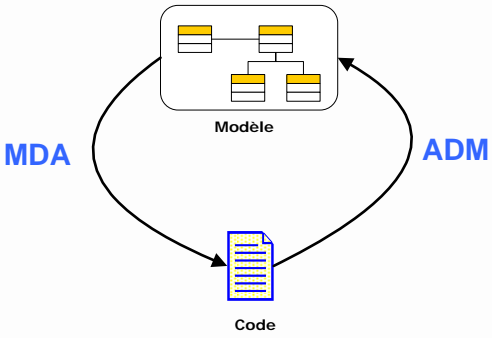
SODIFRANCE
L'inspiration technologique



ADM

Groupe de travail de l'OMG
Architecture-Driven Modernization

- Compréhension
- Evolution
- Refactoring
- Restructuration
- Migration
- ...



SODIFRANCE
L'inspiration technologique



AU-DELA DU GENIE LOGICIEL

La modélisation est une technique universelle

- Constructeurs d'avions, de voitures
- Opérateurs de télécommunication
- Militaires

Normalisée

- ISO (STEP)

Utilisée dans de nombreux outils de conception

- CATIA (modélisation 3D)
- CORE (ingénierie système)
- WINDCHILL (PDM / gestion des données produit)

Quel lien avec le MDA ?

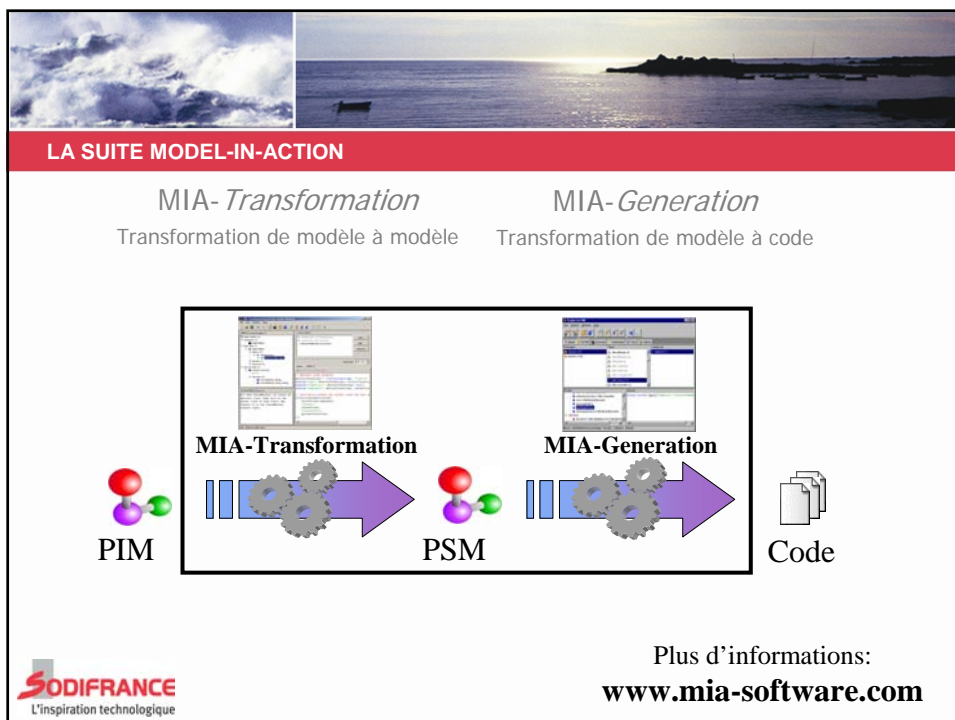
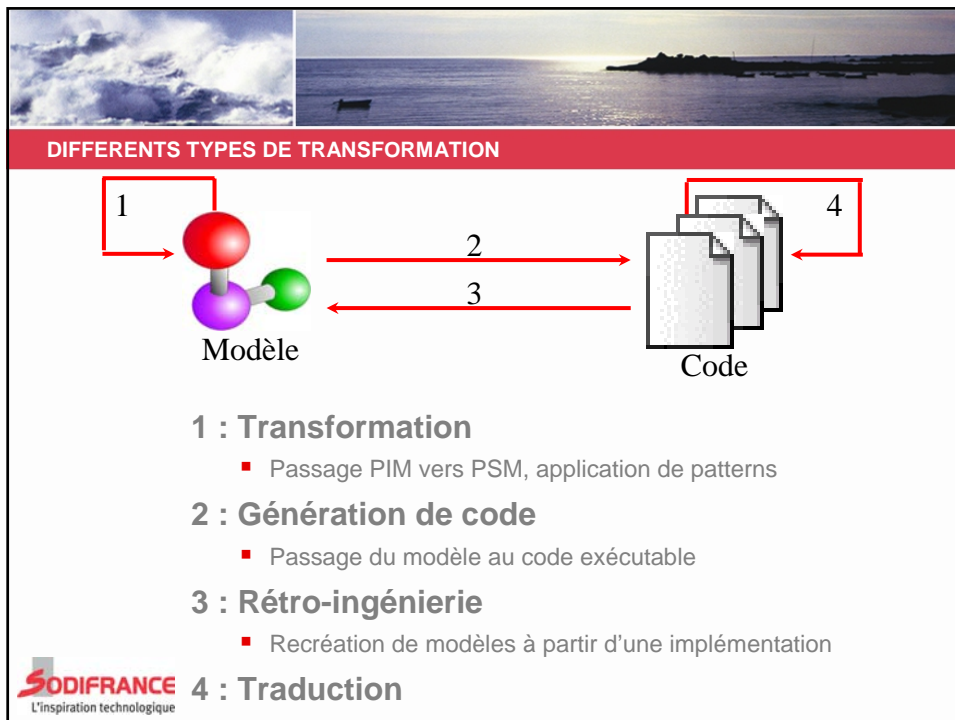
 **SODIFRANCE**
L'inspiration technologique



SOMMAIRE

- Introduction
- **Des outils MDA**
- Processus de développement
- La migration
- Au-delà du génie logiciel ...
- Conclusion

 **SODIFRANCE**
L'inspiration technologique





MIA-TRANSFORMATION

Transformation de modèles à modèles

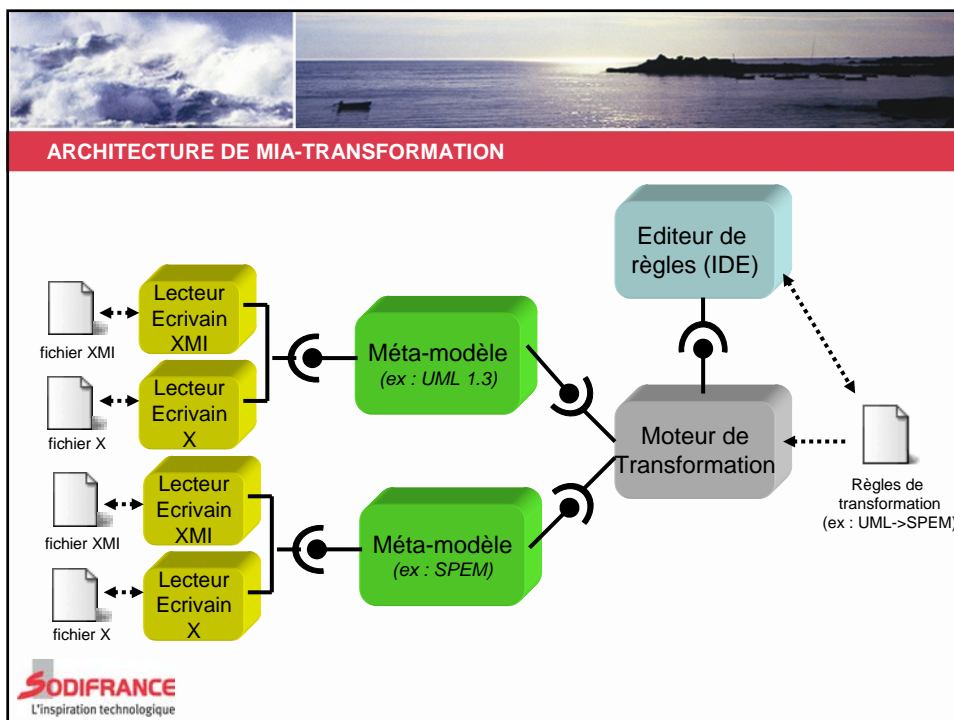
- Plusieurs modèles en entrée comme en sortie
- Basés sur des méta-modèles différents

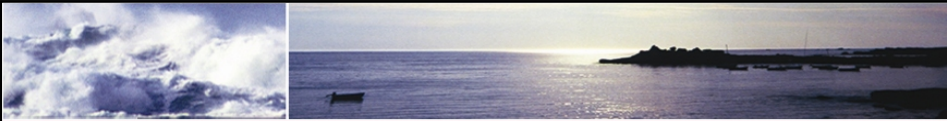
MIA-Transformation, c'est :

- Un éditeur de règles
- Un moteur d'exécution
- Des connecteurs d'import / export de modèles

Intégration d'un méta-modèle

- Environnement généré à partir du méta-modèle
- Lecteur / écrivain XMI



PRINCIPES DE MIA-TRANSFORMATION

Transformation à base de règles

- Une transformation = un ensemble de règles (rule set) entre des méta-modèles

Une règle, c'est

- Un contexte d'exécution
 - déclaration des objets
- Une requête
 - recherche d'objets
- Une action
 - création d'objets

Local context

- ◆ class (uml.Class)
- ◆ constructor (uml.Operation)

Add...
Edit...
Remove


View kind: Split view

[MIA-TL] List Query()

```
class = allOfClass(uml, "Class");
filter(! class.isAbstract);
```

[MIA-TL] Void Action()

```
constructor = createInstance(uml, "Operation");
addLink("name", constructor, class.name);
addLink("owner", constructor, class);
```



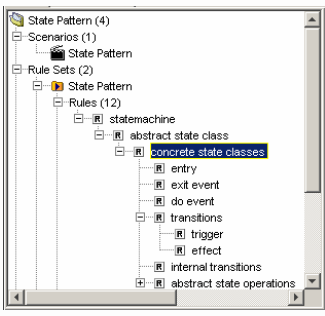

PRINCIPES DE MIA-TRANSFORMATION

Règles arborescentes

- Une règle peut avoir des sous-règles

Evaluation des sous-règles

- Si la règle parente a été exécutée
- Dans le contexte de la règle parente



The screenshot shows a tree view with the following structure:

- State Pattern (4)
 - Scenarios (1)
 - State Pattern
 - Rule Sets (2)
 - State Pattern
 - Rules (12)
 - statemachine
 - abstract state class
 - concrete state classes
 - entry
 - exit event
 - do event
 - transitions
 - trigger
 - effect
 - internal transitions
 - abstract state operations





PRINCIPES DE MIA-TRANSFORMATION

Services


- Macro en langage Java
- Permettent de définir des manipulations complexes

Scénarios

- Enchaînements de transformation
- Permet de livrer des composants de transformation packagés

Projets

- Regroupement de règles, de services et de scénarios

MIA-GENERATION

Objectifs

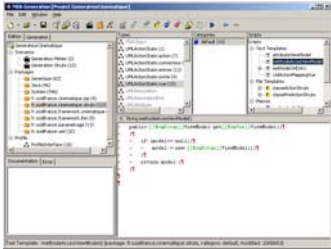

- Production automatique de code
- Permet de créer des générateurs sur mesure,
- dédiés à l'architecture cible

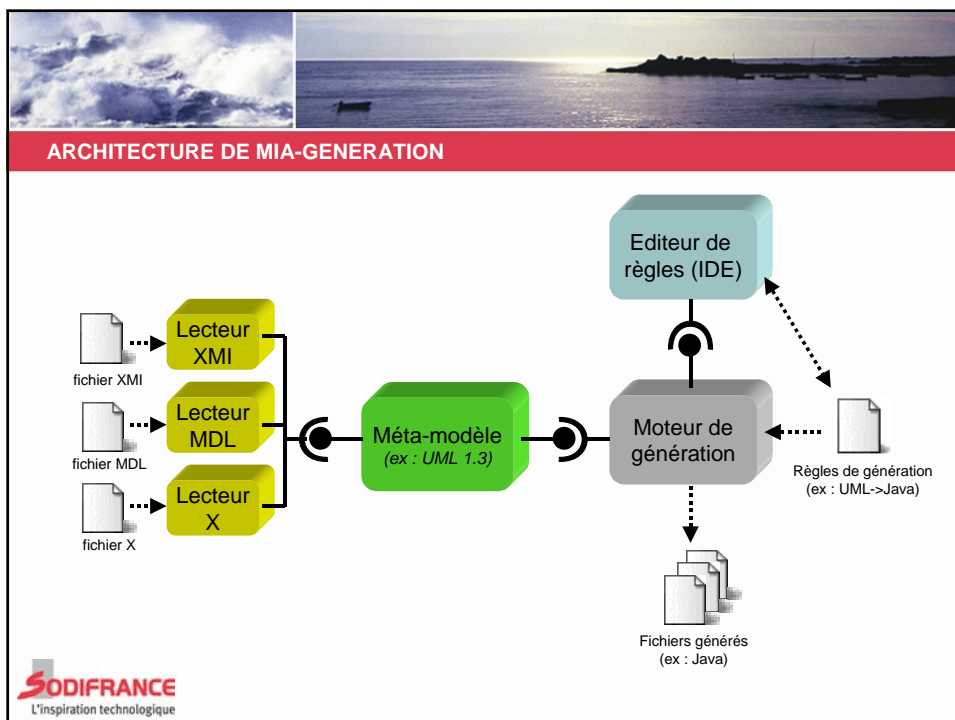
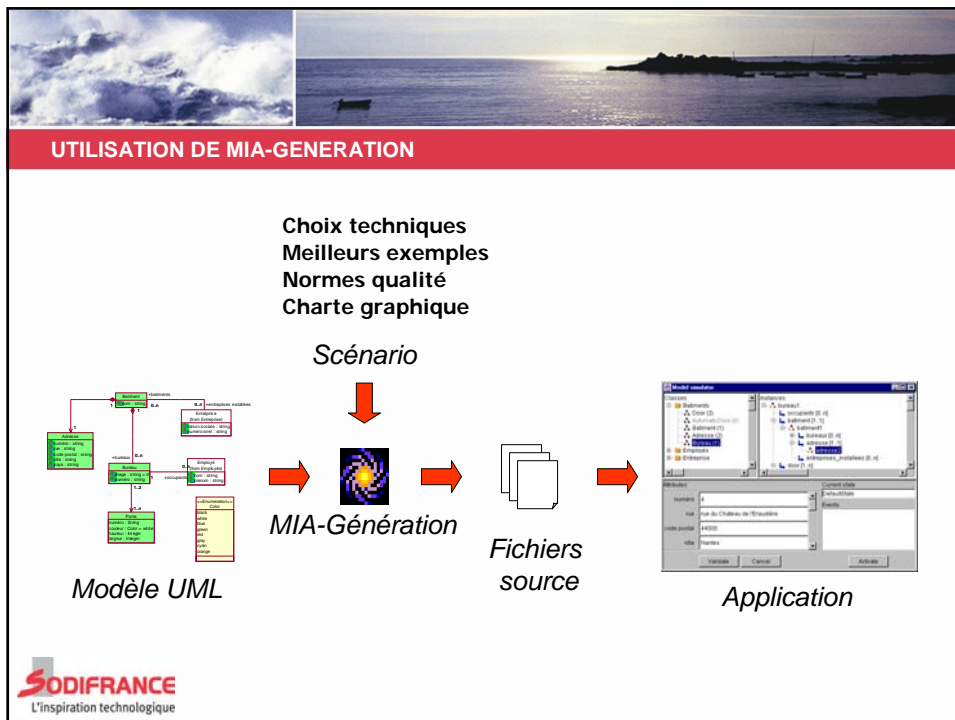
Basé sur les standards

- MOF, XMI
- Java

Points forts

- Indépendant de tout méta-modèle
- Architecture modulaire
- Mécanisme de template





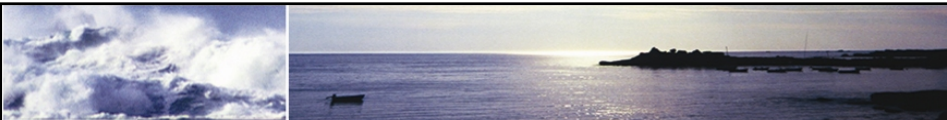
PRINCIPES DE MIA-GENERATION

Les règles de génération sont exprimées à travers des scripts évalués sur des objets d'un modèle

- 2 familles de scripts
 - Templates (WYSIWYG)
 - Code Java


Un projet de génération définit un générateur de code. Un projet =

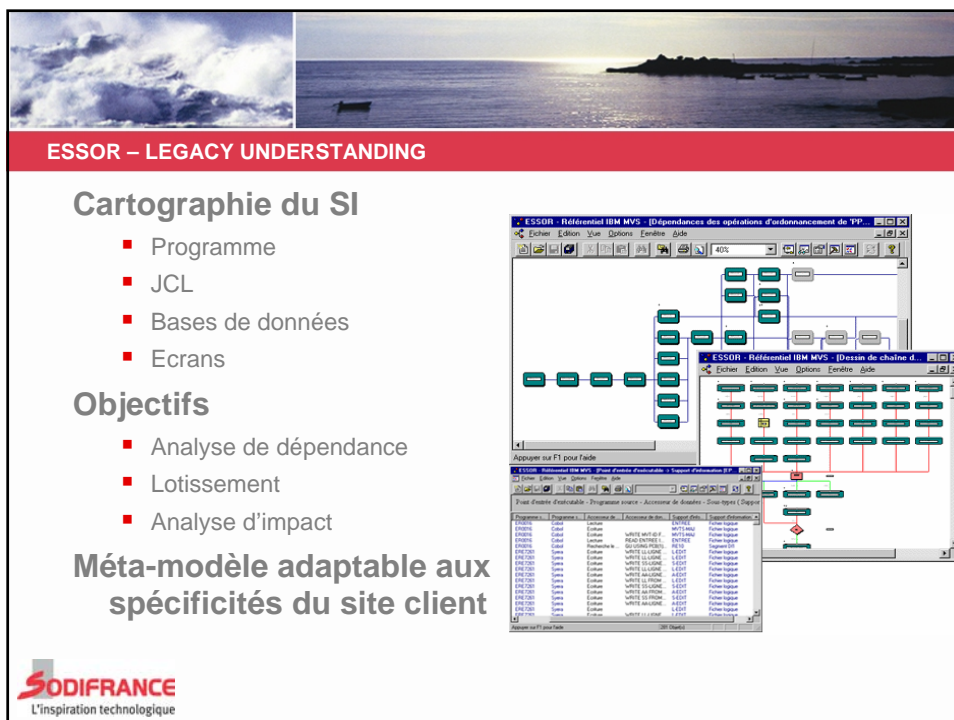
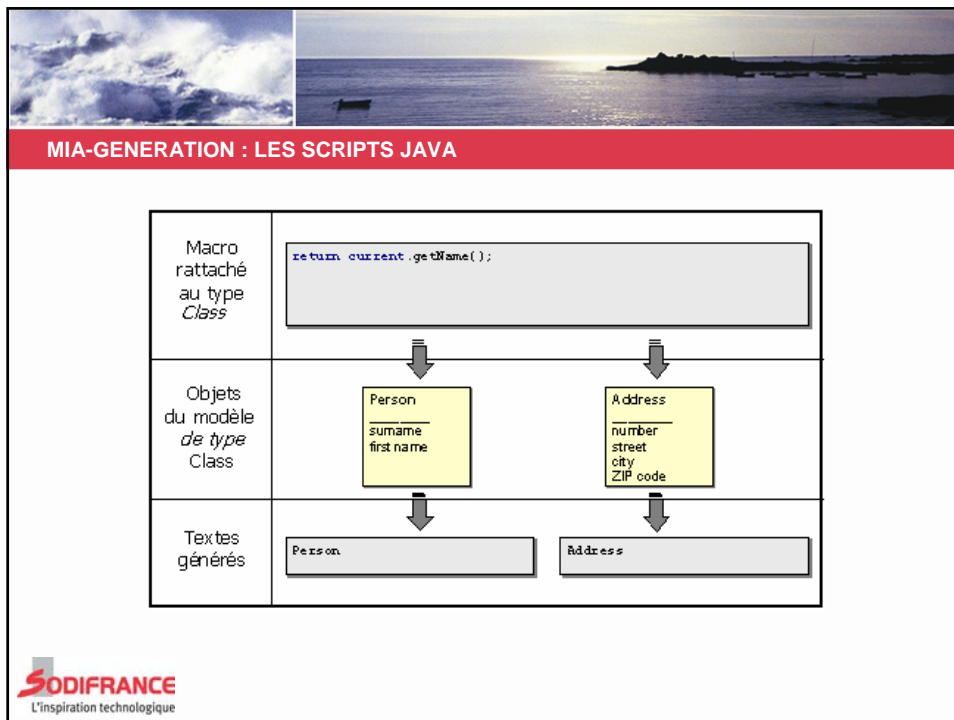
- Des Packages
 - ensemble de scripts de génération
- Des Scénarios
 - descriptions d'enchaînements de générations
- Un Profil
 - personnalisation du méta-modèle

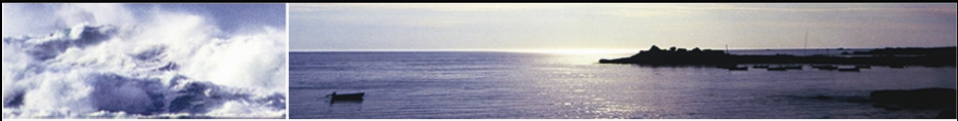



MIA-GENERATION : LES TEMPLATES

Template rattaché au type <i>Class</i>	<pre>Name: [[name]] Attributes: [[attributes]]</pre>
Objets du modèle <i>de type</i> Class	<div style="display: flex; justify-content: space-around; align-items: center;"> <div style="border: 1px solid black; padding: 5px; text-align: center;"> Person ----- surname first name </div> <div style="border: 1px solid black; padding: 5px; text-align: center;"> Address ----- number street city </div> </div>
Textes générés	<div style="display: flex; justify-content: space-around;"> <div style="border: 1px solid black; padding: 5px;"> <pre>Name: Person Attributes: surname first name</pre> </div> <div style="border: 1px solid black; padding: 5px;"> <pre>Nom: Address Attributes: number street city</pre> </div> </div>







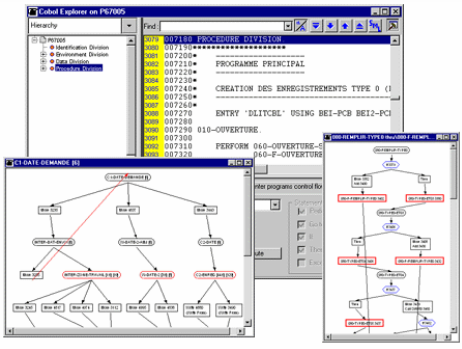
ESSOR – APPLICATION MINING

Analyse fine de programmes



- Cobol
- RPG (AS400)

Objectifs

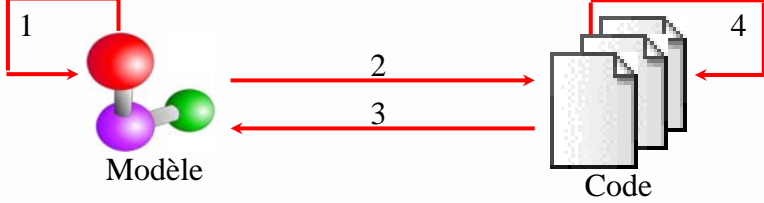
- Flot de contrôle
- Slicing
- Extraction des règles métier
- Analyse qualité




Méta-modèle spécifique au langage

RECAPITULATIF



1 : MIA-Transformation
 2 : MIA-Generation
 3 : Essor, Semantor
 4 : -






SOMMAIRE

- Introduction
- Des outils MDA
- **Processus de développement**
- La migration
- Au-delà du génie logiciel ...
- Conclusion

 **SODIFRANCE**
L'inspiration technologique



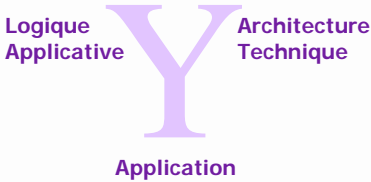
LE CYCLE EN Y


Principe

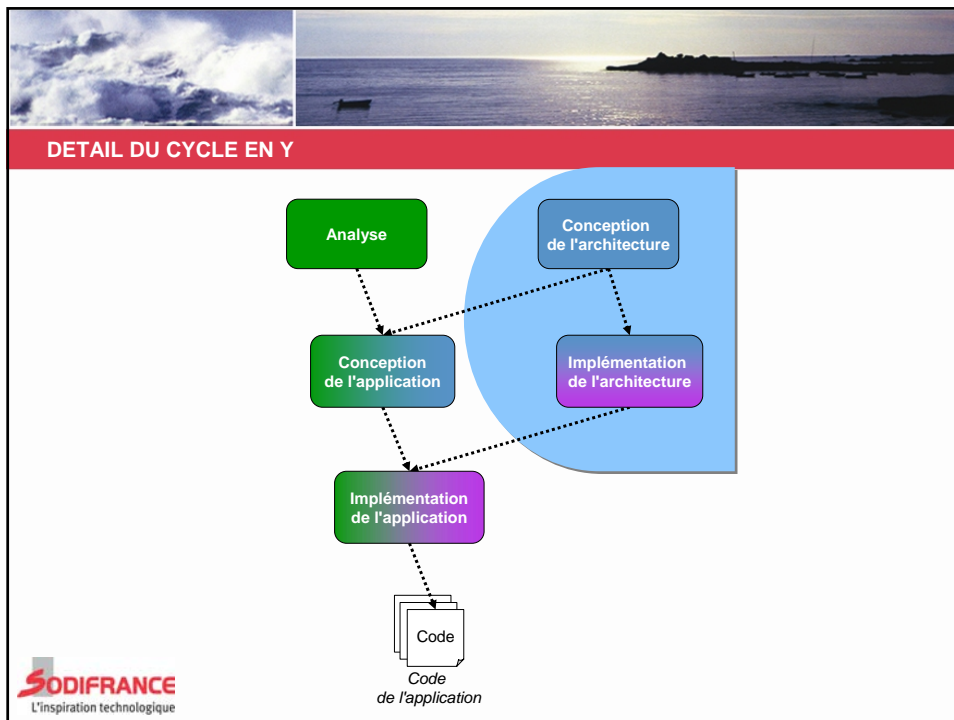
- Dissocier Architecture Technique et Logique Applicative

Intérêts

- Employer au mieux les compétences
- Réutiliser la logique applicative (avec une autre architecture technique)
- Réutiliser l'architecture technique (avec une autre logique applicative)



 **SODIFRANCE**
L'inspiration technologique



LES ROLES – L'EXPERT METIER

L'expert métier


- exprime les besoins fonctionnels de l'application :
 - données manipulées (contrat, cotisation, devis, ...)
 - services rendus (souscription, résiliation, tarification, ...)
 - règles de gestion

Enjeux :

- décrire les besoins d'une application indépendamment d'une architecture technique particulière
 - pérenniser cette description par rapport aux évolutions d'architecture

Expert Métier (with smiley icon) → **Analyse** (green box) → **Modèle fonctionnel** (hierarchical diagram)

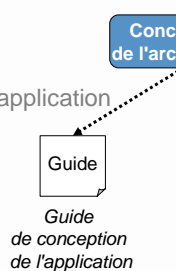
SODIFRANCE
L'inspiration technologique



LES ROLES – L'ARCHITECTE TECHNIQUE

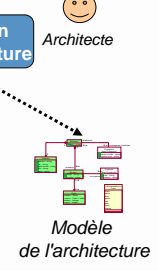
L'architecte technique

- définit les principes architecturaux de l'application
 - client lourd/client léger
 - modèle MVC ?
 - base relationnelle ?



Conception de l'architecture

Guide
Guide de conception de l'application





Modèle de l'architecture

Architecte

Enjeux :

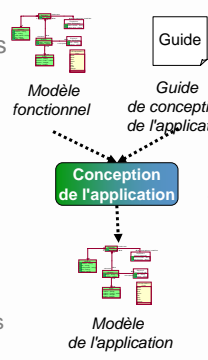
- exprimer l'architecture indépendamment d'une application particulière
 - réutiliser l'architecture pour d'autres applications
- exprimer l'architecture indépendamment d'une technologie d'implémentation
 - pérenniser l'architecture par rapport aux évolutions des technologies

LES ROLES – LE CONCEPTEUR

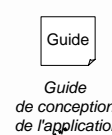
Le concepteur

- décrit l'application en tenant compte des besoins métier et des contraintes architecturales
 - schéma de base
 - composants métier
 - cinématique des écrans



Modèle fonctionnel

Conception de l'application





Guide de conception de l'application

Concepteur applicatif

Enjeux :

- exprimer l'application indépendamment d'une technologie particulière d'implémentation
 - pérenniser l'application par rapport aux évolutions des technologies





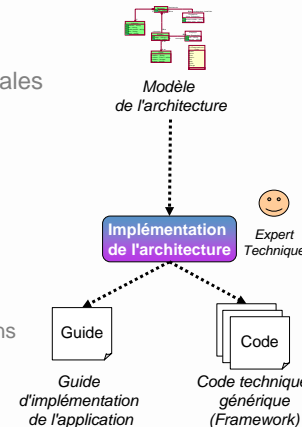
LES ROLES – L'EXPERT TECHNIQUE

L'expert technique

- spécifie les stratégies d'implémentation de l'application à partir des contraintes architecturales
 - framework technique
 - choix des moteurs
 - règles de codage

Enjeux :

- exprimer les stratégies d'implémentation indépendamment d'une application particulière
 - réutiliser cette stratégie pour d'autres applications


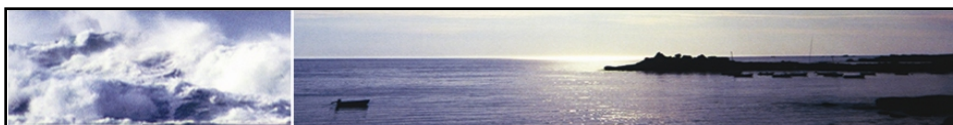


Modèle de l'architecture

Implémentation de l'architecture (Expert Technique)

Guide d'implémentation de l'application

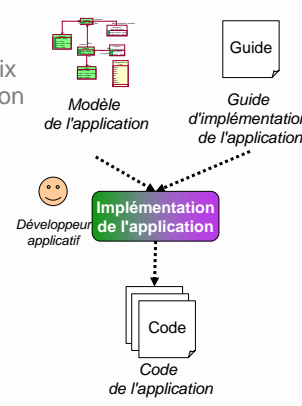
Code technique générique (Framework)

LES ROLES – LE DEVELOPPEUR

Le développeur

- produit le code de l'application à partir des choix de conception et des stratégies d'implémentation
 - scripts de création de la base
 - code des IHM
 - code des composants métier
 - fichiers de déploiement




Modèle de l'application

Guide d'implémentation de l'application

Implémentation de l'application (Développeur applicatif)

Code de l'application





INDUSTRIALISER

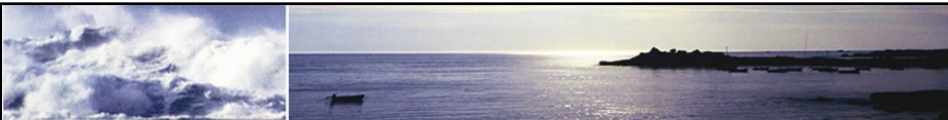
Les objectifs : gagner en ...

- productivité
- qualité
- réactivité

Exemples de moyens

- rationalisation
 - framework
- automatisation
 - génération de code
 - workflow
- assistance
 - environnement de développement
 - outil de modélisation
 - gestion de configuration

**SODIFRANCE**
L'inspiration technologique




EXEMPLE AVEC STRUTS


Struts

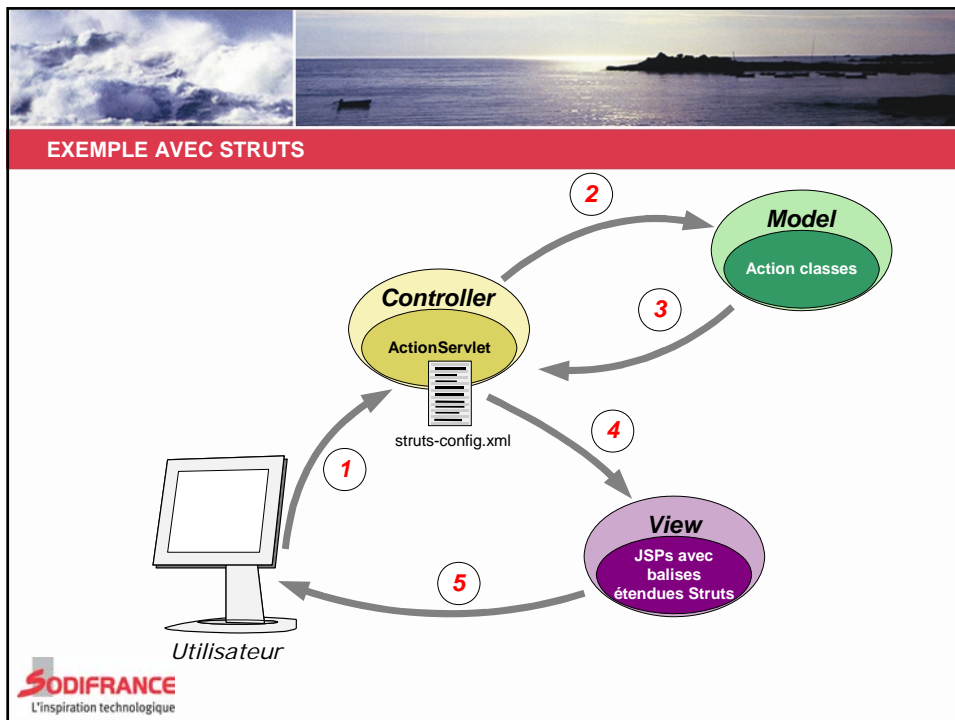
- Framework Java J2EE
- Gestion de la cinématique des applications
- Implémentation du MVC (Model View Controller)

Fonctionnement de Struts

- Des pages JSP
- Des classes Action Java
- Des beans Java pour stocker les données
- Un fichier XML qui définit l'enchaînement des pages et des actions : struts-config.xml

**Struts**

**SODIFRANCE**
L'inspiration technologique



LE FICHIER STRUTS-CONFIG.XML

```

<struts-config>
<form-beans>
  <form-bean name="loginActionForm" type="LoginActionForm" />
  ...
</form-beans>
<action-mappings>
  <action forward="/SaisieMotDePasse.jsp" path="/saisieMotDePasse" />
  <action name="loginActionForm" type="LoginAction"
    path="/loginAction">
    <forward name="saisieMotDePasse" path="/saisieMotDePasse.do" />
    <forward name="erreurSysteme" path="/erreurSysteme.do" />
    <forward name="erreurLogin" path="/erreurLogin.do" />
    <forward name="accueil" path="/accueil.do" />
  </action>
  ...
</action-mappings>
</struts-config>

```

The SODIFRANCE logo is in the bottom left corner.



DIFFICULTES

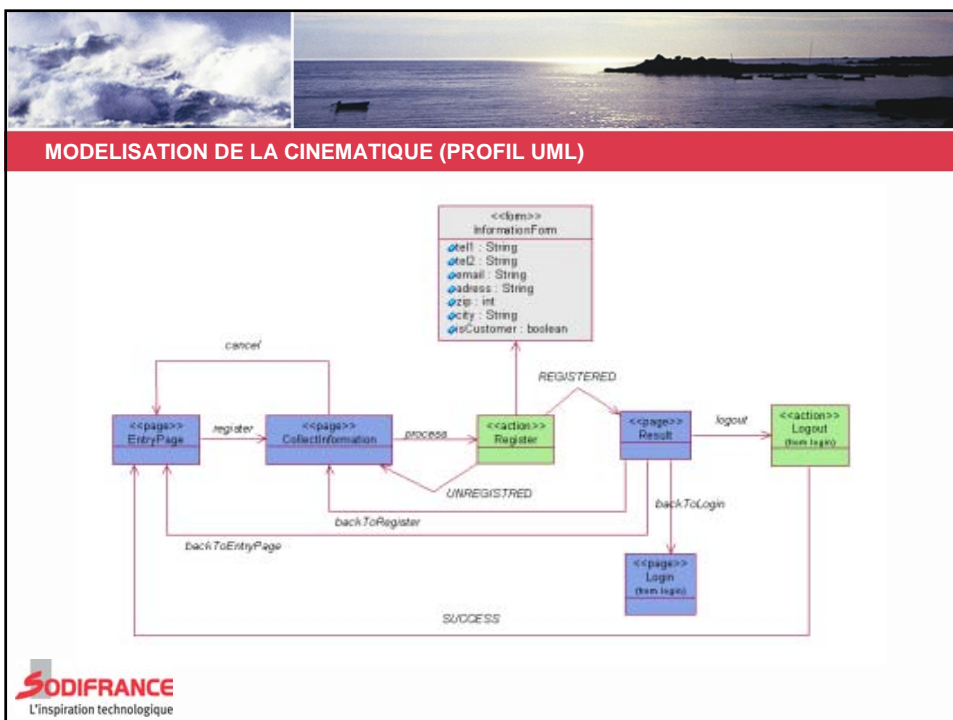
Problèmes

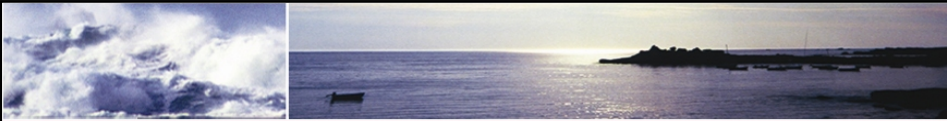
- Volume du fichier
- Manque de lisibilité
- Difficulté à le faire évoluer
- Impossibilité de modéliser la cinématique si l'on ne connaît pas Struts

Solution

- Modéliser
 - Pour masquer la difficulté
- Générer
 - Le fichier de configuration et les classes en indiquant les endroits où introduire du code

SODIFRANCE
L'inspiration technologique





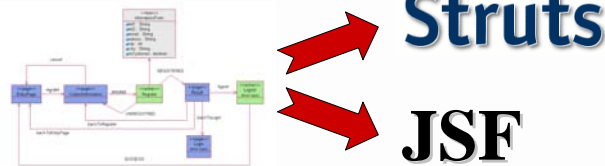
RESULTATS

Simplicité

- Plus besoin d'un expert Struts


Evolutivité

- Du modèle métier
- De son implémentation
 - Passage à JSF transparent



Struts

JSF




MISE EN ŒUVRE DE L'APPROCHE SUR DES PROJETS

Satin : Intranet pour le suivi commercial



- Nombre de personnes : 40
- Architecture : ASP, ActiveX, VB, SQL
- Application : 40 Mo de code applicatif

Opera : Intranet pour le suivi de production

- Nombre de personnes : 20
- Framework et processus identiques à Satin





LE PROJET SATIN

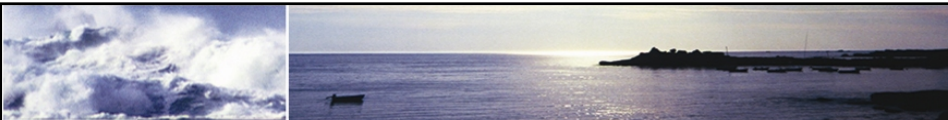



Caractéristiques



- Cycle en Y séparant les activités d'analyse et d'architecture
- Framework d'architecture
- Développements centrés sur le modèle (2500 classes)
- Génération de code, de documentation
- Traçabilité complète depuis les exigences jusqu'au code

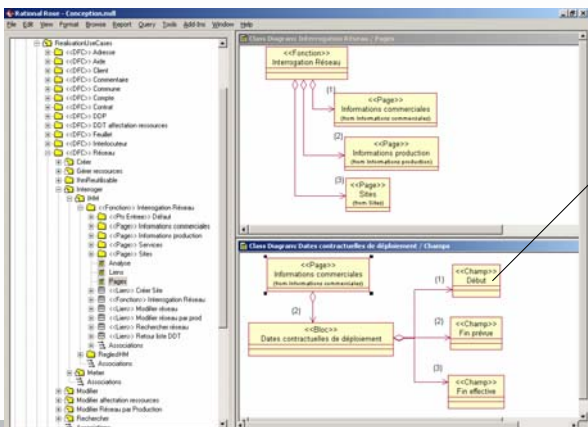
Les principes de modélisation

- Distinction des différents domaines de l'application
- Persistance, objets métier, IHM, navigation, services, ...



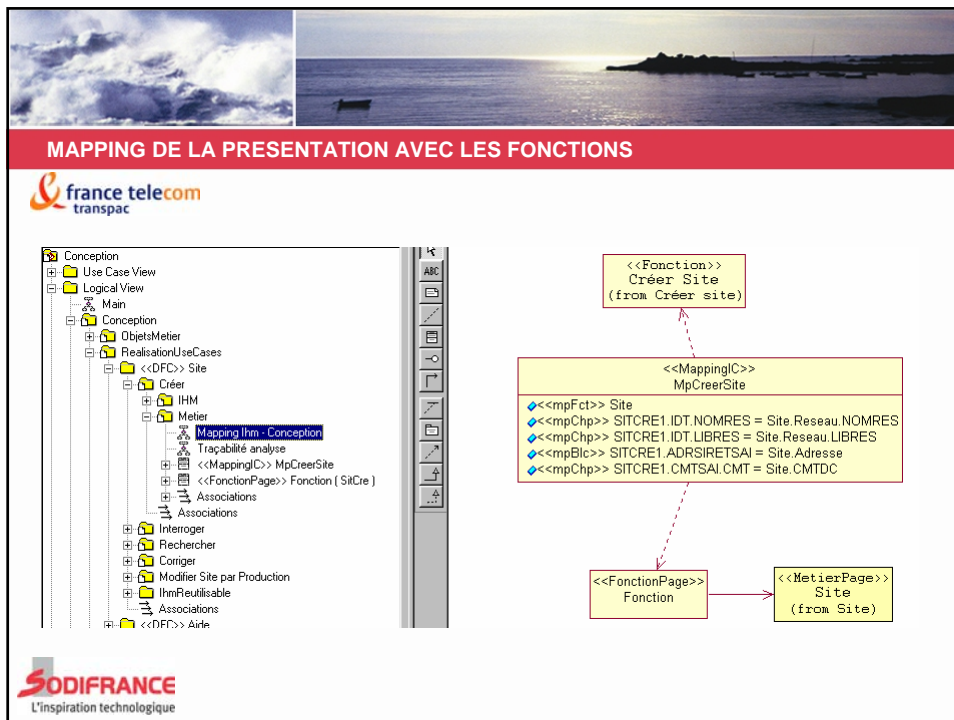
MODELE DE PRESENTATION



Class Specification for Début

Name	Value	Source
Dir	Default	Default
Confidentialité	False	Default
TypeFormat	Standard	Override
Format	DATE	Override
TableA	Default	Default
Longueur	0	Default
Mémoire	DATDEBRES	Override



EXEMPLE D'ECRAN GENERE

france telecom transpac

Satin - Microsoft Internet Explorer

Interrogation d'un compte

Déplacement dans la page

- Identification
- Adresse
- Caract. compte
- Infos société
- Caract. internes
- Gestion des CP
- Commentaires
- Liste contrats

Actions possibles

- Modif compte
- Cloner compte
- Ajout contrat
- Données fin.
- Contenieux

Autres actions

Identification

N° compte	10107 01	Dernière action	(TCTR)	Création	24/02/1983
Etat	FERME (03)	Auteur		Modification	17/10/1990
		Service		Fermeture	26/04/1991
				N° interne	10401

Adresse de facturation

Raison sociale	BANQUE NATIONALE DE PARIS	Téléphone	0140144115
Corresp. / Service	DIVISION FINANCIERE - SGI/OI	Fax	
Adresse	13 RUE LAFFITTE	E-mail	
Complément			
Commune	75009 PARIS	Pays	()

Caractéristiques du compte

Mode de règlement	30 JOURS FIN DE MOIS (30)	RIB	
Devise émission fact.	FRANC FRANCAIS (FRF)	Date effet devise	
Langue émission fact.	FRANCAIS (FRA)		
Facture triée par	NUMERO D'ABONNE (N)	Support facture	PAPIER + FICHIER ELECT. (3)
Traitement facture	NON RETENUE (N)	Facture NTI retenue	NON (N)
Contenieux	()	Type de relevé	B

Informations société

SODIFRANCE
L'inspiration technologique



BILAN



Meilleure productivité

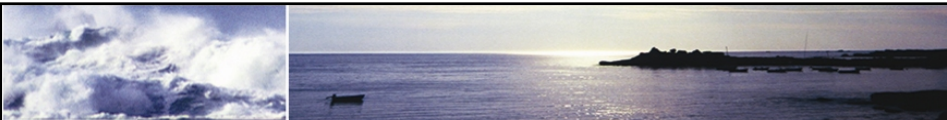
- Fichiers ASP : générés à 97%
- Fichiers VB : générés à 80%
- Fichiers SQL : générés à 100%

Meilleure intégration des évolutions techniques

- Exemple du passage au multilinguisme : 3 étapes
 - Modifier le framework (20 jours)
 - Modifier les générateurs (2 jours)
 - Régénérer toute l'application (1 jour)




SODIFRANCE
L'inspiration technologique



SOMMAIRE

- Introduction
- Des outils MDA
- Processus de développement
- **La migration**
- Au-delà du génie logiciel ...
- Conclusion



SODIFRANCE
L'inspiration technologique



MIGRATION DANS LE CYCLE DE VIE DU PATRIMOINE APPLICATIF

Sources possibles de motivation d'une migration

- Risque d'obsolescence technologique et/ou risque de perte de compétence humaine
- Evolution vers une cible plus pérenne, plus ouverte, plus moderne, plus...
- Volonté de rationalisation-standardisation des technologies / filières de développement
- Gain économique lié au changement d'environnement

Volonté de pérenniser le patrimoine applicatif et d'accroître sa durée de vie.








MIGRATION ET ACCROISSEMENT DE LA VALEUR DU PATRIMOINE APPLICATIF

Une migration se traduit par un accroissement de la valeur technique du patrimoine applicatif



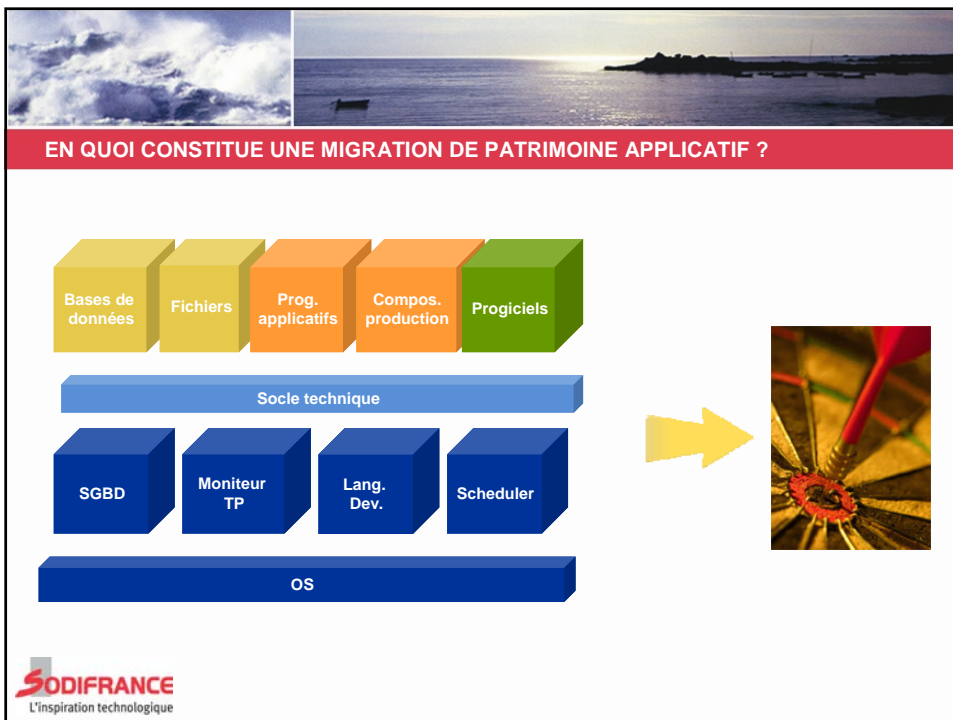
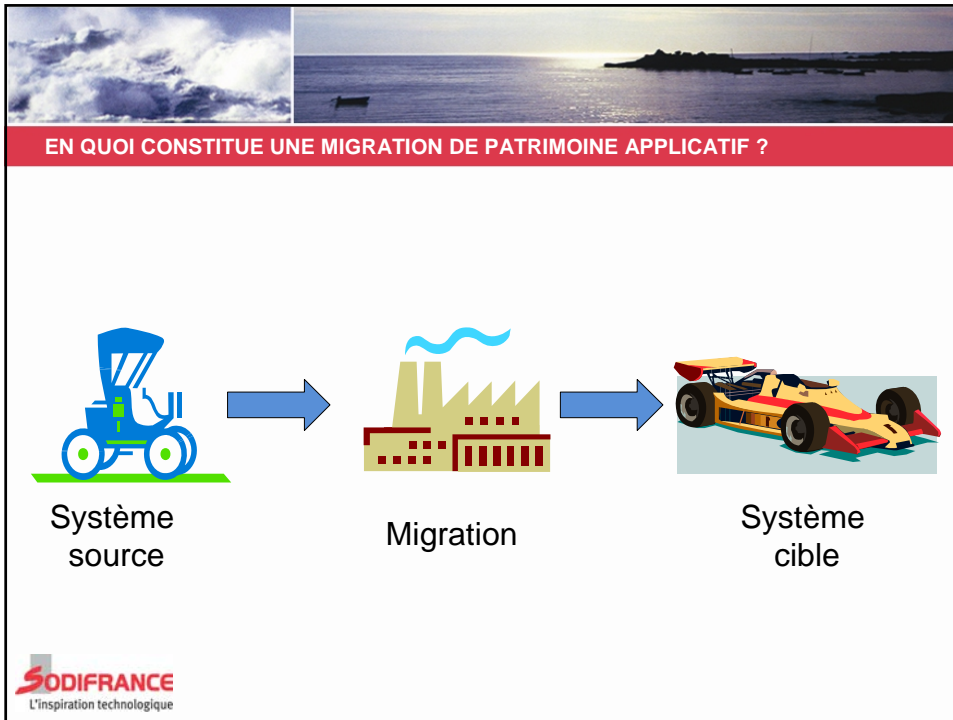
Une migration iso-fonctionnelle

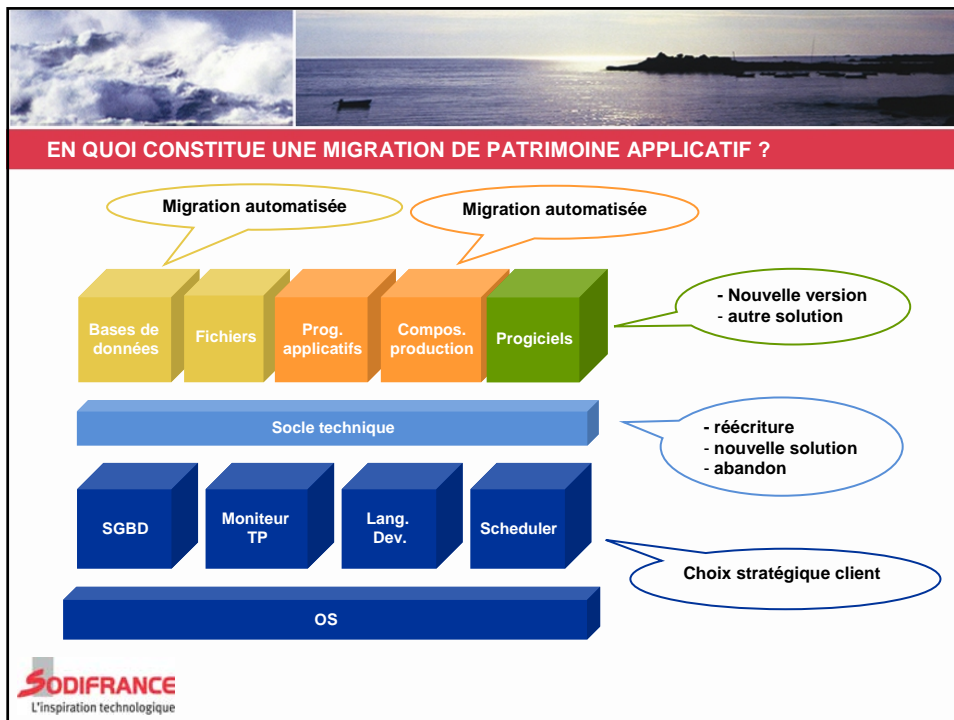
- permet une approche purement technique du projet
- permet une démarche de test basée « simplement » sur la validation de la non régression
- permet une réduction importante des risques, des coûts et des délais

L'accroissement de la valeur fonctionnelle du patrimoine applicatif peut se faire

- en parallèle de la migration => gestion des reports de maintenance
- post-migration => importance de l'iso-maintenabilité







MIGRATION INDUSTRIELLE


Industrialisation

- de l'analyse du patrimoine applicatif à migrer
- des conversions de composants (programmes, jcls, données)
- des tests de non régression
- des processus

Conversion automatisée des composants par application de règles

- Aucune modification manuelle
- La conversion est relançable à tout moment
- Les règles de conversion sont définies et validées en commun

SODIFRANCE
L'inspiration technologique



AVANTAGES DE LA MIGRATION INDUSTRIELLE

Économique


- Un coût de projet divisé par 3 à 5 par rapport à une refonte
- Un délai beaucoup plus court, donc une réactivité plus forte
- Une forte réduction des risques


Technique

- Une garantie de conservation
 - De l'ergonomie
 - Du fonctionnel


Humain

- Réduit le temps d'appropriation
 - par l'utilisateur final
 - par le développeur





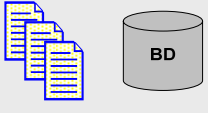
SODIFRANCE
L'inspiration technologique



L'APPROCHE TRADITIONNELLE DE LA MIGRATION

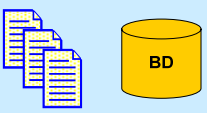
Transformation de code à code :

- besoin initial du client



Environnement source

➔



Environnement cible

SODIFRANCE
L'inspiration technologique



LES LIMITES DE CETTE APPROCHE


Manque de puissance de la technologie

- difficulté de spécifier des règles de migration complexes en particulier pour restructurer complètement le code
- problèmes si les architectures source et cible présentent des écarts importants
- **c'est le cas des migrations client-serveur vers architecture n-tiers**

Maintenabilité

- le patrimoine applicatif n'est toujours pas maîtrisé
- le cycle évolution/migration continue


On ne vient pas s'intégrer dans le processus de développement du client

NOTRE APPROCHE DE LA MIGRATION

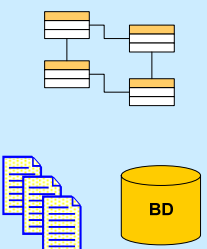
Fournir au client

- les composants migrés
- les outils nécessaires à leur maintenance (**modèles**)




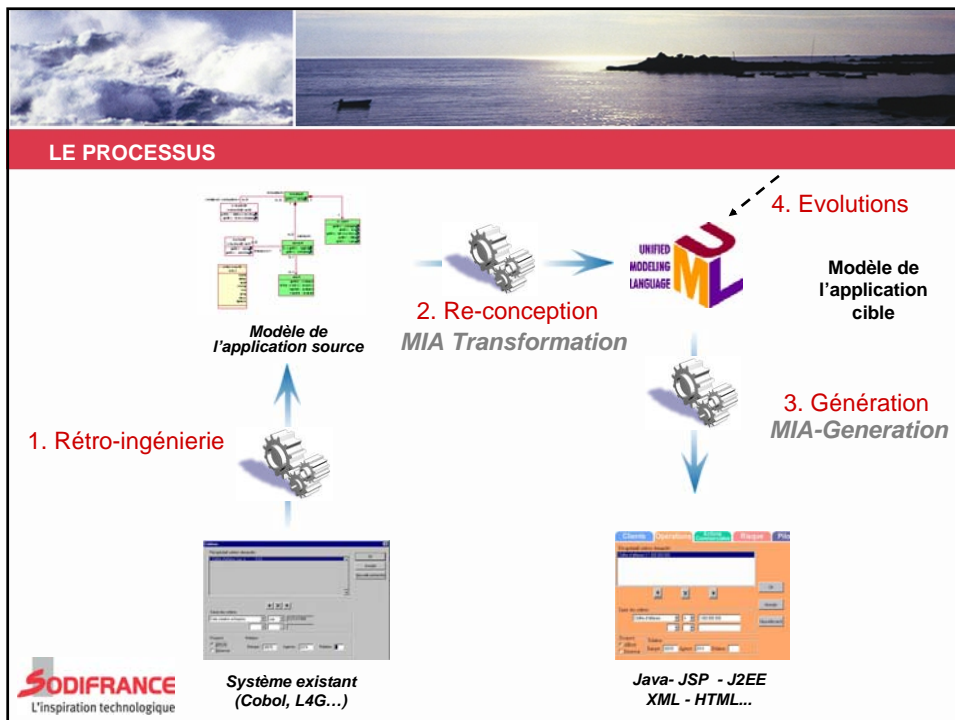
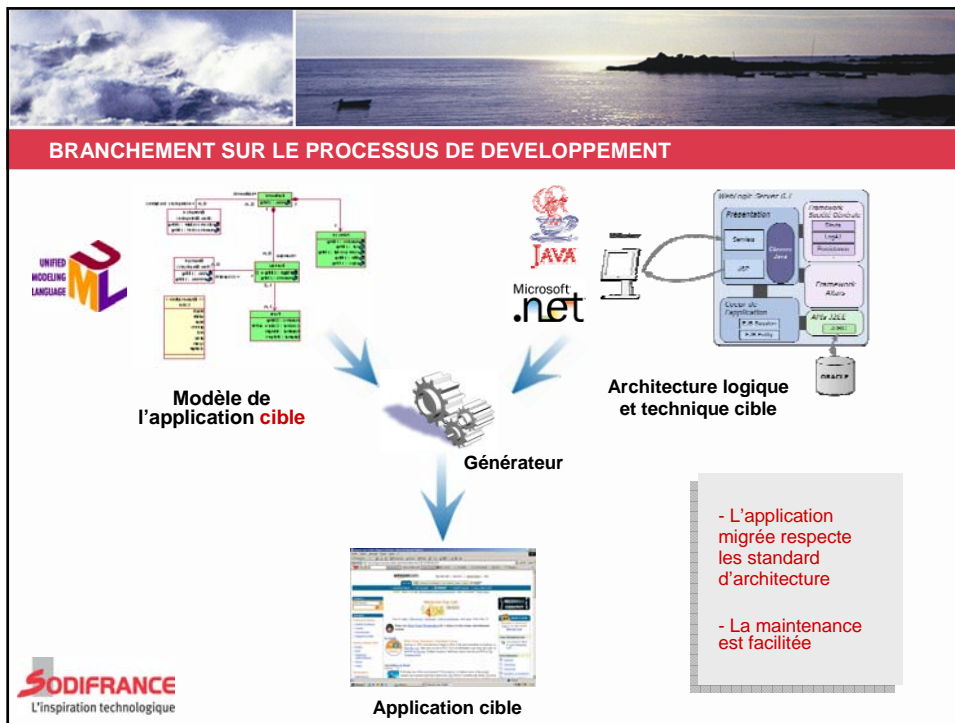
Environnement source


➔ Migration



Environnement cible



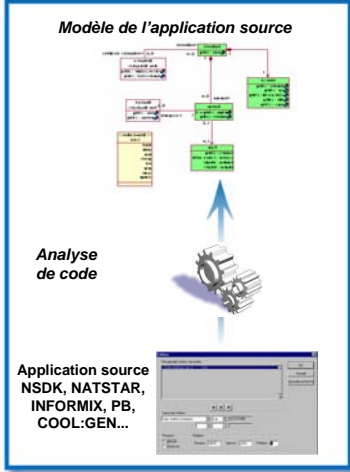




PHASE 1 : RETRO-INGENIERIE



- >> Analyse automatique du code à l'aide d'un parseur
- >> Re-modélisation des informations
- >> Instanciation d'un modèle de l'application source (100% des informations de départ)

Modèle de l'application source



Analyse de code

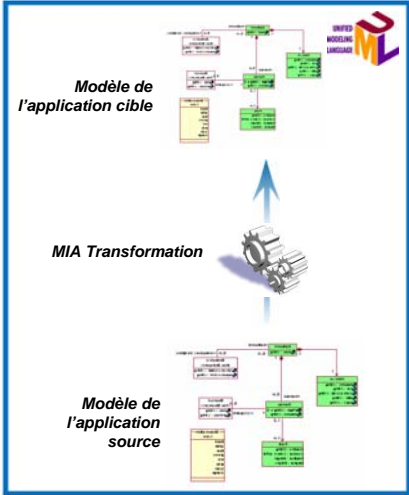
Application source
NSDK, NATSTAR,
INFORMIX, PB,
COOL:GEN...

PHASE 2 : TRANSFORMATION


- >> Processus automatisé de transformation de modèles grâce à MIA-Transformation
- >> Détection de composants particuliers (objets persistants, cinématique, modèles de vue...)
- >> Instanciation d'un modèle de l'application proche de la cible
- >> Prise en compte des normes de modélisation client


Modèle de l'application cible



MIA Transformation

Modèle de l'application source







PHASE 3 : GENERATION

- >> Processus automatisé de génération grâce à MIA-Generation
- >> Génération du code de l'application migrée
- >> Intégration dans l'architecture et le cycle de développement cible


Application cible (J2EE, .Net...)




MIA-Generation



Modèle de l'application cible

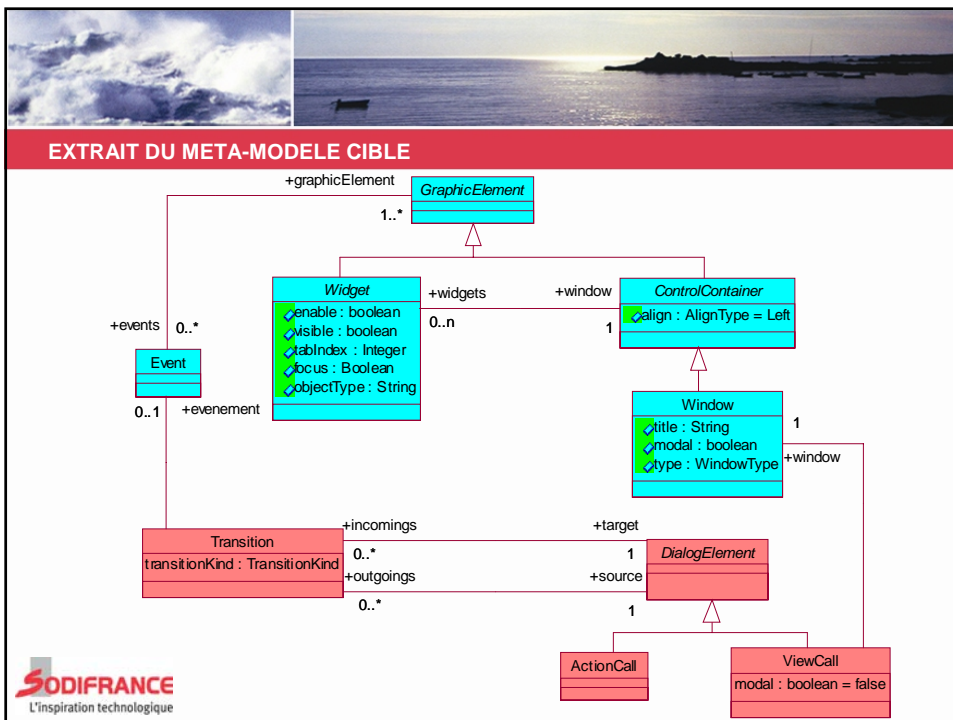
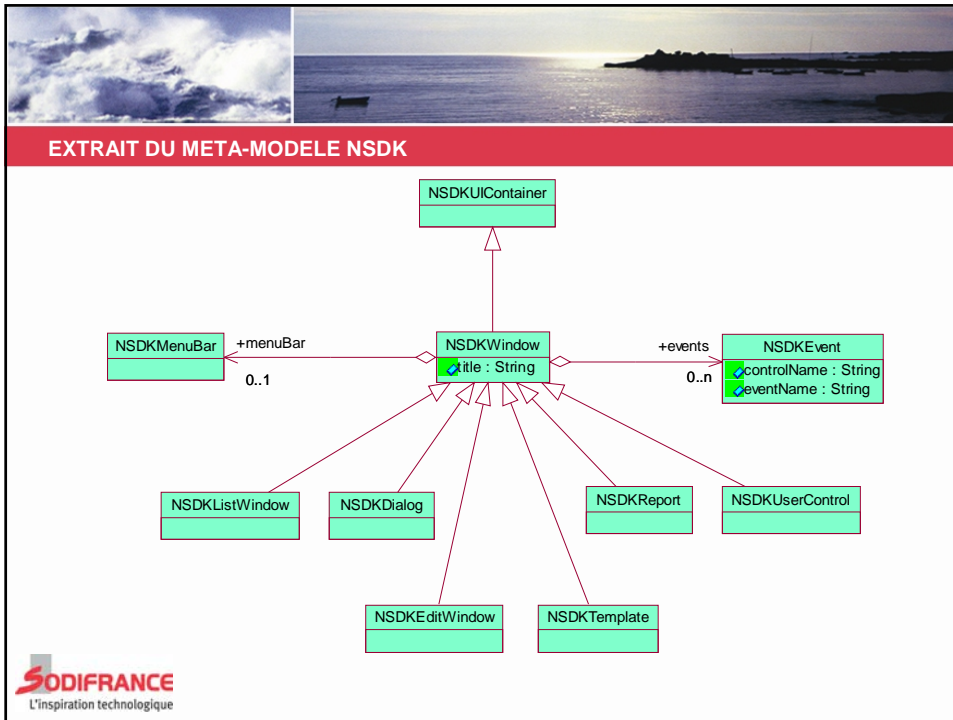





EXEMPLE DE MIGRATION INDUSTRIELLE

- >> Gestion de contrats d'assurance
- >> Architecture de départ **Client/Serveur NSDK** avec une base DB2
- >> Cible **Client léger J2EE**
- >> 160 écrans et 80 000 lignes de code
- >> Plusieurs points techniques délicats dont **interactivité élevée** et **multi fenêtrage**







EXEMPLE DE TRANSFORMATION

PASSAGE DU MODELE EVENEMENTIEL AU MVC

En client-serveur

- Tout événement déclenche un bout de code qui décide des actions à déclencher et de la fenêtre suivante à afficher
- La cinématique n'est pas explicite


En architecture Web

- Tout événement est retransmis à un contrôleur global qui définit les actions à réaliser et l'écran suivant à afficher

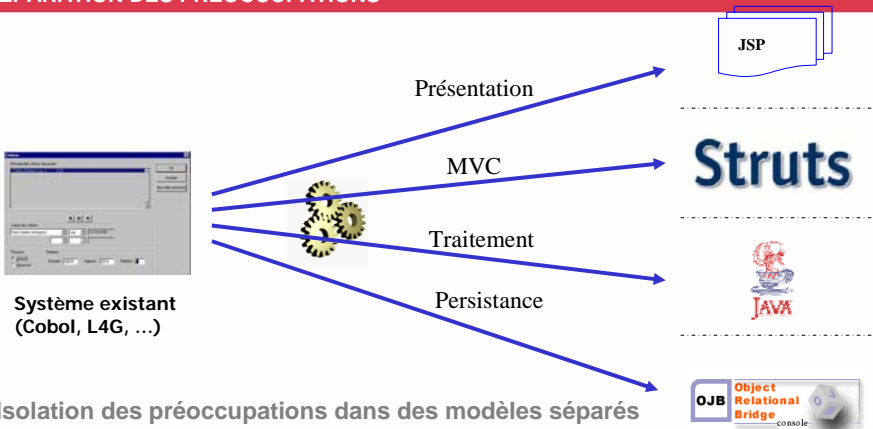
Rôle de la transformation

- Identifier et séparer dans le code ce qui a trait aux traitement et ce qui a trait à la cinématique
- Reconstruire le modèle de cinématique global à partir des fragments épars

SODIFRANCE
L'inspiration technologique



SEPARATION DES PREOCCUPATIONS



The diagram illustrates the decomposition of an existing system into MVC components. On the left, a screenshot of a 'Système existant (Cobol, L4G, ...)' is shown. Four arrows point from this system to a central gear icon representing the MVC pattern. From the gear, four arrows point to specific technologies: 'Présentation' points to 'JSP', 'MVC' points to 'Struts', 'Traitement' points to 'JAVA', and 'Persistance' points to 'OJB Object Relational Bridge console'.

Système existant (Cobol, L4G, ...)

Présentation → JSP

MVC → Struts

Traitement → JAVA

Persistance → OJB Object Relational Bridge console

**Isolation des préoccupations dans des modèles séparés
Pour les traiter isolément
Et faciliter leur recombinaison dans l'architecture cible**

SODIFRANCE
L'inspiration technologique

APPLICATION D'ORIGINE

L'application NSDK d'origine permet notamment de gérer une base de clients

Un menu contextuel riche et interactif

Une interactivité évoluée : Lorsque l'on renseigne le code APE...

Le champ apparaît avec le libellé correspondant

The screenshot shows a window titled 'Informations client' with the following fields: Numéro (6481), Titre (Compagnie), Nom (SODIFRANCE), Auxil nom, Nationalité (FRANCE METROPOLITAINE), Bâtiment, Auxil voie (ALABAMA), Voie (4), Rue (CHATEAU DE L'ERAUDIÈRE), Code postal (44000), Commune (NANTES), Etat province, Pays (FRANCE METROPOLITAINE), Téléphone (0240185212), Télécopie, Indic. télex, Tél. télex, Code APE, Taille (entre 500 et 1000 salariés), SIREN, and NIC. A 'Commentaires' field is at the bottom. Buttons for 'OK', 'Quitter', and 'Aide' are visible.

SODIFRANCE
L'inspiration technologique

APPLICATION MIGREE

L'application finale respecte la charte graphique de l'entreprise

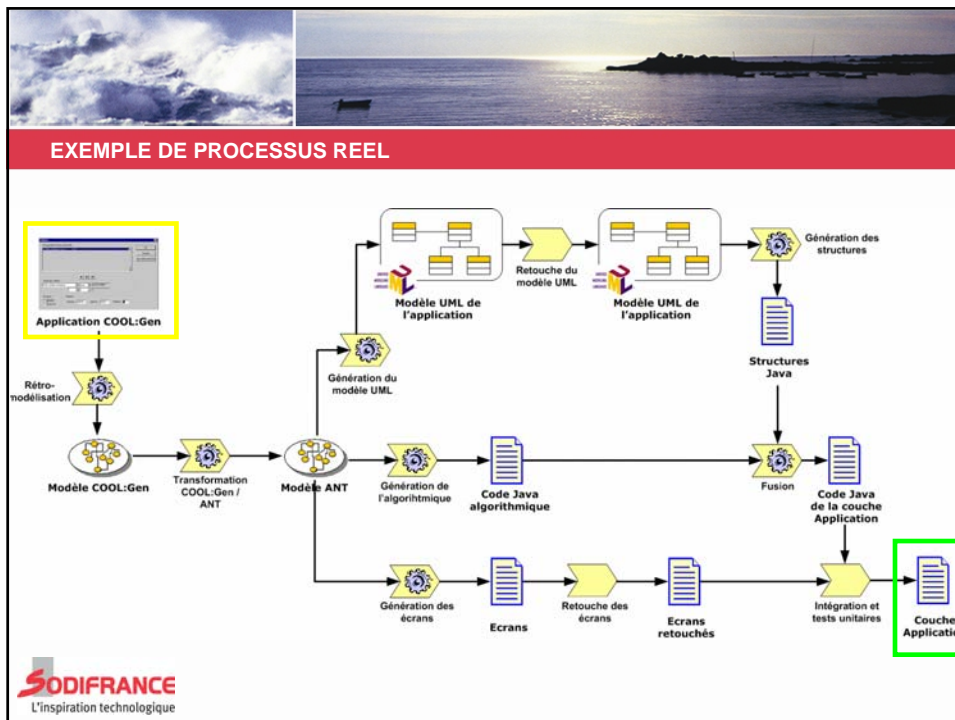
Les fonctionnalités de l'application originale sont reconduites à l'identique

Le menu offre le même niveau d'interactivité

La saisie du code APE provoque l'apparition du champ caché

Les données sont renseignées sans rechargement de la page


The screenshot shows a modernized version of the 'Informations client' form. It features a dark blue sidebar menu with options: Outils, Apparteur, Client, Cie, Proposition, Contrat, Cover, Recherche, and Aide. The main form fields are identical to the original application. The 'Code APE' field is highlighted with a callout indicating that its entry triggers the appearance of the 'entre 500 et 1000 salariés' field. The 'SIREN' and 'NIC' fields are also highlighted, indicating they are populated without a page refresh. Buttons for 'OK', 'Quitter', and 'Aide' are present at the bottom.



SOMMAIRE

- Introduction
- Des outils MDA
- Processus de développement
- La migration
- **Au-delà du génie logiciel ...**
- Conclusion

SODIFRANCE
L'inspiration technologique



BESOINS INDUSTRIELS

Le modèle unique n'existe pas


- Ingénierie système
- Expression des besoins, gestion des exigences
- Modèles d'analyse, de conception, d'implémentation
- Simulation
- Planification des développements


Manipulation de nombreux outils

- Eventuellement basés sur des méta-modèles différents

Problème : l'interopérabilité

- Transfert de l'information (en évitant les re-saisies manuelles coûteuses et sources d'erreurs)

 **SODIFRANCE**
L'inspiration technologique



LES PONTS SEMANTIQUES

Objectif


- Permettre l'échange d'informations entre deux outils
- Eventuellement basés sur des sémantiques différentes

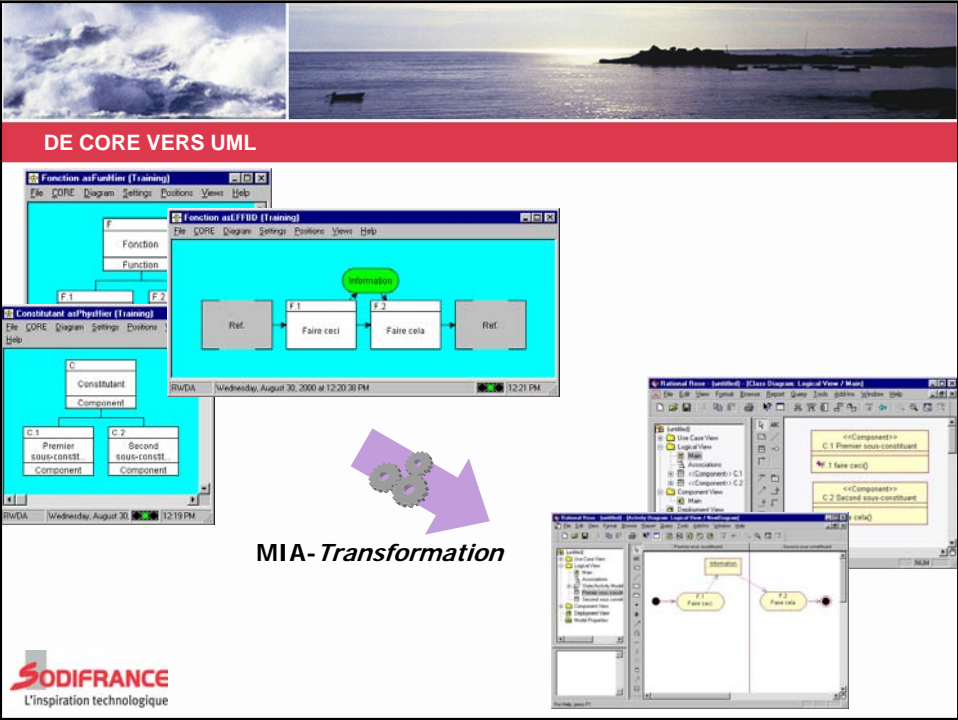
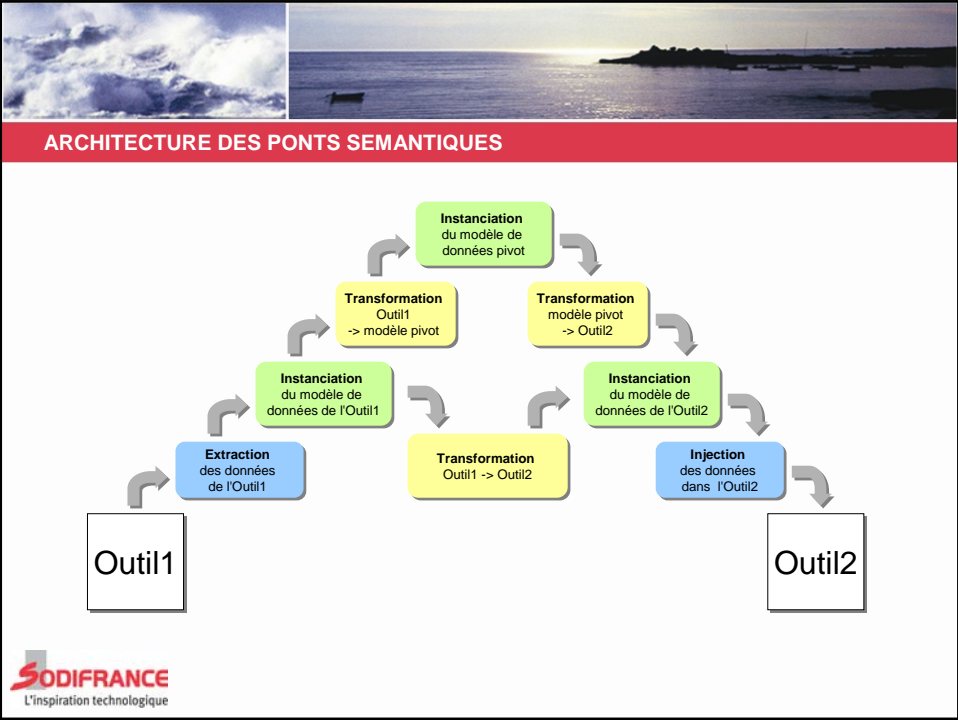
Technique utilisée


- La transformation de modèles
- MIA-Transformation

Contrainte

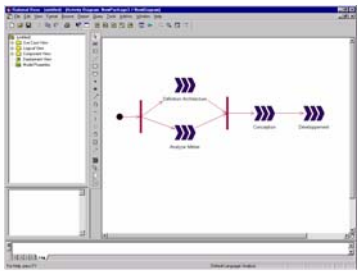
- L'ouverture
- => Architecture à base de composants


 **SODIFRANCE**
L'inspiration technologique



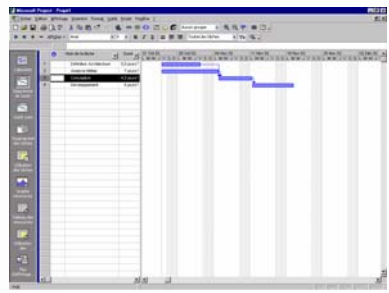


SPEM VERS MS-PROJECT





MIA-Transformation



SODIFRANCE
L'inspiration technologique



PONT STEP / MDA

STEP

- Norme ISO de description de schémas de données
- EXPRESS : langage de modélisation

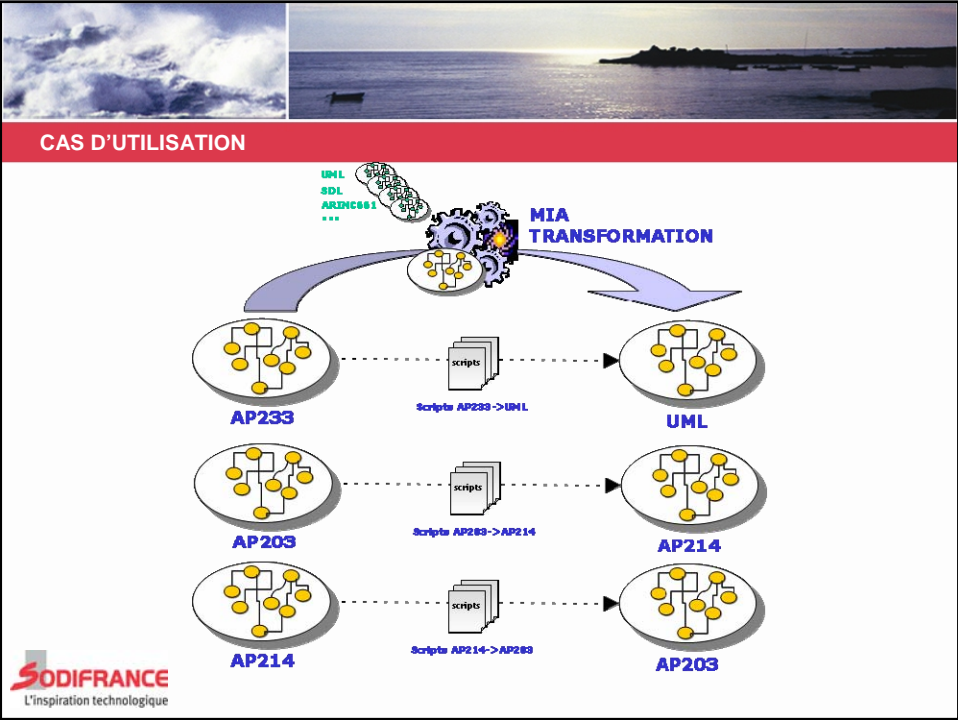
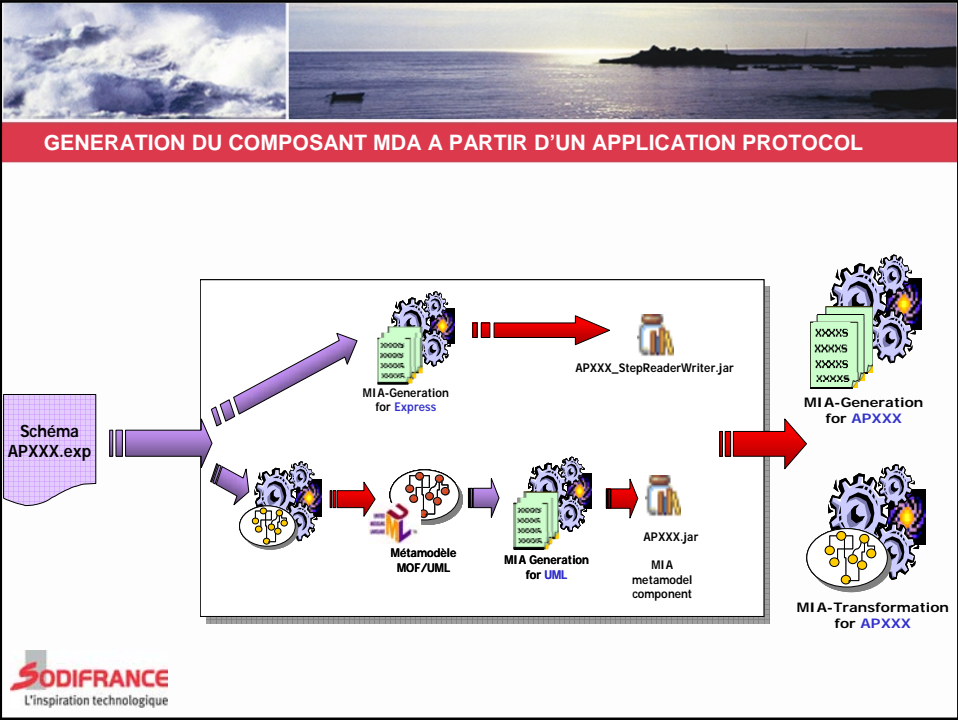
Spécification d'Application Protocol

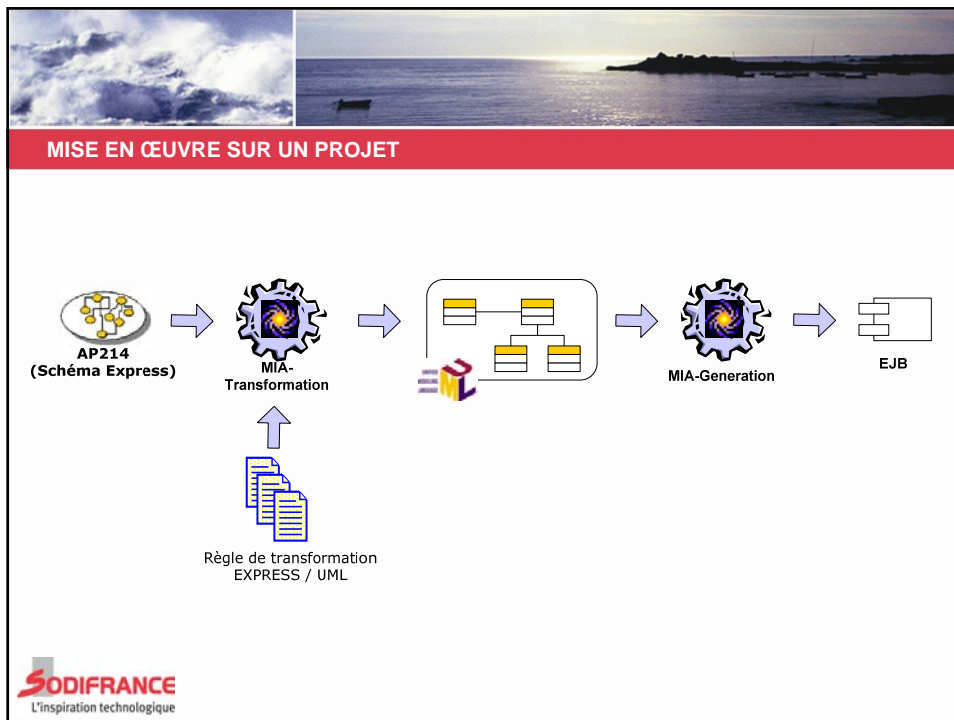
- AP203 : conception 3D
- AP 209 : structures métalliques composites
- AP 214 : processus de conception des véhicules à moteur

Problème

- Comment intégrer ces données dans un processus MDA ?

SODIFRANCE
L'inspiration technologique





SOMMAIRE

- Introduction
- Des outils MDA
- Processus de développement
- La migration
- Au-delà du génie logiciel ...
- **Conclusion**

SODIFRANCE
L'inspiration technologique



UTILISATION ACTUELLE DU MDA DANS L'INDUSTRIE

Percée significative

- De la modélisation UML
- De la génération de code
- La transformation de modèles reste le plus souvent cantonnée à la R&D

Des travaux de méta-modélisation

- Profils UML
- Schémas XML

Les DSLs sont pour la plupart des langages techniques

- Présentation, cinématique, persistance, SOA, etc.
- On reste dans une modélisation technique, proche du code



L'inspiration technologique



LIMITES ACTUELLES

Manque de maîtrise des concepts

- Des méta-modèles redondants et pas toujours bien formés
- Des générations de code complexe qui auraient pu être simplifiées en utilisant la transformation de modèle

Difficultés à trouver le bon niveau de modélisation

- Extreme programming ou extreme modeling ?
- Jusqu'où modéliser ?
 - Faut-il modéliser les traitements ?
 - Faut-il modéliser jusqu'au positionnement des widgets dans un écran ?
- Concilier pérennité et réactivité



L'inspiration technologique



LIMITES ACTUELLES

Quelques problèmes en suspens

- UML 2.0 ou DSLs ?
 - UML 2.0 : l'auberge espagnole de la modélisation (chacun y trouve son bonheur, et si ce n'est pas le cas, il y a encore les profils)
 - DSLs : des langages réduits et dédiés (mais risque de réinventer la poudre)
- Outillage
 - Gestion de configuration
 - Référentiel
- Gestion des évolutions de modèles
 - Comment répercuter les modifications vers les modèles de plus haut niveau et de plus bas niveau en tenant compte des évolutions de ces modèles ?



SODIFRANCE
L'inspiration technologique



AVENIR DU MDA ???

Nouveau paradigme

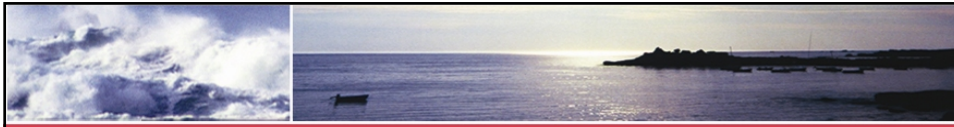
- Ambition initiale
 - Tous les domaines de l'entreprise sont modélisés
 - On utilise des modèles de différents niveaux d'abstraction
 - On passe d'un modèle à l'autre par transformation
- On attend toujours ...

Technologie RAD ++

- Rapid Application Development
- On modélise (à très bas niveau) et on génère du code
- L'ouverture, les standards (et la complexité) en plus



SODIFRANCE
L'inspiration technologique



Merci de votre attention

Pour en savoir plus :
www.sodifrance.fr
ebreton@sodifrance.fr

