

*Soutenance de thèse*

# Approche de métamodélisation pour la simulation et la vérification de modèle

## Application à l'ingénierie des procédés

Benoît Combemale

*Institut de Recherche en Informatique de Toulouse (CNRS UMR 5505)*

Université de Toulouse, France.

<http://combemale.perso.enseeiht.fr>

Encadrant : Xavier Crégut

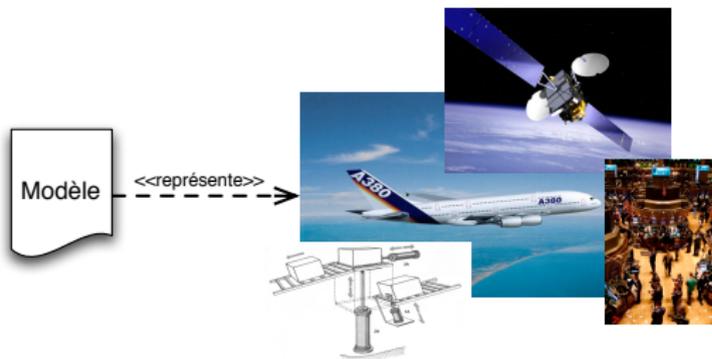
Directeurs : Patrick Sallé et Bernard Coulette



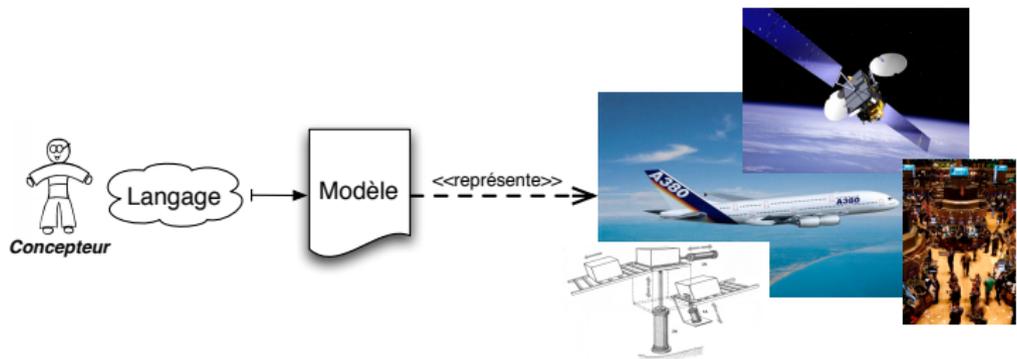
# Contexte scientifique



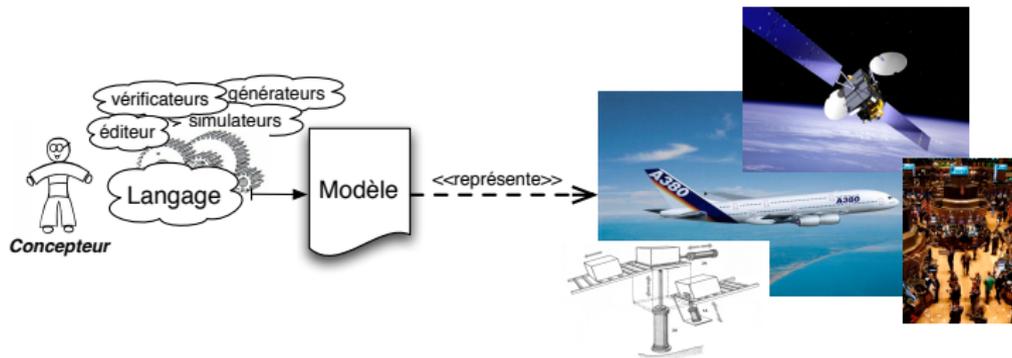
# Contexte scientifique



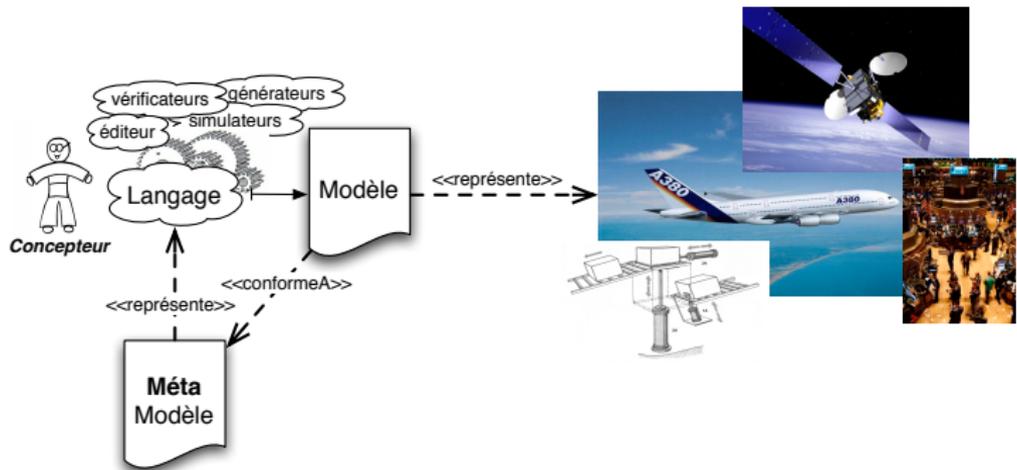
# Contexte scientifique



# Contexte scientifique

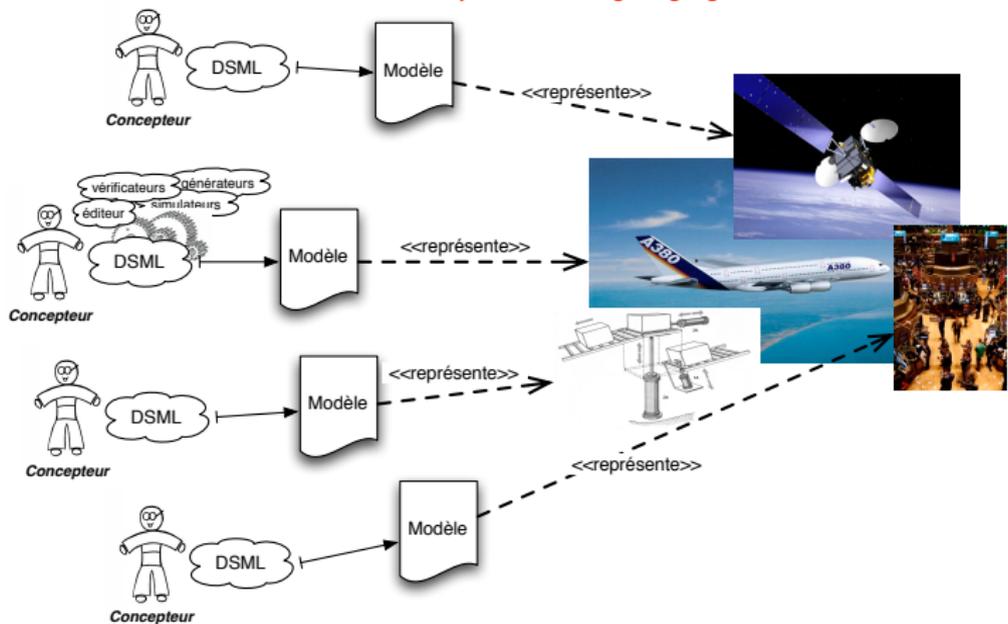


# Contexte scientifique

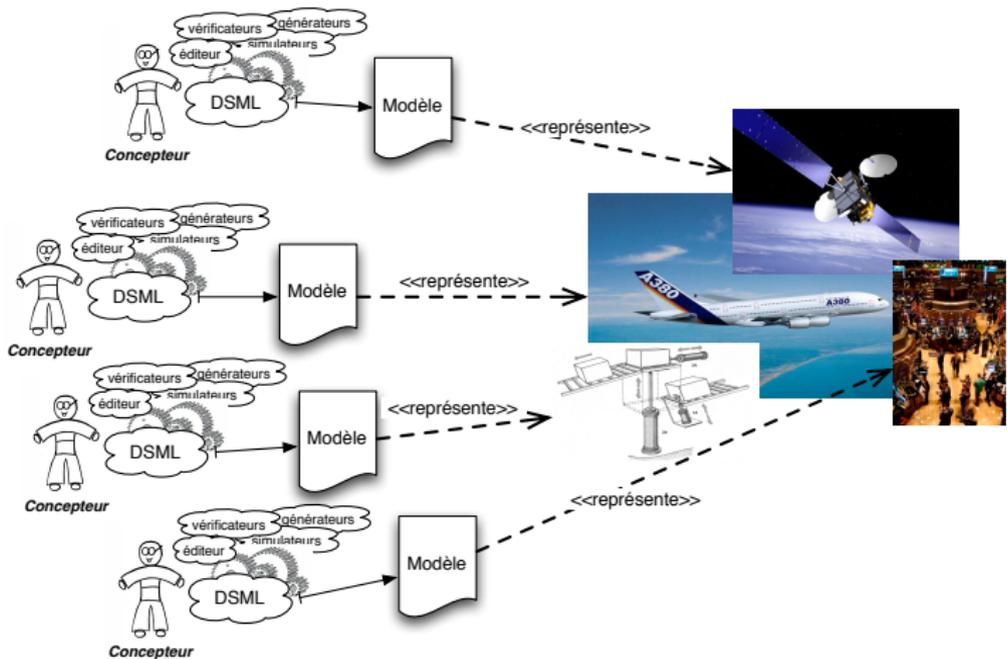


# Contexte scientifique

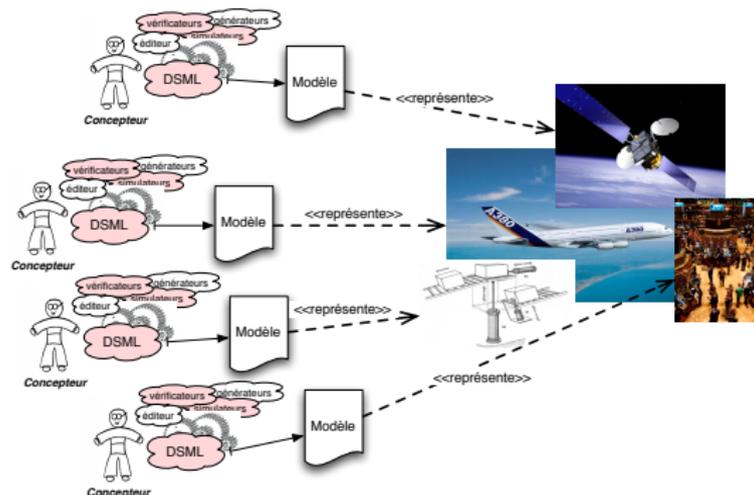
**DSML = Domain Specific Modeling Language**



# Contexte scientifique



## Contexte scientifique



- **Objectif de la thèse** : Démarche outillée de métamodélisation pour la simulation et la vérification de modèle.
- **Problématique** : Comment exprimer la sémantique comportementale d'un DSML et offrir les outils de *méta-niveau* pour faciliter la définition de simulateurs et de vérificateurs de modèle.



**TOPCASED**  
The Open Source toolkit for critical systems

# Contexte applicatif – *industriel*

Le projet TOPCASED

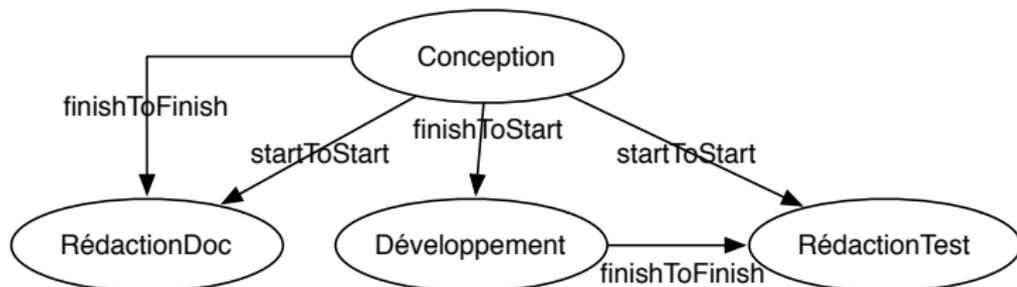
- Atelier *open source* couvrant les différentes phases de développement et intégrant les contraintes de certification
  - Différents langages de modélisation (UML, AADL, SysML, SAM...)
  - Éditeurs graphiques
  - Outils de traçabilité, de génération, de transformation et de V & V
- ⇒ **Application** : démarche et outils de *méta-niveau* pour la définition de simulateurs et de vérificateurs de modèle pour les langages actuels et futurs de l'atelier



Site Internet : <http://www.topcased.org> & Serveur de développement : <http://gforge.enseeiht.fr/>

# Contexte applicatif – *académique*

## L'ingénierie des procédés de développement



- La maîtrise d'un procédé s'apparente à celle du logiciel [Osterweil87, Estublier05]
- Différents langages de modélisation de procédés (p. ex. SPEM de l'OMG)
- Et l'exécution ?

⇒ **Application** : définir une extension du standard SPEM permettant la simulation et la vérification de procédés.

# Plan

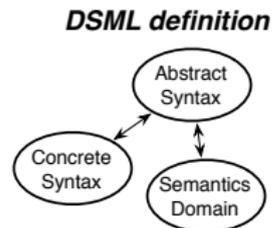
- 1 Contexte
- 2 Vers une opérationnalisation des modèles dans l'IDM
  - La métamodélisation
  - Définition de la sémantique d'exécution d'un DSML
- 3 Contributions pour la simulation et la vérification de modèle
  - Démarche outillée pour la définition d'un DSML exécutable
  - Cadre générique pour la définition de simulateurs de modèle
  - Approche par traduction pour la vérification de modèle
  - Cadre formel pour la métamodélisation
- 4 Conclusion & perspectives

# Plan

- 1 Contexte
- 2 Vers une opérationnalisation des modèles dans l'IDM
  - La métamodélisation
  - Définition de la sémantique d'exécution d'un DSML
- 3 Contributions pour la simulation et la vérification de modèle
  - Démarche outillée pour la définition d'un DSML exécutable
  - Cadre générique pour la définition de simulateurs de modèle
  - Approche par traduction pour la vérification de modèle
  - Cadre formel pour la métamodélisation
- 4 Conclusion & perspectives

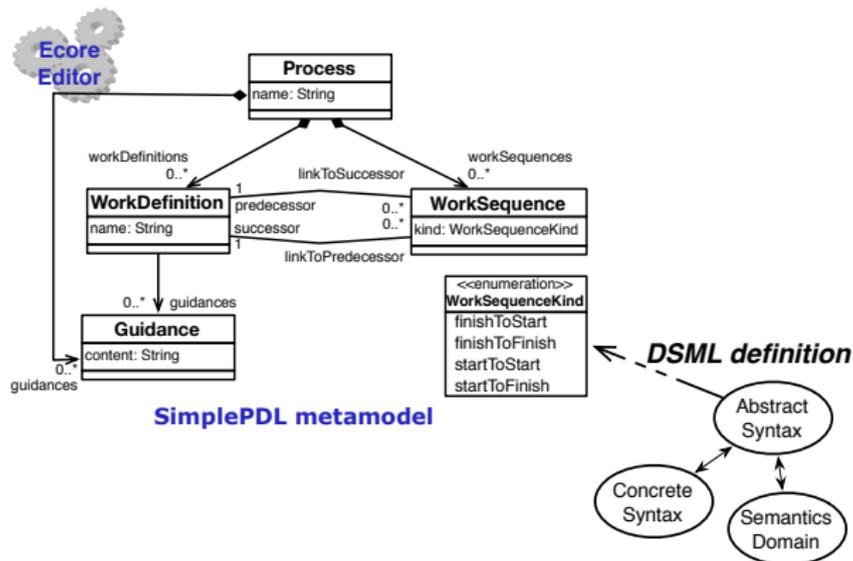
# La métamodélisation

Définition d'un DSML, illustrée avec l'atelier TOPCASED



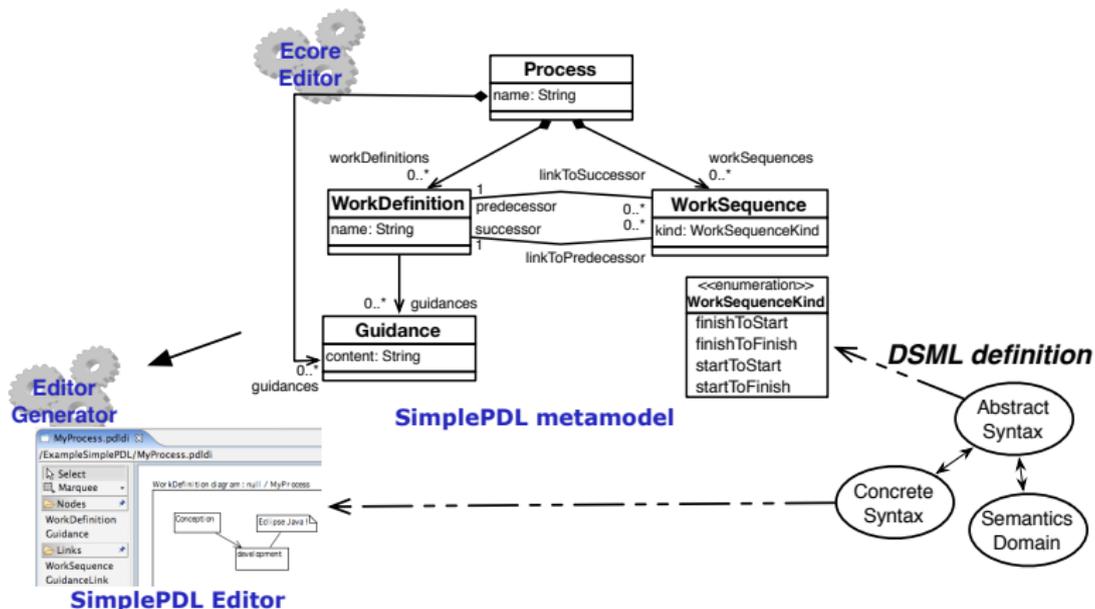
# La métamodélisation

Définition d'un DSML, illustrée avec l'atelier TOPCASED



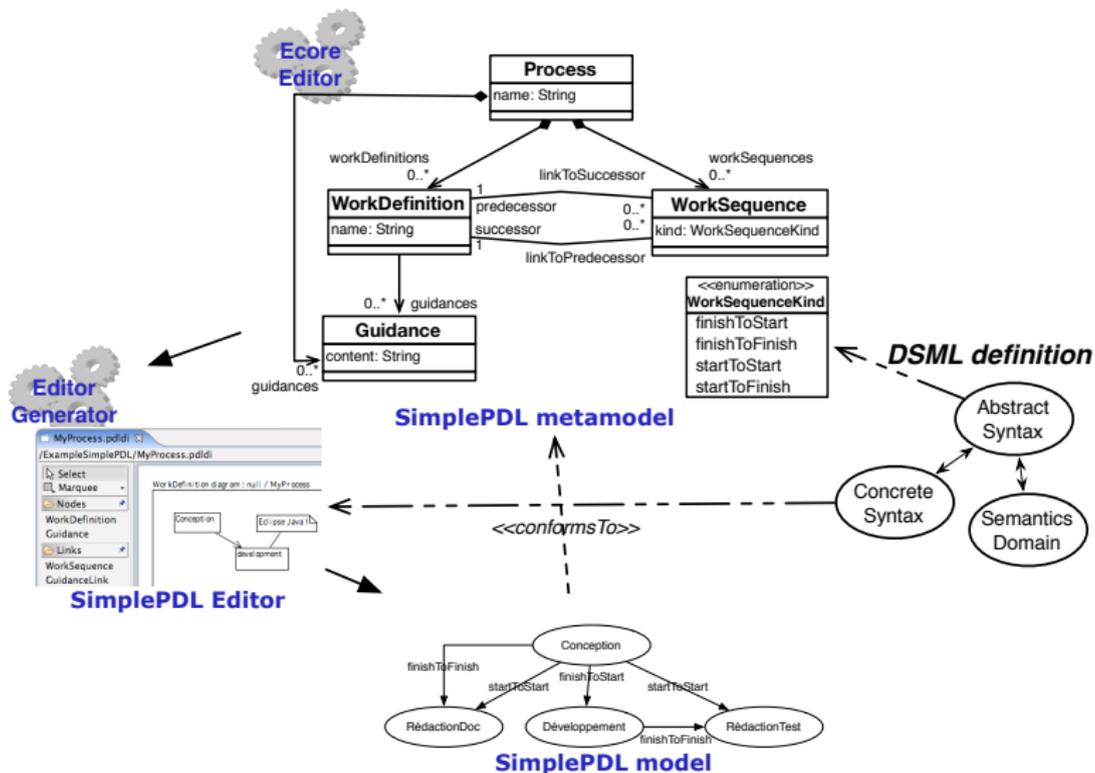
# La métamodélisation

Définition d'un DSML, illustrée avec l'atelier TOPCASED



# La métamodélisation

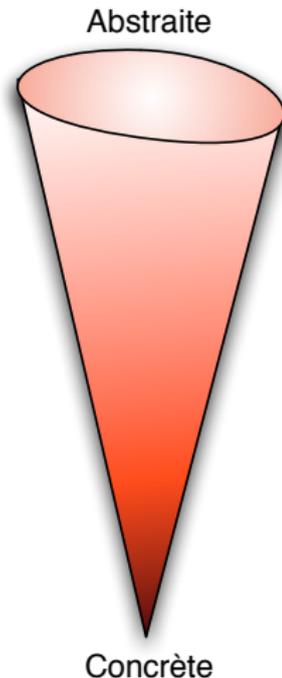
Définition d'un DSML, illustrée avec l'atelier TOPCASED



# Taxonomie des sémantiques dans l'IDM

```
int x;  
void decr () {  
    if ( x>0 )  
        x = x - 1;  
}
```

<b>System</b>
x : Int
decr()



# Taxonomie des sémantiques dans l'IDM

```
int x;
void decr () {
    if ( x>0 )
        x = x - 1;
}
```

<b>System</b>
x : Int
decr()

## Sémantique axiomatique

```
context System::decr() post :
    self.x = if ( self.x@pre > 0 )
                then self.x@pre - 1
                else self.x@pre
    endif
```



# Taxonomie des sémantiques dans l'IDM

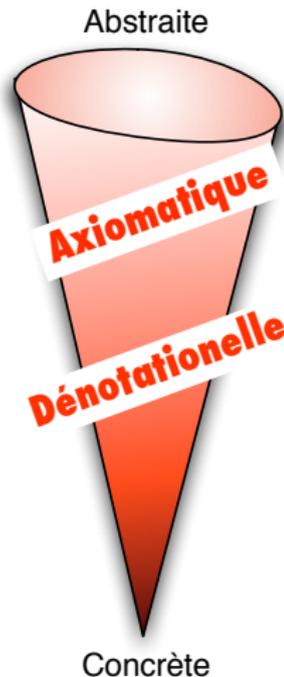
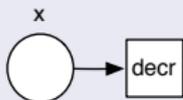
```
int x;
void decr () {
    if ( x>0 )
        x = x - 1;
}
```

<b>System</b>
x : Int
decr()

## Sémantique axiomatique

```
context System::decr() post :
    self.x = if ( self.x@pre > 0 )
                then self.x@pre - 1
                else self.x@pre
endif
```

## Sémantique dénotationnelle (i.e. par traduction)



# Taxonomie des sémantiques dans l'IDM

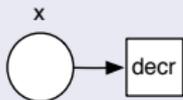
```
int x;
void decr () {
    if ( x>0 )
        x = x - 1;
}
```

<b>System</b>
x : Int
decr()

## Sémantique axiomatique

```
context System::decr() post :
    self.x = if ( self.x@pre > 0 )
                then self.x@pre - 1
                else self.x@pre
    endif
```

## Sémantique dénotationnelle (i.e. par traduction)

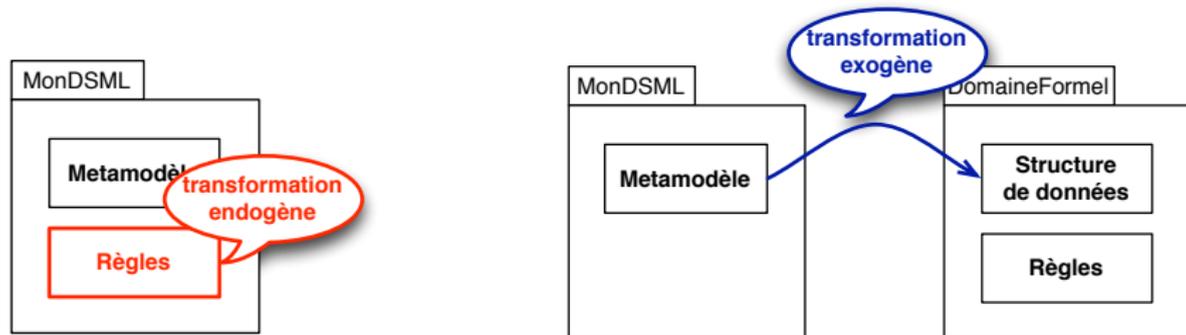


## Sémantique opérationnelle (i.e. langage d'action)

```
operation decr () is do
    if x > 0 then x = x - 1 end
```



# Sémantique opérationnelle et sémantique par traduction



## Sémantique opérationnelle

- + expression naturelle de la sémantique,
- nécessite de définir les outils (p. ex. de V&V).

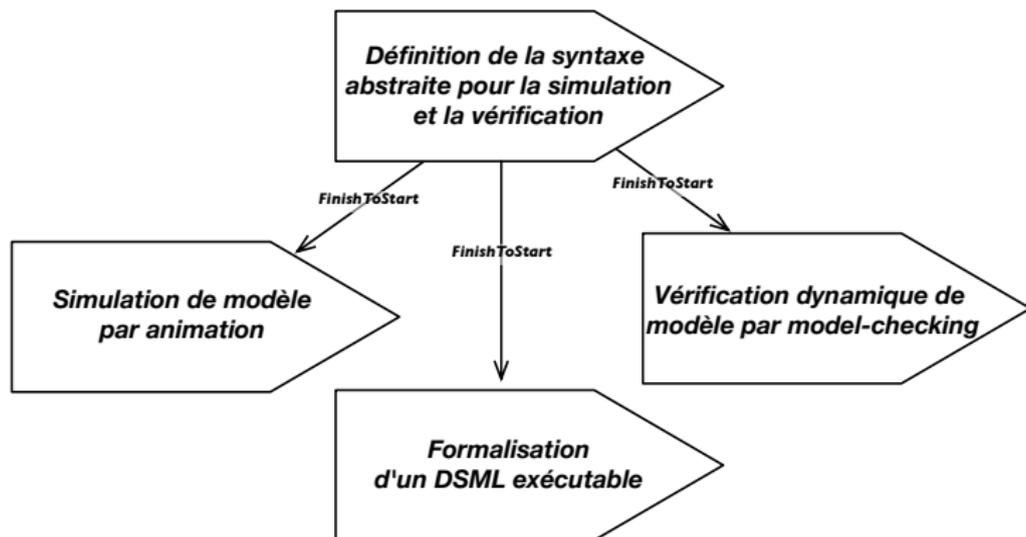
## Sémantique par traduction

- expression d'équivalences sémantiques,
- + permet de réutiliser les outils du domaine cible.

# Plan

- 1 Contexte
- 2 Vers une opérationnalisation des modèles dans l'IDM
  - La métamodélisation
  - Définition de la sémantique d'exécution d'un DSML
- 3 Contributions pour la simulation et la vérification de modèle
  - Démarche outillée pour la définition d'un DSML exécutable
  - Cadre générique pour la définition de simulateurs de modèle
  - Approche par traduction pour la vérification de modèle
  - Cadre formel pour la métamodélisation
- 4 Conclusion & perspectives

# Approche de métamodélisation

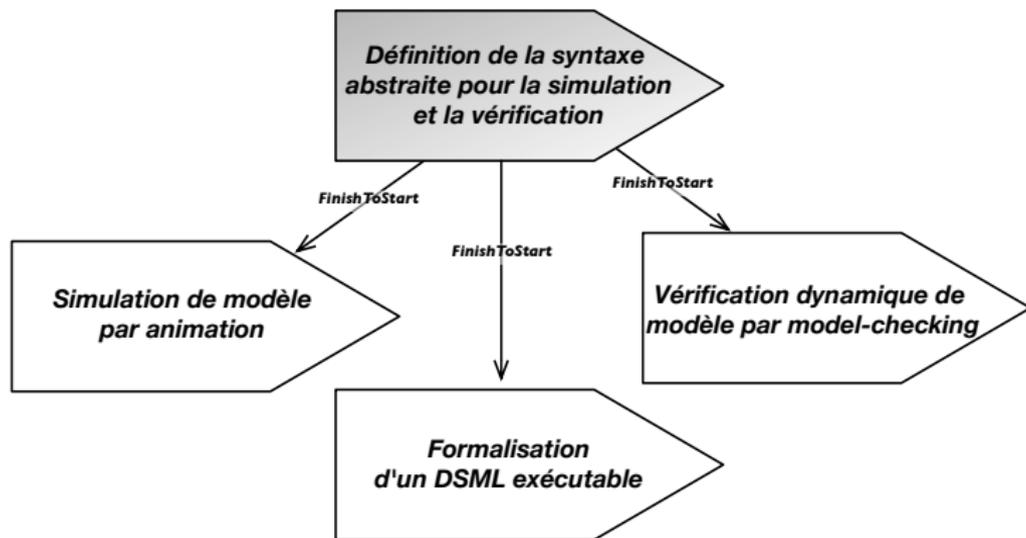


## ■ Objectifs :

*Démarche outillée de métamodélisation pour la définition de DSML « exécutable » et des outils de validation et vérification.*

- maîtriser l'activité de métamodélisation,
- diminuer les coûts,
- offrir des outils génériques ou des approches génératives.

# Démarche outillée pour la définition d'un DSML exécutable



## ■ Problématiques :

- Exprimer et structurer les différentes préoccupations de la syntaxe abstraite d'un DSML exécutable.
  - Expliciter la sémantique comportementale de référence.
- ⇒ Illustré par la définition d'xSPeM, une extension exécutable de SPeM2.0

# *Domain Definition MetaModel (DDMM)*

## Définition (DDMM)

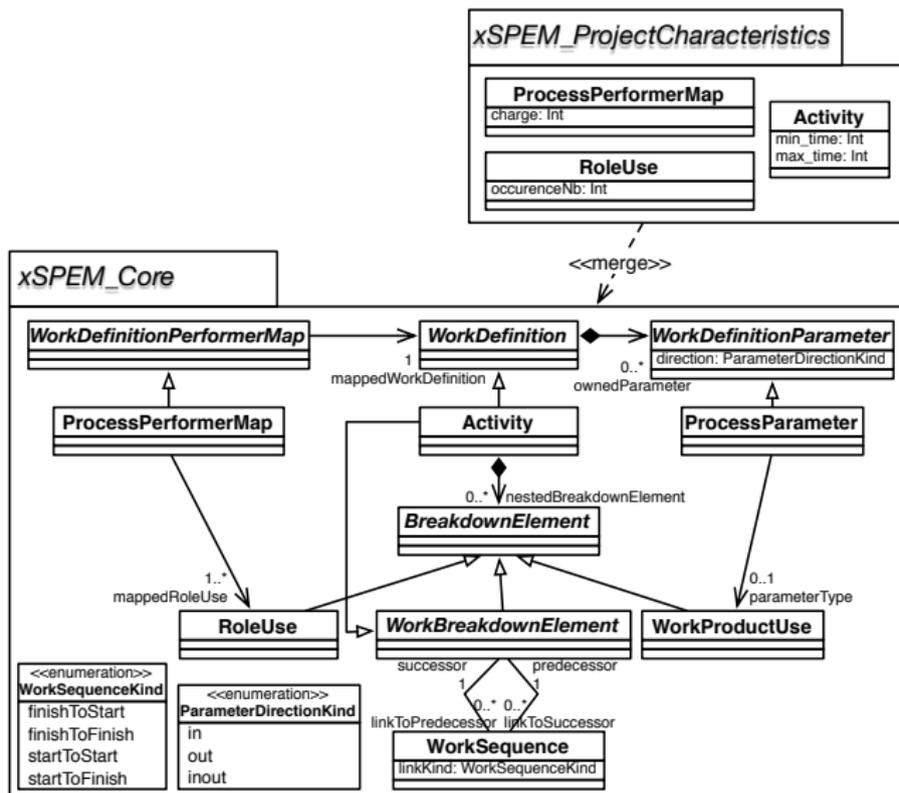
Décrit les informations structurelles du système

## Objectif

Retranscrire la connaissance des experts sur un domaine particulier.

- capture les concepts et les contraintes métiers,
- décrit la structure de données des modèles.

# Domain Definition MetaModel (DDMM) de xSPEM



# States Definition MetaModel (SDMM)

## Définition (SDMM)

Métamodèle complémentaire au DDMM qui :

- capture les informations dynamiques
  - caractérise l'ensemble des états observables possibles du modèle
- ⇒ **Difficulté** : définir le « bon » niveau d'abstraction des informations dynamiques.

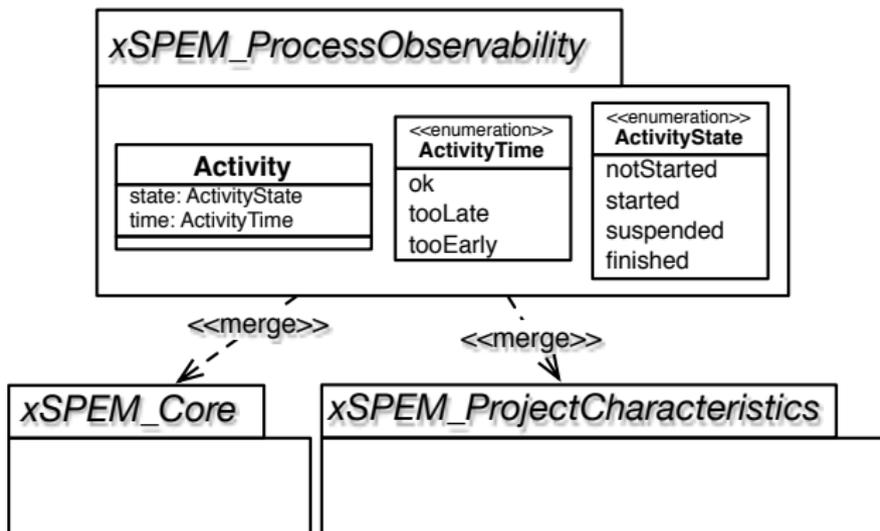
## Approche

Prendre en compte les questions de l'expert sur l'exécution pour définir les états et événements observables (souhaités).

- ⇒ spécifier le comportement en utilisant un langage d'expression de contraintes temporelles.
- Promotion de TOCL (Gogolla & al., 2002) au niveau de Ecore,
  - Définition du modèle de propriétés pour chaque DSML.

# States Definition MetaModel (SDMM) de xSPeM

- "toute activité non optionnelle doit démarrer",
- "chaque activité non optionnelle doit être réalisée dans l'intervalle de temps  $t_{min}$  et  $t_{max}$ ".



# Events Definition MetaModel (EDMM)

## Définition

Métamodèle décrivant les événements (et leurs paramètres) qui font évoluer l'exécution du système.

## Comportement de xSPEM (*Activity*)

Soit  $a$  une activité,

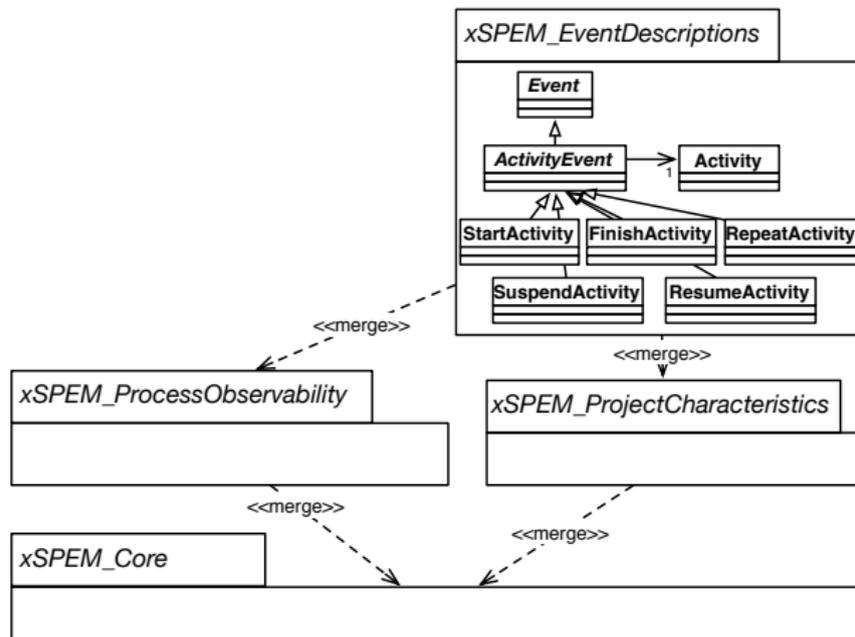
$$\forall ws \in a.predecessor,$$

$$\left( ws.linkKind = startToStart \ \&\& \ ws.linkToPredecessor.state = started \right)$$

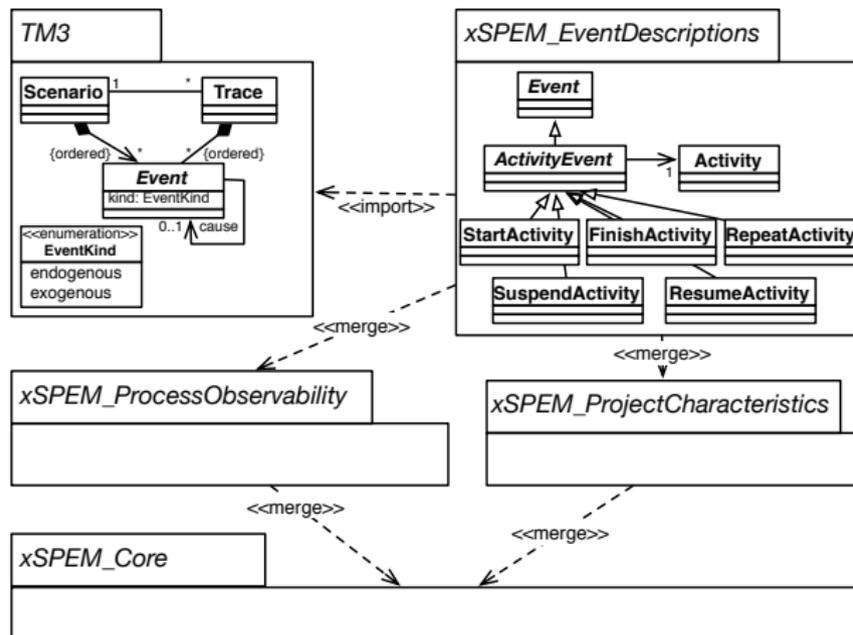
$$\| \left( ws.linkKind = finishToStart \ \&\& \ ws.linkToPredecessor.state = finished \right)$$

$$notStarted, ok, clock \xrightarrow{StartActivity} started, ok, 0$$

# Events Definition MetaModel (EDMM) de xSPEM



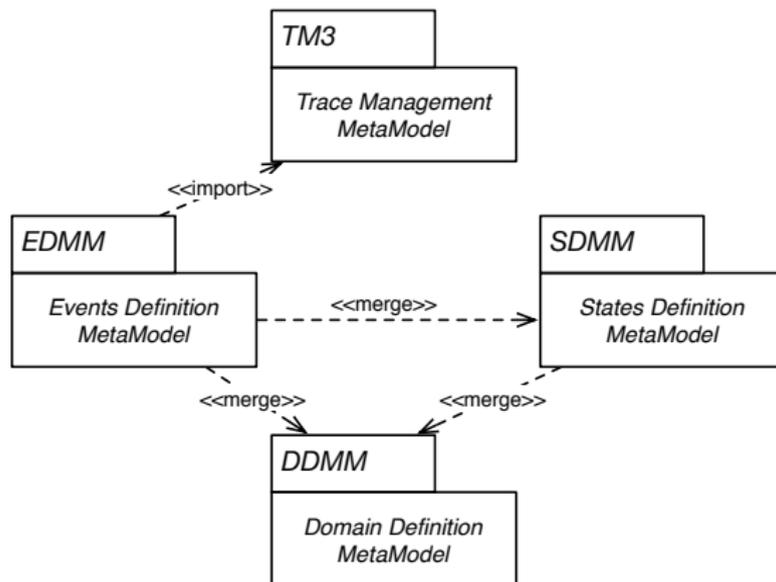
# Trace Management MetaModel (TM3)



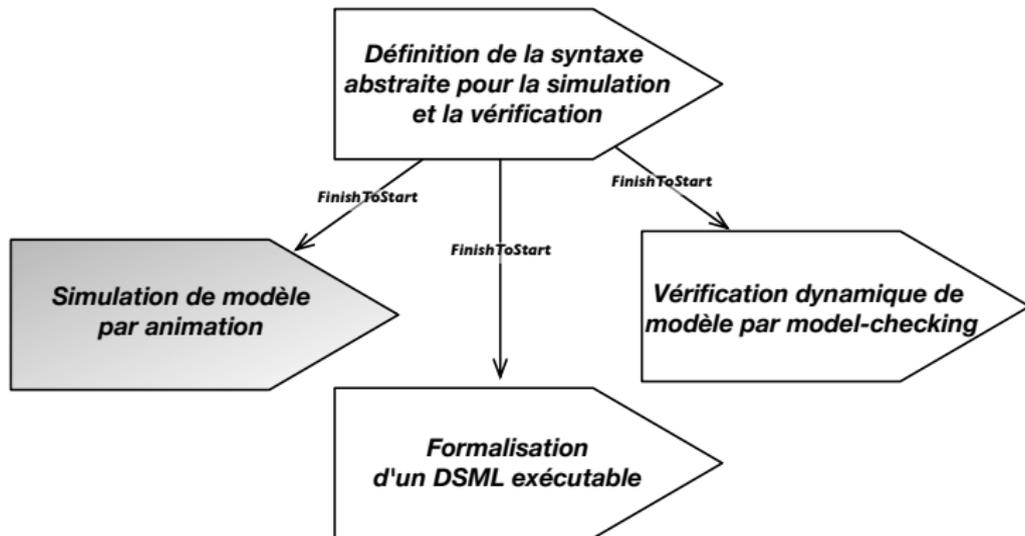
# Architecture d'un DSML exécutable

## Définition d'un langage de modélisation « exécutable »

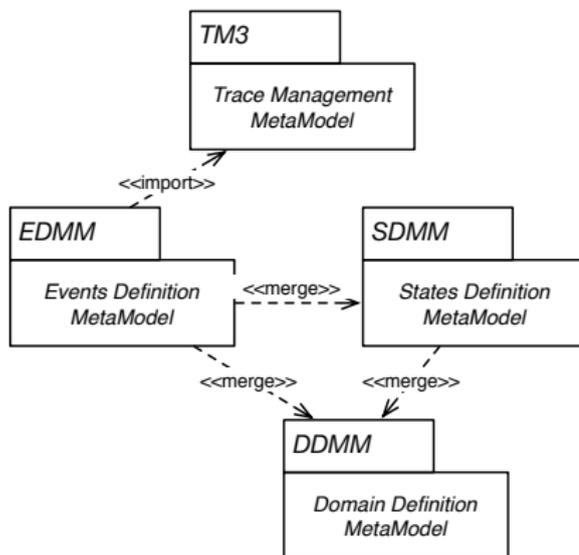
$L_m = \langle AS, CS^*, M_{ac}^*, SD, M_{as} \rangle$  où  $AS = \{DDMM, SDMM, EDMM\} \cup \{TM3\}$



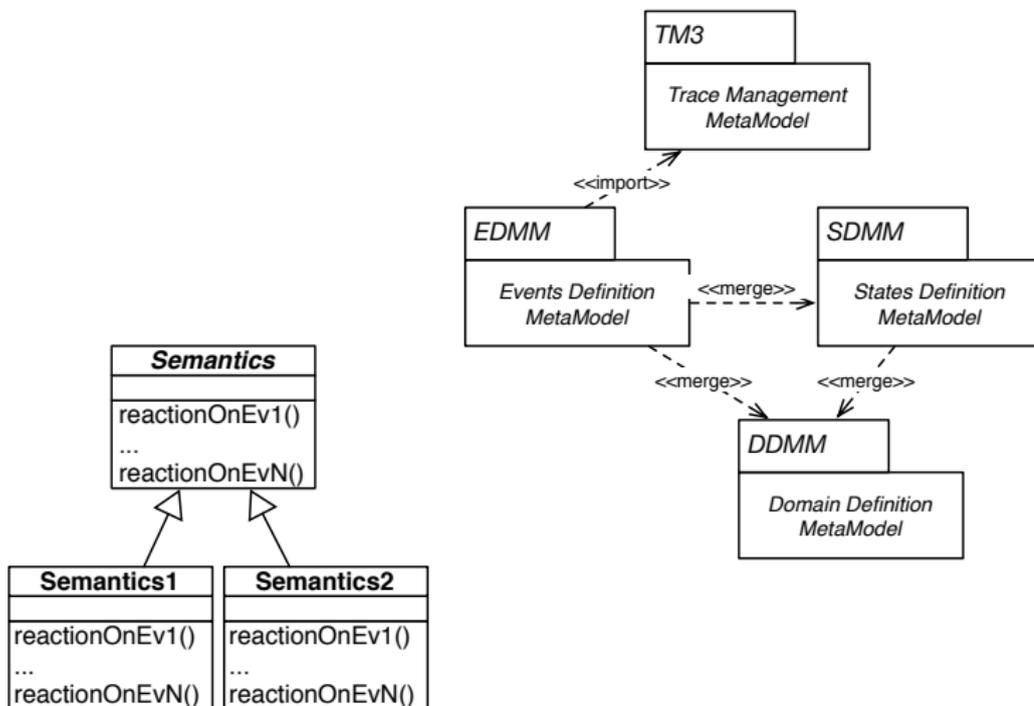
# Cadre générique pour la définition de simulateurs de modèle



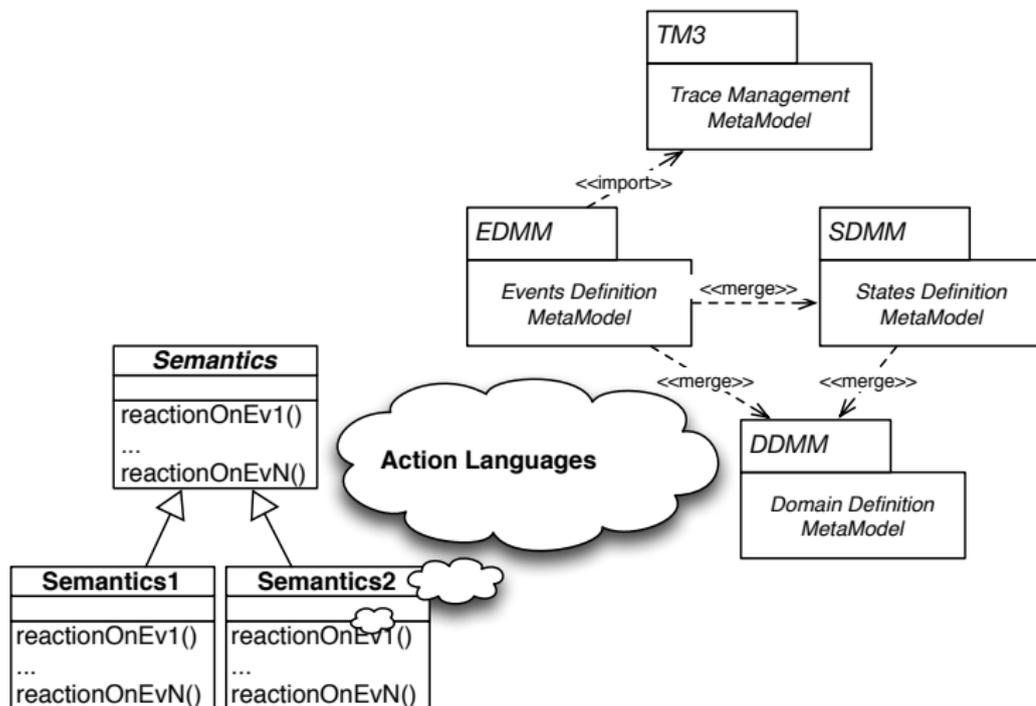
# Objectifs et approches pour la simulation de modèle



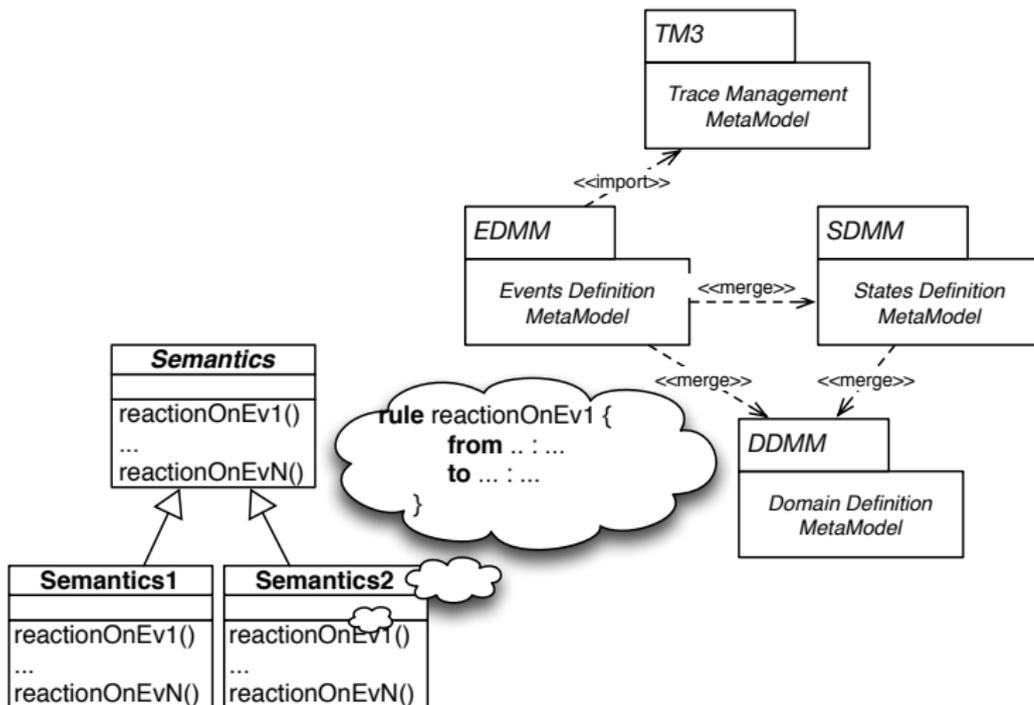
# Objectifs et approches pour la simulation de modèle



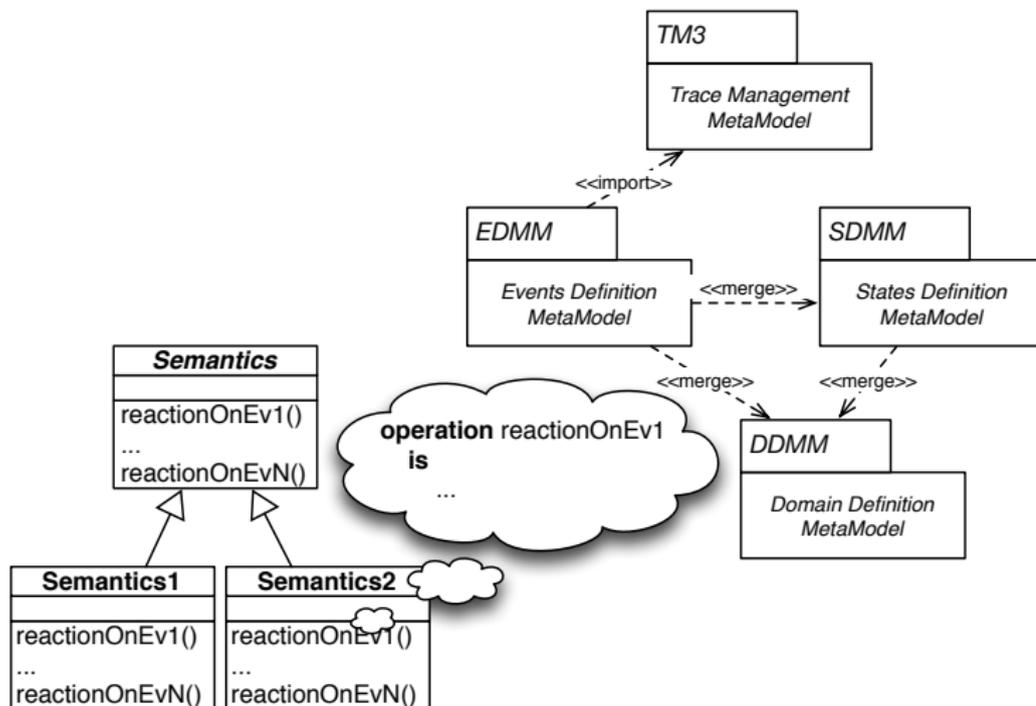
# Objectifs et approches pour la simulation de modèle



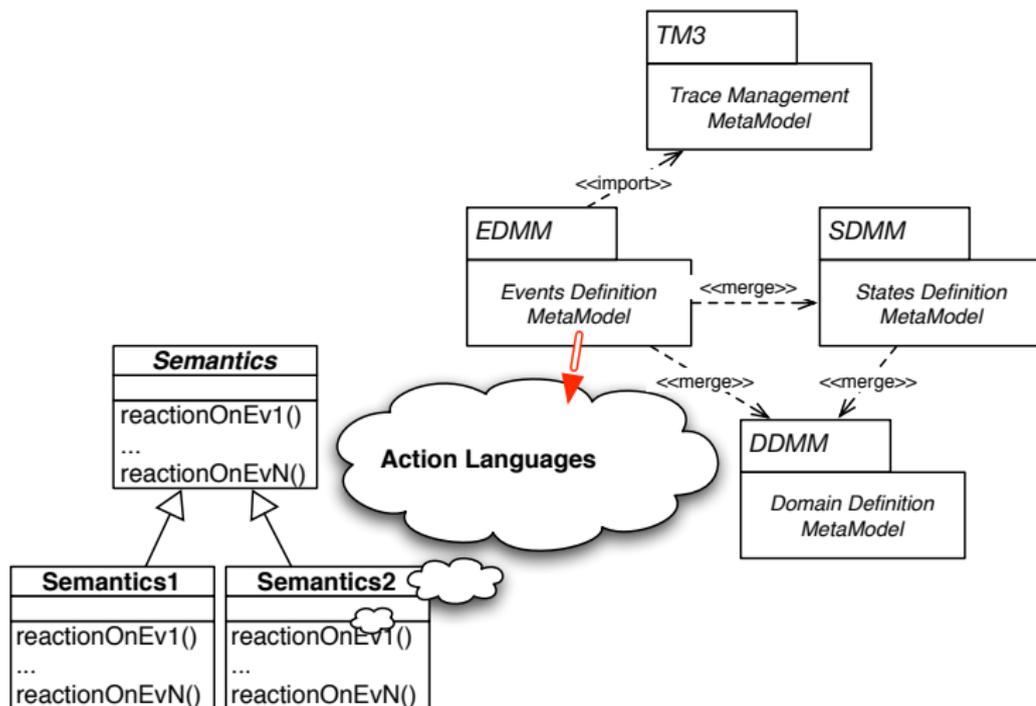
# Objectifs et approches pour la simulation de modèle



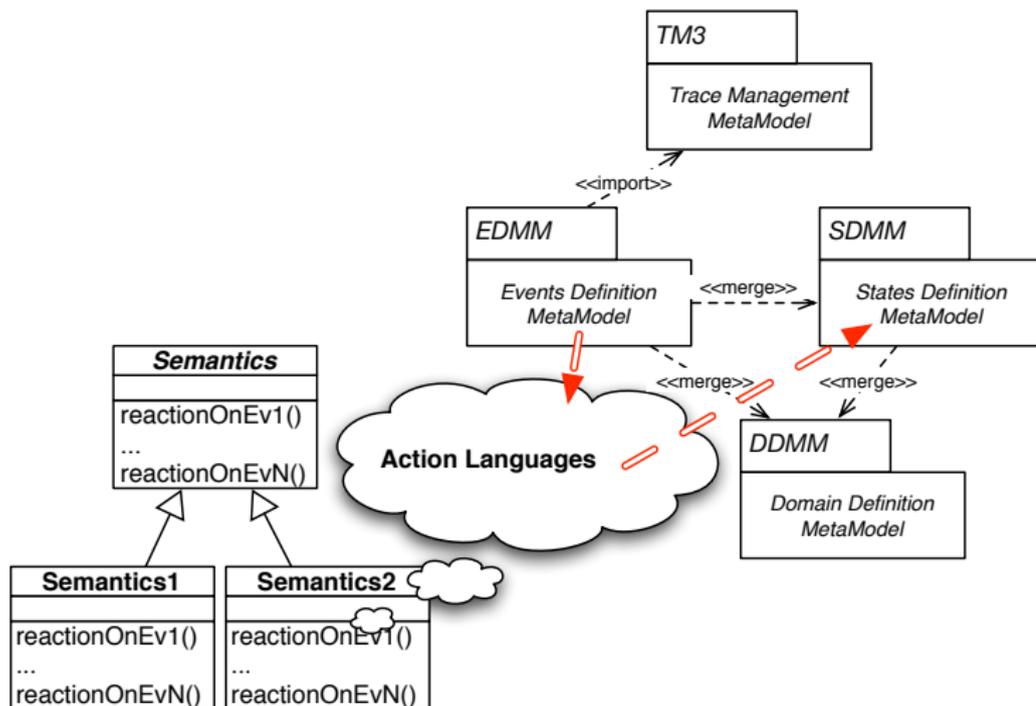
# Objectifs et approches pour la simulation de modèle



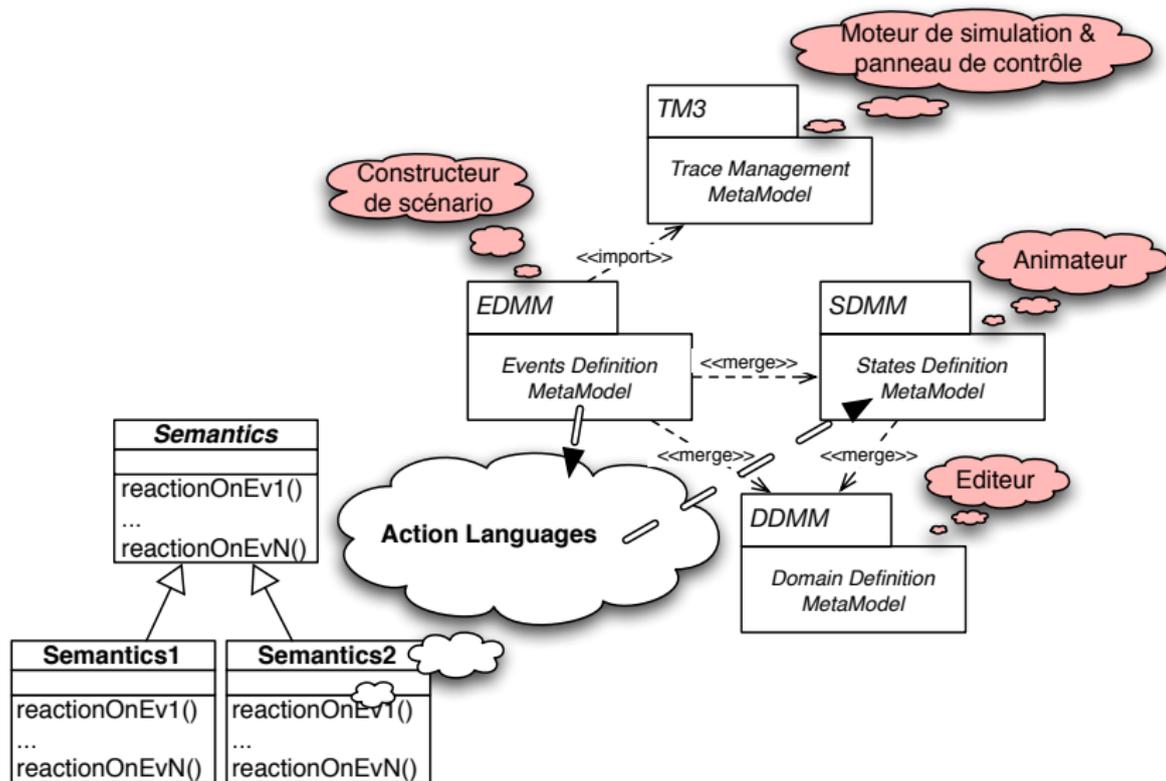
# Objectifs et approches pour la simulation de modèle



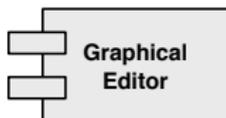
# Objectifs et approches pour la simulation de modèle



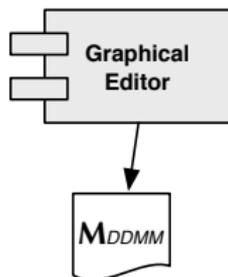
# Objectifs et approches pour la simulation de modèle



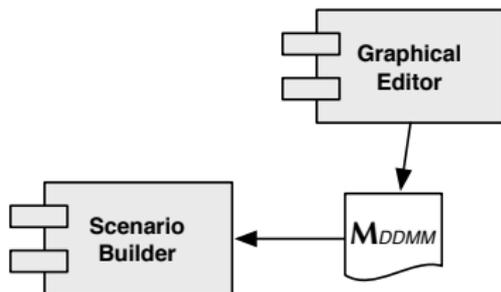
# Communication entre les composants



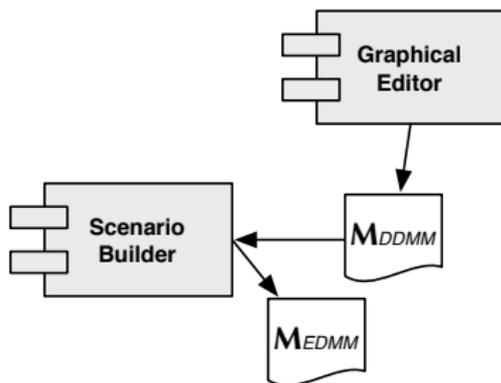
# Communication entre les composants



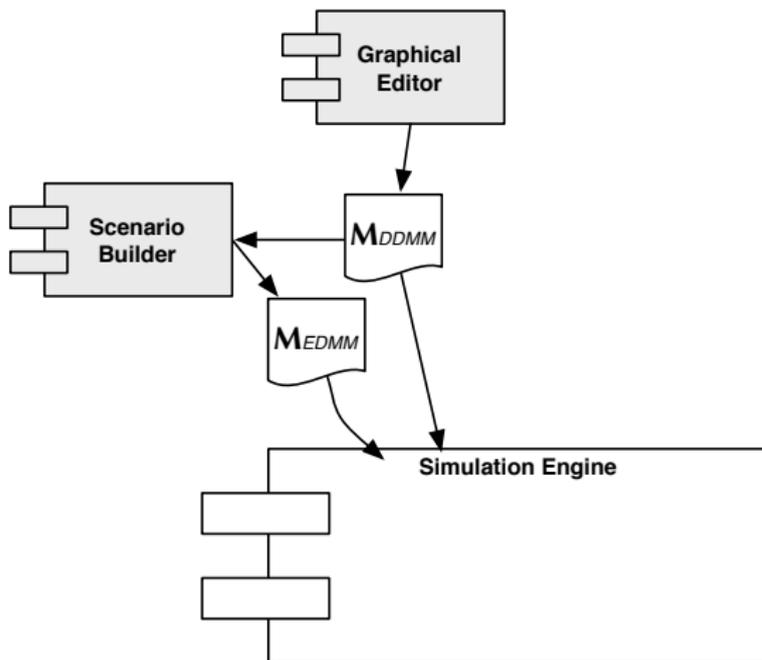
# Communication entre les composants



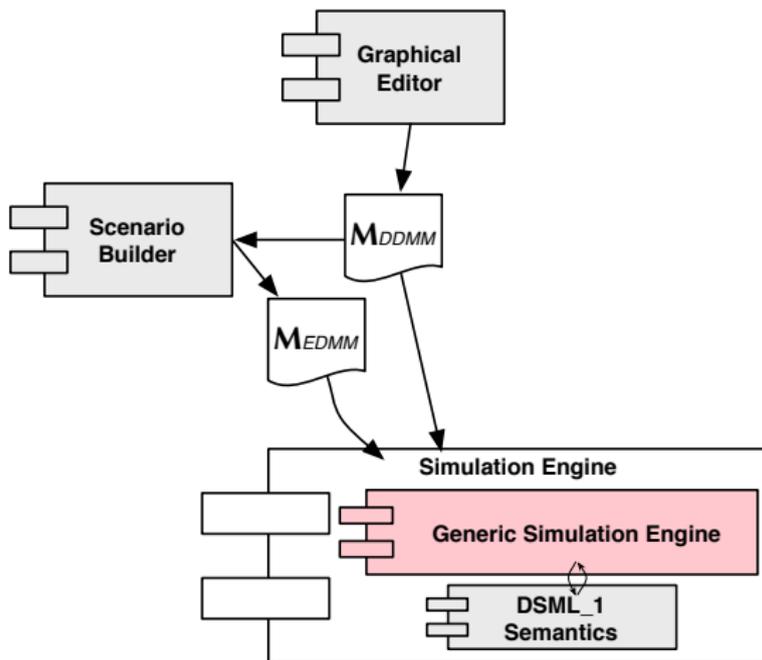
# Communication entre les composants



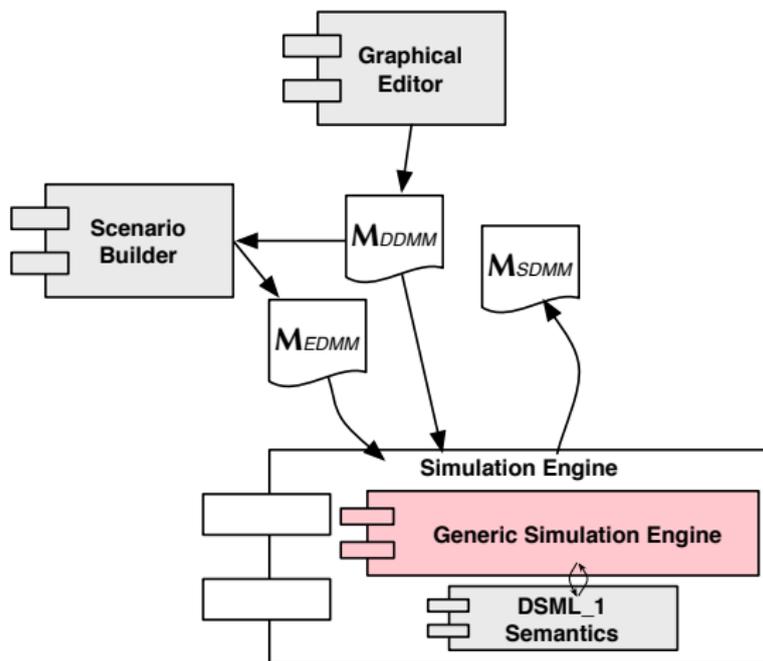
# Communication entre les composants



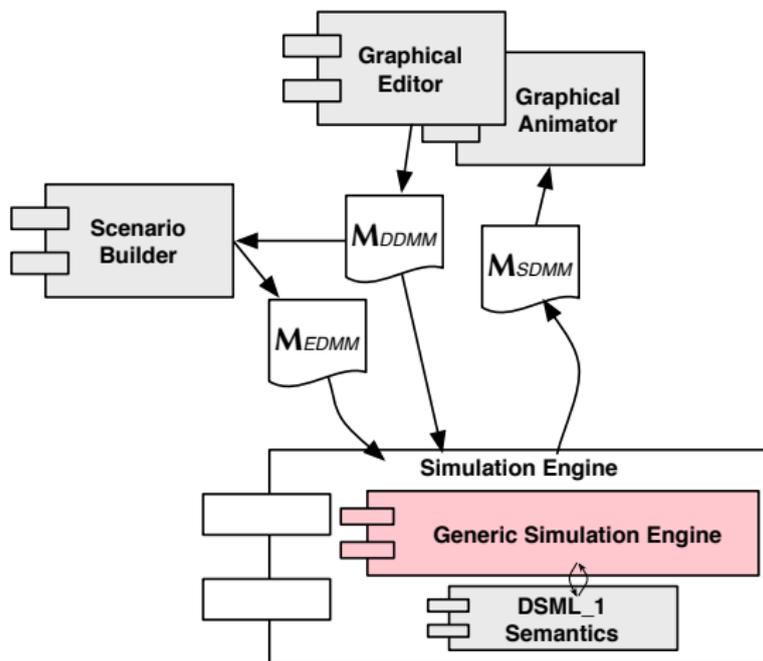
# Communication entre les composants



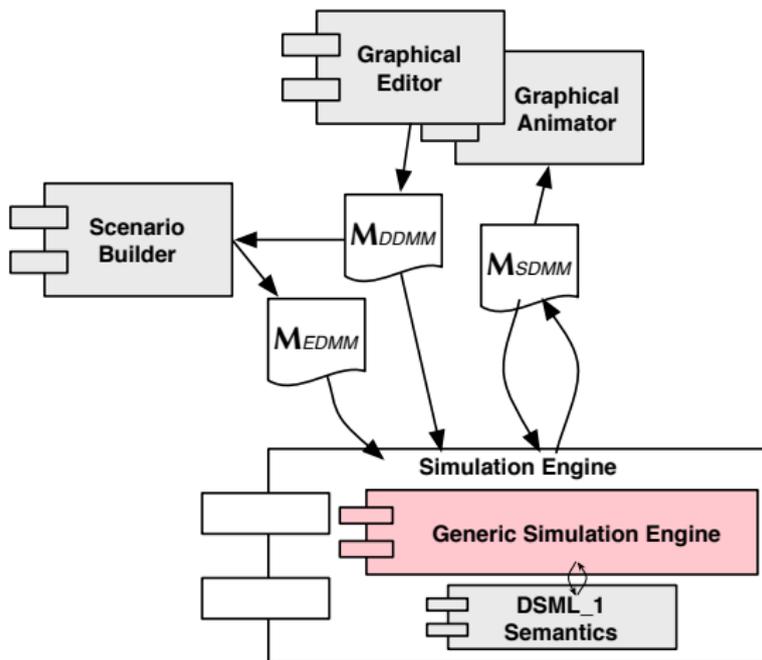
# Communication entre les composants



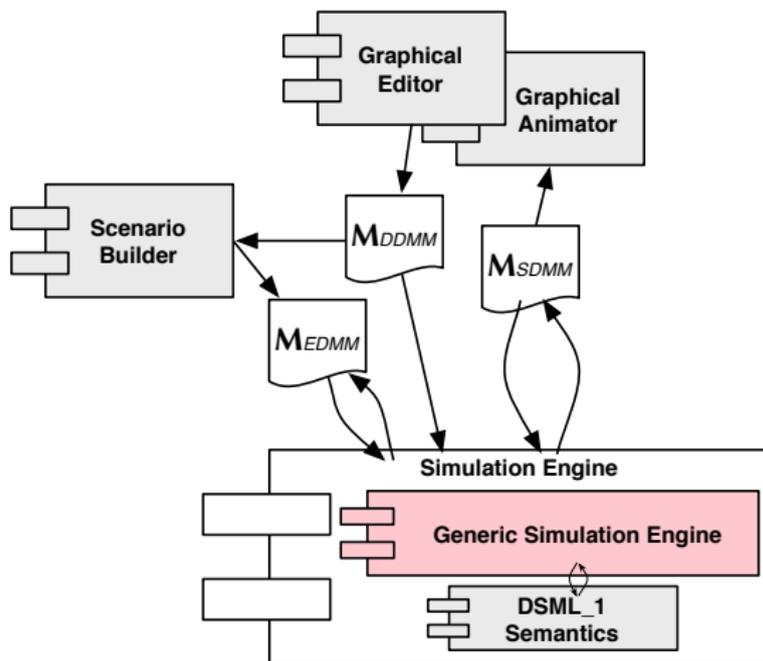
# Communication entre les composants



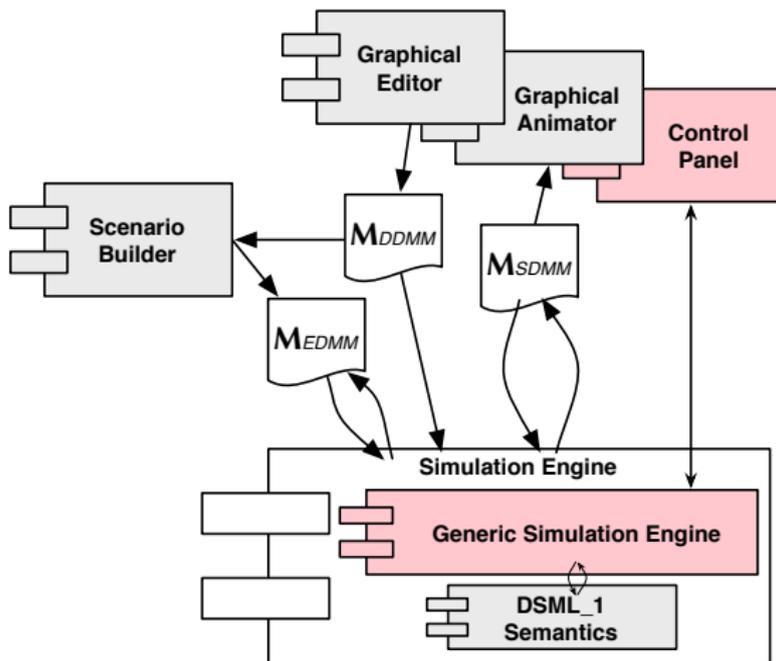
# Communication entre les composants



# Communication entre les composants

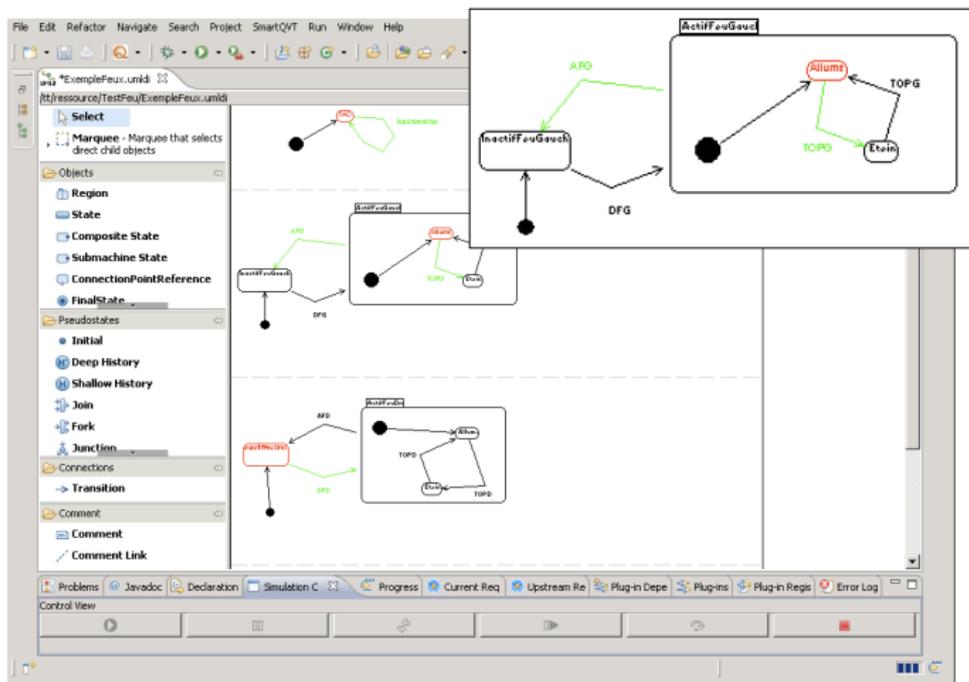


# Communication entre les composants





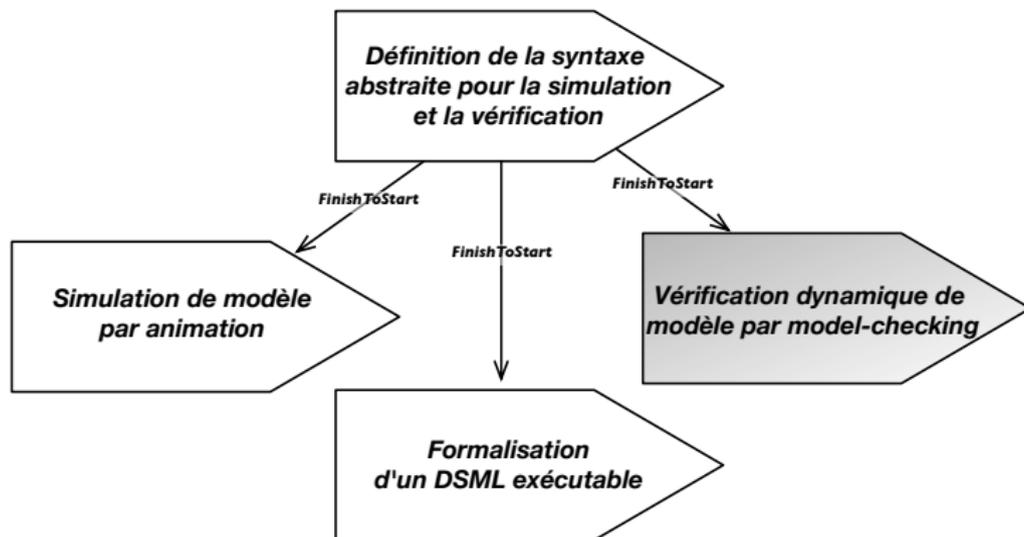
# Simulateur de machines à états UML2.0



Inclus dans l'atelier TOPCASED à partir de la version 2.0.

Synchronisé avec la version Ganymede d'Eclipse : **aujourd'hui !**

# Approche par traduction pour la vérification de modèle



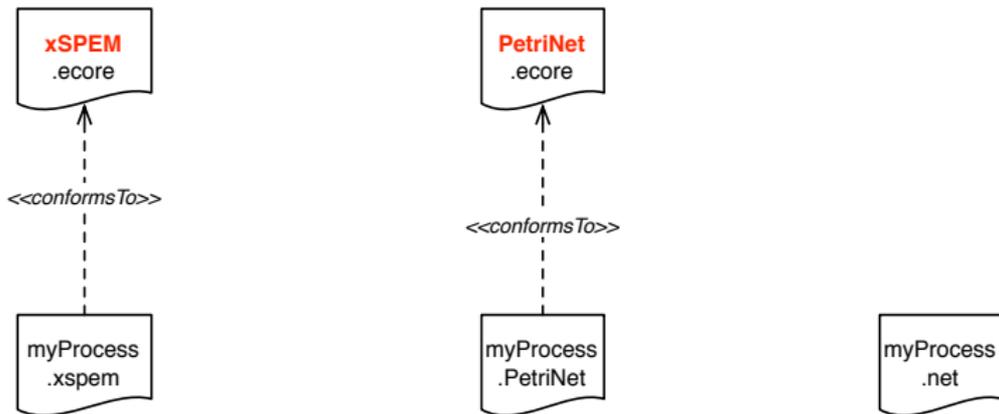
# Démarche générale pour la vérification de modèle



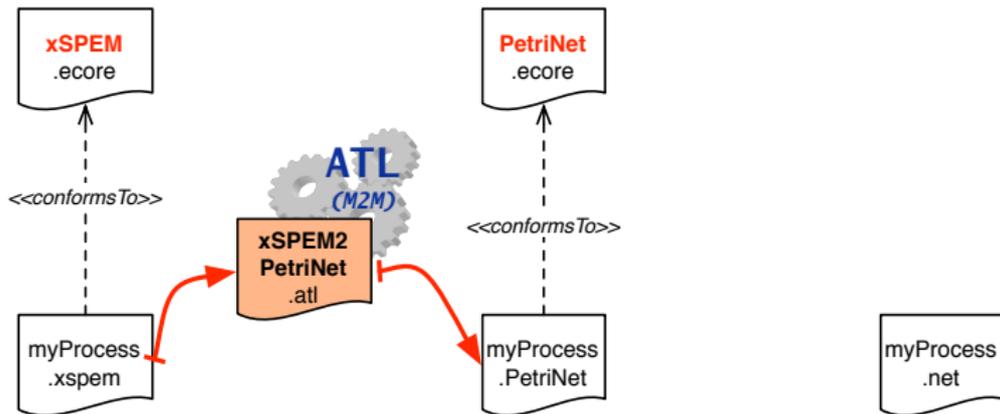
# Démarche générale pour la vérification de modèle



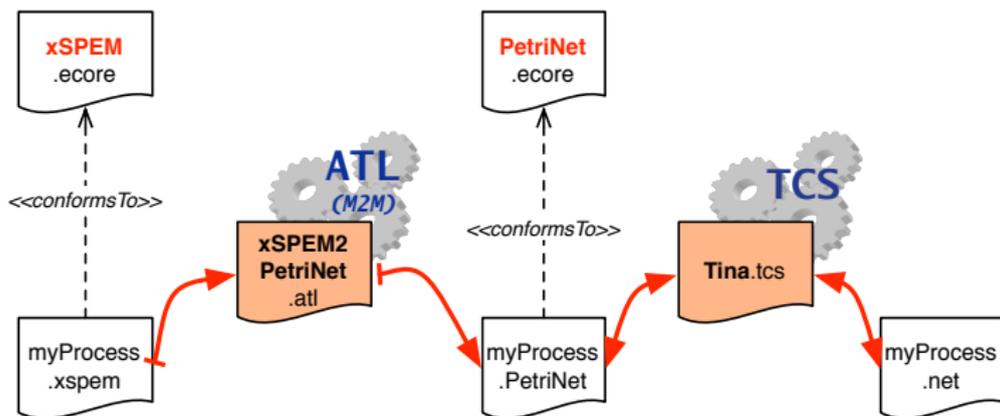
# Démarche générale pour la vérification de modèle



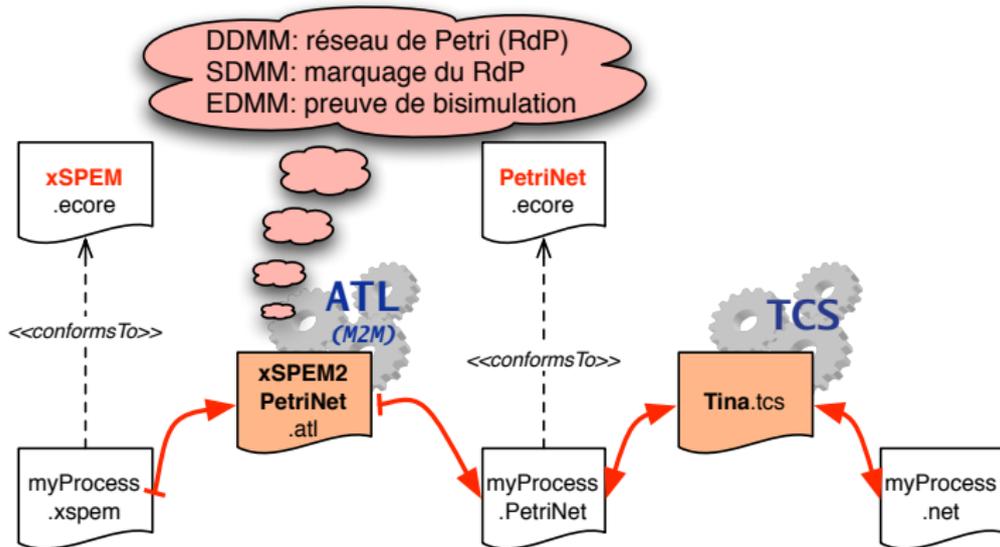
# Démarche générale pour la vérification de modèle



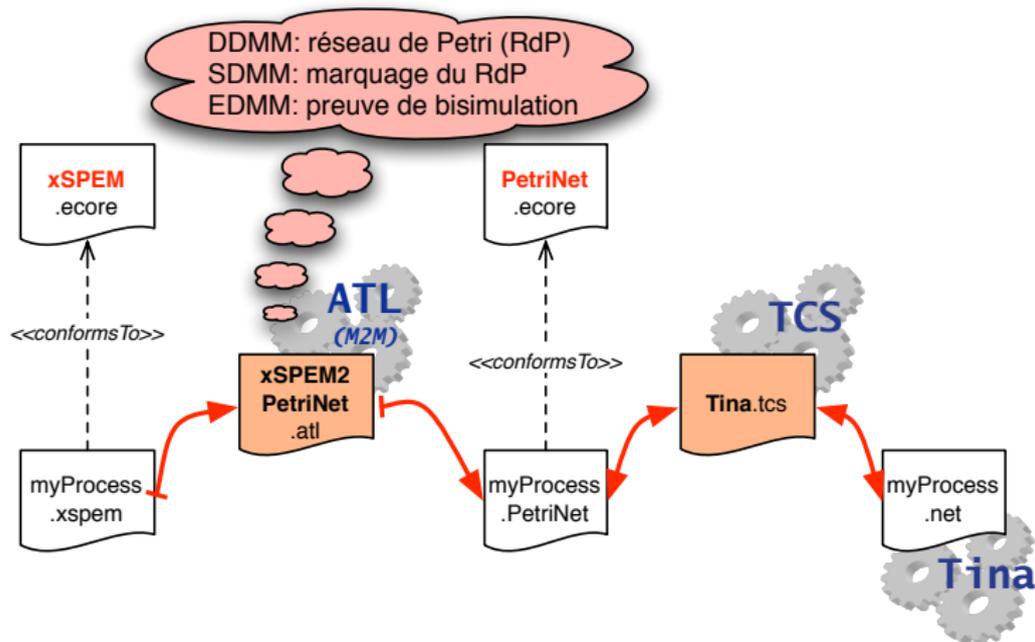
# Démarche générale pour la vérification de modèle



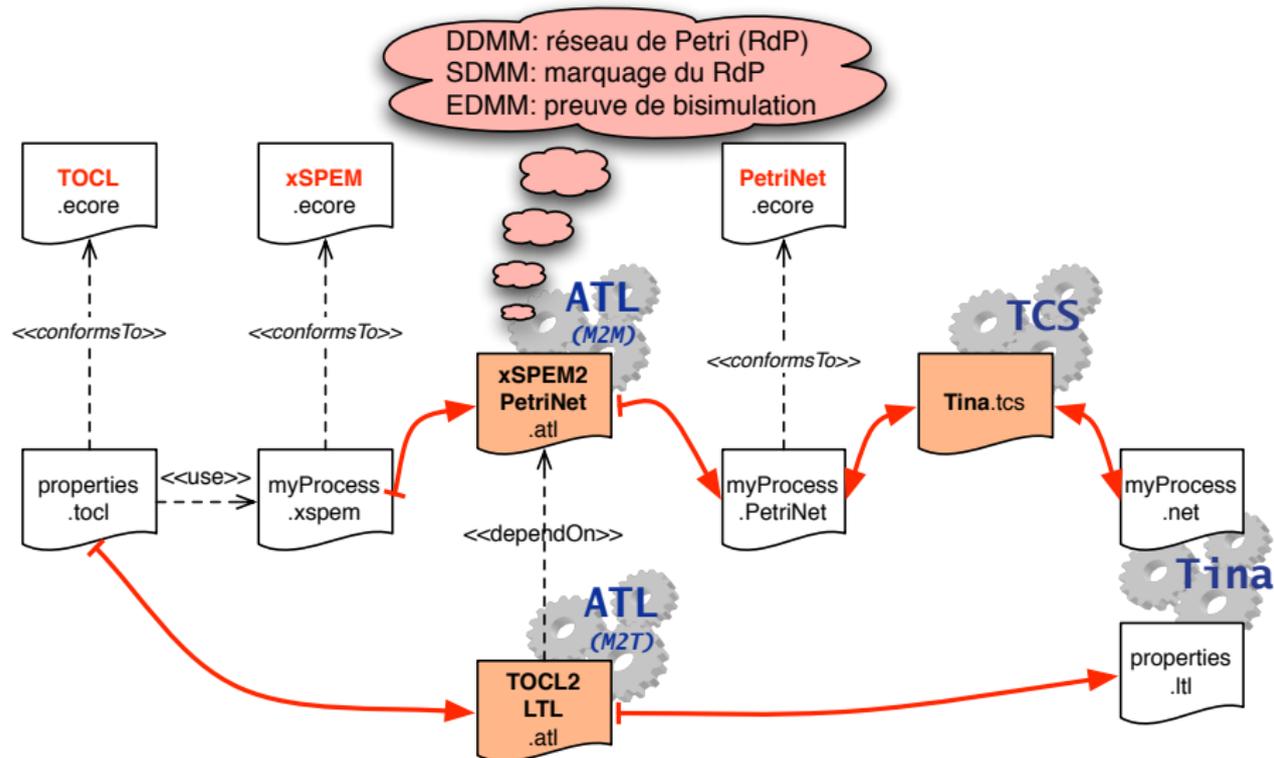
# Démarche générale pour la vérification de modèle



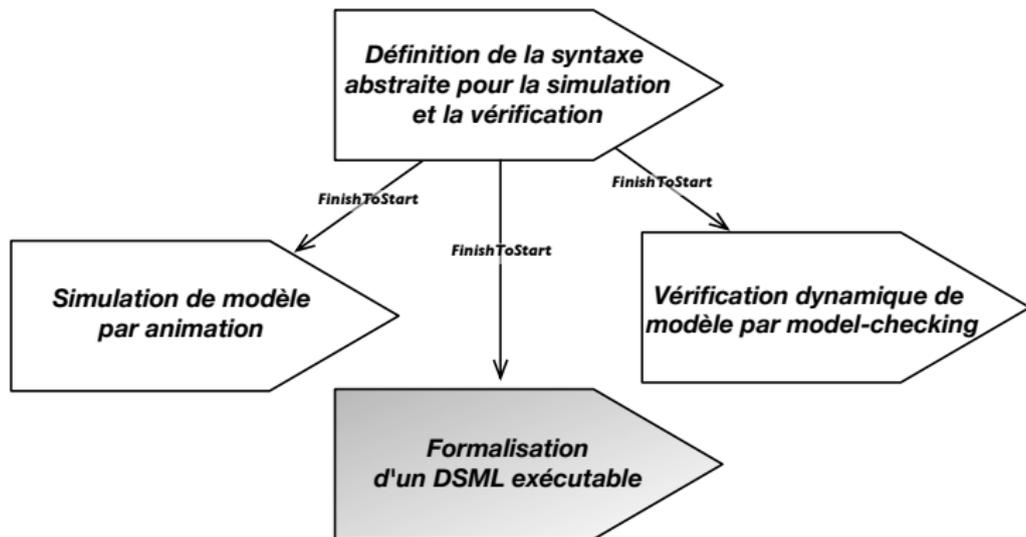
# Démarche générale pour la vérification de modèle



# Démarche générale pour la vérification de modèle



# Cadre formel pour la métamodélisation



# Cadre formel pour la métamodélisation

- **Objectif** : formaliser la sémantique comportementale de référence
  - ⇒ valider les sémantiques par traduction,
  - ⇒ engendrer les sémantiques opérationnelles.
- **Approche** : définition d'un *framework*, basé sur un outil formel comme l'assistant de preuve COQ, permettant de :
  - traduire de manière transparente et automatiquement la syntaxe abstraite (en MOF) et la sémantique statique (en OCL) d'un DSML,
  - définir la sémantique opérationnelle d'un DSML (sémantique de référence).

# Travaux relatifs : Jouault & Bézivin [FMOODS'06]

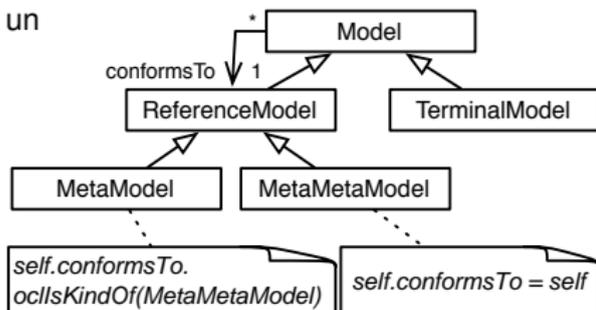
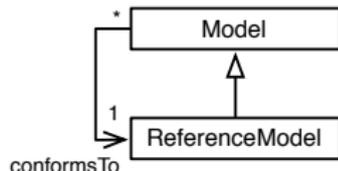
## ■ Notion de *ReferenceModel*

- « Un modèle est conforme à un modèle de référence »
- « Un modèle de référence est un modèle »

- ⇒ un *ReferenceModel* est conforme à un *ReferenceModel*,
- ⇒ un *ReferenceModel* initial doit être réflexif.

## ■ Illustré par la formalisation de KM3

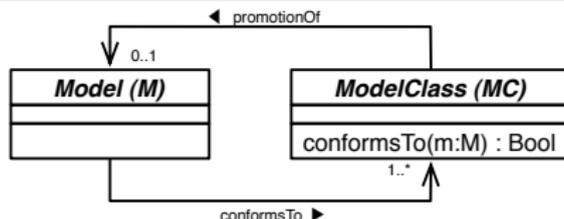
## ■ Implanté en Prolog



# Description *intuitive* de COQ4MDE

## Modèle (MODEL) et classe de modèles (MODELCLASS)

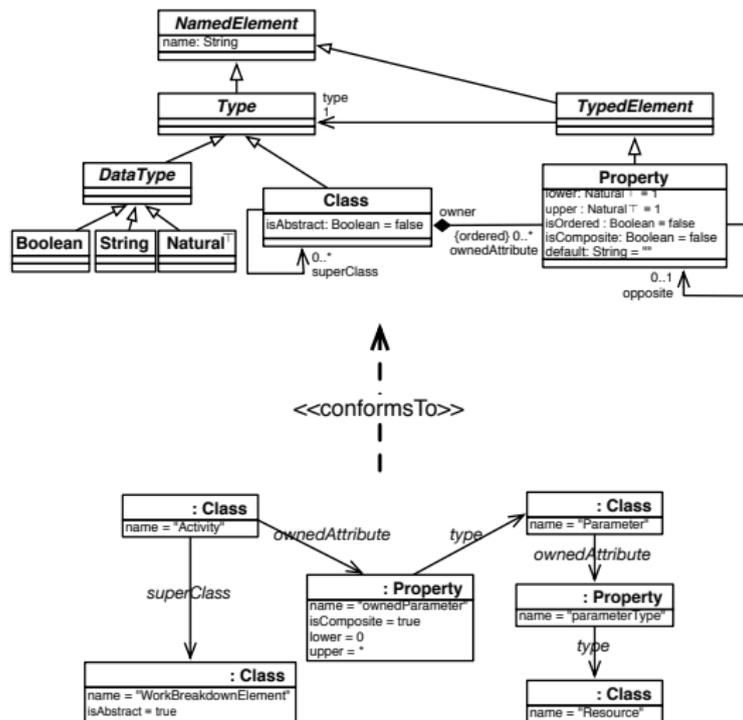
- MODELCLASS ( $\langle \textit{concepts}, \textit{relations}, \textit{semantique} \rangle$ ) :
  - langage de modélisation permettant de définir une famille de modèles,
  - spécifie les propriétés sémantiques de ces modèles.
- MODEL ( $\langle \textit{objets}, \textit{liens} \rangle$ ) : le niveau des instances.



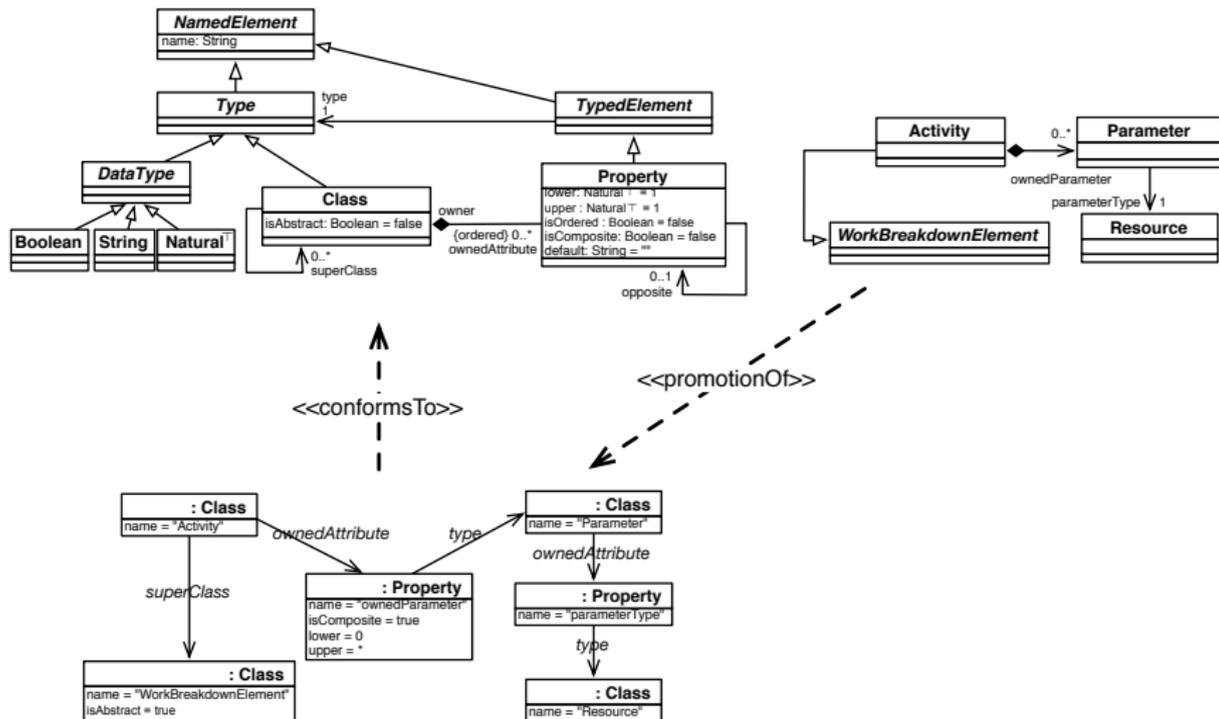
- Un modèle est conforme à une ou plusieurs classe(s) de modèles.
- Une classe de modèles est obtenue directement ou par promotion.



# Formalisation de xSPEM



# Formalisation de xSPEM



# Évaluation de COQ4MDE

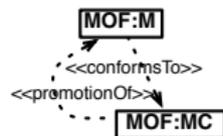
- Formalisation de la classe de modèles  
*EMOF\_Core* ( $MC_{MOF}$ ).

MOF:MC

# Évaluation de COQ4MDE

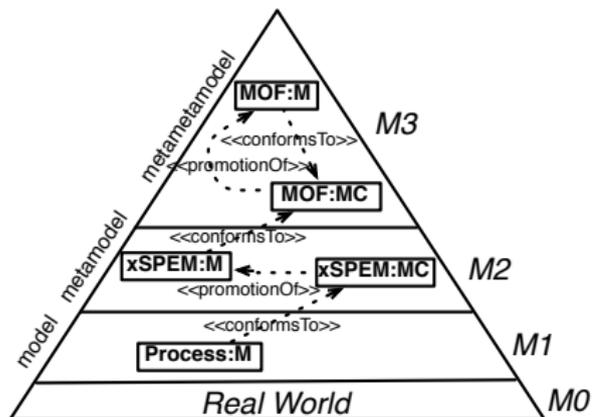
- Formalisation de la classe de modèles  $EMOF\_Core$  ( $MC_{MOF}$ ).
- Vérification de la métacircularité de  $EMOF\_Core$  : définition du modèle  $M_{MOF}$  conforme à  $MC_{MOF}$  et de la promotion  $P$ , t. q. :

$$P(M_{MOF}) = MC_{MOF}$$



# Évaluation de COQ4MDE

- Formalisation de la classe de modèles  $EMOF\_Core$  ( $MC_{MOF}$ ).
- Vérification de la métacircularité de  $EMOF\_Core$  : définition du modèle  $M_{MOF}$  conforme à  $MC_{MOF}$  et de la promotion  $P$ , t. q. :
 
$$P(M_{MOF}) = MC_{MOF}$$
- Formalisation de la pyramide de l'OMG.



# Plan

- 1 Contexte
- 2 Vers une opérationnalisation des modèles dans l'IDM
  - La métamodélisation
  - Définition de la sémantique d'exécution d'un DSML
- 3 Contributions pour la simulation et la vérification de modèle
  - Démarche outillée pour la définition d'un DSML exécutable
  - Cadre générique pour la définition de simulateurs de modèle
  - Approche par traduction pour la vérification de modèle
  - Cadre formel pour la métamodélisation
- 4 Conclusion & perspectives

# Contributions

- Taxonomie des techniques pour implanter la sémantique d'exécution d'un DSML [IDM'06,MDEIS'06],
- Démarche de métamodélisation pour la simulation et la vérification de modèle,
  - Approche dirigée par les propriétés pour la définition de la syntaxe abstraite d'un DSML exécutable [EIS vol.IX],
  - Approche pour la vérification formelle de modèle par sémantique de traduction [IDM'07,ECMDA'08],
  - Architecture et outils génériques pour la simulation de modèle par sémantique opérationnelle [TopcasedWP2.10 D01, TopcasedWP2.10 D02],
- Cadre formel de métamodélisation et API en COQ pour la définition des modèles et de la sémantique statique [Towers'07].



# Applications et transfert technologique

- **xSPEM**, une extension eXécutable de SPEM2.0 [APSEC'07],
- **TopProcess** (WP1 & Tectosages) : outils pour la modélisation, la vérification et la simulation de modèles xSPEM [ERTS'08a]
  - ⇒ <http://combemale.perso.enseeiht.fr/xSPEM/>
  - ⇒ <http://gforge.enseeiht.fr/projects/topcased-pe/>
- **Topcased Tina Bridges** (WP3 & LAAS CNRS et SopraGroup) : intégration sur le bus de modèles TOPCASED des projecteurs pour TINA
  - ⇒ <http://gforge.enseeiht.fr/projects/tina-bridges/>
- **Topcased Model Simulation** (WP2 & ONERA, Anyware et Atos Origin) : simulateur de machines à états UML2.0 [ERTS'08b], *et bientôt SAM...*
  - ⇒ <http://gforge.enseeiht.fr/projects/topcased-ms/>

# Perspectives

- Environnement d'expression des propriétés temporelles
- Analyse des résultats de vérification
- Extension du cadre de simulation (points d'arrêt, outils d'analyse, etc.)
- Sémantique d'exécution avec création dynamique
- Environnement formel de métamodélisation
- Administration de systèmes dirigée par les modèles

## *Soutenance de thèse*

# Approche de métamodélisation pour la simulation et la vérification de modèle

## Application à l'ingénierie des procédés

Benoît Combemale

*Institut de Recherche en Informatique de Toulouse (CNRS UMR 5505)*  
Université de Toulouse, France.

<http://combemale.perso.enseeiht.fr>