Praxeme, le sens de l'action

*Praxime, initiative pour une méthode publique*

*« We can't solve problems by using the same kind of thinking we used when we created them. » . »*
Albert Einstein

# MDA, le retour d'une bonne idée

✉ dvau@praxeme.org
☎ +33 (0) 6 77 62 31 75
http://www.praxeme.org

Référence : SLB-19 Version : 30/11/07

MDA réactive une idée présente à l'aube du génie logiciel.

Cette idée est simple :

•Avant de produire un logiciel – *a fortiori* un système –, il est nécessaire d'en faire un modèle.

•Un modèle ne suffit pas : il en faut plusieurs, chacun relatif à une problématique particulière.

Cependant, cette bonne idée a été perdue. Si, maintenant, le standard MDA de l'OMG la réhabilite, il est bon que nous nous penchions sur notre histoire pour comprendre et dépasser les raisons qui ont conduit cette bonne idée à l'échec.

Cf. la citation.

# Objectif de la présentation

- **Objectif**

  **Identifier les modèles nécessaires pour couvrir la chaîne d'activité**

- **Thèmes**
  - Les fondamentaux, l'évolution des idées
  - Les types de modèles
  - L'enchaînement des modèles
  - La production du code

Durée de la présentation : 40 mn





MDA est un cadre formel. C'est dire qu'il ne s'intéresse pas au contenu et, particulièrement, ne définit pas les types de modèles nécessaires à nos activités. Cette limite permet, justement, d'établir un standard !

Définir les catégories de modèles revient à la méthodologie. C'est ce que nous allons proposer ici.

## Contenu de la présentation

1. **Une brève histoire des modèles**
2. **À la recherche des modèles**
3. **L'articulation des modèles**



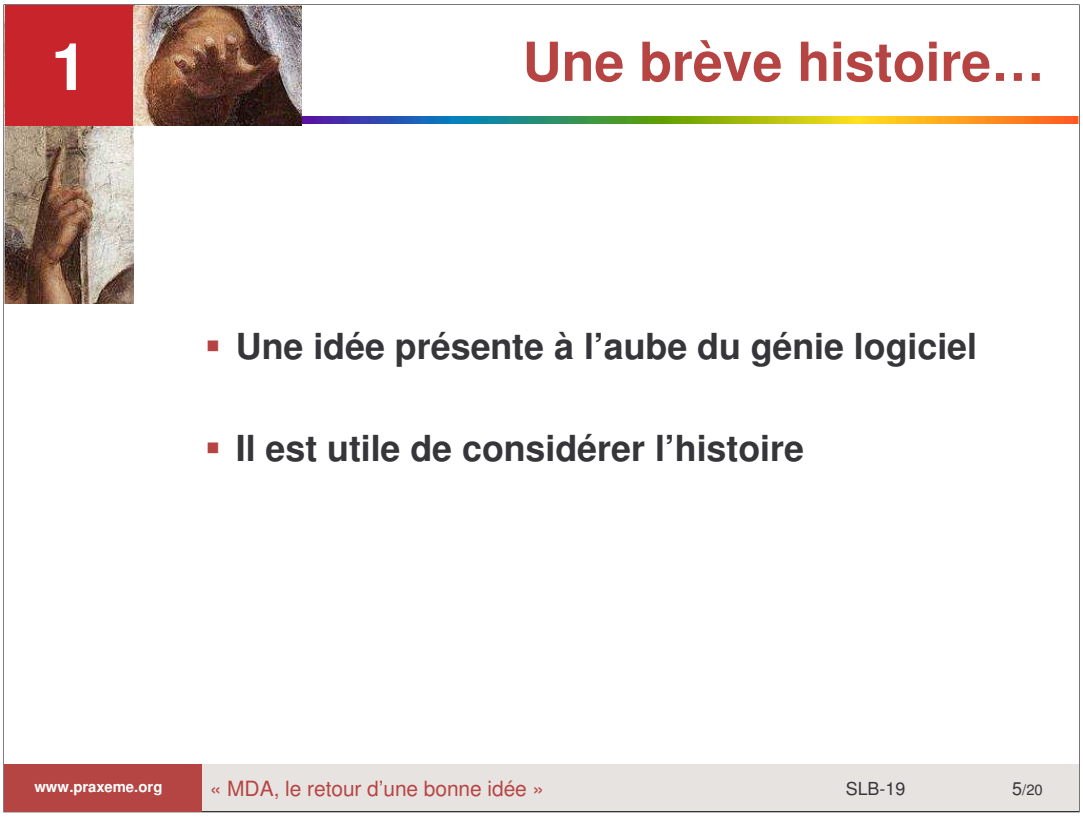
Tout d'abord, un exercice de mémoire (sans lequel nous ne pouvons apprendre de nos erreurs passées, en tant que communauté professionnelle).

La partie centrale proposera les types de modèles tels que la méthodologie Praxeme les identifie.

Ce n'est pas suffisant d'identifier les modèles, il faut encore les articuler. Nous illustrons ce point en évoquant les règles de dérivation qui permettent de construire un système SOA à partir des modèles amont.

# Agenda

| Partie | Durée | Horaire |
|---|---|---|
| Une brève histoire des modèles | 5 mn | 14h – 14h05 |
| À la recherche des modèles | 20 mn | 14h05 – 14h25 |
| L'articulation des modèles | 10 mn | 14h25 – 14h35 |
| Conclusion | 5 mn | 14h35 – 14h40 |

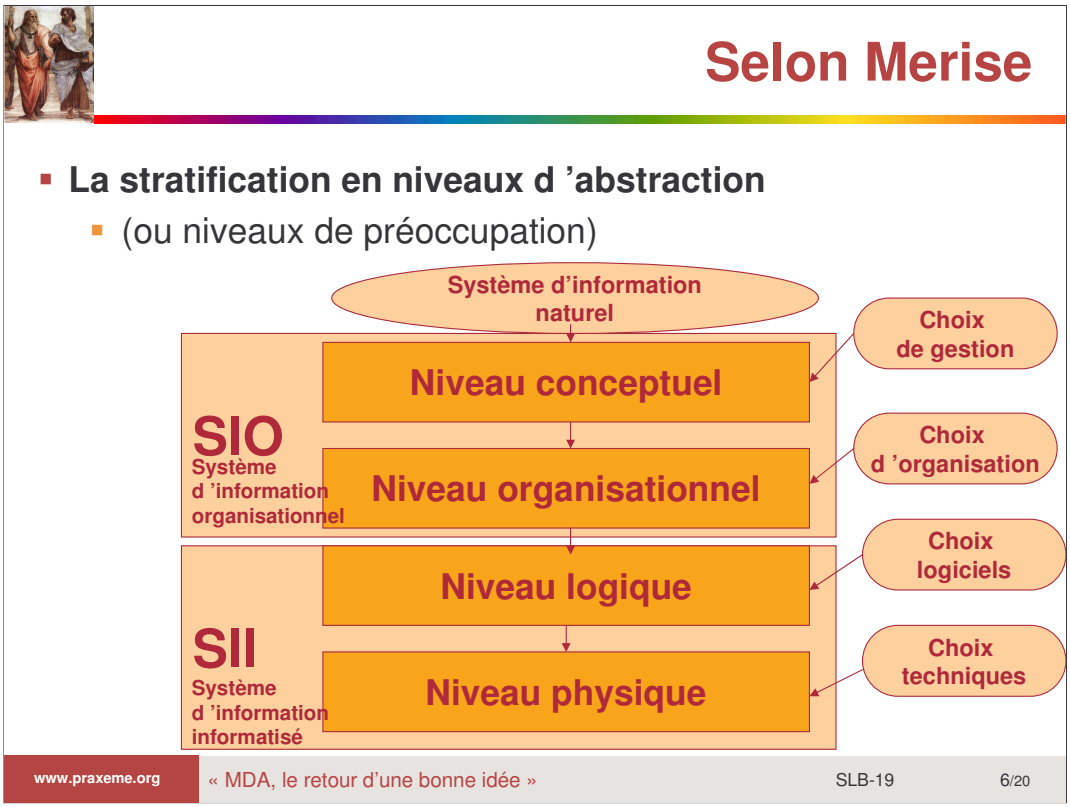



La communauté informatique présente une curieuse particularité : c'est, sans doute, le seul métier intellectuel sans histoire, du moins sans mémoire. Nous sommes tellement obnubilés par la nouveauté que nous ne regardons jamais derrière nous, sauf pour repousser, dans un mépris amusé, l'expérience de nos prédécesseurs. Et, à chaque nouvelle « technologie », nous nous empressons de jeter aux orties les quelques bonnes habitudes que nous avions péniblement acquises.

Ce faisant, il nous arrive de jeter le bébé avec l'eau du bain. Ainsi, les apports des méthodes antérieures ont été perdus et nous devons laborieusement réinventer certaines évidences. MDA reprend, avec quelques termes nouveaux, des idées bien en place dans les années 80.

Ce constat, en soi, ne nous apporte rien. Nous devons nous demander pourquoi les méthodes ont régressé, pourquoi – avant même qu'elles tombent en désuétude – leurs préceptes n'étaient pas réellement appliqués, pourquoi l'industrialisation du logiciel rêvée depuis des décennies ne s'est pas concrétisée…

Ces questions nous sensibiliseront aux risques qui guettent la nouvelle vague libellée « MDA ». Faute d'un tel questionnement, il n'y a aucune raison que nous ne tombions pas dans les mêmes panneaux que nos prédécesseurs.




Critique

L 'apport

Il importe, en effet, de séparer les plans (niveaux d'abstraction)

Pour une meilleure évolution (« *separation of concerns* »)

Pour gérer les compétences et responsabilités

Les limites

Le postulat de la séparation données-traitements

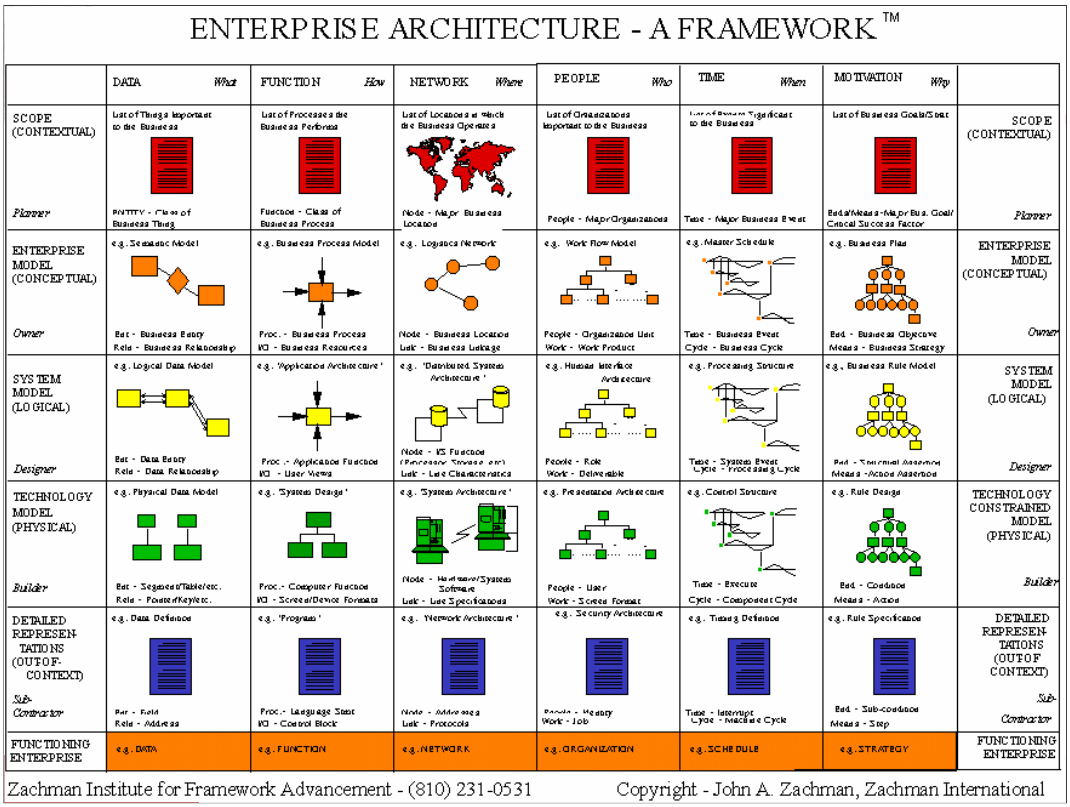
Ce n 'est pas un axiome valide pour tous les acteurs !

La focalisation sur l 'information

Ce n 'est qu 'un des aspects de l 'activité

Les niveaux s 'étagent selon selon un processus de production logiciel

…plutôt que par une logique comprise par tous



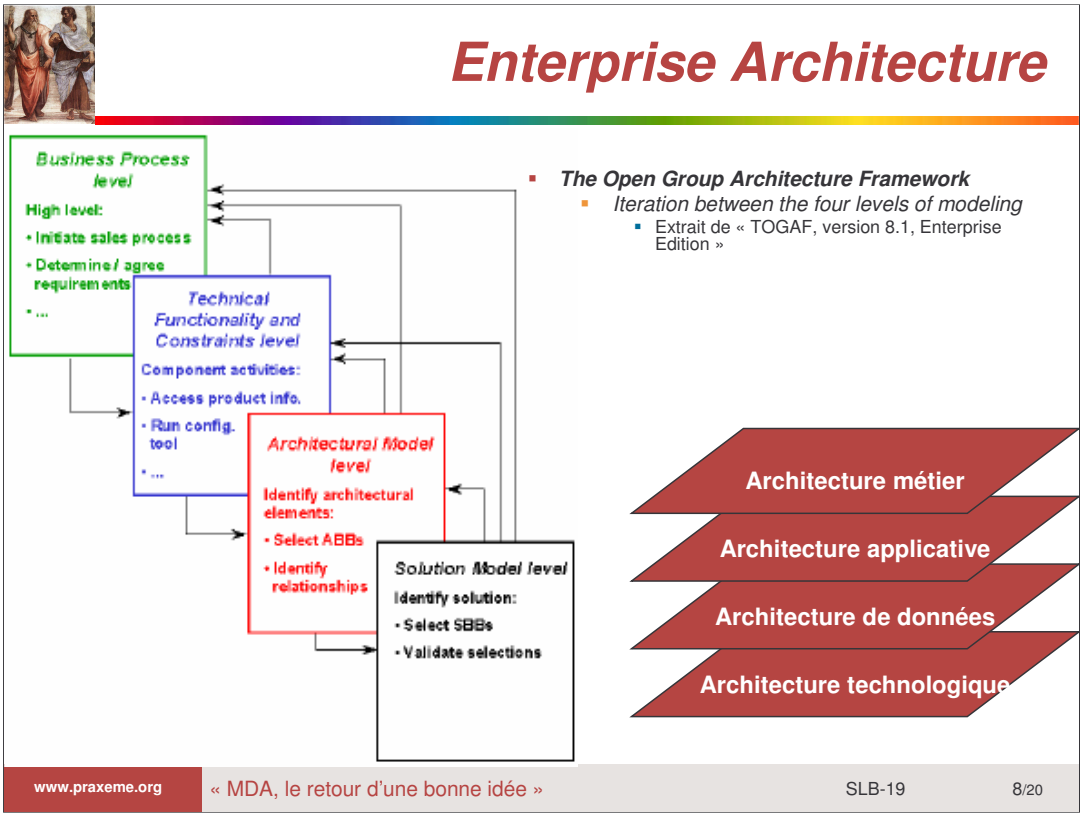

Critique

Les apports

- Un cadre complet

Les limites

- Une coloration fortement informatique
  - Données, fonctions
  - Conséquence = réduction sur la représentation
- Les catégories sont posées *a priori*
  et non justifiées
- Une inflation en nombre de modèles
  - Par croisement de deux critères
  - L ’effet pervers des matrices...




Le courant *Enterprise Architecture*, sans trop insister sur les modèles, ressent le besoin de séparer les représentations. Les plans de représentation, généralement au nombre de quatre, révèlent :

•un flottement théorique (les plans peuvent différer d'une méthode à l'autre) ;

•une orientation qui reste très informatique.

Par ailleurs, ce courant n'a pas encore pris en compte les avancées d'UML et de MDA. Le niveau souvent très général auquel se situe les pratiques d'EA explique le peu de rigueur des représentations utilisées. De ce fait, se pose le problème de l'ajustement entre la grande vision de l'architecte d'entreprise (ou de l'urbaniste de SI) et les modèles détaillés nécessaires au niveau des projets.

Des travaux de rapprochement sont en cours.

## Bilan

- **La bonne idée : les niveaux d'abstraction**
  - *"separation of concerns"*
- **Les difficultés dans les pratiques**
  - Trop orienté informatique
    - Mélange des considérations dans les modèles
- **Les limites des méthodes antérieures**
  - Liées au paradigme de l'époque
    - La séparation données-traitements



Cet aperçu historique nous enseigne que :

•l'idée de base de MDA n'est pas nouvelle ;

•elle a été, jadis, promue par les méthodes ;

•elle a été rejetée par les informaticiens qui n'en ont pas toujours vu les apports et n'ont pas toujours toléré l'abstraction exigée ;

•la tendance ethno-centrique de la profession a rapidement évacué les modèles « amont ».

Bien sûr, les choses ont changé :

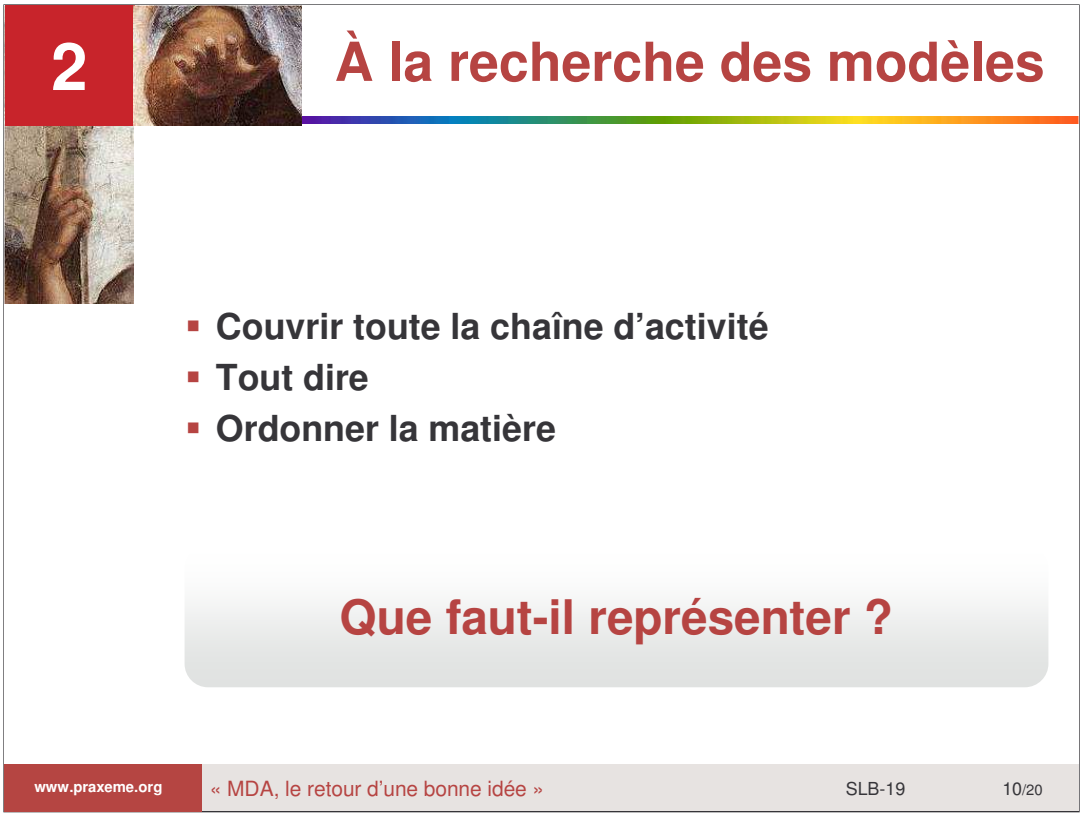
•nous nous sommes dotés d'une approche du réel un peu plus naturelle, la logique objet (est-elle si bien intériorisée, aujourd'hui ?) ;

•nous avons renoncé à la séparation données-traitements (pourtant, le « modèle sémantique » est perçu, presque toujours, comme un modèle de données) ;

•nous avons de nouveaux instruments (des outils de modélisation UML, des dispositifs d'agilité comme le MDM et les moteurs de règles).

Pour autant, nous pouvons toujours commettre les erreurs du passé, si nous n'y prenons garde.

Pour reprendre la citation d'Einstein et puisque l'idée des modèles n'est pas neuve : quel changement d'approche proposons-nous pour échapper à nos anciennes erreurs ?

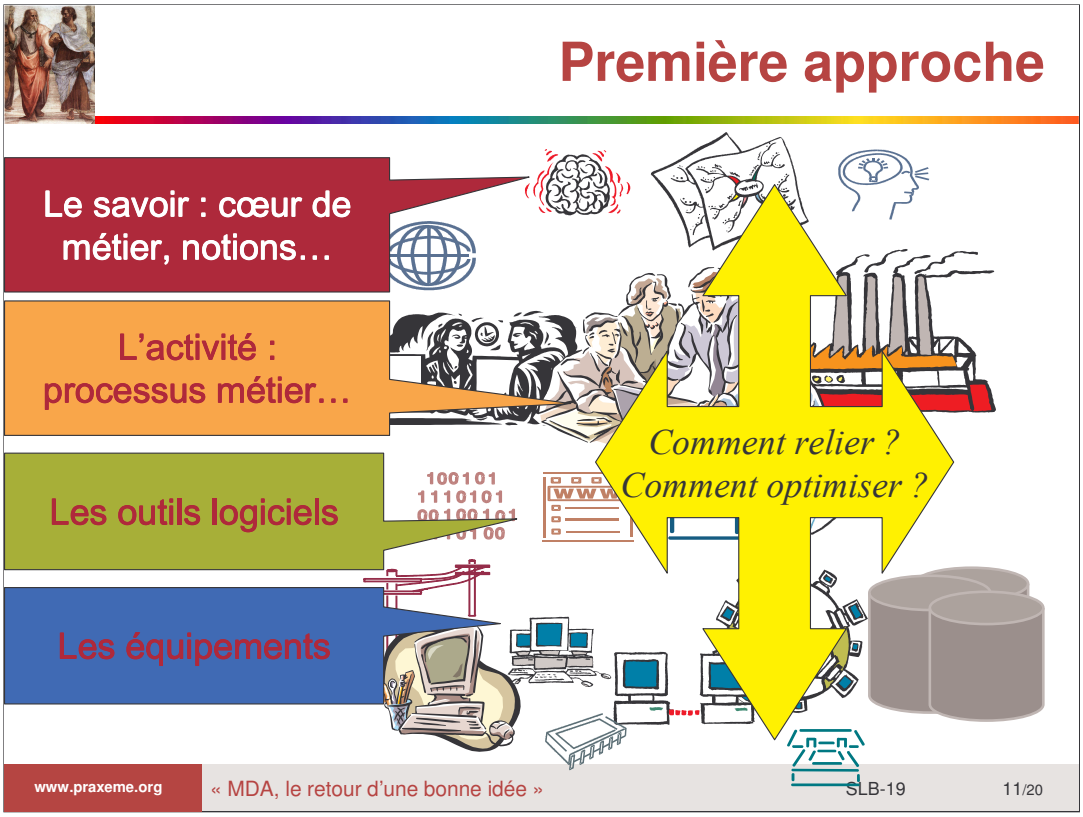



La question fondamentale est : « que faut-il représenter ? ».

Que ce soit au niveau des projets ou à celui des entreprises, nous sommes confrontés à la complexité du fait de l'énormité et de la disparité des informations à brasser, des questions à poser, des expertises à convoquer et des décisions à prendre.

Si nous considérons non seulement le développement du logiciel mais toute la chaîne d'activité qui va de la stratégie au déploiement, nous sommes pris de vertige !

MDA seul ne répond pas à cette question mais le standard nous met sur la voie en nous suggérant que certains des modèles devraient être indépendants des choix techniques.




Le premier piège à éviter : ne considérer le « métier » qu'en tant qu'activités.

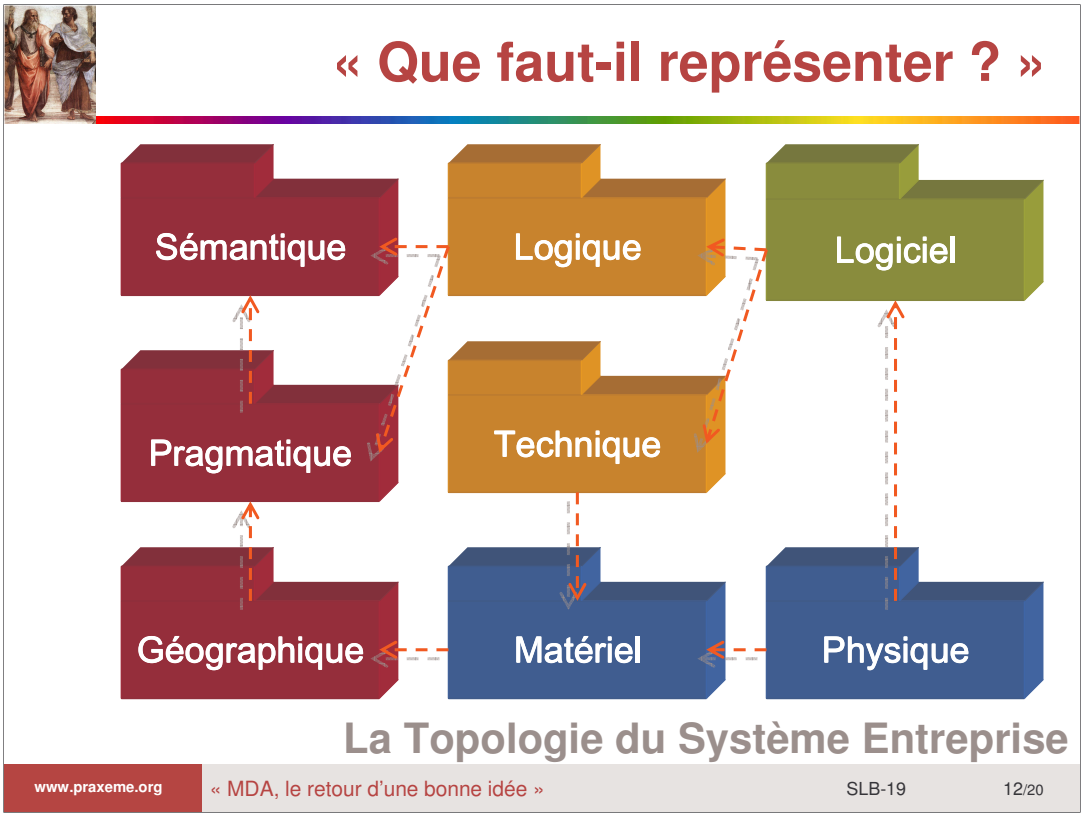
C'est une erreur commune puisque la plupart des méthodes indiquent, comme point d'entrée, soit les cas d'utilisation (méthodes de développement logiciel), soit les processus métier (courant *Enterprise Architecture* ou urbanisation des SI).

Or, il est un niveau d'abstraction préalable, que nous nommerons « sémantique ». Le positionnement de ces niveaux ne suffit pas pour les définir. Il s'agit aussi – et surtout – d'une différence d'approche. La description des activités (cas d'utilisation ou processus) applique l'approche fonctionnaliste, centrée sur l'activité et la décomposition hiérarchique des activités. Pour la description des fondamentaux du métier, nous adoptons une approche orientée objets. Les objets sont, ici, les objets du monde réel ou les « *business entities* », les concepts que manipulent les acteurs de l'entreprise, leurs partenaires, leurs clients… Ceci n'a rien à voir avec l'informatique, bien sûr.

Albert Einstein réclamait une nouvelle façon de penser. Voilà la réponse, du moins en ce qui concerne notre façon de penser l'entreprise, de la transformer et de la doter de nouveaux moyens :

1. introduire un nouveau niveau d'abstraction (en fait, réhabiliter le niveau « conceptuel » défini par Merise) ;
2. appliquer une approche radicalement orientée objet, comme instrument pour exprimer formellement la connaissance du métier.

Nous irons plus loin : les informations et décisions que nous séparons dans ces différents niveaux, nous devons les relier de façon à maîtriser les flots d'activités.

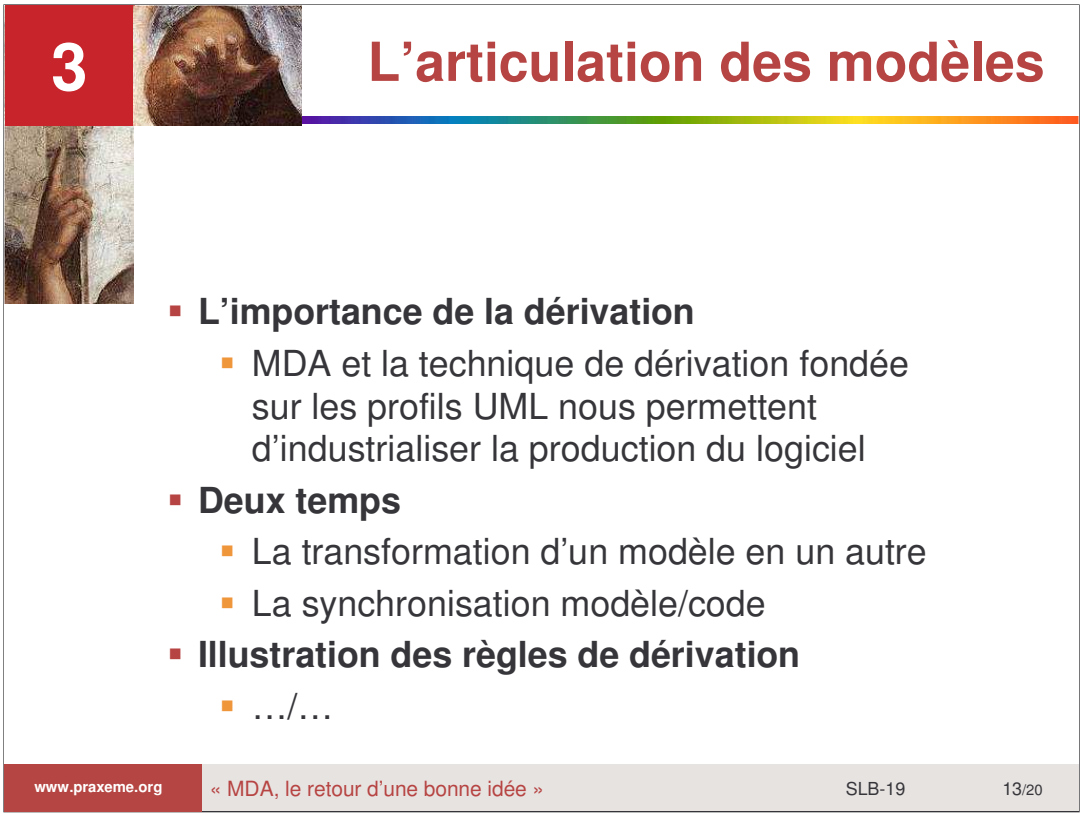



La Topologie du Système Entreprise fournit le socle théorique de la méthodologie Praxeme.

Elle identifie et articule huit aspects pour tout dire formellement de l'entreprise. Chaque aspect est susceptible d'une modélisation, c'est-à-dire d'une représentation formelle, rigoureuse. La méthodologie prévoit l'existence d'informations qui resteront dans le domaine de l'informel : objectifs de l'entreprise, exigences, vocabulaires… Ces informations ressortissent à la pré-modélisation. Elles sont conservées car précieuses. Le modélisateur les dirigera, comme justifications, vers les éléments des modèles appropriés (traçabilité).

Pour représenter chaque aspect, la méthodologie sélectionne les catégories de représentation les mieux adaptées. Le méta-modèle de Praxeme les expose en détail. Il est fondé sur le méta-modèle UML, dont il donne une lecture simplifiée, et s'inspire d'autres méta-modèles standards comme BMM.

Pour une justification de la Topologie du Système Entreprise, voir, par exemple, le « Guide général » de Praxeme, et son annexe.

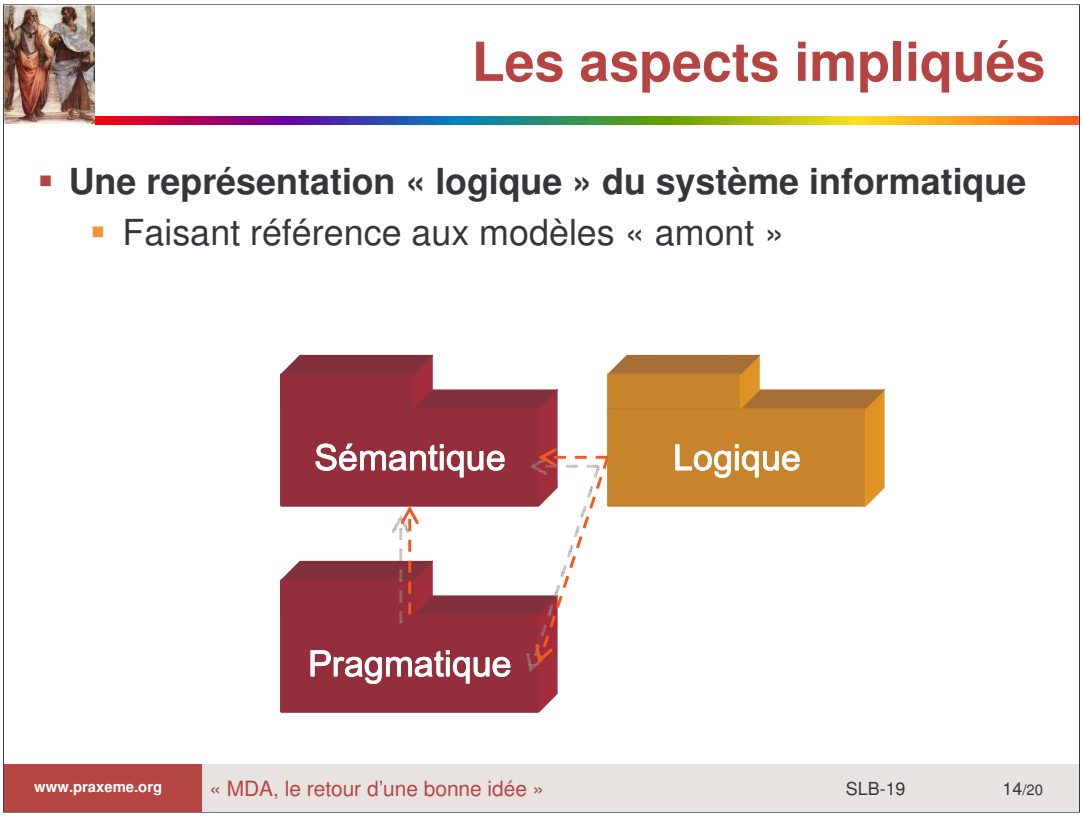



La Topologie du Système Entreprise ordonne les éléments d'information nécessaires à la description de l'entreprise et de son système d'information.

Pour aller jusqu'au bout de l'idée MDA et en faire un outil pratique, nous établissons des règles qui permettent de convertir un élément d'un modèle amont en un ou plusieurs éléments d'un modèle aval, puis en code. C'est en cela que réside toute la force de MDA, fondée sur la technologie des profils UML.

Ce que Philippe DESFRAY, bien avant la standardisation sous l'appellation « *UML profiles* », nommait l'hypergénéricité, constitue la grande nouveauté. Combinée avec les lignes directrices de MDA, elle concrétise enfin le projet d'industrialisation du logiciel.

Cette révolution semble passer inaperçue ! C'est bon signe : cela révèle un changement plus profond que les effets de mode. Une diffusion lente et discrète dans les pratiques professionnelles vaut bien mieux que le battage superficiel auquel nous sommes habitués. Le changement de paradigme, au sens fort de ce terme, ne s'opère que lentement, même s'il est marqué par quelques rares ruptures tonitruantes.



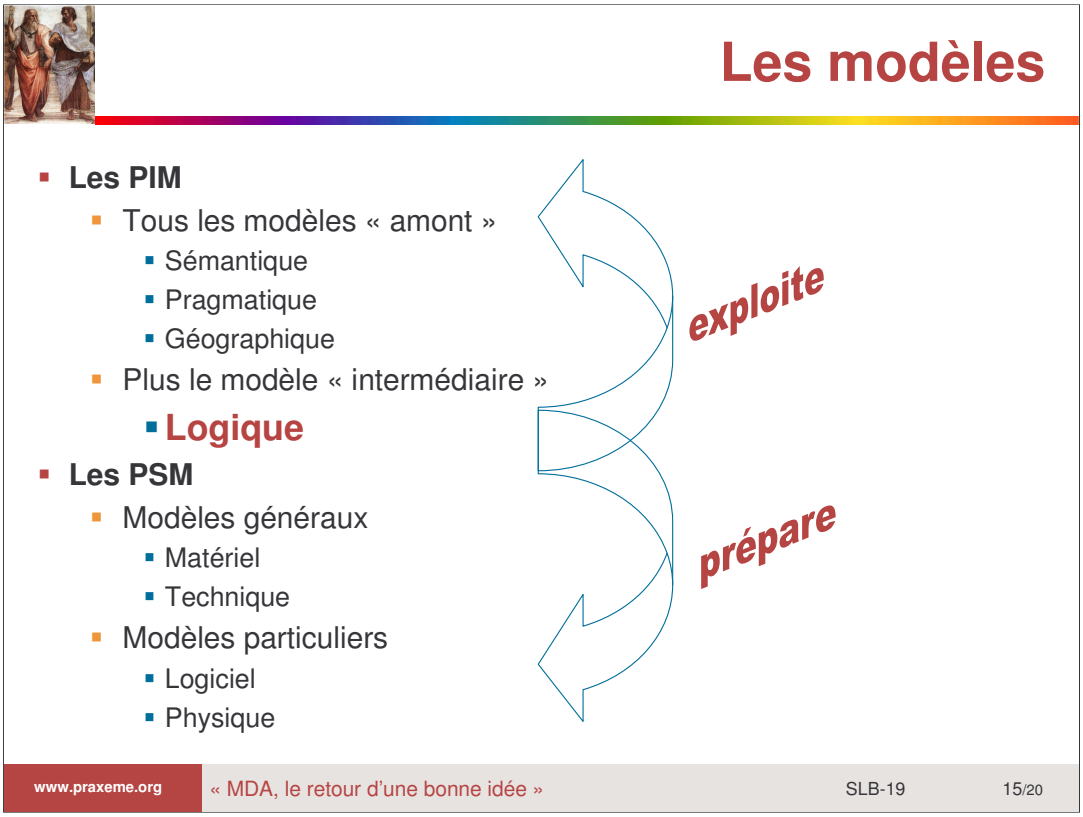

Pour illustrer la dérivation, nous ne nous intéresserons, dans cette conférence, qu'à deux des articulations fixées par la Topologie du Système Entreprise :

1. du modèle sémantique au modèle logique ;
2. du modèle pragmatique au modèle logique.

Rappel

- Le modèle sémantique décrit les fondamentaux du métiers. Ses principaux moyens d'expression sont les classes et les automates à états. Il incorpore non seulement la description des informations, mais aussi celles des actions et transformations que subissent les objets. Un modèle sémantique ne peut être considéré comme terminé que si les règles de gestion qui contraignent les objets sont encapsulées ou, au moins, localisées. Ceci, d'ailleurs, est un bon moyen de dégager des opérations à valeur sémantique.
- Le modèle pragmatique décrit la façon selon laquelle les acteurs de l'entreprise manipulent les objets « métier ». On y trouve les choix d'organisation, les cas d'utilisation, les processus…

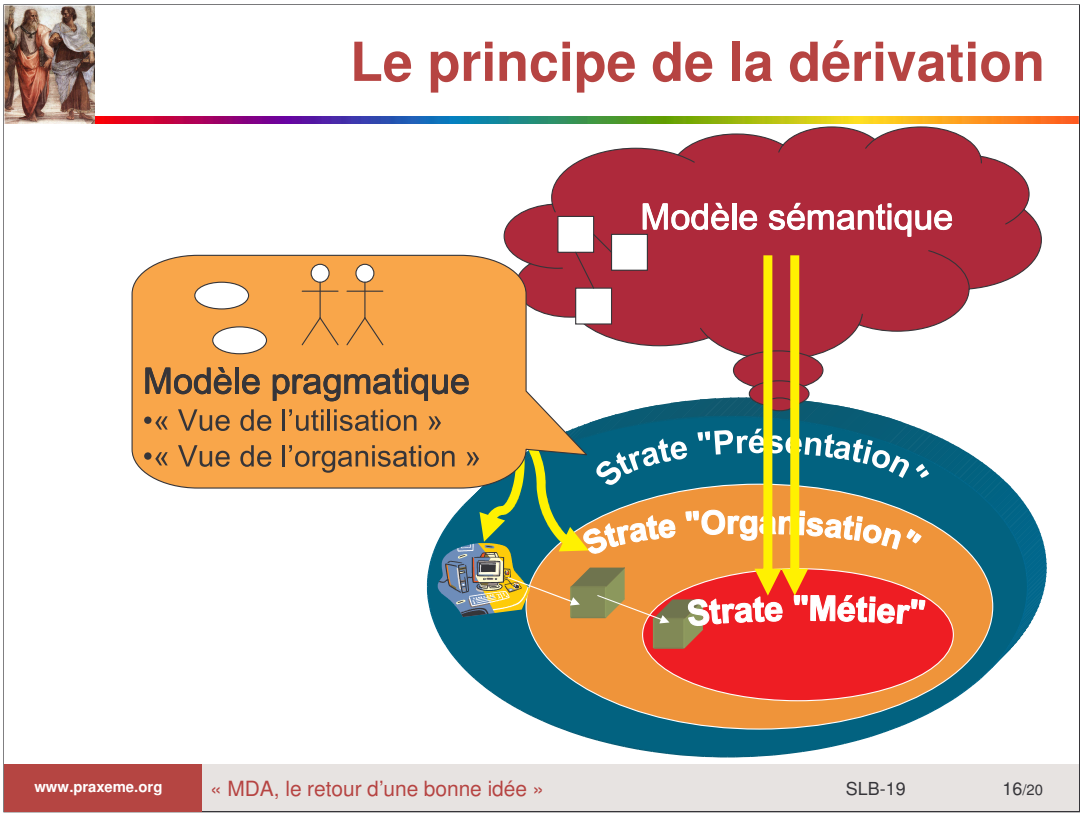
Pour aller plus loin : cf. les supports de formation dans le corpus Praxeme.




Praxeme, appuyé sur les principes fixés par MDA, va au-delà du standard en définissant les modèles.

L'aspect logique joue un rôle d'intermédiaire entre les aspects purement « métier » et l'informatique.

Le fait qu'il soit indépendant des choix techniques a des implications considérables. En effet, cette indépendance le rend stable : on peut l'exploiter et l'enrichir sur le long terme. Il a donc la durée de vie nécessaire à l'urbanisation du SI et aux politiques de convergence et de réutilisation.




L'approche SOA soulève une question pratique toute simple : comment trouver les services ?

Il est vain d'espérer construire une architecture de services, « par le bas », c'est-à-dire en pensant en termes logiciels.

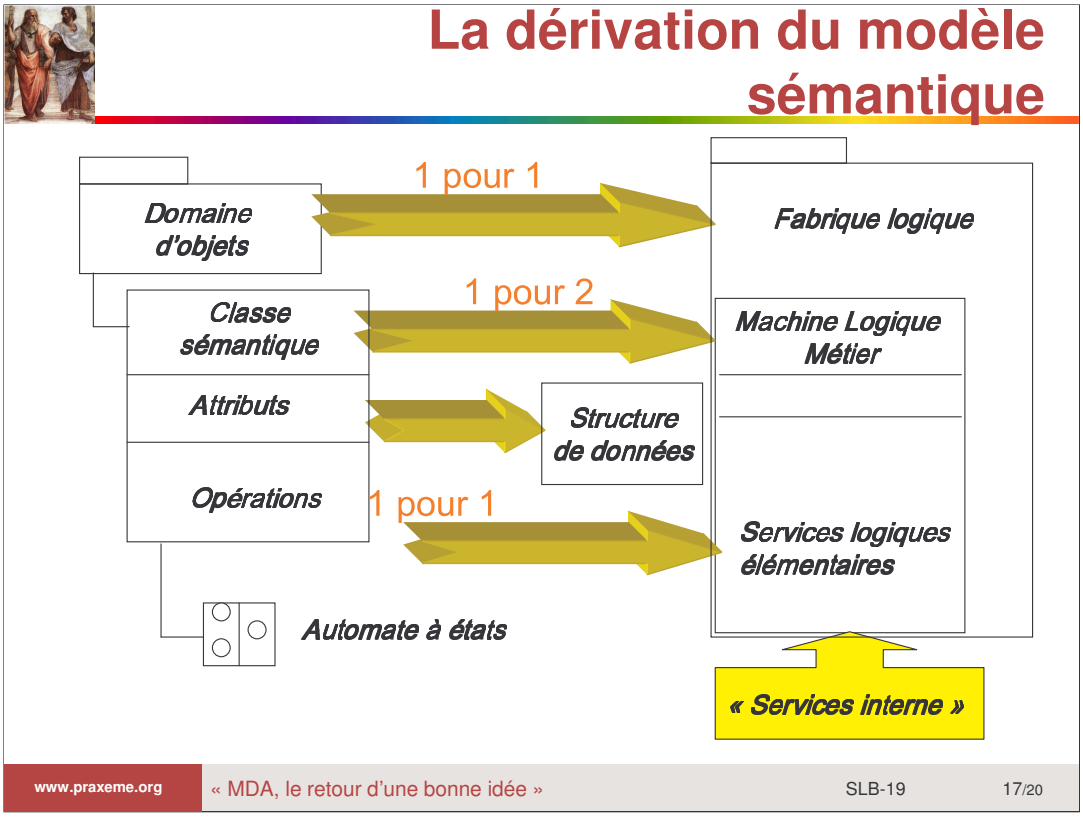
Dans le cadre proposé par Praxeme, la réponse se formule en termes de dérivation. Le modèle logique s'obtient « mécaniquement » en dérivant le modèle sémantique et le modèle pragmatique.

De ce fait :

•Le système d'information – au moins sa représentation logique – est stratifié.

•Les services sont découverts en référence au métier et possèdent un fort contenu fonctionnel.

Les pages suivantes, extraites des supports de formation, donnent une idée des règles de dérivation pour une SOA.




À gauche : les catégories de représentation du modèle sémantique.

À droite : celle de l'aspect logique, dans un style SOA.

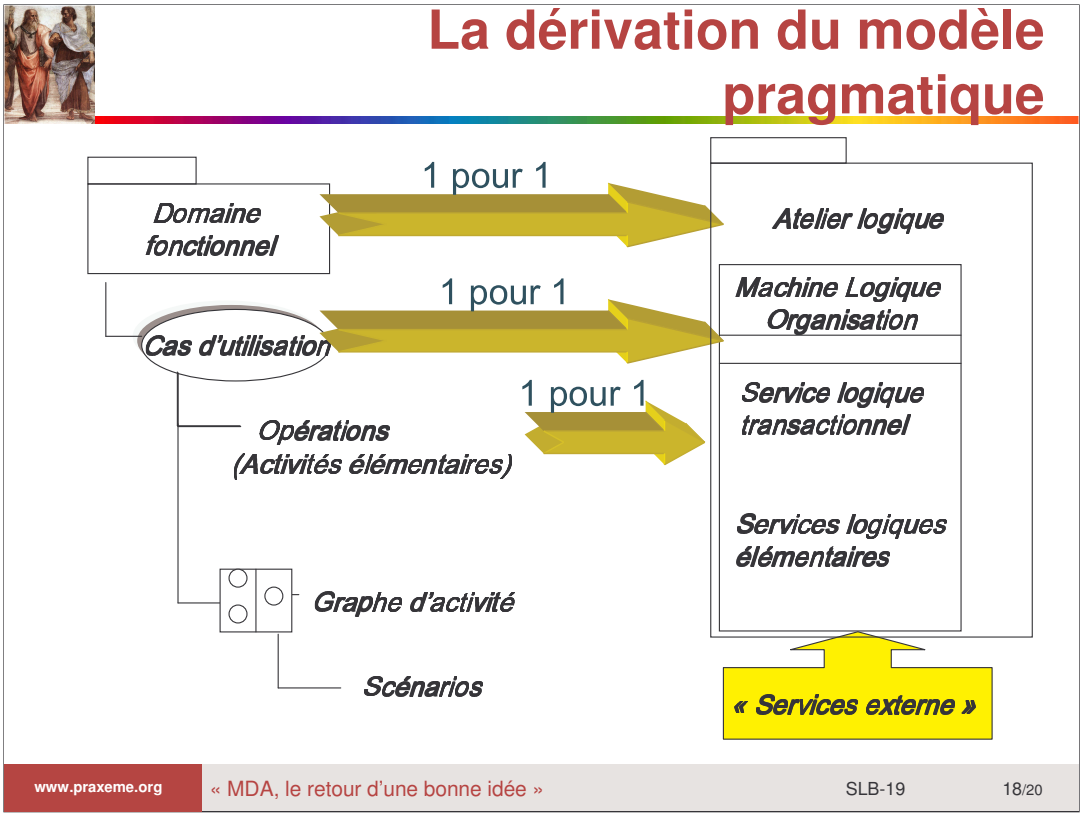
Praxeme prend le terme « service », au pied de la lettre, dans le respect de la métaphore. C'est pourquoi il correspond à l'opération d'UML. Le constituant logique qui offre les services est nommé « machine ». Les machines sont elles-mêmes rangées dans des ateliers (non représentés ici). Ces ateliers s'ordonnent en « fabriques logiques » qui correspondent aux domaines d'objets du modèle sémantique.

Les domaines d'objets diffèrent des domaines fonctionnels du modèle pragmatique. Autant le domaine fonctionnel résulte d'une décomposition d'activité et tend à reconduire l'organisation de l'entreprise, autant le domaine d'objets se constitue autour d'un objet central. L'introduction des domaines d'objets conduit à changer la physionomie des systèmes informatiques, en dégageant un noyau stable, composé de services hautement réutilisables.

La dérivation transporte aussi les automates à états.

Elle introduit de nouveaux éléments. Par exemple, la signature d'un service logique (opération stéréotypée « service ») obéit à des règles précises que la dérivation peut prendre en compte.

Le concepteur logique intervient pour compléter le résultat de la dérivation, notamment pour prendre en compte les décisions d'architecture logique. En effet, l'architecture logique est moins couplée que le modèle sémantique…




Il est important d'appliquer ces règles de dérivation sur un modèle des cas d'utilisation correctement structuré. Dans le cas contraire, les défauts de structure sont reportés dans le système informatique.

Or, la première version d'un modèle des cas d'utilisation (« Vue de l'utilisation » dans l'aspect pragmatique) exprime les pratiques existantes et révèle, très souvent, un taux de redondance important. Il faut considérer cette première version comme un acte d'analyse. Il convient de le faire suivre d'un effort de conception. La conception utilisera toutes les ressources du diagramme des cas d'utilisation pour évacuer la redondance. Sur cette deuxième version, bien structurée, nous pouvons appliquer les règles de dérivation qui vont peupler la strate « Organisation » de l'architecture logique.

À noter que, sur cette strate, la « fabrique logique » correspond à l'entreprise. Cette approche prévoit la construction de fédérations de systèmes, pour outiller les réseaux d'entreprises.

Le sens
de l'action

# Conclusion

- **Praxeme, une méthodologie d'entreprise**
  - Publique et ouverte
  - Fondée sur les standards
  - Soutenues par des acteurs publics et privés
- **Pour en savoir plus**
  - Le site de l'association *Praxeme Institute*
    - www.praxeme.org
  - **Le Symposium**
    - Le vendredi 14 décembre 2007
      - *Enterprise Architecture*, MOA, témoignages…

**MDA permet de réactiver notre héritage méthodologique**



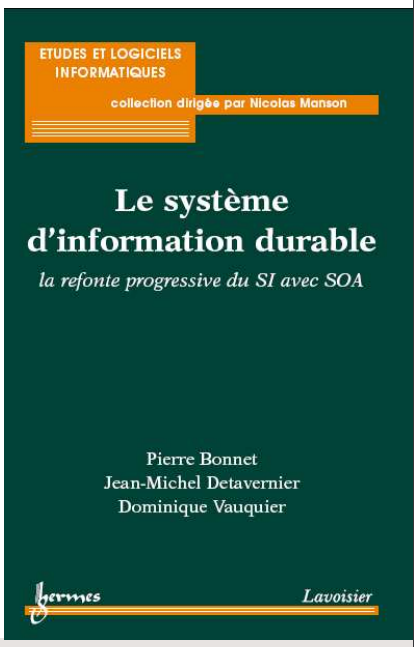
Le site publie les guides méthodologiques et une partie du corpus de la méthode, notamment, le « Guide de l'aspect logique » qui détaille les procédés pour SOA.

Les inscriptions au Symposium sont gratuites et se font en ligne sur le site.

Il est possible de s'inscrire également à une liste de diffusion qui permet de se tenir informé des principaux événements publics, liés à la méthode Praxeme.

## L'ouvrage

- **Cas concret d'une refonte de système d'information en SOA**
  - SMABTP
  - Application de Praxeme
    - Mise au point des procédés pour SOA
  - Outillage
    - Framework
    - Profil UML
    - Génération
      - Cf. conférence de Philippe DESFRAY




www.praxeme.org « MDA, le retour d'une bonne idée » SLB-19 20/20

Sorti en novembre 2007, cet ouvrage fait le bilan de l'application de Praxeme et de MDA, à grande échelle, sur le projet de refonte du SI de la SMABTP.

Co-écrit avec :

•Pierre Bonnet, d'Orchestra Networks, spécialiste de SOA et de MDM,

•Jean-Michel Detavernier, directeur informatique adjoint de la SMABTP.

Le projet a été outillé avec Objecteering. Des travaux se poursuivent pour automatiser les règles de dérivation.

La spécification des services logiques utilise un pseudo-code. Dans la perspective MDA, ce fait ouvre de nouvelles voies…

Voir aussi le site de la communauté S-IT-A :
**http://www.sustainableitarchitecture.com**