

Quels sont les meilleurs outils UML 2 pour les bases de données ?

Cet article confronte les 14 principaux logiciels du marché permettant de modéliser du niveau conceptuel à la génération de script SQL, une base de données à l'aide de la notation UML 2. Chaque logiciel est évalué selon différents critères relatifs à la qualité de traduction d'éléments d'un diagramme de classes (associations, classes-associations, agrégations, contraintes inter relations, héritage et rétro-conception d'une base de données).

Sommaire

Introduction	1
Associations binaires	3
Associations <i>n</i> -aires	5
Classes-associations	7
Contraintes	9
Agrégations	11
Héritage	13
La rétro-conception	15
Le classement	16
Quelques mots sur les outils	17
Conclusion	30
Produits	31
Pour en savoir plus	31

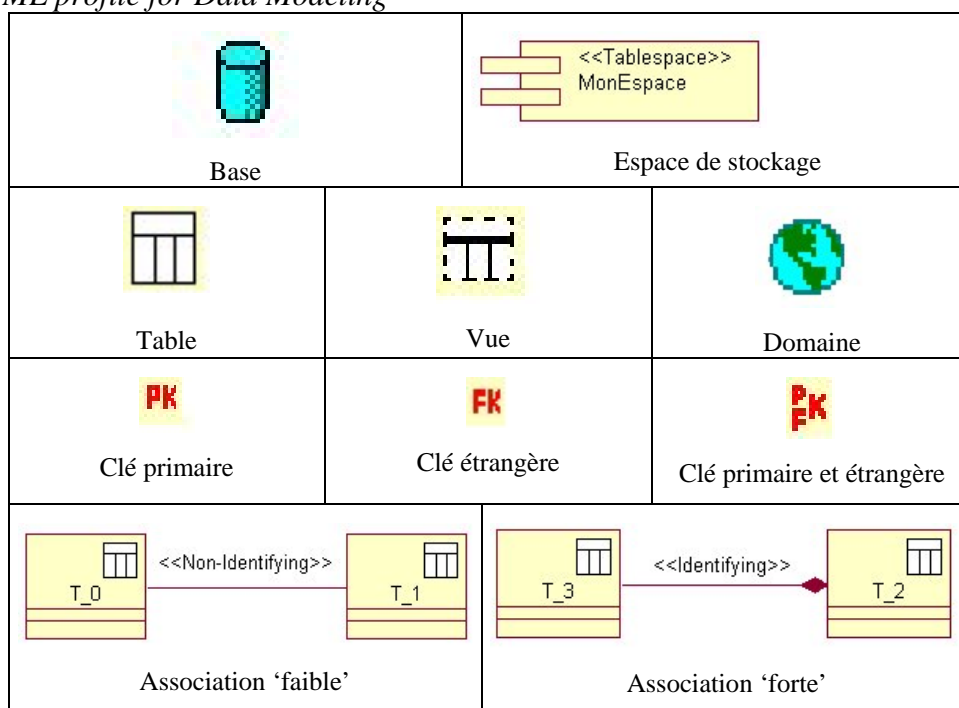
Introduction

Suivant l'approche MDA¹ (*Model Driven Architecture*), les objectifs des outils sont de permettre de modéliser un système dans un modèle PIM (*Platform Independent Model*) en utilisant un langage de spécifications approprié, puis de pouvoir traduire dans un ou plusieurs PSM (*Platform Specific Model*) réalisant ainsi l'implémentation concrète du système. Quand le domaine d'implémentation se cantonne aux bases de données, UML peut être utilisé comme modèle PIM, alors que SQL le sera au niveau du PSM.

Depuis plus de 30 ans, la conception des bases de données était réalisée à l'aide du modèle entité-association. La notation UML qui s'impose dans bien des domaines de l'informatique s'adapte de mieux en mieux aux bases de données. Le premier pas a été réalisé par Rational Rose avec son profil² UML (*UML profile for Data Modeling*). Ce profil permet de décrire des bases de données notamment à l'aide de stéréotypes prédéfinis (Table, RelationalTable, View, etc.).

¹ démarche de réalisation de logiciel, proposée et soutenue par l'OMG

² Un profil, proposition d'une communauté, regroupe un ensemble d'éléments (composants, stéréotypes, icônes, propriétés...) qui s'appliquent à un contexte particulier tout en conservant le métamodèle d'UML intact.

Fig. 1 UML profile for Data Modeling

La notation UML n'a pas été initialement prévue pour les bases de données offre toutefois, dans sa version 2, une panoplie d'éléments nécessaires à la modélisation (classes, attributs, associations, contraintes, stéréotypes, etc.). Seul le diagramme de classes est concerné dans lequel tous les concepts peuvent être employés à modéliser une base tout en respectant la normalisation. Ce diagramme de classes pourra être manipulé soit au niveau conceptuel (les anciens concepteurs devront faire l'analogie avec le MCD de Merise) ou au niveau logique (manipulation des relations, clés primaires et étrangères au profil UML).

La partie consacrée aux bases de données n'est pas prépondérante pour la majorité des outils. D'autres fonctionnalités sont offertes en ce qui concerne la modélisation de processus métier BPM (*Business Process Models*) qui peuvent être importés ou exportés conformément au langage BPEL4WS (*Business Process Execution Language for Web Services*). Bon nombre d'entre eux fournissent un référentiel pour le contrôle des données métiers utiles aux architectes des systèmes d'information qui en seront les principaux utilisateurs. Les plus récents s'inscrivent davantage dans l'architecture MDA, incluent le standard QVT (*Query View Transformation*) pour la transformation de modèles.

La majorité des outils proposent un processus de conception basé sur les différents niveaux du conceptuel au physique (forward engineering), un processus de rétroconception (reverse engineering) et un processus d'interéchange (round-trip engineering) entre chaque modèle. Les outils permettent également aussi la génération de code en différents langages (SQL, PowerBuilder, C++, C#, Java, XML, IDL-CORBA, Visual Basic, IHM ou site Web).

Seuls Together et MyEclipse sont totalement intégrés à Eclipse (Objecteering et MagicDraw proposent toutefois un plug-in).

Chaque logiciel sera évalué selon la qualité d'implémentation de différents diagrammes de classes incluant les critères suivants :

- associations binaires et n-aires, classes-associations et agrégations ;
- contraintes (partition, inclusion et relatives à l'héritage) ;
- héritage (décomposition au niveau logique et héritage multiple) ;
- rétro-conception d'une base de données.

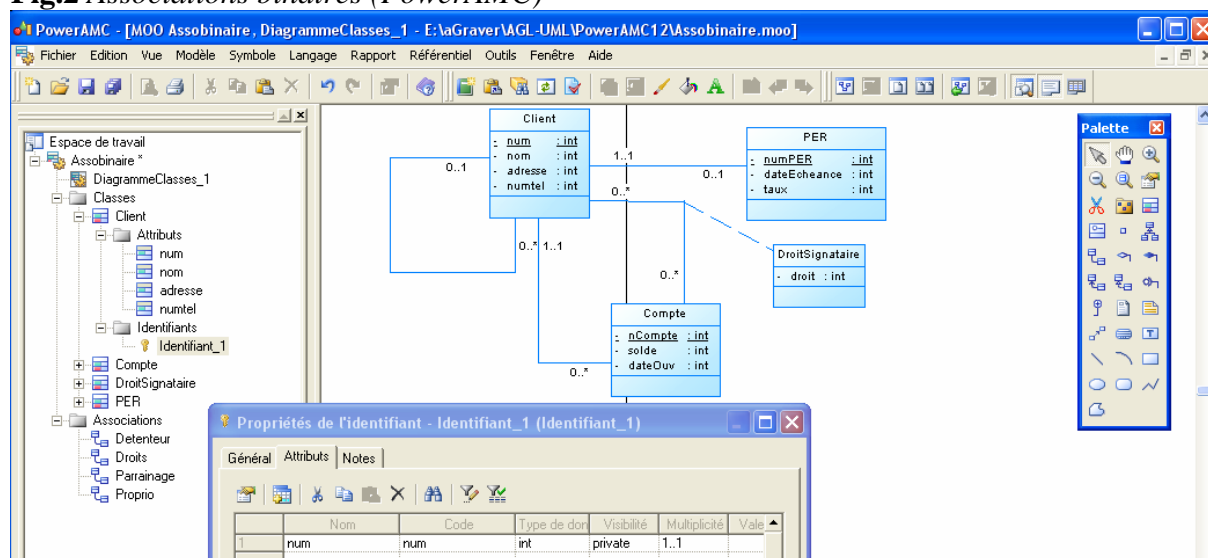
Pour chaque critère, un tableau illustre la mise en oeuvre des outils avec une signalisation à trois états (satisfaisant : feu vert, moyen : feu orange, insatisfaisante et absente : feu rouge). Certaines particularités intéressantes seront illustrées par des copies d'écran au fil des exemples. Le classement final qui prend également en compte la robustesse et l'ergonomie de chaque logiciel n'engage que moi naturellement.

Chaque logiciel sera évalué selon sa qualité à traduire différents éléments d'un diagramme de classes (associations, classes-associations, agrégations, contraintes inter relations, héritage et rétro-conception d'une base de données). Pour chaque critère, un tableau résume le résultat de chaque outil (feu vert : satisfaisant, feu orange : moyen, feu rouge insatisfaisant ou fonctionnalité absente). La notation finale prend également en compte la robustesse et l'ergonomie de l'outil. Sont exclus de ce comparatif, les outils qui n'utilisent pas UML au niveau conceptuel, citons DB Designer, Database Design Studio, DeZign, AllFusion ERWin, xCase, CASE Studio et ER/Studio.

Associations binaires

La figure 2 décrit une modélisation UML de comptes bancaires qui sont la propriété de clients. Un client peut parrainer d'autres clients. Un client n'a droit qu'à souscrire à un seul plan d'épargne retraite. Enfin, un client peut être désigné comme signataire de comptes ne lui appartenant pas (un seul droit lui est alors affecté). Au niveau conceptuel, les outils sont évalués sur les moyens mis en oeuvre pour représenter une classe (avec ses attributs et son identifiant), une association binaire ou réflexive (nommage, désignation des rôles et des multiplicités).

Fig.2 Associations binaires (PowerAMC)







































Au niveau logique, les outils sont évalués sur leur capacité à générer un schéma relationnel correct et sur la possibilité de définir d'éventuelles contraintes de répercussion de clés étrangères (CASCADE) pour la suite de processus de conception.







Fig.3 Modèle relationnel attendu (associations binaires)

```
Client[num, nom, adresse, numtel, numParrain#]
PlanEpargneRetraite[numPER, dateEcheance, taux, num#]
DroitSignataire[num#, nCompte#, droit]
Compte[nCompte, solde, dateOuv, num#]
```

Au niveau de SQL, les scripts sont évalués sur la dénomination des contraintes (utilisation du nom des associations ou de celui des rôles) et la position de la clé étrangère de l'association *un-à-un* (qui devrait, en théorie, se situer dans la table PlanEpargneRetraite).

La majorité des outils permettent une saisie correcte au niveau conceptuel. ModelSphere et MyEclipse pêchent par manque de fonctionnalités tandis que Visual Paradigm nécessite une manipulation complexe inter-modèles. Au niveau logique, des différences d'implémentation (ou des insuffisances) apparaissent. Seuls Win'Design, Rational Rose et Objectteering font un sans faute. Un bémol pour MagicDraw, Visual Paradigm, Together et Enterprise Architect qui ne disposent pas du mécanisme d'identifiant de classe. A noter que Together représente une classe-association avec le même symbole que celui de l'association *n-aire*.

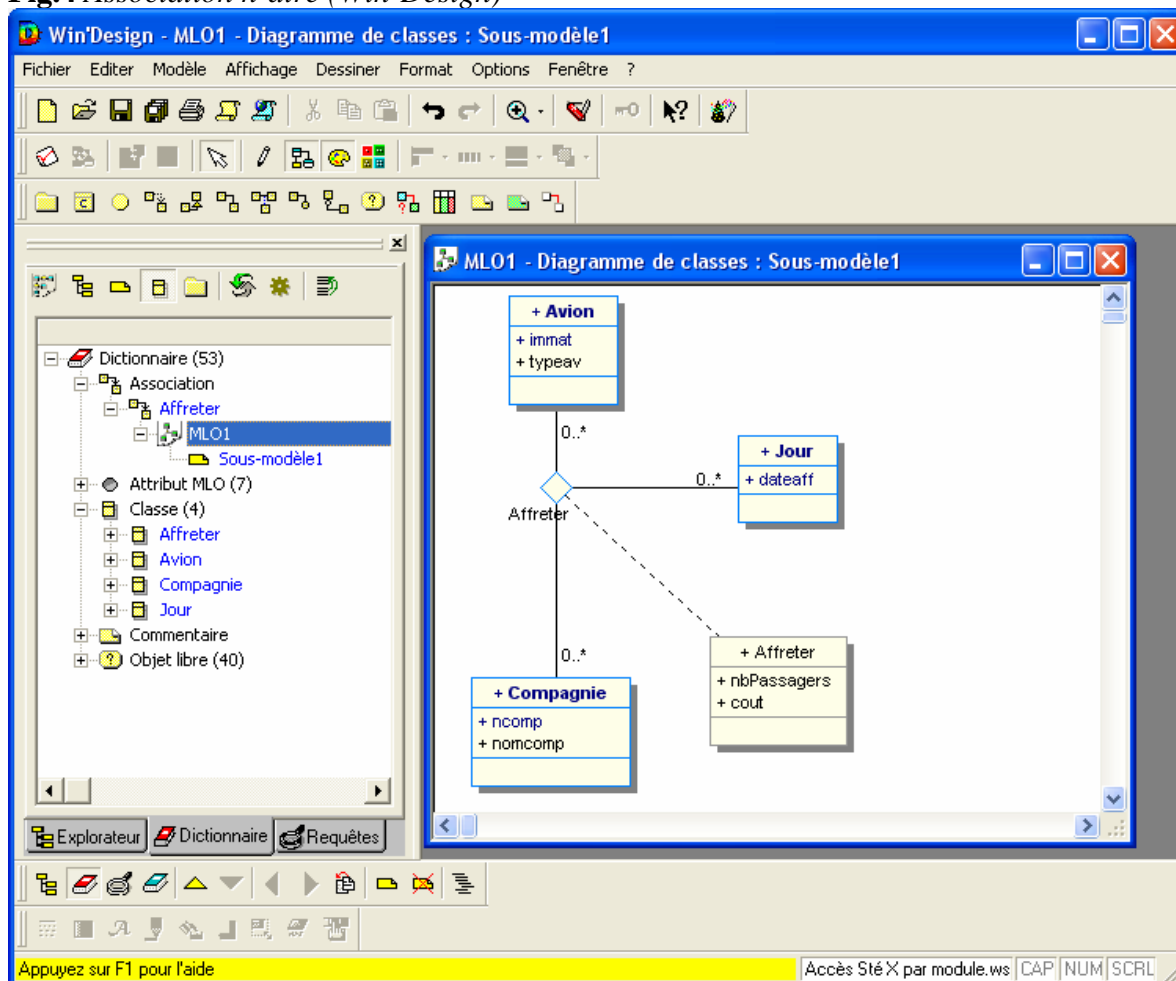
Associations binaires	Niveau conceptuel avec UML	Modèle logique	Code SQL (contraintes de clés étrangères)
Enterprise Architect			
MagicDraw			
MEGA Designer			
ModelSphere			
MyEclipse			
Objectteering			
Poseidon			
PowerAMC			
Rational Rose Data Modeler			
Together			
Visio			
Visual Paradigm			

Visual UML			
Win'Design			

Associations n -aires

La figure 4 décrit une modélisation UML des affrètements d'avions par différentes compagnies au cours du temps. Au niveau conceptuel, les outils sont évalués sur la possibilité de représenter une association n -aire à l'aide du symbole losange (*diamond*), de nommer l'association et d'y apposer différentes multiplicités maximales (1 ou plus généralement *).

Fig.4 Association n -aire (Win'Design)



Au niveau logique, les outils sont évalués sur leur capacité à transformer une association n -aire selon la valeur des multiplicités (en particulier pour 0..1 et 1, le degré de la clé primaire de la relation dérivée doit être réduit).

Fig.5 Modèle relationnel attendu (association n -aire)

Avion[immat, typeAv]







Compagnie[comp, nomComp]



Affretement[immat#, comp#, dateAff#, nbPassagers, cout]

Jour[dateAff]

Au niveau de SQL, les outils sont évalués sur la possibilité de choisir de ne pas traduire une relation en table (dans notre exemple, il n'est pas souhaitable de transformer la relation Jour en une table) et la traduction d'éventuelles multiplicités 0..1 et 1 (clé primaire réduite ou contrainte UNIQUE).

Seuls cinq outils sont vraiment à l'aise au niveau conceptuel avec le concept d'association *n*-aire. Le diagramme test ajoutait à la difficulté le fait de connecter une classe-association. Le grand gagnant de cet exercice est Win'Design qui maîtrise le processus de bout en bout.

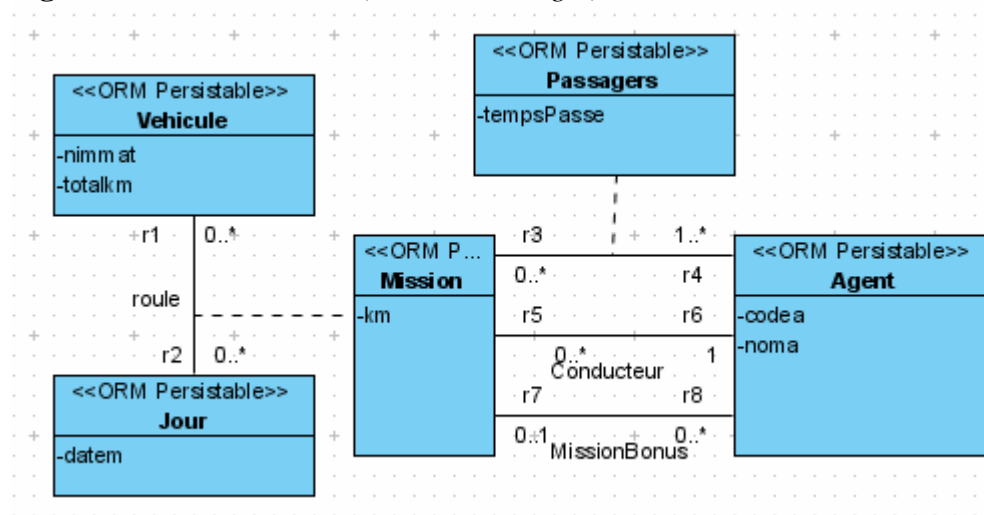
Associations <i>n</i> -aires	Niveau conceptuel avec UML (symbole losange, multiplicités)	Modèle logique	Code SQL
Enterprise Architect			
MagicDraw			
MEGA Designer			
ModelSphere			
MyEclipse			
Objectteering			
Poseidon			
PowerAMC			
Rational Rose Data Modeler			
Together			
Visio			
Visual Paradigm			
Visual UML			

Win'Design			
------------	---	---	---

Classes-associations

La figure 6 regroupe sur une modélisation UML des missions qu'un employé peut être amené à suivre (en tant que passager ou conducteur). Un employé pourra choisir une mission en particulier pour calculer un bonus par exemple. Au niveau conceptuel, les outils sont évalués sur la capacité de représenter toutes les multiplicités et plusieurs liens sur la même classe-association.

Fig.6 Classes-associations (Visual Paradigm)



Au niveau logique, les outils sont évalués sur la capacité de générer correctement un schéma relationnel en fonction des multiplicités des classes-associations.
































Fig.7 Modèle relationnel attendu (classes-associations)

```

Vehicule[nimmat, totalkm]
Agent[codea, noma, (nimmat, datem)#]
Mission[nimmat#, datem#, km, codea#]
Jour[datem]
Passagers[(nimmat, datem)#, codea#, tempsPasse]
  
```

Au niveau de SQL, les outils sont évalués sur la dénomination des clés étrangères (on devrait pouvoir retrouver le nom de certaines associations liées à la classe-association).

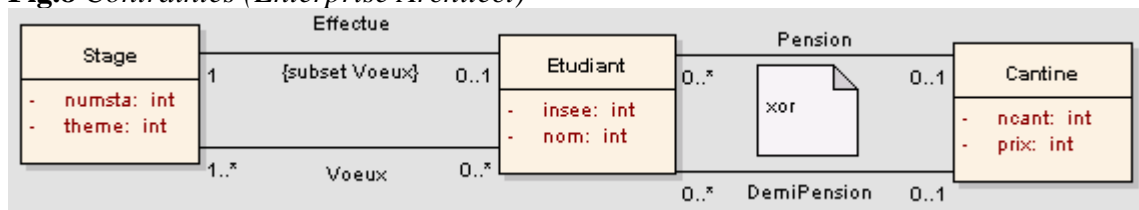
C'est au niveau logique, durant la phase de migration des clés étrangères, que les disparités se produisent. Seuls trois outils passent ce test haut la main, il s'agit de Objecteering, Rational Rose et PowerAMC.

Classes-associations	Niveau conceptuel avec UML	Modèle logique	Code SQL
Enterprise Architect			
MagicDraw			
MEGA Designer			
ModelSphere			
MyEclipse			
Objectteering			
Poseidon			
PowerAMC			
Rational Rose Data Modeler			
Together			
Visio			
Visual Paradigm			
Visual UML			
Win'Design			

Contraintes

La figure 8 illustre deux contraintes UML 2 (partition : xor et inclusion : subsets). La partition exprime qu'un étudiant est soit pensionnaire soit demi-pensionnaire, l'inclusion signifie que le stage d'un étudiant doit être au préalable souhaité. Les outils sont évalués sur la capacité de représenter ces contraintes (stéréotypes prédéfinis, menu contextuel, etc.). Dans la majorité des cas, la contrainte doit être saisie manuellement ou le stéréotype doit être créé pour pouvoir être réutilisé par la suite. Peu d'outils proposent des stéréotypes prédéfinis.

Fig.8 Contraintes (*Enterprise Architect*)



Ces deux contraintes n'ont pas d'influence sur la structure du modèle relationnel (idéalement, une clé étrangère devrait être toutefois ajoutée entre Stage et Voeux pour assurer l'inclusion).

Fig.9 Modèle relationnel attendu (contraintes)

```














Stage[numsta, theme, ninsee#]
      |
      v
Voeux[numsta#, ninsee#]

Etudiant[ninsee, nom, ncantineP#, ncantineDP#]

Cantine[ncantine, prix]
  
```

Au niveau de SQL, les outils sont évalués sur la génération de directives de vérification (CHECK) pour implémenter la contrainte de partition (non nullité exclusive des clés étrangères liant Etudiant à Cantine). Cette dernière contrainte pourrait aussi être assurée par un déclencheur (qui pourrait être déclaré automatiquement mais qu'il faudrait par la suite programmer manuellement).

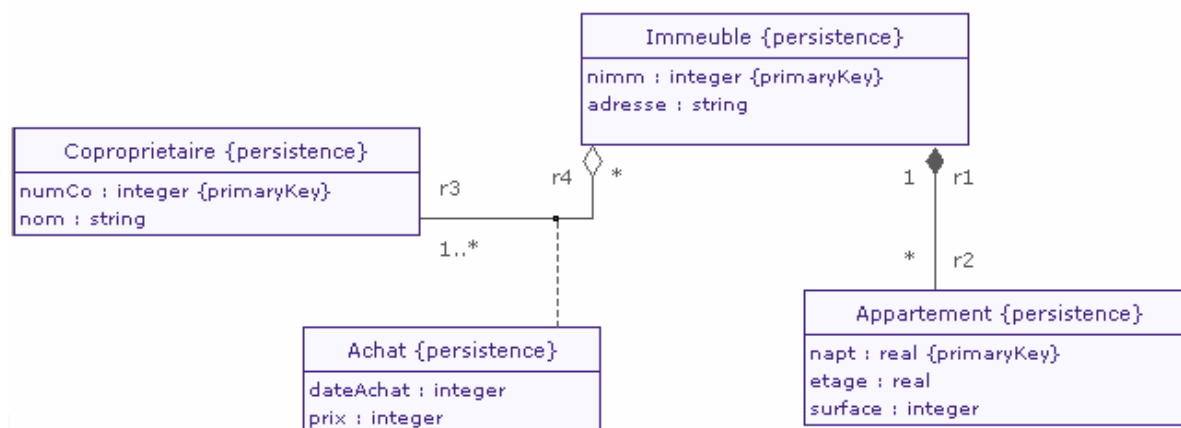
Bien que certaines contraintes prédéfinies de UML 2 soient déjà répertoriées par certains outils, la traduction du conceptuel au physique n'est pas encore automatisé. Il sera donc de votre charge de programmer explicitement au niveau du code SQL l'enrichissement sémantique de vos diagrammes de classes.

Contraintes	Niveau conceptuel avec UML (partition et inclusion)	Code SQL (clé étrangère, contrainte CHECK ou déclencheur)
Enterprise Architect		
MagicDraw		
MEGA Designer		
ModelSphere		
MyEclipse		
Objectteering		
Poseidon		
PowerAMC		
Rational Rose Data Modeler		
Together		
Visio		
Visual Paradigm		
Visual UML		
Win'Design		

Agrégations

La figure 10 décrit une composition qui exprime qu'un appartement est un composant d'un immeuble et une agrégation partagée qui renforce l'association entre un copropriétaire et un immeuble. Au niveau conceptuel, les outils sont évalués sur la capacité de représenter ces deux formes d'agrégation.

Fig.10 Agrégations (*Objectteering*)



Au niveau logique, les outils sont évalués sur la capacité de générer un schéma relationnel traduisant correctement la composition (clé primaire de la table composant enrichie par la clé primaire de la table composite).


































Fig.11 Modèle relationnel attendu (*agrégations*)

```

Coproprietaire[numco, nom]
Immeuble[nimm, adresse]
Achat[nimm#, numco#, prix, dateAchat]
Appartement[nimm#, napt, surface, etage]
  
```

Les scripts SQL sont évalués sur la capacité d'inclure la directive CASCADE au niveau de la clé étrangère de la tables composites et de celle liée à l'agrégation partagée (ici Achat et Appartement)

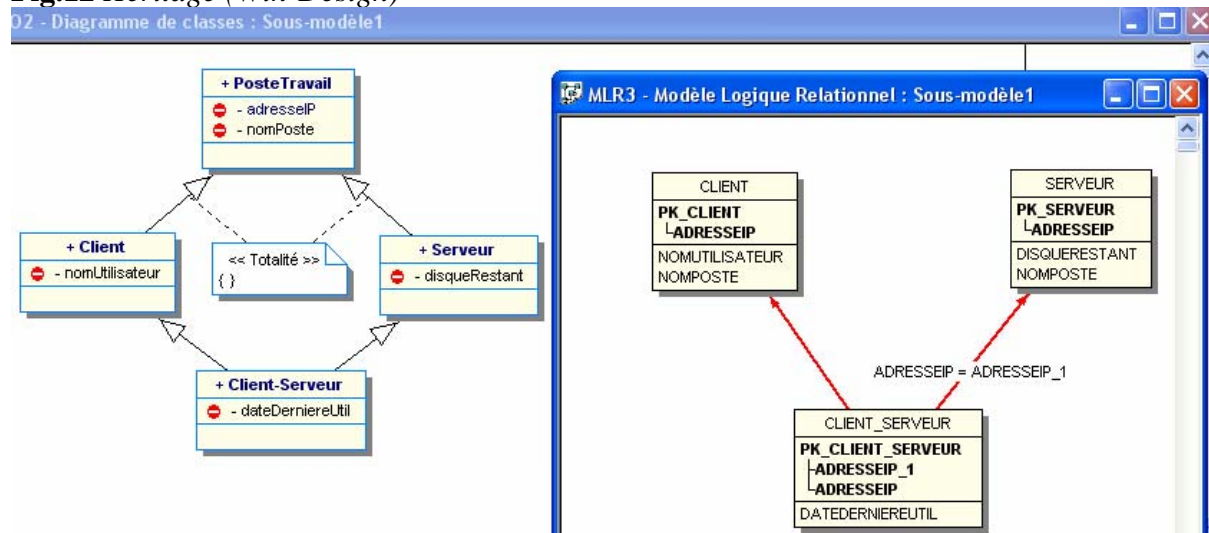
C'est encore une fois au niveau logique, durant la phase de transformation des agrégations (dont une couplée à une classe-association), que les disparités apparaissent. Seuls trois outils réussissent à construire un modèle logique correct il s'agit de Objectteering, PowerAMC et Win'Design. Rational Rose et Visio ne proposent qu'une seule forme d'agrégation. En revanche, aucun outil ne génère de directive CASCADE au niveau du code SQL.

Agrégations	Niveau conceptuel avec UML (symboles de l'agrégation partagée et de la composition)	Modèle logique (clé composite)	Code SQL (contraintes CASCADE)
Enterprise Architect			
MagicDraw			
MEGA Designer			
ModelSphere			
MyEclipse			
Objectteering			
Poseidon			
PowerAMC			
Rational Rose Data Modeler			
Together			
Visio			
Visual Paradigm			
Visual UML			
Win'Design			

Héritage

La figure 12 décrit une hiérarchie à deux niveaux avec un héritage multiple. La contrainte de totalité exprime qu'il n'existe pas de poste de travail n'étant ni client ni serveur.

Fig.12 Héritage (Win'Design)



Au niveau logique, les outils sont évalués sur la capacité de proposer les trois cas de décompositions pour chaque niveau du graphe d'héritage. On devrait pouvoir choisir par exemple, la décomposition descendante pour le premier niveau et par distinction pour le second niveau.

Fig.13 Modèle relationnel attendu (héritage)










Client[adresseIP, nomPoste, nomUtilisateur]

Serveur[adresseIP, nomPoste, disqueRestant]

Client-Serveur[adresseIPC#, adresseIPS#, dateDerniereUtil]

Aucun outil ne génère actuellement du code SQL implémentant des contraintes d'héritage (idéalement il faudrait générer soit un déclencheur, soit des directives CHECK en fonction du type de la contrainte).

Alors que MagicDraw et MEGA maîtrisent bien les contraintes UML 2 relatives à l'héritage, seuls PowerAMC et Win'Design (en passant par un MCD Merise) construisent un modèle logique correct en proposant les trois types de décomposition.

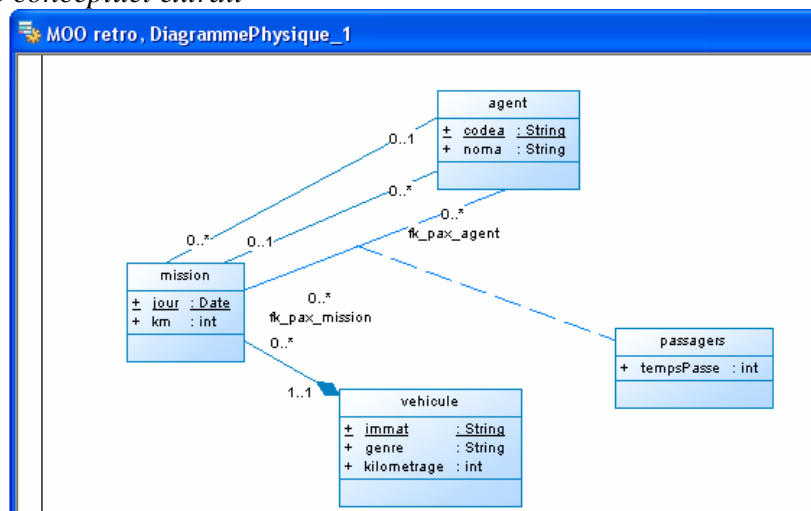
Héritage	Niveau conceptuel avec UML (contraintes et héritage multiple)	Modèle logique (3 cas de décomposition)
Enterprise Architect		
MagicDraw		
MEGA Designer		
ModelSphere		
MyEclipse		
Objectteering		
Poseidon		
PowerAMC		
Rational Rose Data Modeler		
Together		
Visio		
Visual Paradigm		
Visual UML		
Win'Design		

La rétro-conception











Les outils sont évalués sur la capacité à générer un diagramme UML à partir de la base de données MySQL suivante. Le processus devrait produire, en théorie, un diagramme contenant trois associations sur une classe-association.

Bon nombre de solutions implémentent une composition lors du processus de *reverse engineering*. Ce faisant, une mission est identifiée par un numéro de véhicule et un jour. Chaque mission pourra être connectée à un agent de trois manières distinctes.

Fig.14 Modèle conceptuel extrait



Seuls quatre outils sont capables de produire un diagramme UML correct suite à la rétro conception de la base. La majorité des outils permettent une connexion via ODBC ou JDBC. PowerAMC, Rational Rose, Together et Win'Design sont encore plus puissants car ils permettent d'analyser en plus d'une base, le script SQL de création des tables, index et contraintes.

Rétroconception	Sources de données (connexion base, script SQL)	Diagramme UML produit
Enterprise Architect		
MagicDraw		
MEGA Designer		
ModelSphere		
MyEclipse		

Objectteering		
Poseidon		
PowerAMC		
Rational Rose Data Modeler		
Together		
Visio		
Visual Paradigm		
Visual UML		
Win'Design		

Le classement

La note finale est déterminée à partir des résultats des tests précédents, de la robustesse, de l'ergonomie, de la documentation et du support éventuel que j'ai du solliciter.

BILAN	Note	Prix indicatif pour 1 licence	Version évaluée
PowerAMC	★★★★☆	2800 €	12
Win'Design	★★★★☆	1800 €	7
Rational Rose Data Modeler	★★★★☆	3500 €	7.0
Objectteering	★★★★☆	2000 €	6
MagicDraw	★★★★☆	4250 €	12.0
Enterprise Architect	★★★★☆	118 €	6.5
Visual Paradigm	★★★★☆	540 €	3
MEGA Designer	★★★★☆	NC	2005 SP3
Together	★★★★☆		2006 R2
Visual UML	★★★★☆	230 €	5.0
MyEclipse	★★★★☆	45 €/an	5.1
Visio	★★★★☆	630 €	2007
Poseidon	★★★★☆	700 €	5.x
ModelSphere	★★★★☆	1160 €/an	2.5

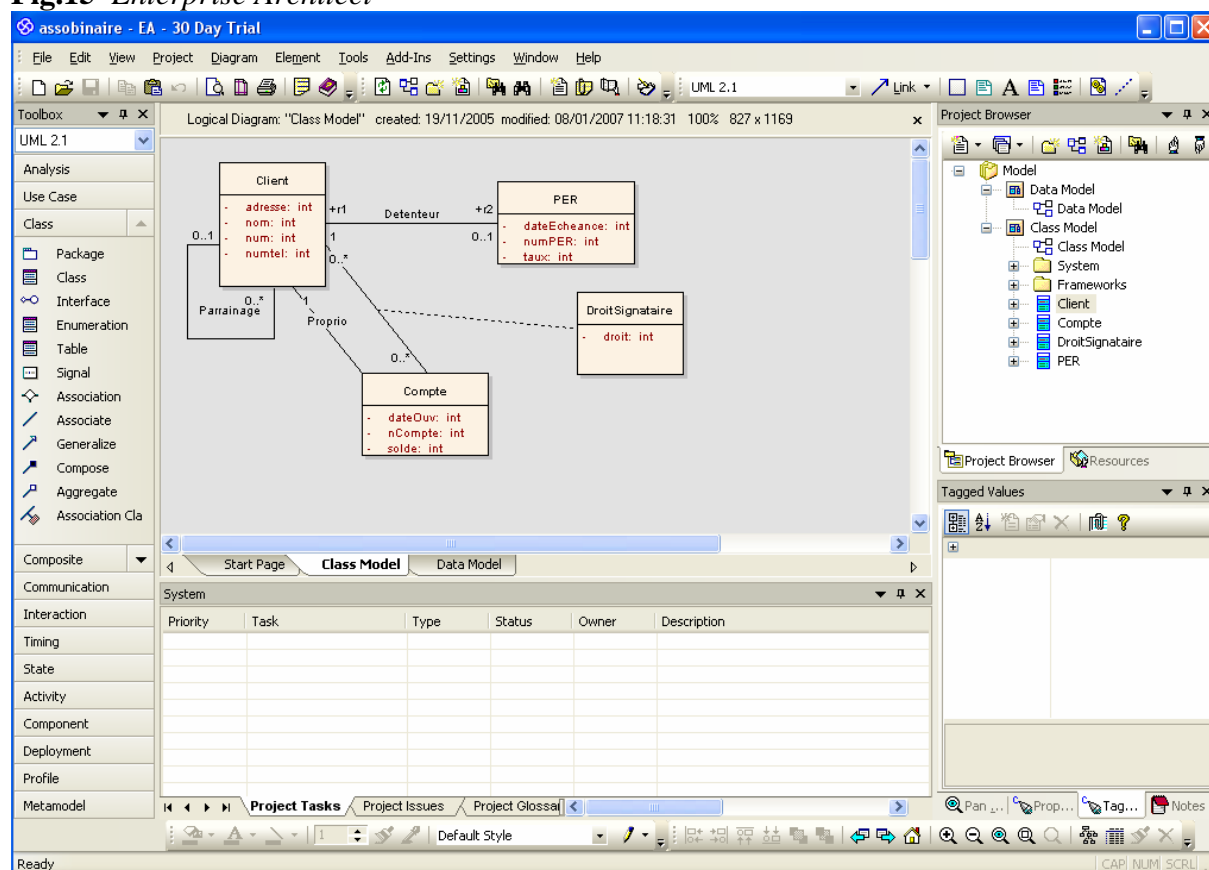
Quelques mots sur les outils

Si vous envisagez d'évaluer vous même un des outils, ces quelques remarques vous feront sans doute gagner du temps à la mise en œuvre.

Enterprise Architect

Ergonomie soignée et robustesse pour cet outil Australien, qui pêche malheureusement dans les transformations de modèles comme nous le verrons par la suite. Créez un nouveau projet puis choisissez les modules Class et Database. Sélectionner Data Architect dans la fenêtre Toolbox pour disposer des icônes nécessaires à la construction de diagrammes de classes. Ajouter un nouveau diagramme dans la fenêtre de droite. Il n'est pas possible de définir un attribut identifiant par classe. Pour créer une classe-association (clic droit sur la classe Advanced/Make Association Class...). Si vous supprimer des éléments pensez à vérifier dans la fenêtre du browser qu'ils sont bien effectivement ôtés du modèle.

Fig.15 Enterprise Architect



Pour générer un modèle logique exprimé dans le formalisme UML utilisant le profil pour les bases de données (celui initié par Rational Rose), il faut d'abord le transformer en paquetage (Project/Source Code Engineering/Generate Package Source Code...) puis Project/Model Transformation/Transform Current Package (choisir DDL et le répertoire cible).

Pour générer un modèle physique suivez l'assistant qui apparaît après avoir sélectionné : Project/Database Engineering/Generate Package DDL...

Dans les points forts de ce produit, citons la documentation de bonne qualité avec un lien direct vers le forum Internet dédié, la possibilité de personnaliser un grand nombre de paramètres et processus et la facilité de travailler avec différents paquetages au sein d'un même diagramme de classes. Les points faibles principaux sont l'absence de transformation d'un modèle logique en conceptuel et la faible qualité du mécanisme de transformation de

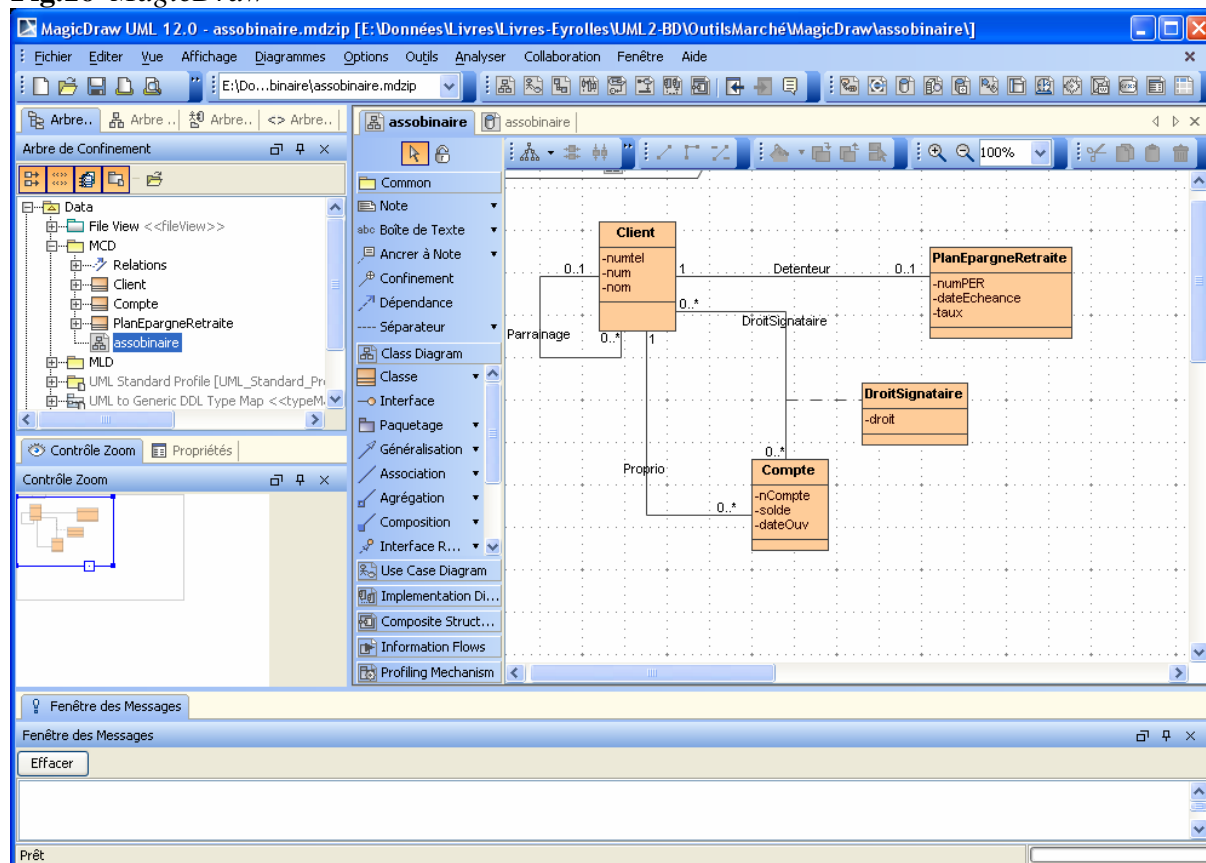
modèles (génération très basique des clés étrangères). En effet, seules les associations binaires sont correctement traduites. Pour les associations n -aires, les classes-association et les agrégations, le concepteur doit enrichir le processus de transformation (reprogrammation du *mapping*). Chose compliquée et hasardeuse. L'héritage ne propose qu'un seul cas de décomposition. La rétro conception est correcte, seul un schéma logique au formalisme UML est produit (pas de remontée d'un modèle conceptuel).

MagicDraw

Outil robuste à l'ergonomie ressemblant à celle de Visual Paradigm, MagicDraw se targue sur son site Web, par de nombreuses nominations, d'avoir été élu à plusieurs reprises meilleur outil de modélisation. Bien qu'il propose des assistants puissants de conversion, son mécanisme de transformation de classes-association n'est pas encore au point.

Créez un nouveau projet dans lequel vous installerez un nouveau paquetage contenant votre diagramme de classes. La saisie des éléments du diagramme de classes est intuitive mais il n'y a pas malheureusement de possibilité de définir d'identifiant de classe. Pensez à créer un autre paquetage qui contiendra le modèle logique (modèle relationnel exprimé au profil UML). Pour transformer les classes en relations, passer par Outils/Transformer qui lance un assistant.

Fig.16 MagicDraw



La génération du script SQL s'opère via le répertoire Jeux de Code d'Ingénierie (New/DDL...), créez une base puis faites glisser vos relations du paquetage du niveau logique. Ensuite, il suffit de sélectionner la base puis de lancer la génération.

Pour la rétro-conception, il faut créer une base dans ce même répertoire, puis Editor... lance un menu qui précise l'accès à la base (souvent une connexion JDBC). Ensuite, il suffit de lancer le processus par le menu Inverser. Le schéma relationnel obtenu par rétro-conception est au profil UML. Un assistant vous permet de le convertir en diagramme (conceptuel) de classes UML (menu Outils/Transformer/DDI to UML). Pour visualiser les associations,

positionnez-vous sur une des classes du schéma conceptuel (clic droit Eléments reliés/Afficher les éléments liés). Le diagramme risque fort d'être de qualité moyenne si votre base de données implémente des associations plusieurs-à-plusieurs ou d'agrégation que vous attendiez légitimement visualiser en tant que classes-associations.

Des bons points pour la possibilité de transformation d'une base à une autre, les contraintes prédéfinies d'un graphe d'héritage, le choix de pilotes SGBD et les démos thématiques animées qui sont téléchargeables sur le site de l'éditeur. Les limitations concernant UML sont l'absence d'identifiant de classe et la transformation des classes-associations (qui n'est pas du tout prise en compte). De plus, aucun cas de décomposition n'est prévu pour l'héritage, l'éditeur semble attendre la tendance du marché pour se décider à implémenter une solution.

MEGA Designer

Faisant partie des solutions relatives à l'architecture et gouvernance du système d'information de la société Française MEGA International, l'outil Designer se consacre aux différents modèles de données servant à la conception.

Fig.17 MEGA Designer



Pour débuter, créez une base de données (dans le répertoire Databases) dans laquelle vous définirez un nouveau modèle de données (clic droit New...). La création de ce modèle de données va entraîner automatiquement la création d'un paquetage (portant le nom de la base). Dans le répertoire Paquetages, sélectionner ce paquetage et créez un diagramme de classes (New/Generate a Class Diagram). Pour créer une classe-association, ne pas définir de classe mais cliquez sur l'association puis Association class... Comme PowerAMC et Win'Design, MEGA permet l'identification relative.

Afin de générer un modèle logique, positionnez-vous sur votre base dans le répertoire Databases et choisissez Editier puis Outils (synchroniser du conceptuel au logique). Un assistant en quatre étapes est alors lancé. Le script SQL est créé via l'option Générer.

Pour la rétro-conception, créez une base puis Reverse Database, sélectionner l'entrée ODBC. Il faudra définir un diagramme (New/Diagram/Relational Diagram) pour faire un glisser déposer des tables obtenues. Pour obtenir l'équivalent UML du modèle, il faut synchroniser

(clic droit sur la base puis Edit/Tools/Synchronisation...). Le schéma obtenu par rétro-conception peut être de qualité moyenne. A noter enfin l'existence d'un modèle de données hybride entre UML et le niveau logique relationnel appelé Data diagram.

La principale limitation concernant UML est l'absence d'association *n*-aire et la qualité de transformation des classes-associations et de l'héritage multiple. Par ailleurs, aucun pilote de SGBD *open source* n'est pris en compte dans le processus de conception. Les points forts sont la robustesse de son ergonomie, son outil de recherche d'éléments inter-objets (processus, paquetages, bases de données, etc.), son outil de synchronisation par étapes guidées par un assistant et les passerelles possibles avec PowerAMC et Rational Rose.

ModelSphere

Outil de la société québécoise Grandite, ModelSphere permet de créer des modèles conceptuels de type entité-association, des modèles relationnels (associés ou non à une base) selon différents formalismes et notamment UML. La partie UML de l'outil permet de créer un diagramme de classes. Le processus de transformation vers le niveau logique n'est pas finalisé notamment au niveau des identifiants et des clés étrangère car des modèles de liens (pas encore opérationnels) doivent être mis en oeuvre. Le niveau logique est donc composé de tables sans clés liées entre elles graphiquement.

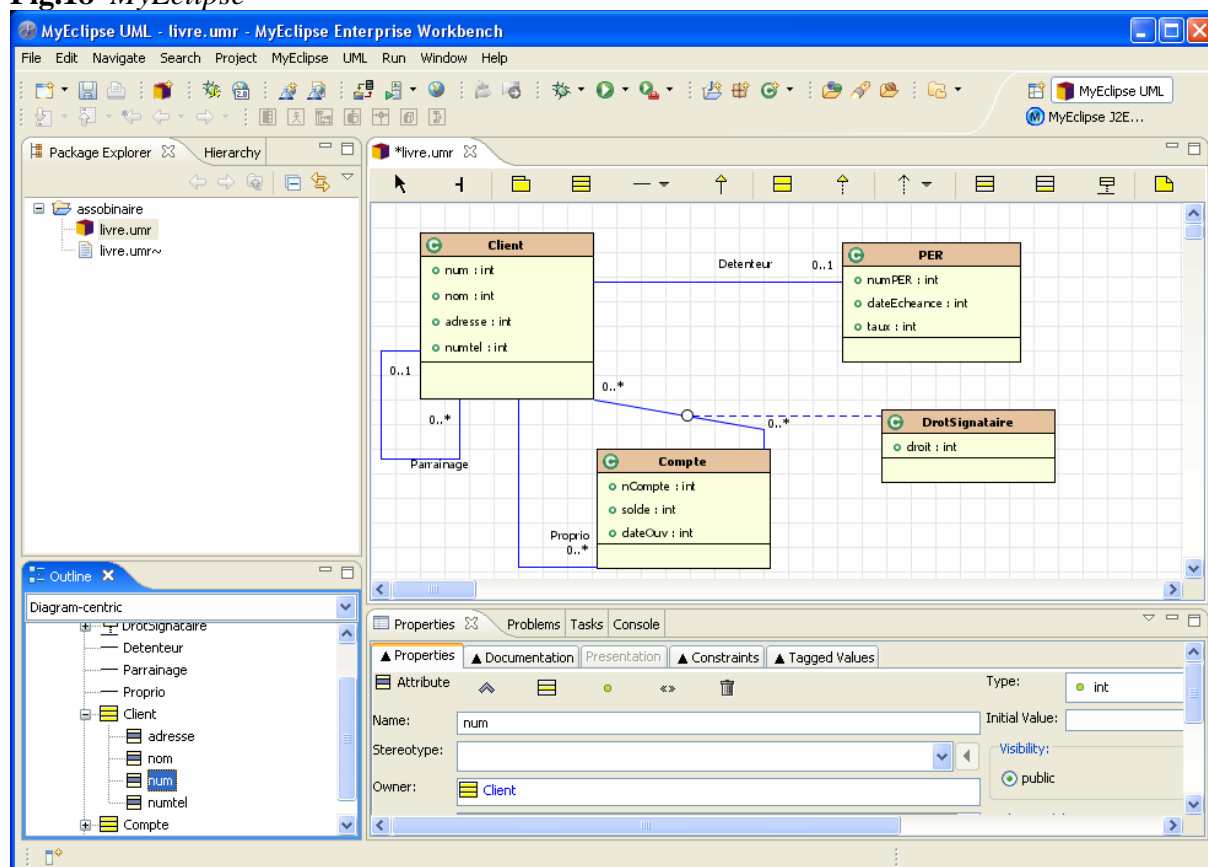
Pour générer un modèle logique, sélectionnez le modèle de classes (clic droit Générer Modèle de données...). Par la suite, positionnez-vous dans le répertoire contenant les tables générées et ajoutez un diagramme (clic droit Ajouter). Faites glisser les tables et les associations générées pour découvrir le schéma relationnel généré.

Les principales limitations concernant UML sont l'absence d'identifiant de classe, de notation graphique pour les classes-associations et les associations *n*-aires. Le processus de rétro-conception d'une base est correct (via ODBC ou JDBC) mais le schéma généré UML est de qualité moyenne puisque le processus traduit toute table en une classe. Pas de moyen pour l'heure de traduire un MCD en diagramme de classes et inversement.

MyEclipse

Comme Poseidon, cet outil n'est pas vraiment fait pour les bases de données (niveau logique inexistant de même que la génération de code SQL à partir d'un diagramme UML). L'ergonomie est toutefois mieux réussie que Poseidon. Cet outil oblige l'utilisateur à se servir du modèle Entity-Relationship pour concevoir ses schémas conceptuels.

Fig.18 MyEclipse



Les principales limitations côté UML sont l'absence d'identifiant de classe et de notation pour les associations n-aires. Le processus de rétro-conception d'une base est correct (grand choix de SGBD) mais le schéma généré est de type Entity-Relationship (notation *crow's foot*).

Pour vous faire une idée de ces possibilités Preferences/MyEclipse/Database Explorer, configurer votre accès à la base puis créez une connexion Window/Open Perspective/Other/MyEclipse/Database Explorer. Créez ensuite un diagramme (clic droit New ER Diagram), enfin sélectionnez les tables à rétro-concevoir.

Objecteering

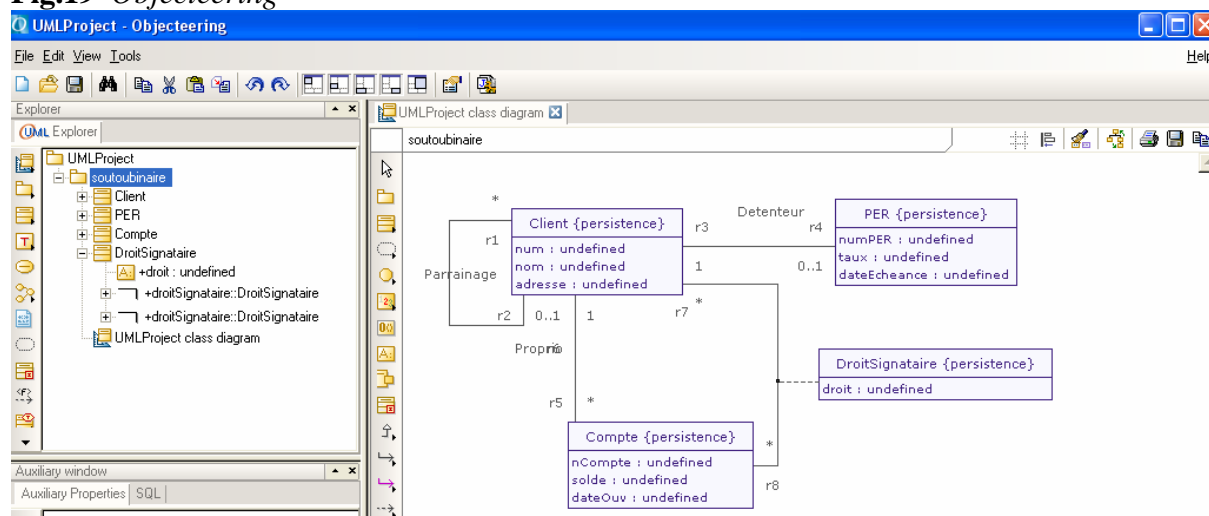
Objecteering est un outil français de la société éponyme, filiale de SOFTEAM qui a été acteur majeur dans la communauté des technologies objet et premier membre Européen de l'OMG. Dans ce logiciel, le terme physical model désigne un schéma relationnel (niveau logique) et logical model désigne un script SQL.

Avant tout, créez un projet puis configurez l'affichage des tags qui serviront à annoter un diagramme de classes pour sa transformation (Tools/Diagram graphic options/Properties Class Diagram, rendre visible les *tagged values*). Ensuite déployez le module SQL (Tools/Deploy an MDAC... choisir SQLDesigner). Concernant ce module (Tools/MDAC options...), configurez l'accès à votre base. Dans l'option Diagram generation... cochez la case Generates diagram...

Vous pouvez créer un paquetage (dans la racine) qui contiendra votre diagramme de classes. La transformation nécessite d'enrichir le diagramme UML de *tagged values*. Ainsi vous devrez annoter chaque classe du tag *persistence* (avec la propriété *persistent*). Les identifiants sont choisis via un clic droit sur la classe MDA Components/SQL Designer/Primary key...

Pendant la saisie du diagramme de classes servez vous plutôt de la fenêtre de gauche (UML Explorer) pour supprimer les éléments superflus.

Fig.19 Objectteering



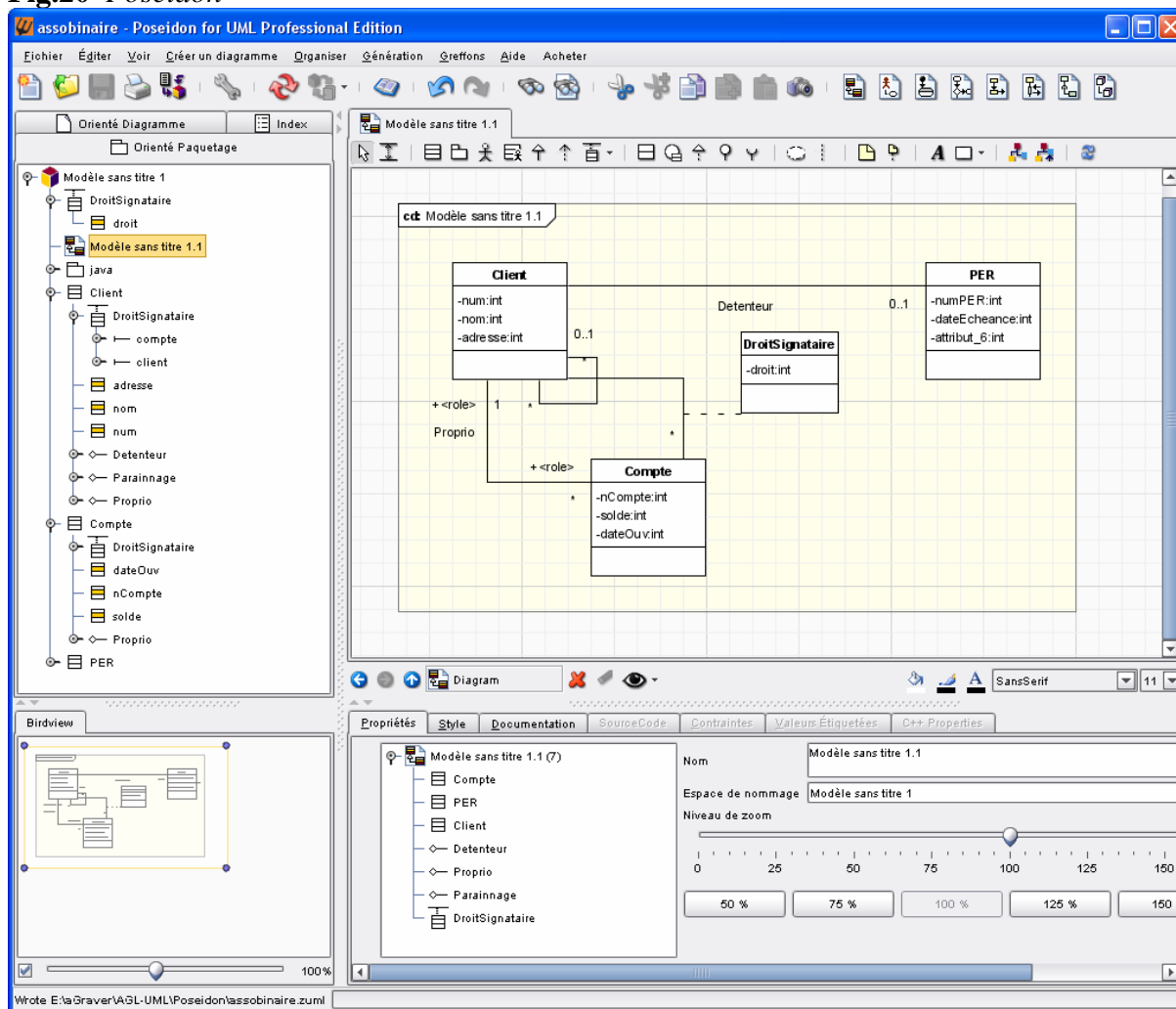
La génération d'un schéma relationnel se fait au niveau du paquetage contenant le diagramme UML (clic droit MDA Components/SQL Designer/Generate physical model). La génération d'un script SQL se fait au niveau du paquetage contenant le modèle physique (clic droit MDA Components/SQL Designer/Generate SQL files).

Un grand nombre de points forts : rigueur de la démarche et ergonomie à la fois sobre et puissante, documentation détaillée du module SQL Designer (principes de transformations), réactivité et efficacité du support et l'existence d'un forum dédié. Les points faibles concernent le faible nombre de SGBD pris en compte (notamment SQL92 et tous les *open source*), l'absence de processus de rétro-conception et la non prise en compte de l'héritage multiple. Enfin, certaines transformation de diagrammes ont posé des problèmes qui sont désormais identifiés et en cours de correction.

Poseidon

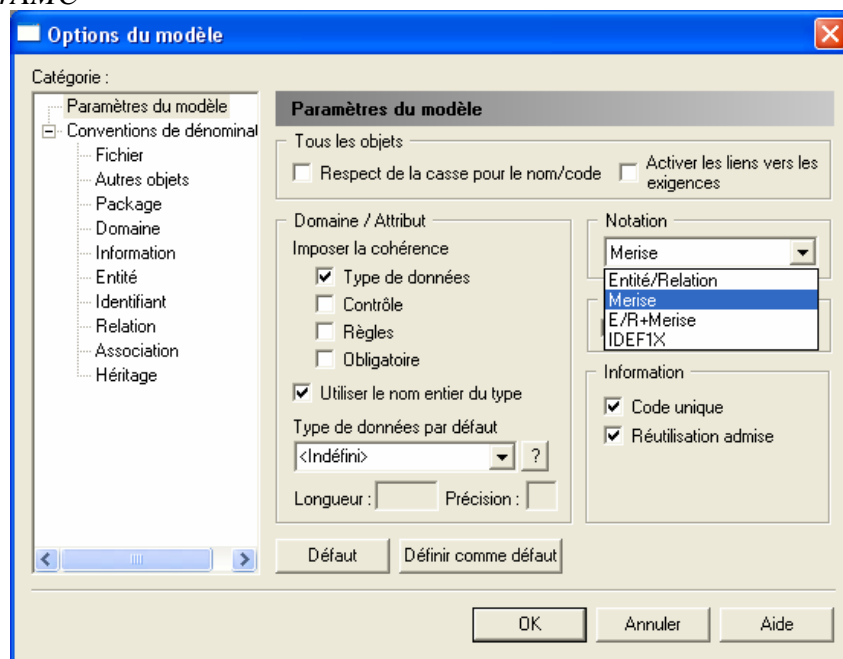
Cet outil n'est pas vraiment fait pour les bases de données. Le niveau logique n'est pas supporté, la génération de code SQL est plus que limite (les classes-associations ne sont pas traduites et les associations sont des fois inexistantes). L'ergonomie surprend au début mais on s'y fait. La création de classes-associations et d'associations réflexives n'est pas aisée. Les options ne sont pas toutes traduites en français, les associations *n*-aires ne sont pas prises en compte, il n'y a pas de processus de rétro-conception d'une base, n'en jetez plus !

Fig.20 Poseidon



PowerAMC

PowerAMC (anciennement AMC*Designer) est la version française de l'outil de modélisation PowerDesigner de Sybase. Concernant les bases de données, l'outil prend en charge trois types de modèles qu'on peut transformer entre eux (le modèle conceptuel de données de type Merise, entité-relation ou IDEF1X, le modèle physique de données qui correspond au niveau logique, le modèle orienté objet (MOO) au formalisme UML). L'ergonomie de l'outil est très intuitive.

Fig.21 PowerAMC

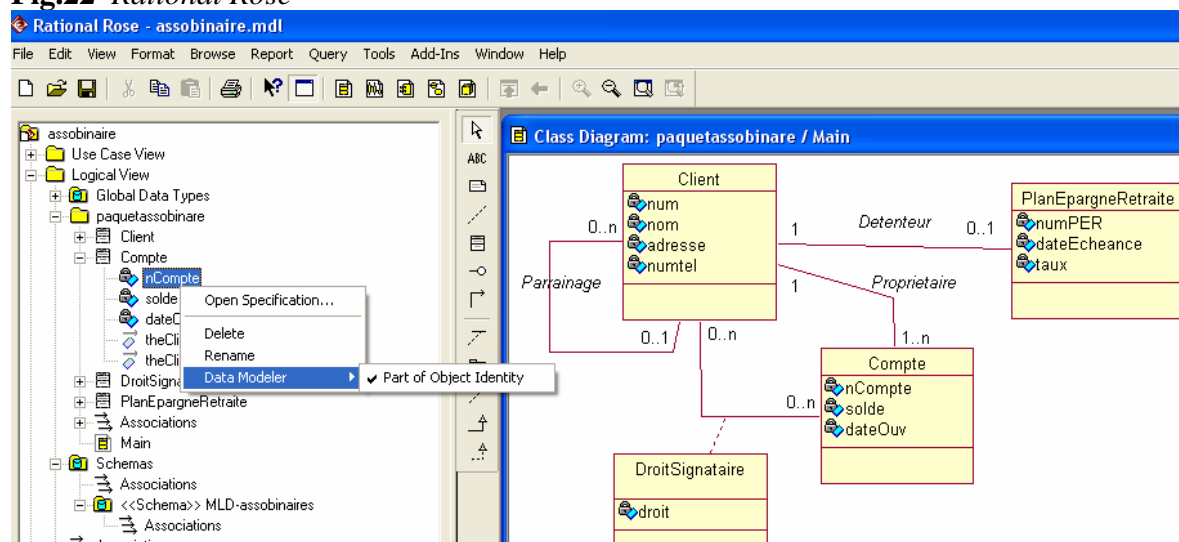
Les transformations de modèles se font très facilement en ouvrant le diagramme puis Outils/Générer un Modèle...). Pour manipuler une base de données (génération ou rétro-conception), il faut travailler avec un modèle physique pour que l'option SGBD soit active. La rétro-conception d'une base de données peut s'opérer, comme Win'Design à partir de l'extraction de tables ou d'un script SQL existant.

Quelques points forts : un grand choix de SGBD est pris en compte, la robustesse et la qualité de l'ergonomie qui permet de travailler facilement avec différents diagrammes dans le même environnement. Les seuls points faibles côté UML se réduisent à l'absence de contraintes prédéfinies et du concept d'association *n*-aire.

Rational Rose Data Modeler

L'outil de Rational est propriété d'IBM depuis le rachat de la société en 2001. Sa manipulation est assez intuitive et son interface est sobre. Pour débiter, créez un composant Database dans le répertoire Component View (clic droit sur un répertoire, Data Modeler/New/Database). Nommez la base et précisez la nature du SGBD cible (Name et Target). Créer un schéma dans le répertoire Logical View (clic droit sur le répertoire, option Data Modeler/New/Schema). Associez ensuite ce schéma à la base de données cible précédemment créée.

Le diagramme de classes UML doit être situé dans un paquetage (clic droit sur le compartiment Logical View, New/Package). Les classes doivent être marquées Persistent (clic droit, Open Specification, onglet Detail).

Fig.22 Rational Rose

La transformation du modèle objet se fait en sélectionnant le paquetage des classes UML à transformer (clic droit puis Data Modeler/Transform to Data Model...). Pour visualiser le modèle logique créé dans le schéma un diagramme (Data Modeler/New/Data Model Diagram) puis faites glisser chaque relation obtenue par la transformation.

Le processus de rétro conception démarre du schéma (compartiment Logical View/Schemas) qu'on sélectionne par un clic droit (Data Modeler/Transform to Object Model). Un schéma se transforme en un paquetage à créer initialement. Un assistant permet de sélectionner le nom du paquetage en sortie et si oui ou non les clés primaires doivent être transformées en identifiant de classe. Le fait de transformer un modèle de données dans un paquetage modifie les nouveaux éléments du paquetage mais ne détruit ni ne modifie les éléments mis à jour au niveau du modèle de données. Si cette option du logiciel fonctionne bien pour les associations binaires, classes-associations et l'héritage, elle n'est pas opérationnelle pour les associations *n*-aires.

Bien que peu de modifications aient été apportées à cet outil depuis la première version de cet ouvrage, le point fort du logiciel est sa très grande robustesse. On peut toutefois regretter qu'il n'inclut pas encore de pilotes pour les SGBD *open source*, qu'il ne propose pas l'agrégation simple et les associations *n*-aires.

Together

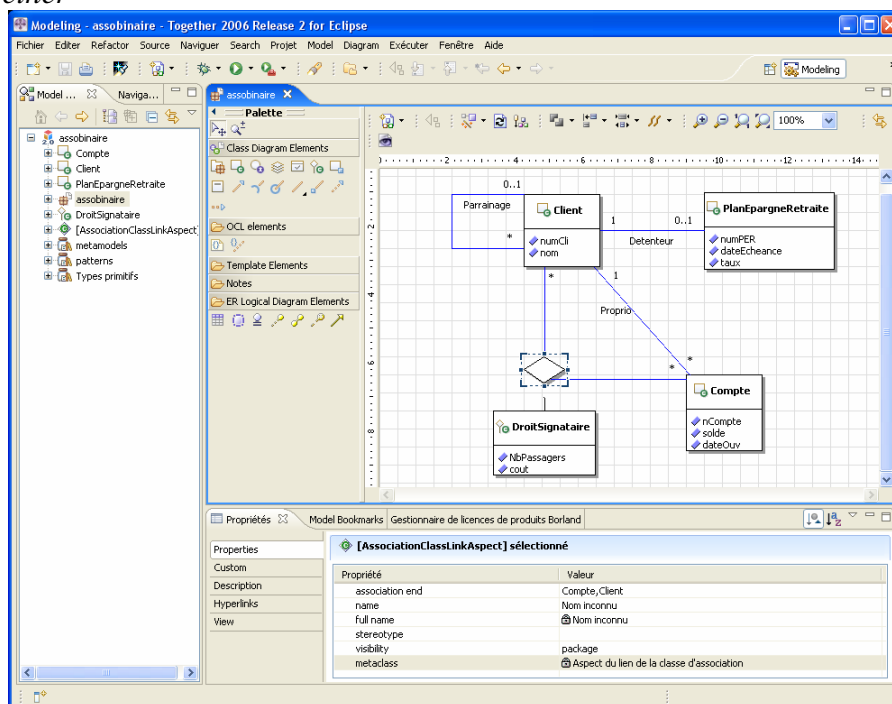
Together est l'outil de conception de Borland. La partie relative aux bases de données est nommée *Data Modeling* et est présente dans la version pour Eclipse. Une autre version existe pour Visual Studio mais elle ne prend pas en compte l'aspect modélisation de bases de données.

Dans la partie *Data Modeling*, seuls les niveaux logique et physique sont présents. Le formalisme des modèles logiques s'apparente au profil UML pour les bases de données. Dans ce modèle, deux types d'associations sont prévues : celles qui doivent donner lieu à la création d'une table (*many-to-many relationship*) et celles qui génèrent seulement une clé étrangère (*relationship*). Pour ces dernières, l'entité cible du lien créé graphiquement détermine l'entité parent. Les modèles physiques font apparaître les clés étrangères sous la forme graphique (formalisme IDEF1X). La transformation entre les deux modèles se fait par la fonction Importer/DB schema from ER Logical Diagram. La génération de script SQL s'opère par la fonction Exporter/DDI SQL script.

La création d'un diagramme de classes nécessite de créer un projet (Nouveau/Autres/ULML 2.0 Project). Une palette de symboles est alors mise à disposition. Pour créer une classe-association (ou une association *n*-aire), sélectionner l'icône Association Class. Ensuite relier le symbole losange (*diamond*) aux différentes classes avec le lien Association End.

La transformation automatique du diagramme de classes en modèle logique n'est pas nativement proposée. On ne peut avoir accès à la base de données que par le modèle logique et UML n'est pas encore exploité au niveau conceptuel. Pour ce faire, il faudra programmer des transformateurs de modèles (similaires aux règles de mapping de Enterprise Architect). La documentation fait toutefois preuve d'optimisme "*The concept of entities and relationships in logical data modeling maps to the concept of classes and associations in the UML 2.0 class diagram*". Il n'empêche que la programmation des règles de transformation des classes-association, associations *n*-aires et l'héritage par les éléments du modèle logique profilé par Borland risque de ne pas être tout repos.

Fig.23 Together



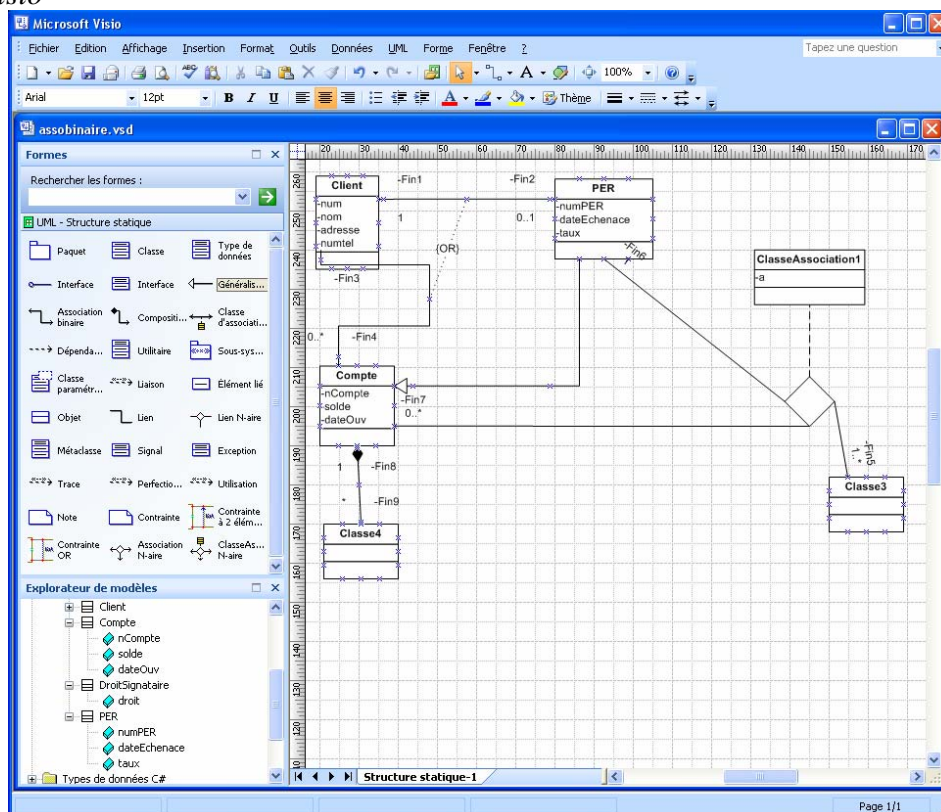
La rétro-conception consiste à importer (Fichier/Importer, la source étant soit un script, soit une connexion JDBC à une base qu'il faudra paramétrer). Le schéma généré est un modèle physique dont le formalisme s'apparente au profil UML et qui fait apparaître explicitement les clés étrangères.

Les points forts de l'outil : sa qualité et sa robustesse, l'efficacité du support et l'expertise dans les fonctionnalités de transformation de modèles par l'utilisation de QVT sur la base de EMF (*Eclipse Metamodel Framework*). Le point faible majeur concerne l'absence de processus natif de transformation entre schémas conceptuels et modèles logiques.

Visio

Visio fait partie de la suite Office de Microsoft. En utilisant la notation UML, il n'est pas possible de générer de modèle ni de code. Pour ce faire, vous devrez travailler soit avec le modèle ORM (modèle conceptuel binaire graphique), soit avec IDEF1X (modèle relationnel graphique). Ce sont les seuls formalismes qui permettent de se connecter à une base. Dans ces deux cas, vous devrez utiliser Visio au sein de l'environnement de Visual Studio. Si vous utilisez une base *open source*, il faudra trouver le bon pilote (en utilisant un pilote générique ODBC pour MySQL j'ai eu la surprise de trouver que les colonnes de mes tables étaient inscrites en caractères japonais !).

Fig.24 Visio

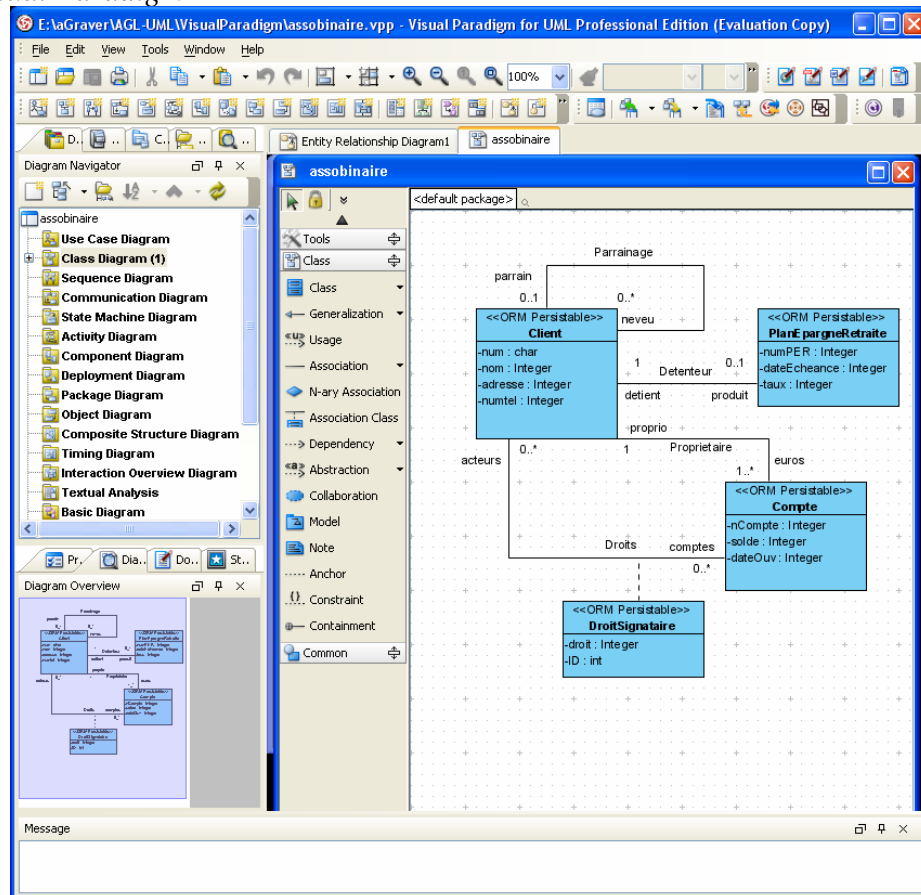


Visual Paradigm

Une fois saisi le diagramme de classes UML, nommez impérativement tous les rôles des liens d'associations (sinon la génération au niveau logique ne sera pas possible). Ensuite configurer votre transformation Tools/Object...(ORM)/Wizards... Rendez persistantes vos classes, choisissez un identifiant pour chacune et paramétrez votre connexion à la base de données cible. Une fois le modèle logique généré, il est nécessaire de le synchroniser Tools/Object...(ORM)/Synchronize... avec le diagramme de classes avant de pouvoir générer un script SQL. De toutes façons, vous devrez souvent agir sur le modèle logique (surtout pour les identifiants des classes-associations) avant de générer une base de données.

Quelques points forts : un grand choix de SGBD est pris en compte, la réactivité et l'efficacité du support. Les points faibles concernent le déplacement des objets et des liens sur un graphique du niveau logique (Entity-Relationship) qui n'est pas toujours réussi et l'absence d'identifiant de classe et d'association *n*-aire.

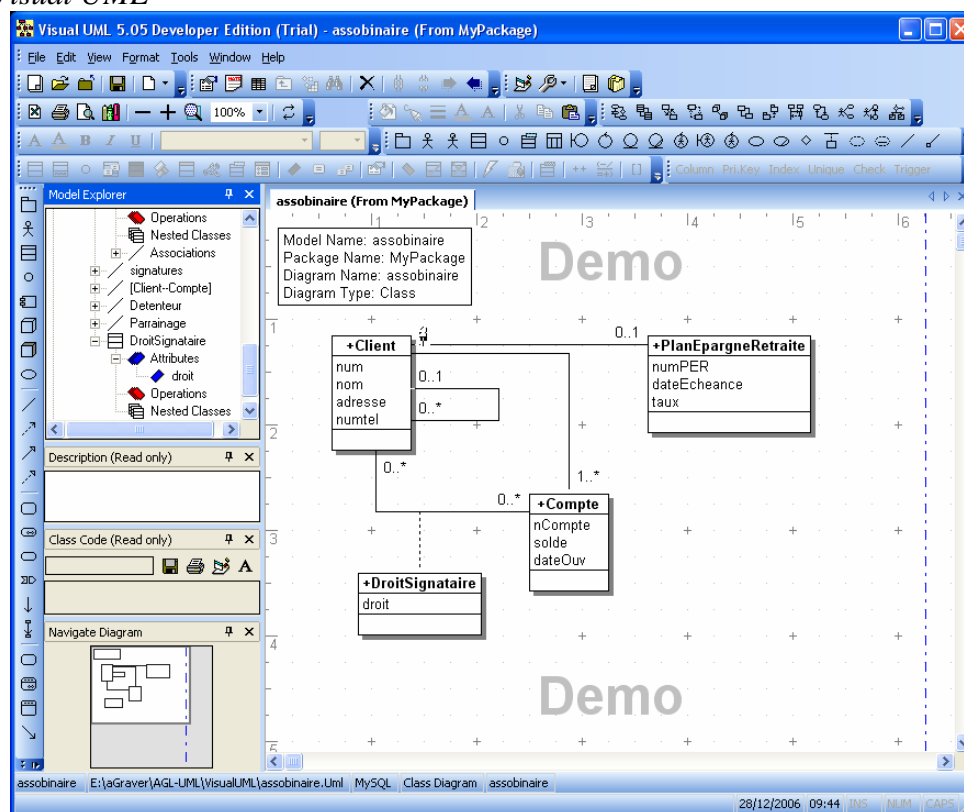
Fig.25 Visual Paradigm



Visual UML

Pas grand chose à dire sur cet outil qui ne prend en compte que le niveau logique et qui nécessite donc par conséquent de définir manuellement les clés étrangères pour chaque association! Si vous êtes réfractaire au niveau conceptuel, rendez vos classes persistantes, munissez-vous d'un pilote ODBC pour votre base de données et bon courage pour la suite.

Fig.26 Visual UML

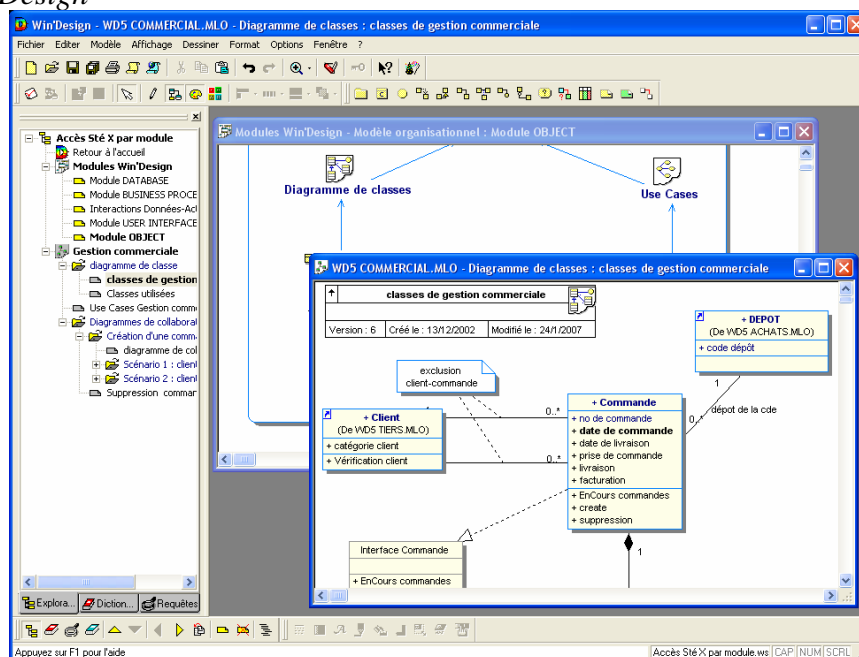


Win'Design

Win'Design est le produit de la société CECIMA basée à Aix en Provence. Il est présent sur le marché français depuis 1995. Développé initialement pour Merise/2, la notation UML arrive en 2002 avec la version 5. Depuis l'outil est en évolution constante.

La Gamme comprend quatre modules autonomes et complémentaires, qui s'articulent autour d'un référentiel (Database pour la conception et le reverse des bases de données, Business Process pour la modélisation des processus métier, Object pour la modélisation UML et UserInterface pour le maquetage des IHM). Vous devrez disposer du premier et du troisième module pour traduire des diagrammes de classes en script SQL. Comme PowerAMC, l'outil permet la double notation Merise/2 et UML 2 (sans le mode mixte de PowerAMC qui peut porter à confusion). Cet outil est le plus complet en ce qui concerne les contraintes Merise/2.

Fig.27 Win'Design



Win'Design est probablement l'outil le plus facile à prendre en main et son interface est très intuitive. Citons d'autres points forts : un très grand choix de SGBD est pris en compte, un grand nombre d'options de transformation de modèles (à tous les niveaux), la réactivité et l'efficacité du support. Notez que Win'Design est le seul à traiter correctement la transformation des associations n -aires et qu'avec PowerAMC et Rational, il est capable d'extraire un modèle conceptuel sans connexion à la base (à partir du seul script SQL de création des tables et contraintes). Le point faible de la version évaluée concerne la transformation de diagramme UML complexes (quelques cas spéciaux relevés lors de ces tests) en MLD et MCD. Les problèmes sont connus et en cours de correction.

Conclusion

Tous les outils proposent une version d'évaluation limitée dans le temps (allant de 7 jours pour MEGA à 2 mois pour MagicDraw, 30 jours en moyenne pour les autres). Seuls Together et MyEclipse sont intégrés à Eclipse (Objectteering et MagicDraw proposent toutefois un plug-in). Tous les outils ne sont opérationnels au niveau conceptuel car les éditeurs outre-Atlantique ont souvent privilégié des modèles ne proposant que peu d'éléments aux concepteurs (*entity* et *relationship* par exemple).

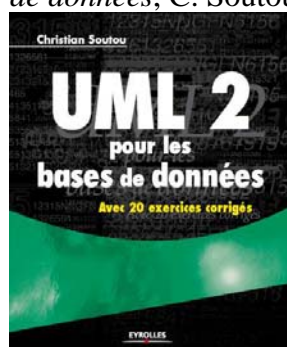
Depuis 5 ans, le nombre d'outils capable de générer des scripts SQL a plus que doublé et les fonctionnalités deviennent de plus en plus puissantes. L'évolution d'UML se concrétise car le *Request for Proposal* de l'OMG nommée *Information Management Metamodel* a pour but de définir, pour le courant de l'année 2007, un métamodèle incluant différentes sous-parties: *Common Warehouse Metamodel*, *UML2 Profile for Relational Data Modeling*, *UML2 Profile for Logical (Entity Relationship) Data Modeling*, *UML2 Profile for XML Data Modeling* et *UML2 Profile for Record Modeling* (COBOL). De nouvelles spécifications à propos de la modélisation sont sans doute à attendre prochainement comme par exemple la notion d'identifiant de classe.

Produits

Produit	URL
Enterprise Architect	http://www.sparxsystems.com.au/products/ea.html
MagicDraw	http://www.magicdraw.com/
MEGA Designer	http://www.mega.com/en/product/mega_designer/
ModelSphere	http://www.silverrun.com/modelsphere.html
MyEclipse	http://myeclipseide.com
Objectteering	http://www.objectteering.com/
Poseidon	http://gentleware.com/index.php?id=30
PowerAMC	http://www.sybase.com/products/developmentintegration/poweramc
Rational Rose Data Modeler	http://www-306.ibm.com/software/awdtools/developer/datamodeler/
Together	http://www.borland.com
Visio	http://www.microsoft.com/france/office/visio
Visual Paradigm	http://www.visual-paradigm.com/product/vpuml/productinfovpumlse.jsp
Visual UML	http://www.visualuml.com/Products.htm
Win'Design	http://www.win-design.com/fr/

Pour en savoir plus

- Livre *UML 2 pour les bases de données*, C. Soutou, Eyrolles 2007.



- Spécification UML 2.1, www.omg.org/cgi-bin/doc?ptc/2006-04-02
- *Database Modelling in UML*, Geoffrey Sparks, www.sparxsystems.com.au