Démarche dirigée par les modèles pour la conception d'entrepôts de données multidimensionnelles

F.Atigui, F.Ravat, O.Teste, G.Zurfluh

IRIT (SIG/ED)
Université Paul Sabatier
118 route de Narbonne
31062 Toulouse cedex 09

{atigui, ravat, teste, zurfluh}@irit.fr

Résumé. Ce travail présente notre démarche dirigée par les modèles pour la conception d'entrepôt de données (ED) multidimensionnelles. Notre approche a l'avantage de modéliser les données de l'ED tout en incluant les processus ETL (extraction, transformation, chargement) associés. Cette démarche permet de couvrir tout le cycle de conception de l'ED : l'analyse des besoins ainsi que les modélisations conceptuelle, logique et physique (avec les traitements ETL). Elle est définie dans le cadre méthodologique de l'Ingénierie Dirigée par les Modèles (IDM). Ceci est particulièrement avantageux car l'IDM offre des mécanismes de transformation automatique entre les différents niveaux présentés par des modèles. La phase d'analyse permet de construire le modèle multidimensionnel initial à partir des besoins des décideurs. La phase de modélisation conceptuelle sert à valider ce modèle par rapport aux sources et à le compléter afin de garantir un passage automatique vers les niveaux inférieurs (logique et physique).

Mots clés : Démarche de conception d'ED multidimensionnelles, ETL, IDM.

1 Contexte et problématique

Nos travaux se situent dans le cadre des systèmes d'aide à la décision reposant sur un Entrepôt de Données (ED) multidimensionnelles. Un ED est construit à partir des données sources d'une organisation via des processus ETL (extraction, transformation, chargement) (Vassiliadis, 2009). Les données de l'entrepôt sont souvent représentées sous un format multidimensionnel afin de favoriser la prise de décision.
Les approches de conception d'entrepôts peuvent être classées en trois catégories (Rizzi et al., 2006) :

- Les approches « descendantes » (Kimball & Ross, 2002 et Prat et al., 2006) permettent de construire le schéma de l'entrepôt à partir d'une analyse détaillée des besoins des décideurs, les problèmes de correspondances entre ces besoins et les sources existantes sont traités à postériori lors du chargement des données dans l'entrepôt.

- Les approches « ascendantes » (Golfarelli & Rizzi, 1998 et Husemann et al., 2000) partent d'une analyse détaillée des sources de données, les besoins des décideurs ne sont pris en compte que vers la phase d'analyse en ligne des données.

- Les approches « mixtes » (Zepda et al., 2008 et Romero & Abello, 2010) considèrent à la fois les besoins des décideurs et la disponibilité des données sources. Elles

présentent donc l'avantage de concevoir des schémas multidimensionnels valides par rapport aux sources de données.

Dans le processus de conception d'entrepôt, les démarches actuellement utilisées s'avèrent fastidieuses et nécessitent une expertise humaine importante. Nous souhaitons apporter des solutions à cette problématique.

De part les avantages des approches mixtes, nous souhaitons positionner nos travaux dans cette catégorie. Plus précisément, pour répondre aux limites des approches existantes, nous souhaitons proposer une démarche (i) *mixte*, (ii) *unifiée* et (iii) *automatique* pour la conception d'ED multidimensionnelles. Il s'agit d'une démarche de modélisation « mixte » puisque le schéma de l'entrepôt doit être construit à partir des besoins des décideurs et validé par rapport aux sources de données. Nous qualifions cette démarche d'« unifiée » car elle doit permettre de fusionner deux cycles de développement : la conception d'ED et la conception des processus ETL. Enfin, cette démarche est dite « automatique » car elle doit proposer un ensemble de règles permettant la génération automatique des schémas logiques et physiques à partir du schéma conceptuel.

Dans ce contexte, il est reconnu que l'Ingénierie Dirigée par les Modèles (IDM) est une approche de modélisation qui vise à couvrir le cycle de développement du logiciel en proposant un ensemble de modèles et de processus de transformation automatique. L'IDM applique les principes de l'Architecture Dirigée par les Modèles (Model Driven Architecture : MDA). MDA est standardisée par l'OMG[1] et basée sur l'utilisation de modèles aux différentes phases du cycle de développement d'une application. En effet, MDA préconise l'élaboration de trois types de modèles : 1) Le modèle des exigences (Computation Independent Model : CIM) décrit les services que doit fournir l'application pour répondre aux besoins des utilisateurs, sans spécifier les détails de sa construction. 2) Le modèle d'analyse et de conception (Platform Independent Model : PIM) définit la structure et le comportement du système sans indiquer la plateforme d'exécution. 3) Le modèle de code ou de conception concrète (Platform Specific Model : PSM) est la projection d'un PIM sur une plateforme donnée (Blanc & Salvatori, 2005). MDA permet d'aboutir au code de l'application en partant du PIM grâce à la définition d'une série de transformations. Le choix de l'IDM comme cadre formel présente plusieurs avantages. En effet, le processus de développement de l'entrepôt exige moins d'effort améliorant ainsi la productivité. L'IDM permet également de fournir un support pour l'évolution, l'intégration, l'interopérabilité, l'adaptabilité, la portabilité et la réutilisabilité des systèmes d'information (Mazon & Trujillo, 2008).

Nous souhaitons donc proposer une démarche de conception d'ED multidimensionnelles reposant sur l'enchainement de différents modèles CIM, PIM et PSM. En section, 2 nous présentons les travaux existants relatifs à l'IDM d'ED. En section 3, nous présentons les différents modèles de notre démarche. La section 4 détaille les différents modèles PIMs et PSMs proposés. La section 5 se centre sur les transformations automatiques de modèles.

[1] OMG : Object Management Group

2 Etat de l'art

La conception d'ED a fait objet de nombreux travaux (Teste, 2009). Dans la littérature, elle a été traitée de deux points de vue différents (Rizzi et al., 2006). Le premier point de vue concerne la conception du schéma multidimensionnel qui vise à spécifier la structure de l'entrepôt. Le second point de vue a pour objet la modélisation des traitements ETL afin de représenter les processus responsables de l'alimentation et de la mise à jour de l'entrepôt.

Dans cet article, nous ne présentons que les approches de conception dirigées par les modèles. En ce qui concerne la modélisation conceptuelle des ED, la première proposition dirigée par les modèles date de 2003. Les travaux de (Poole, 2003) proposent une approche basée sur les différents paquetages du Common Warehouse Metamodel (CWM) (OMG, 2003). Les Méta-modèles du CWM définissent l'interopérabilité entre les composants physiques des entrepôts (Kadima 2005). (Zepda et al., 2008) proposent une approche de modélisation mixte. Ils présentent une démarche qui examine les schémas des sources et définit les règles de transformations des méta-modèles Entité/Association vers les méta-modèles OLAP[2]. Par la suite, une phase d'analyse des besoins est appliquée. Enfin, le schéma conceptuel de l'entrepôt est construit en prenant en compte les schémas sources et les besoins décideurs. Les travaux de (Mazon & Trujillo, 2009) proposent une démarche de conception reposant sur différents modèles : un CIM multidimensionnel, un PIM initial, un PIM hybride (validé par rapport aux schémas sources) et un PSM (modèle logique). (Essaidi & Osmani, 2009) proposent d'utiliser MDA pour définir un cadre de modélisation des différents composants d'un entrepôt et les processus 2TUP[3] afin de décrire le processus d'entreposage. En conclusion, l'approche de (Poole, 2003) est intégralement basée sur les méta-modèles du CWM. Ces derniers décrivent des détails d'implantation techniques qui ne relèvent pas de l'aspect conceptuel. Ils sont difficilement compréhensibles par les concepteurs et les utilisateurs finaux. Les travaux de (Mazon & Trujillo, 2009) ne traitent pas de l'aspect physique ; de même les travaux de (Zepda et al., 2008) se limitent uniquement à la modélisation conceptuelle. Enfin, les travaux de (Essaidi & Osmani, 2009) ne proposent qu'une démarche globale sans spécification précise des modèles. Les travaux de recherches sur la modélisation des processus ETL supposent que le schéma de l'entrepôt est défini au préalable. A notre connaissance, il existe un seul travail de (Munoz et al., 2009) proposant la conception dirigée par les modèles pour les processus ETL. Les auteurs présentent un ensemble de règles de transformations permettant la génération automatique des activités et des actions ETL à partir des modèles conceptuels.

Dans cette section, nous avons présenté un état de l'art sur la conception dirigée par les modèles des entrepôts d'une part et des processus ETL d'autre part. Ces deux sujets, pourtant si dépendants, font l'objet de travaux séparés. De manière générale, les recherches sur la modélisation de l'entrepôt visent à proposer un schéma pour l'entrepôt en fonction des besoins et/ou des schémas sources. Alors que, la modélisation des processus ETL se contente d'établir les relations de correspondance entre les éléments de ce schéma et les éléments sources. Cependant, à ce niveau, nous avons souvent besoin de modifier la structure du

[2] OLAP : On Line Analytical Processing.

[3] 2TUP : 2 Track Unified Process : est un processus de développement logiciel qui implémente le Processus Unifié.

schéma de l'ED pour qu'il soit en adéquation avec les données sources (Ravat et al., 2006). En outre, les travaux existants proposent des modèles, voire des méthodes de conception différentes, pour décrire le schéma multidimensionnel (Mazon & Trujillo, 2009) ou les processus ETL (Simitsis & Vassiliadis, 2003). Ces méthodes partagent plusieurs étapes souvent coûteuses, particulièrement lorsqu'il s'agit des méthodes mixtes ; c'est le cas par exemple de la phase de validation du schéma multidimensionnel par rapport aux sources de données. Ainsi, il est crucial que la conception des processus ETL soit directement adossée à la conception du schéma de l'entrepôt (Annoni et al., 2006).

Par rapport aux différentes propositions citées au début de cette section, notre approche présente plusieurs avantages.

1. Elle permet de couvrir tout le cycle de développement, de l'analyse des besoins jusqu'à la modélisation physique généralement omise ;
2. Elle valide le schéma par rapport aux sources et le revalide par rapport aux besoins ;
3. Elle réduit le temps de développement de l'entrepôt, étant donné que :
    i. Les deux cycles de développements des entrepôts et des processus ETL sont unifiés et des étapes redondantes ne sont plus nécessaires ;
    ii. Le schéma conceptuel est bien validé par rapport aux besoins des décideurs ;
    iii. Les transformations formelles sont définies pour la génération automatique des modèles logiques et physiques.

3 IDM pour ED multidimensionnelles

Notre démarche débute par l'analyse des besoins (modèle des exigences) pour aboutir aux modèles physiques spécifiques à différentes plateformes. Les principes de l'IDM simplifient la tâche du concepteur : ce dernier doit construire un modèle des exigences (CIM) et les transformations automatiques traduisent ce modèle en une succession de modèles de façon à obtenir un modèle physique adapté à la plateforme choisie. Le CIM permet de spécifier les exigences des décideurs pour un système d'information décisionnel qui fournit l'information la plus adéquate (Gam, 2008). La modélisation des exigences doit permettre d'identifier et de décrire les buts et les objectifs de l'organisation afin que la déduction automatique du modèle conceptuel soit possible.

Dans le cadre de cet article, nous ne traitons pas la première étape, à savoir la définition du modèle des exigences. Les deux étapes suivantes permettent de construire les modèles indépendants des plateformes (PIM). Il à noter que MDA ne donne aucune indication sur le nombre de modèles PIM à élaborer (Blanc & Salvatori, 2005). Nous proposons deux niveaux différents de PIM. Le premier niveau représente le modèle conceptuel (PIM_1) et permet de décrire la structure multidimensionnelle de l'entrepôt (Ravat et al., 2007) et de spécifier les traitements ETL associés. Le second niveau (PIM_2) présente différentes alternatives de modèles logiques déduits automatiquement à partir du modèle conceptuel. Enfin, les

différents modèles logiques sont traduits de manière automatique en un ensemble de modèles de plateformes d'implantation physiques (PSMs). La figure suivante présente les différentes étapes de notre approche.

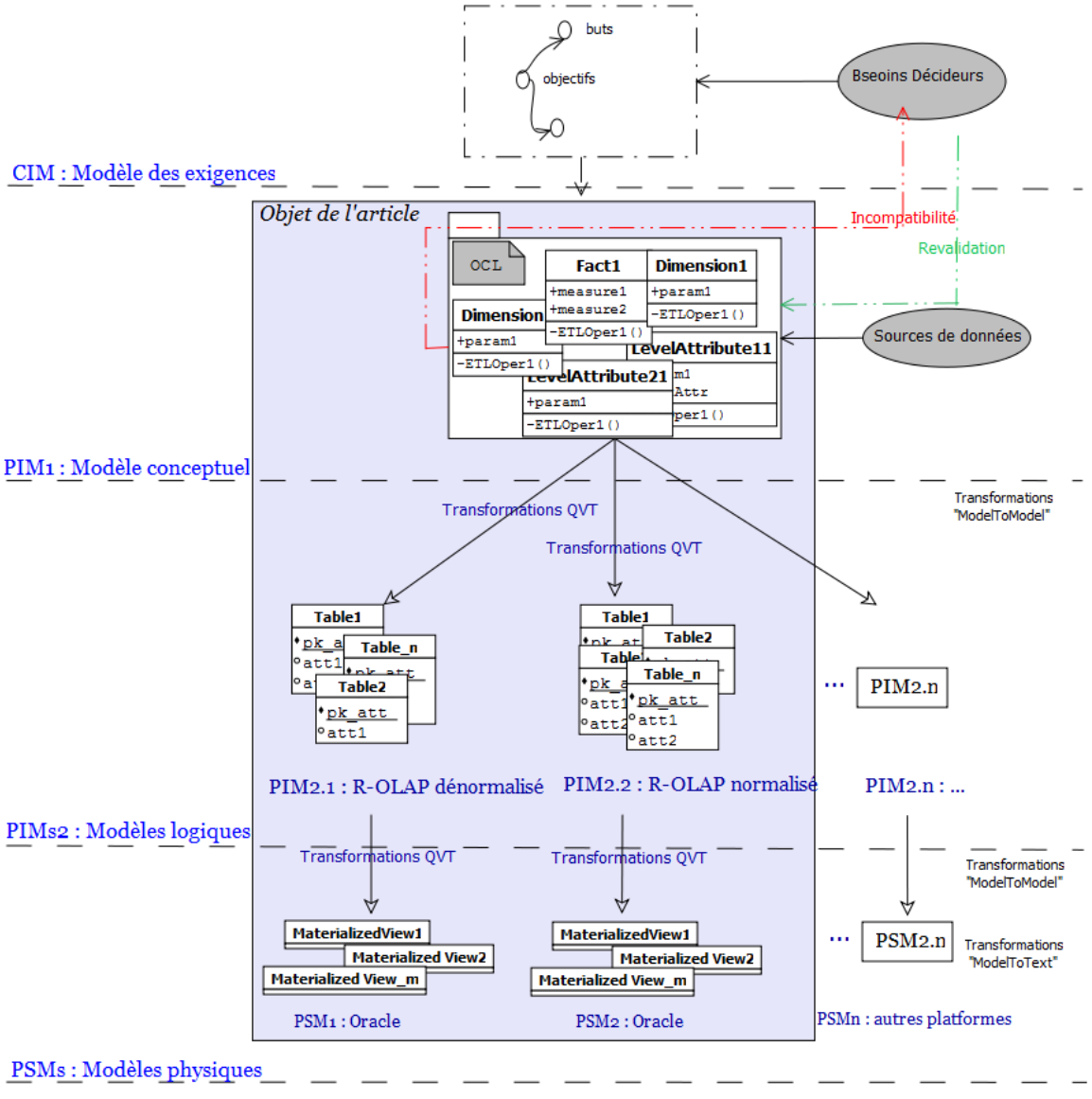

Figure 1. Démarche dirigée par les modèles pour la conception d'ED

4 Les modèles

4.1 PIM$_1$: Modélisation conceptuelle

4.1.1 Profil UML pour les ED

Le modèle conceptuel de l'entrepôt (PIM$_1$) permet une représentation multidimensionnelle des besoins issus du CIM et des traitements ETL associés. Il est caractérisé par un aspect structurel statique (attributs multidimensionnels) et un aspect dynamique (opérations de chargement). Afin, de représenter ces deux aspects, nous avons recours aux modèles de

classes UML (Unified Modeling Language) qui s'impose comme langage d'expression de modèles à objets (Kadima, 2005). UML permet de modéliser les aspects structurel et comportemental d'un système via un ensemble de modèles. Particulièrement, les modèles de classes permettent de représenter simultanément ces deux aspects. Ils explicitent la structure d'un système grâce aux attributs et leur comportement grâce aux opérations. L'utilisation de ces modèles permet de réduire l'effort du concepteur dans l'apprentissage de nouveaux modèles et méthodes. Cependant, les modèles de classes s'avèrent inadaptés à la modélisation multidimensionnelle. Pour pallier cette insuffisance et expliciter la sémantique des concepts multidimensionnels, nous proposons un profil UML. Un profil est un ensemble de mécanismes et de techniques permettant d'adapter UML à un domaine d'application spécifique. D'un point de vue technique, un profil est un ensemble de stéréotypes. Un stéréotype peut être défini comme un concept spécifique à un domaine (Kadima, 2005). Dans le contexte des ED, nous définissons le profil « Profil UML pour les ED » (DWP) explicité par la figure 2. Ce profil est composé du stéréotype « Constellation » (extension de la méta-classe « package »). Un schéma en constellation est composé de sujets et d'axes d'analyse (Ravat et al., 2007) présentés respectivement par les stéréotypes : « Fact » et « Dimension » (extensions de la méta-classe « Class »). Une dimension est composée d'une ou plusieurs hiérarchies (stéréotype « Hierarchy »). Chaque hiérarchie est composée d'attributs (« LevelAttribute ») qui étendent la méta-classe « Class ». Chaque niveau d'attributs est composé d'un paramètre (stéréotype « Parameter ») et éventuellement d'un ou plusieurs attribut(s) faible(s) (stéréotype « WeakAttribute »). Ces derniers servent à compléter la sémantique d'un paramètre. Le lien de composition réflexif au niveau des « LevelAttribute » exprime le fait qu'un niveau de granularité i est composé du niveau de granularité i-1. Une dimension doit être liée à au moins un fait qui à son tour doit avoir un lien avec au moins une dimension. Un fait est composé d'une ou plusieurs mesure(s) représentant les indicateurs d'analyse (stéréotype « Measure » étendant la méta-classe « StructuralFeature »). Une classe UML est composée d'une liste de propriétés statiques et d'un ensemble d'opérations dynamiques. Dans le contexte des ED, les propriétés statiques sont les mesures pour les faits, les paramètres et les attributs faibles des hiérarchies composants les dimensions. Les opérations concernent particulièrement les traitements de chargement de l'entrepôt modélisées par le stéréotype « ETLOperation » (extension de la méta-classe « Operation »). Nous supposons que chaque fait, dimension, hiérarchie et niveau d'attributs doit posséder au moins une opération ETL. Ces opérations sont décrites par une liste de contraintes définies par le stéréotype « ETLConstraint ». Tous les stéréotypes ainsi définis ont une visibilité publique à l'exception des opérations et des contraintes ETL (« ETLOperation » et « ETLConstraint »). Ces dernières doivent être invisibles vis-à-vis du décideur.

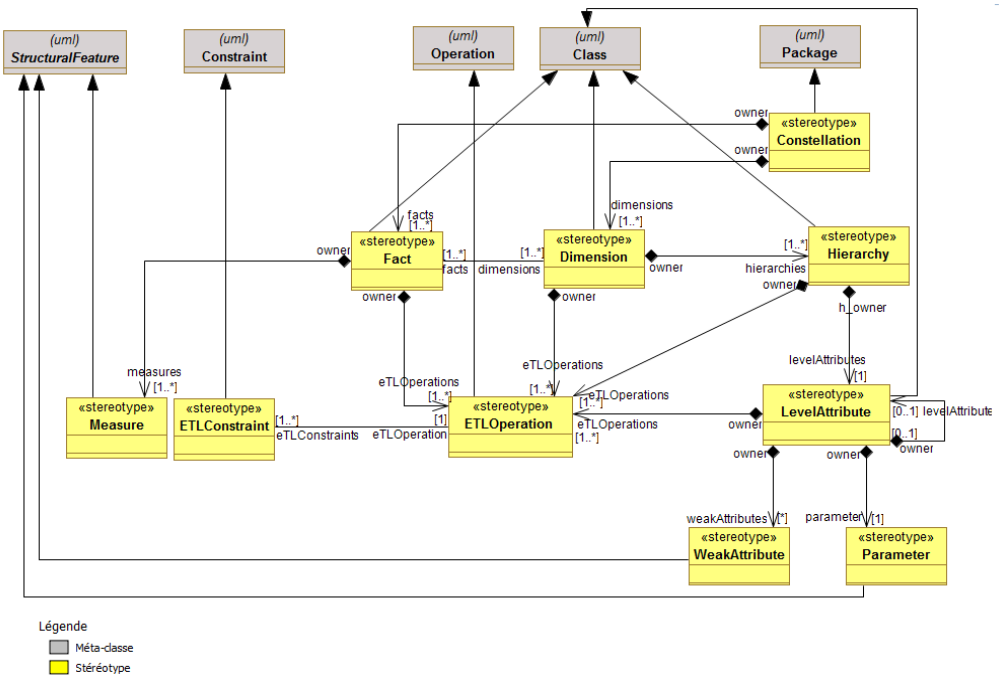

Figure 2. Profil UML pour les ED (DWP)

4.1.2 Modélisation statique

Le modèle conceptuel (PIM_1) permet de représenter les données répondant aux besoins des décideurs sous un format multidimensionnel. Dans cette section nous présentons un exemple explicitant la structure statique de ce modèle. Cette représentation repose sur les concepts de fait, de dimension, de mesure, de hiérarchie, de paramètre et d'attribut faible. L'exemple suivant présente une instanciation du profil explicité par la figure 2 sans prendre en compte les opérations de chargement.

Exemple 1. L'étude de cas illustrant les propositions de l'article concerne une société de vente par correspondance qui souhaite analyser les ventes de produits réalisées auprès de ses clients en France. Le schéma multdimensionnel permet d'analyser les quantités et les montants de la vente d'un produit à un client à une date donnée. Le modèle conceptuel (PIM_1) statique est présenté par la figure 3. Les mesures « quantité » et « montant » du fait « Ventes » peuvent être analysées suivant les dimensions « Client », « Temps » et « Produit ». Ces dimensions sont composées respectivement des hiérarchies « H_Géo », « H_Temps » et « H_Prod ». Chaque hiérarchie est attachée avec un lien de composition au niveau de granularité plus faible (appelé niveau racine). Chaque niveau d'attributs est composé du niveau de granularité inférieure (une « Classe » de produit est composée d'une ou plusieurs « Catégorie »).

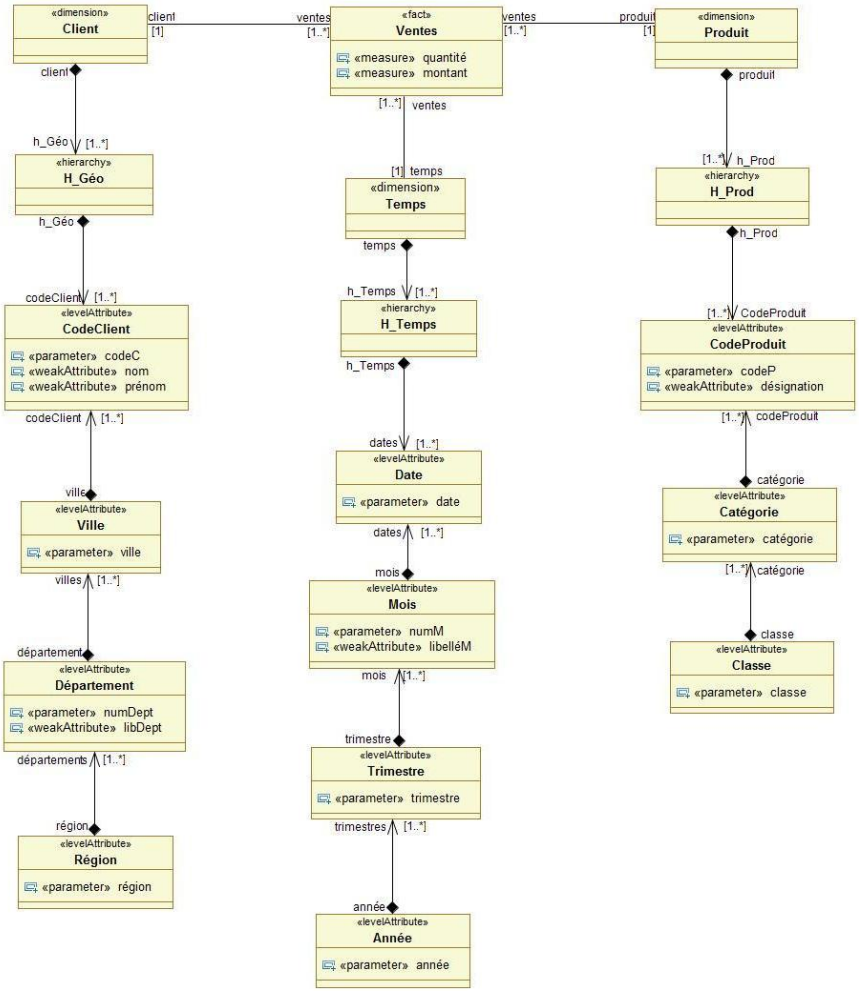

Figure 3. PIM_1 statique de l'étude de cas « Ventes »

4.1.3 Modélisation dynamique et contraintes

L'aspect dynamique du modèle conceptuel décrit les traitements de chargement associés aux différents éléments multidimensionnels. Ces traitements sont représentés via des opérations UML. Ces dernières servent à valider les éléments multidimensionnels par rapport aux différentes sources opérationnelles. Afin de mieux décrire ces opérations, le modèle conceptuel est complété par des contraintes OCL (Object Constraint Langage) permettant de spécifier les différentes conditions sur ces opérations.

- *Opérations ETL*

Le modèle de l'entrepôt construit à partir des besoins des décideurs a l'avantage de répondre à la totalité des exigences des décideurs. Cependant, ces derniers n'ont pas toujours une vue complète sur les sources de données. Par conséquence, les éléments de ce modèle peuvent être incompatibles, voire inexistants dans les schémas des sources. La modélisation conceptuelle unifiée du schéma multidimensionnel et des processus d'alimentation permet de fusionner les étapes de confrontations par rapport aux sources de données. Les processus de chargement sont décrits par des opérations UML spécifiques appelées « opérations ETL ». Ces dernières sont responsables (i) de vérifier l'adéquation des éléments multidimensionnels avec les

sources, (ii) de les valider, (iii) d'établir les relations de correspondances entre les éléments sources et les éléments cibles de l'entrepôt et (iv) de stocker les différentes formules d'extraction des éléments multidimensionnels à partir des sources opérationnelles.

- *Contraintes*

Dans notre cas d'étude, les contraintes OCL servent à décrire le PIM conceptuel en précisant les post et les pré-conditions associées aux opérations de chargement. OCL présente l'avantage de compléter les modèles UML via des descriptions formelles tout en faisant en sorte qu'ils soient indépendants des plateformes. Une expression OCL doit être définie dans un contexte (mot clé : « context »). Une contrainte OCL peut exprimer un invariant sur un objet (« inv »), une pré-condition à respecter avant l'appel de l'opération (« pré ») ou une post-condition à respecter après l'appel de l'opération (« post ») (OMG, 2010). Le méta-modèle « Profil UML pour les ED », contient le stéréotype « ETLConstraint » dont l'instance est un ensemble de contraintes OCL. Ces contraintes permettent en particulier de définir l'ordre d'exécution des opérations de chargement en décrivant les pré et les post conditions sur chaque opération.

Exemple 2. La figure 4 présente le modèle conceptuel de l'étude cas « Ventes », ce modèle comprend en plus des attributs déterminés précédemment au niveau du PIM_1 statique (figure 3), les opérations d'extraction et les contraintes OCL. L'opération « extraireVentes() » du fait « Ventes » permet d'extraire et de charger les mesures « quantité » et « montant ». Les pré-conditions définies sur cette opération permettent de spécifier l'ordre d'exécution des opérations de chargement. En effet, l'opération de chargement des mesures ne doit être exécutée qu'après le chargement des dimensions en relation avec le fait. Les post-conditions vérifient qu'après l'appel de cette opération, les mesures « quantité » et « montant » sont chargées convenablement. La contrainte OCL associé au fait « Ventes » est la suivante.

context Ventes ::extraireVentes()
pre : self.produit-> notEmpty() and self.temps-> notEmpty() and self.client-> notEmpty()
post : self.quantité.notEmpty() and self.montant.notEmpty()

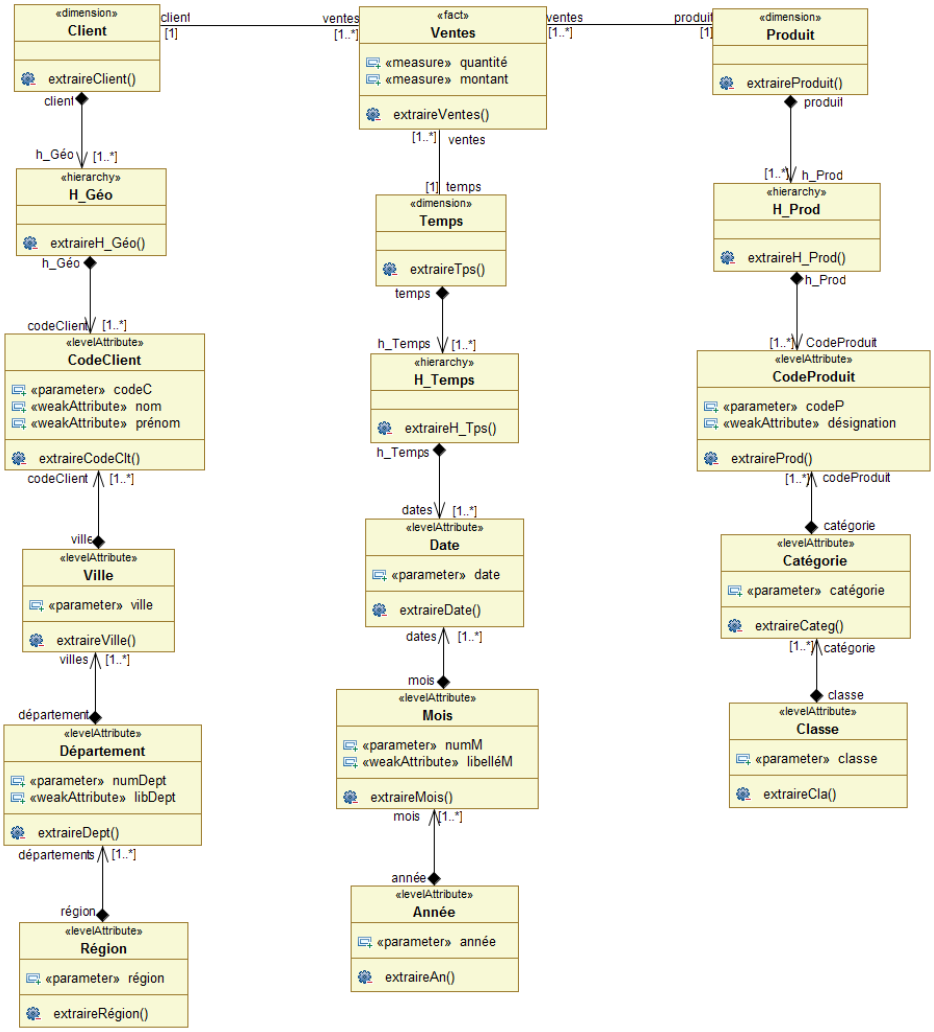

Figure 4. PIM conceptuel de l'étude de cas « Ventes »

4.2 PIM$_2$: Modélisation logique

Une fois le PIM conceptuel défini, le modèle logique peut être déduit automatiquement en appliquant un ensemble de règles. Afin de faciliter la tâche du concepteur, nous définissons un ensemble de règles permettant la génération automatique des modèles logiques à partir du modèle conceptuel. Le concepteur peut ainsi choisir le modèle logique adapté à l'application : R-OLAP[4] normalisé, dénormalisé ou optimisé (R-OLAP avec treillis de vues), O-OLAP[5], H-OLAP[6] ou encore les schémas XML, etc. A ce niveau nous définissons les règles de passage vers les modèles logiques souvent utilisés : le R-OLAP dénormalisé et le R-OLAP normalisé. Les règles de transformation du modèle multidimensionnel vers le R-OLAP dénormalisé sont détaillées dans la section suivante. Le passage vers le R-OLAP normalisé est présenté en annexe.

Exemple 3. Le schéma relationnel suivant présente le modèle R-OLAP dénormalisé généré à partir du PIM$_1$ (figure 3) en appliquant les règles détaillées dans la section 5.1.3.

[4] R-OLAP : Relational-On Line Analytical Processing.

[5] O-OLAP : Object-On Line Analytical Processing.

[6] H-OLAP : Hybrid- On Line Analytical Processing.

Client_dim (codeC, nom, prénom, ville, numDept, libDept, région)
Produit_dim (codeP, désignation, catégorie, classe)
Temps_dim (date, numM, libelléM, trimestre, année)
Ventes (codeC#, codeP#, date#, quantité, montant)

4.3 PSM : Modélisation physique

De manière générale, les démarches de conception des ED s'arrêtent au niveau logique. La démarche que nous proposons va au-delà en couvrant tout le cycle de développement du niveau analyse jusqu'au niveau physique. En effet, chaque modèle logique construit est transformé en un modèle physique en appliquant un ensemble de règles. Nous présentons les règles de passage vers les vues matérialisées (Hobbs, 2005). L'implantation utilisant les vues matérialisées s'avère une solution avantageuse puisque le calcul, le stockage, la mise à jour et le rafraîchissement sont effectués automatiquement par le S.G.B.D[7]. Le PSM correspondant présente le code textuel de l'implantation, de l'alimentation et de la mise à jour des structures mutltidmensionnelles. L'exemple suivant explicite la commande de création de la vue matérialisée qui correspond à la table de fait « Ventes » de l'exemple 3. Chaque attribut de la vue peut être construit à partir d'un seul ou d'une combinaison d'attribut(s) source(s). Dans cet exemple, chaque attribut correspond à un attribut source unique provenant d'une seule source : les codes client, les codes produit et les dates sont construit respectivement à partir des attributs sources « codeC », « codeP » et « dateC ». Les quantités et les montants des ventes sont créés respectivement à partir des sommes des quantités et des sommes des montants de commandes.

Exemple 4. A partir d'une source de données relationnelle, le code simplifié de la création d'une vue matérialisée est le suivant :

```
CREATE MATERIALIZED VIEW Ventes
BUILD IMMEDIATE
REFRESH COMPLETE ON DEMAND
AS SELECT    codeC, codeP, dateC AS date,
             SUM (quantité) AS quantite, SUM (montant) AS montant
FROM commande co, ligne_com lc
WHERE co.refc = lc.refc
GROUP BY codeC, codeP, dateC;
```

5 Transformation de modèles basée sur QVT (Query/View/Transformation)

Cette section présente la formalisation en QVT des règles de transformation permettant un passage automatique entre le modèle conceptuel et les modèles logiques de notre démarche. QVT est un langage déclaratif standardisé par l'OMG. Une transformation QVT entre deux modèles candidats est spécifiée grâce à un ensemble de relations. Chaque transformation est composée des éléments suivants :

- « Domains » : chaque domaine désigne un modèle candidat et un ensemble d'éléments à relier.

[7] S.G.B.D : Système de Gestion de Bases de Données.

- « Relation Domain » : permet de spécifier le type de relation entre les domaines, elle peut être marquée comme « Checkonly » (C) ou « Enforced » (E). Un domaine « Chekonly » permet de vérifier s'il existe une correspondance valide qui satisfait la relation ; alors qu'un domaine « Enforced » permet de créer un élément dans le modèle si le lien de correspondance n'est pas vérifé. Pour chaque domaine le nom de son méta-modèle sous-jacent doit être spécifié.

- La clause « When » : décrit les pré-conditions qui doivent être remplies pour réaliser la transformation.

- La clause « Where »: détermine les post-conditions qui doivent être remplies par tous les éléments du modèle participant à la relation.

Une transformation contient deux types de relations : les relations « top-level » et les relations « non-top-level ». L'exécution d'une transformation nécessite que toutes les relations « top-level » soient exécutées, alors que les relations « non-top-level » doivent être exécutées quand elles sont invoquées directement ou transitivement à partir de la clause « Where » d'une autre relation (OMG, 2009).

5.1 Transformation PIM_1-PIM_2

5.1.1 Méta-modèle source

Une transformation permet la génération automatique d'un modèle cible à partir d'un modèle source en appliquant un ensemble de règles. Elle nécessite de spécifier les méta-modèles décrivant ces modèles. Pour transformer le PIM conceptuel de l'ED vers le PIM logique R-OLAP, le méta-modèle source est défini par le profil UML présenté par la figure 2. Notons que toutes les méta-classes de ce profil sont descendantes directes ou indirectes de la méta-classe « ModelElement ». Cette dernière est caractérisée par un attribut « name » (nom de l'élément). Ceci justifie l'utilisation de cet attribut lors de la formalisation des règles de transformation en QVT dans la section 5.1.3.

5.1.2 Méta-modèle cible

Le modèle conceptuel de l'entrepôt est transformé en un modèle logique R-OLAP. Ce dernier est décrit par le méta-modèle relationnel du CWM (OMG, 2003) explicité par la figure 5. Ce méta-modèle présente les tables relationnelles comme un ensemble de colonnes, de clés primaire et d'éventuelles clés étrangères.

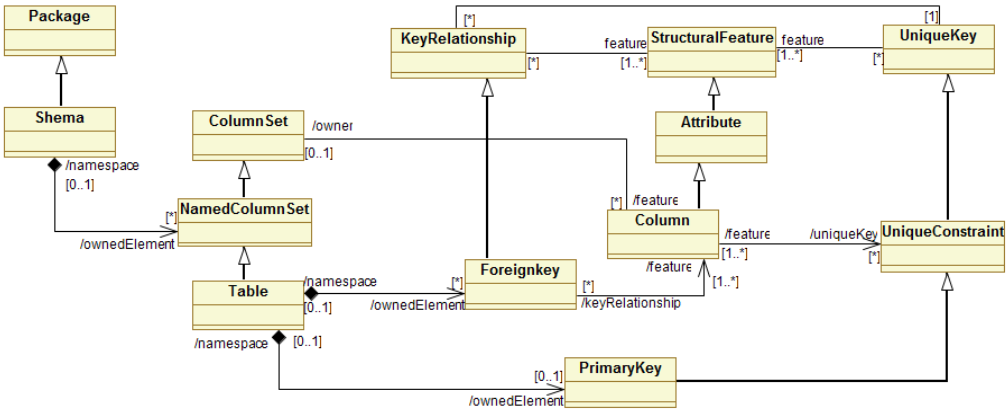

Figure 5. Extrait du méta-modèle CWM relationnel

5.1.3 Transformation QVT

Le modèle conceptuel de l'entrepôt est traduit en R-OLAP dénormalisé en appliquant les règles suivantes :

- R1 : Tout schéma en constellation est transformé en un schéma relationnel ;
- R2 : Toute dimension est transformée en une table où :
  - Les colonnes = tous les paramètres et les attributs faibles relatifs aux différentes hiérarchies composant cette dimension ;
  - Clé primaire = paramètre du plus bas niveau (appelé paramètre racine).
- R3 : Tout fait est transformé en une table où :
  - Les mesures et les paramètres racines relatifs aux dimensions en relation avec le fait sont transformés en colonnes de cette table ;
  - Les clés étrangères correspondent aux paramètres racines des dimensions en relation avec le fait ;
  - La clé primaire correspond à la concaténation des clés étrangères référençant les dimensions auxquels est lié le fait.

La traduction de ces règles en QVT est constituée de plusieurs relations, nous ne présentons que les relations de type « top level » :

Relation « Main ». Cette relation est le point d'entrée au processus de transformation. La partie gauche de la figure 6 montre les éléments du modèle source (md : DWP) transformés en éléments du modèle R-OLAP dénormalisé (r-olap-d : Relational) présenté par la partie droite de la figure. Un schéma en constellation est transformé en un schéma relationnel de même nom. Chaque fait est transformé en une table via la relation « FactToTable ». Chaque dimension est traduite en une table du schéma relationnel par la relation « DimensionToTable » spécifiée au niveau de la clause « Where ».

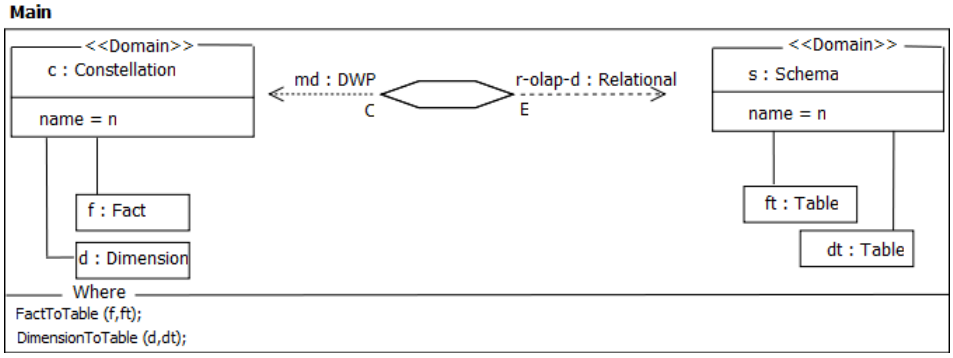

Figure 6. Relation « Main »

Relation « DimensionToTable ». Une dimension est transformée en une table relationnelle dont le nom est la concaténation du nom de la dimension et du suffixe '_dim'. Tous les attributs (les paramètres et les attributs faibles) de la (ou des) hiérarchie(s) composant cette dimension sont transformés en colonnes de la table grâce à la relation « LevelAttributeToColumn ». Le paramètre racine détermine la clé primaire de cette table via la relation « ParameterToPrimaryKey » définie par la clause « Where ».

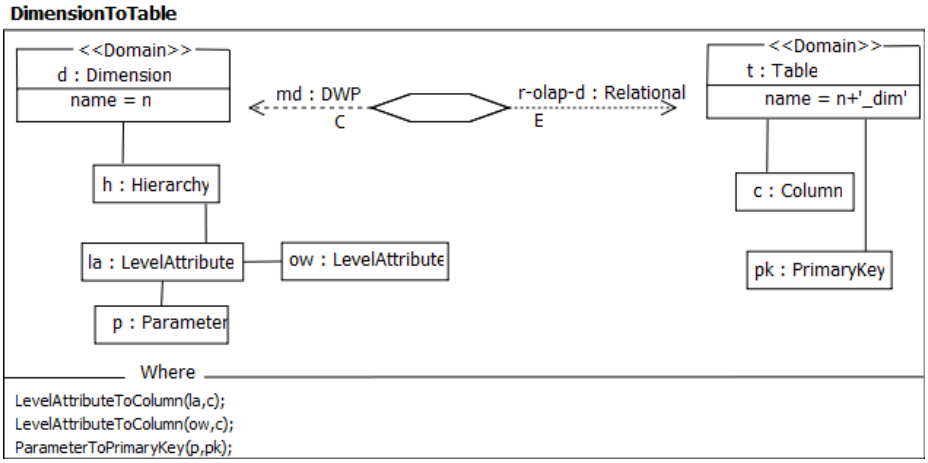

Figure 7. Relation « DimensionToTable »

Relation « FactToTable ». Une fois les dimensions liées au fait transformées (la pré-condition « DimensionToTable » de la clause « When »), le fait est converti en une table ayant le même nom. Les mesures sont transformées en colonnes au moyen de la relation « MeasureToColumn » de la clause « Where ». Les paramètres racines (« la.h_owner = 1 ») des dimensions liées au fait sont transformés en clés étrangères par la relation « ParameterToForeignKey » et par la suite en clés primaires grâce à la relation « ParameterToPrimaryKey ».

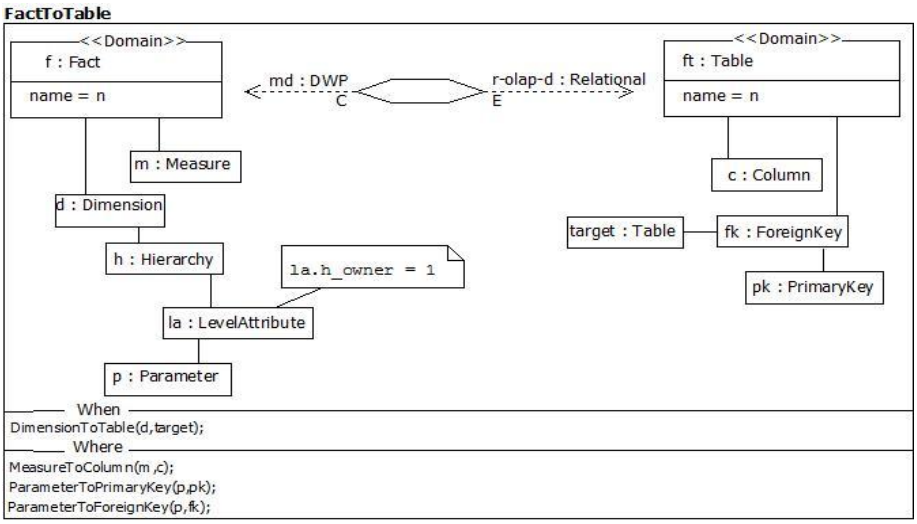

Figure 8. Relation « FactToTable »

5.2 Transformation PIM-PSM

Les modèles physiques présentés par les vues matérialisées sont déduits par composition des modèles conceptuel (PIM_1) et logique (PIM_2). Il s'agit de transformations multi-modèles : un ou plusieurs modèle(s) en entrée sont transformés en un ou plusieurs modèle(s) en sortie. En effet, chaque dimension du modèle conceptuel est traduite en une dimension physique implanté sous oracle (Hobbs, 2005) : les hiérachies, les paramètres et les attributs faibles sont transformés respectivement en hiérachies, niveaux et attributs. Le second modèle en entrée est le modèle logique. A ce niveau chaque table (de fait ou de dimension) est transformée en une vue matérialisée, les colonnes relatives à chaque table sont traduites en attributs de la vue matérialisée correspondante. Les formules d'extraction stockées au niveau des opérations ETL du PIM_1 définissent les attributs sources invoqués par les commandes de création des vues matérialisées.

6 Conclusion

L'approche présentée dans cet article vise à faciliter la tâche de conception d'entrepôts et à fournir plusieurs plateformes d'implantation possibles. Pour répondre à cet objectif, nous avons proposé une démarche unifiée, mixte et automatique pour la conception d'ED. Cette démarche permet de couvrir le cycle de modélisation d'ED : l'analyse des besoins ainsi que les modélisations conceptuelle, logique et physique. L'avantage de cette approche est qu'elle permet de décrire la conception d'ED et des processus ETL associés. Cette démarche est fondée sur une approche mixte permettant de construire le modèle multidimensionnel à partir des besoins des décideurs et de le valider par rapport aux sources opérationnelles. Il s'agit d'une approche dirigée par les modèles permettant d'automatiser le passage entre les différents niveaux de modélisation et de fournir nombreuses alternatives pour l'implantation physique. Les règles de transformation ont été définies en utilisant le langage QVT.

Nous envisageons dans les travaux futurs d'étendre la phase d'analyse des besoins et d'automatiser le passage du modèle des exigences vers le modèle conceptuel en définissant un ensemble de règles QVT. Nous envisageons également de formaliser en QVT les règles de

passage vers les vues matérialisées et de définir des règles de transformations vers d'autre plateformes. Actuellement, une partie des règles de transformation (du PIM conceptuel vers le R-OLAP) est implantée en utilisant le plugin Eclipse « MediniQVT ». Nous envisageons l'implantation de nouvelles règles et l'évaluation de la démarche globale auprès d'un ensemble d'utilisateurs de profils différents. Il est aussi envisageable de prendre en compte des sources de données de nature différentes (structurées, semi-structurées, voire non structurées). Ceci nécessite la définition de différents modèles en fonction de la nature des sources (Ravat, 2007). Il serait indispensable de définir des opérations ETL différentes en fonction du type de données contenues dans les sources.

Références

(Annoni et al., 2006) E. Annoni, F. Ravat, O. Teste, G. Zurfluh "Towards Multidimensional Requirement Design". 8th international conference on Data Warehousing and Knowledge Discovery (DAWAK'06), p.75-84, Krakow (Poland), September 2006.

(Blanc & Salvatori, 2005) X. Blanc, O. Salvatori "MDA En Action : Ingénierie Logicielle Guidée Par Les Modèles". Paris : Eyrolles, 2005.

(Essaidi & Osmani, 2009) M. Essaidi, A. Osmani "A Data Warehouse Development Using MDA and 2TUP". 18th international conference on Software Engineering and Data Engineering (SEDE'09), Las Vegas (Nevada, USA), June 22-24, 2009.

(Gam, 2008) I. Gam "Ingénierie des Exigences pour les Systèmes d'Information Décisionnels : Concepts, Modèles et Processus, la méthode CADWE". Thèse de doctorat en Informatique de l'Université Paris I - Panthéon - Sorbonne, octobre 2008.

(Golfarelli &.Rizzi, 1998) M. Golfarelli, S. Rizzi "Methodological framework for data warehouse design". 1st international workshop on Data Warehousing and OLAP (DOLAP'98), Bethesda (Maryland, USA), p. 3-9, November 7, 1998.

(Husemann et al., 2000) B. Husemann, J. Lechtenborger, J. Vossen "Conceptual data warehouse Modelling". 2nd international workshop on Design and Management of Data Warehouse (DMDW'00), Stockholm (Sweden), p. 6.1-6.11, June 5-6, 2000.

(Hobbs, 2005) L. Hobbs "Oracle Materialized Views & Query Rewrite". An Oracle White Paper, mai 2005.
http://www.oracle.com/technetwork/database/features/bi-datawarehousing/twp-bi-dw-materialized-views-10gr2--131622.pdf

(Kadima, 2005) H. Kadima "MDA : Conception Orientée Objet Guidée Par Les Modèles". Paris : Dunod, 2005.

(Kimball & Ross, 2002) R. Kimball, M. Ross "The data warehouse toolkit : the complete guide to dimensional modeling". 2nd edition, Wiley, 2002.

(Mazon & Trujillo, 2008) J-N Mazon, J. Trujillo "An MDA approach for the development of data warehouses". Decision Support Systems (DSS'08), Vol. 45 (1), p.41-58, April 2008.

(Mazon & Trujillo, 2009) J-N. Mazon, J. Trujillo "A Hybrid Model Driven Development Framework for the Multidimensional Modeling of Data Warehouses". Special Interest Group on Management Of Data (SIGMOD'09), Providence (Rhode Island), Vol. 32 No.2, June 29-July 2, 2009.

(Munoz et al., 2009) L. Munoz, J-N Mazon, J. Trujillo "Automatic Generation of ETL processes from Conceptual Models". 12th international workshop on Data warehousing and OLAP (DOLAP'09), Hong Kong (China), p.33-40, November 2009.

(OMG, 2003) Object Management Group, "Common Warehouse Metamodel (CWM) Specification", version 1.1, Vol. 1. http://www.omg.org/spec/CWM/1.1/, visité février 2010.

(OMG, 2009) Object Management Group, "Meta Object Facility (MOF) 2.0 Query/View/Transformation", version 1.1, http://www.omg.org/spec/QVT/1.1/Beta2/, December 2009, visité février 2010.

(OMG, 2010) Object Management Group, "Object Constraint Langage", (version 2.2) http://www.omg.org/spec/OCL/2.2/ , visité février 2010.

(Poole, 2003) J. Poole "Model Driven Data Warehousing (MDDW)", OMG Announces Integrate, Burlingame (California, USA), January 28-29, 2003, http://www.cwmforum.org/POOLEIntegrate2003.pdf , revisité aoûtt 2010.

(Prat et al., 2006) N. Prat, J. Akoka, I. Comyn-Wattiau. "A UML-based data warehouse design method", Decision Support Systems (DSS'06), Vol. 42(3), p. 1449-1473, 2006.

(Ravat et al., 2006) F. Ravat, O. Teste, G. Zurfluh "A multiversion-based multidimensional model". 8th international conference on Data Warehousing and Knowledge Discovery (DAWAK'06), p.65-74, Krakow (Poland), September 2006.

(Ravat, 2007) F. Ravat "Modèles et outils pour la conception et la manipulation de systèmes d'aide à la décision". Habilitation à diriger des recherches, Université Toulouse 1 Capitole, décembre 2007.

(Ravat et al., 2007) F. Ravat, O. Teste, R. Tournier, G. Zurfluh " Querying Multidimensional Databases". 11th East-European conference on Advances in Databases and Information Systems (ADBIS'07), p.298-313, Varna (Bulgarie), septembre 2007.

(Rizzi et al., 2006) S. Rizzi., J. Trujillo, A. Abello "Research in Data Warehouse Modeling and Design: Dead or Alive?". 9th international workshop on Data warehousing and OLAP, (DOLAP'06), Arlington (Virginia, USA), p. 6-10, November 2006.

(Romero & Abello, 2010) O. Romero, A. Abello, "Automatic validation of requirements to support multidimensional design". Data & Knowledge Engineering (DKE'10), Vol. 69(9), p. 917-942, September 2010.

(Simitsis & Vassiliadis, 2003) A. Simitsis, P. Vassiliadis “A methodology for the conceptual modeling of ETL processes”. 15th international Conference on Advanced Information Systems Engineering workshops (CAiSE’03), Vol. 30 (7), p. 492-525, November 2005.

(Teste, 2009) O. Teste “ Modélisation et manipulation des systèmes OLAP : de l'intégration des documents à l'usager ”. Habilitation à diriger des recherches, Université Paul Sabatier, décembre 2009.

(Vassiliadis, 2009) P. Vassiliadis “A survey of Extract-transform-Load technology”. International Journal of Data Warehousing & Mining (IJDWM’09), Vol. 5(3), p. 1-27, July-September, 2009.

(Zepda et al., 2008) L. Zepda, L. Celma, R. Zatarain “Mixed Approach for Data Warahouse Conceptual Design with MDA”. International Conference on Computational Science and Its Applications (ICCSA’08), p. 1204-1217, 2008.

Annexe

La modélisation logique en R-OLAP normalisé se distingue de la modélisation en R-OLAP dénormalisé ; les tables dimensions sont normalisées conformément aux principes de la 3ème forme normale. Les règles de transformations restent donc les mêmes pour le fait. Alors que les dimensions sont normalisées ; chaque niveau d’attribut est transformé en une table relationnelle. Les relations de composition entre les différents niveaux d’une même hiérarchie sont traduites par des clés étrangères référençant le niveau de granularité suivante. Les relations QVT « top-level » sont définies comme suit.

Relation « Main ». Cette relation permet de transformer un schéma en constellation en un schéma relationnel. De la même manière qu’en R-OLAP dénormalisé les faits et les dimensions sont transformés en tables en appliquant respectivement les relations « FactToTable » et « DimensionToTable » de la clause « Where ». Les niveaux d’attributs non racines (définis par la contrainte « la.h_owner = 0 ») sont également transformés en tables via la reltation « LevelAttributeToTable ».

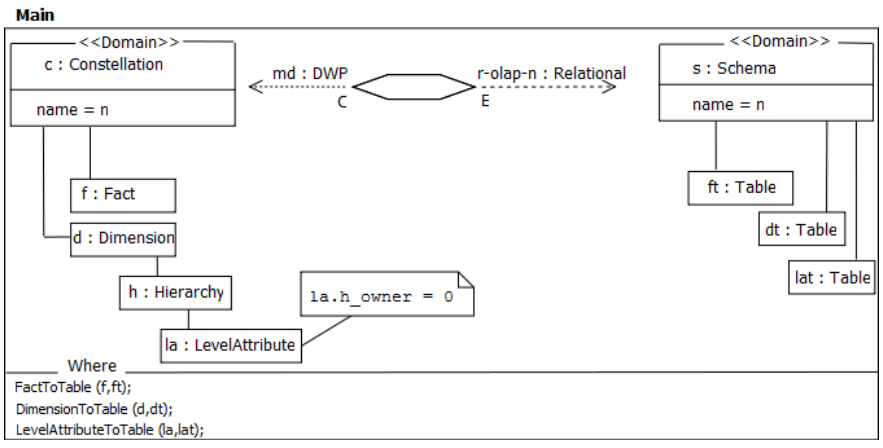

Figure 9. Relation « Main »

Relation « DimensionToTable ». Cette relation permet de transformer une dimension en une table ayant le même nom suffixé par '_dim'. Le paramètre racine (défini par la contrainte « la.h_owner = 1») est transformé en une clé primaire de cette table (« ParameterToPrimaryKey » de la clause « Where »). Les attributs faibles éventuellement liés à ce paramètre sont transformés en colonnes via la relation « LevelAttributeToColumn ». Une table dimension contient également une clé étrangère vers la table du niveau de granularité suivant (défini par la contrainte « la.owner = ow.levelAttribute »). La définition d'une clé étrangère exige que le niveau à référencer soit déjà transformé en une table. Ceci est spécifié par la pré-condition de la clause « When ».

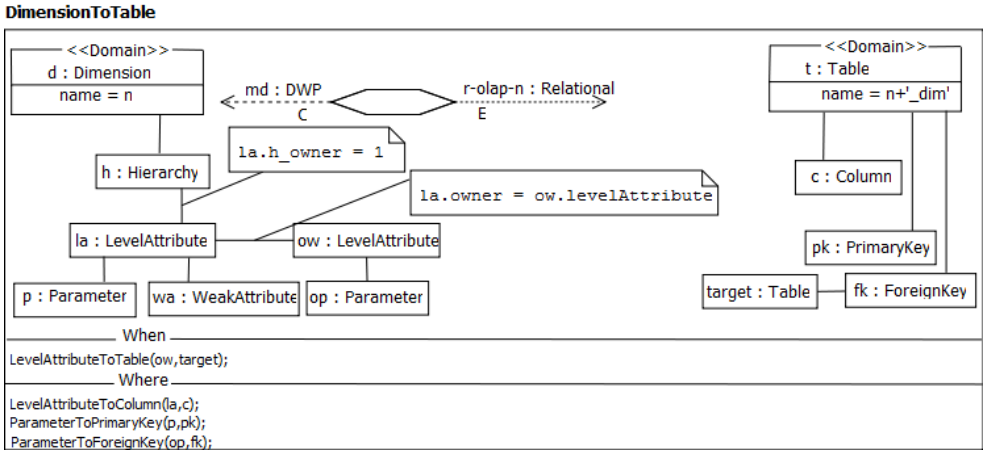

Figure 10. Relation « DimensionToTable »

Relation « FactToTable ». Identique au R-OLAP dénormalisé (figure 8).

Relation « LevelAttributeToTable ». Tous les niveaux non racines sont transformés en une table ayant le même nom concaténé avec le suffixe '_level'. La clé primaire de cette table est le paramètre associé à ce niveau (transformé via la relation « ParameterToPrimaryKey » de la clause « Where »). Les attributs faibles liés à ce paramètre sont transformés en colonnes de cette table (« WeakAttributeToColumn »). Cette table contient également une clé étrangère vers la table du niveau de granularité suivante et appartenant à la même hiérarchie. Ceci n'est possible que si le niveau en cours de transformation n'est pas celui de plus haute granularité (défini par les conditions des clauses « When » et « Where »).

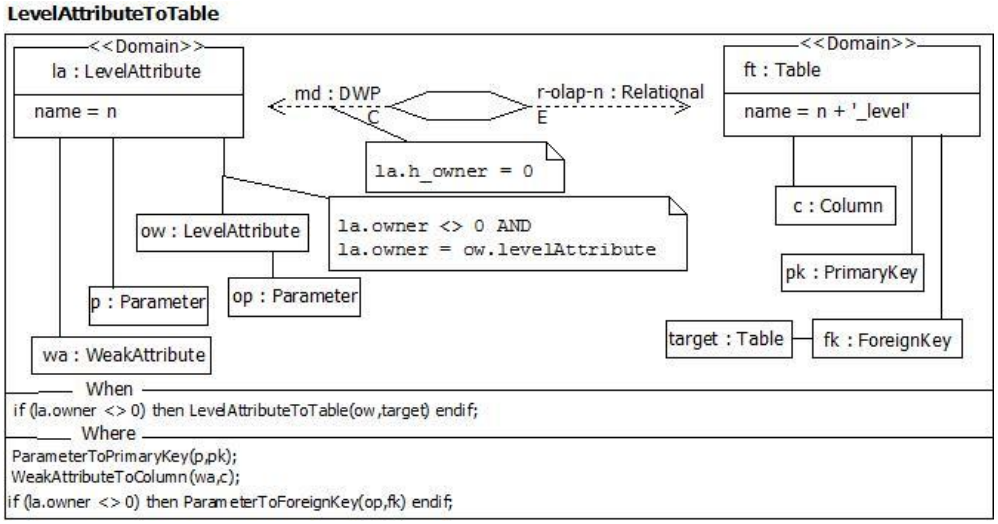

Figure 11. Relation « LevelAttributeToTable »