Un cadre conceptuel basé sur UML et Spatial OCL pour la définition des contraintes d'intégrité dans les systèmes SOLAP

Kamal Boulil*, Sandro Bimonte*
François Pinet*

*Irstea, UR TSCF, 24 Avenue des Landais, 63172 AUBIERE, France
kamal.boulil@irstea.fr,
sandro.bimonte@irstea.fr,
francois.pinet@irstea.fr

Résumé. Les entrepôts de données spatiales et les systèmes Spatial OLAP sont des technologies d'aide à la décision permettant l'analyse en ligne des gros jeux de données géo-référencées. Dans un tel type de systèmes, la qualité de l'analyse dépend : de la qualité des données entreposées, de comment les agrégations sont effectuées, et de comment les données entreposées sont explorées. Dans cet article, nous proposons une méthode pour garantir la qualité de ces trois facteurs, fondée sur un profil UML et sur des contraintes d'intégrité définies avec Spatial OCL. Afin de valider notre proposition, nous proposons également une implémentation automatique dans une architecture ROLAP.

1 Introduction

Les Entrepôts de Données Spatiales (EDS) et les systèmes Spatial OLAP (SOLAP) fournissent de nouvelles possibilités d'analyse spatio-multidimensionnelle de l'information géolocalisée (Bédard et al., 2006). Les EDS ont été définis comme "une collection de données spatiales et non spatiales orientées sujet, intégrées, non volatiles et variant dans le temps disponible pour le support du processus de prise de décision" (Stefanovic et al., 2000). Les données spatiales entreposées sont analysées au moyen de systèmes SOLAP qui ont été définis par Yvan Bédard comme des "plates-formes visuelles conçues spécialement pour faciliter une analyse spatio-temporelle rapide et simple, ainsi qu'une exploration des données, en suivant une approche multidimensionnelle basée sur des niveaux d'agrégation visualisés dans des affichages cartographiques ainsi que dans des tableaux et des diagrammes" (Bédard et al., 2006). Comme les EDS intègrent des données issues de plusieurs sources hétérogènes, plusieurs problèmes liés à leur qualité émergent. Afin d'améliorer la qualité des données dans les EDS, certaines approches ont été proposées pour "réparer" les données au moyen de techniques statistiques, de techniques de fouille de données, etc. (Ribeiro et al., 2011). Dans le même temps, les Contraintes d'Intégrité (CI) sont reconnues comme des méthodes efficaces pour exprimer des règles pour contrôler la cohérence logique des EDS (Salehi, 2009). En outre, la qualité des analyses spatio-multidimensionnelles dépend aussi de l'agrégation des mesures. Cette agréga-

tion doit se faire selon des conditions d'agrégeabilité (ou contraintes d'agrégation) (Lenz et Shoshani, 1997). Toutefois, dans les systèmes SOLAP, il faut également mettre en place un contrôle au cours de l'exploration des données (agrégées), ceci afin d'éviter des interprétations erronées des résultats des requêtes SOLAP (Levesque et al., 2007). D'autre part, la modélisation conceptuelle des ED est reconnue comme un élément indispensable à la réussite des projets de Businnes Intelligence (Torlone, 2003). Dans ce contexte, UML (Unified Modeling Language) est considéré comme le standard orienté-objet pour la modélisation des différents aspects des systèmes logiciels, et également des systèmes d'EDS (Aguila et al., 2011). En effet, UML offre un outil puissant de formalisation aux concepteurs et décideurs, lors des phases d'élaboration et de mise en oeuvre des schémas de données. Il peut être également interprété par des Ateliers de Génie Logiciel (AGL). De la même manière, la définition des CI à un niveau conceptuel permet de considérer les problèmes de qualité dès les premiers stades des développements (Boulil et al., 2011). Nous pensons que le langage de contraintes OCL (Object Constraint Language), constitue une solution efficace pour définir les CI pour les EDS (Boulil et al., 2011) à un niveau conceptuel, de manière claire, non ambiguë et indépendante des plateformes d'implantation cibles. Ainsi, dans ce papier, nous présentons trois contributions principales.

Pour la première, nous étendons/reformulons la définition de CI dans les systèmes d'EDS afin de considérer tous les aspects qui peuvent impacter la qualité de l'analyse (spatio-) multidimensionnelle dans ces systèmes. Nous utilisons les CI pour effectuer trois types de contrôle de qualité : (A) pour garantir la cohérence logique des données (spatiales) entreposées (e.g., les géométries des villes doivent être incluses dans les géométries de leurs pays) ; (B) pour veiller à ce que l'agrégation des mesures soit correctement faite et ait bien un sens (e.g., la somme des prix unitaires des produits n'a pas de sens) ; (C) pour éviter les problèmes d'interprétation liés à des requêtes insensées (e.g., indiquer à l'utilisateur que demander les ventes en URSS après le 26 Décembre 1991 ne peut pas retourner de résultat) (Levesque et al., 2007)).

Deuxièmement, motivés par un manque de cadre conceptuel unifié pour définir les CI des systèmes SOLAP, nous proposons une méthode de modélisation conceptuelle entièrement basée sur UML et Spatial OCL (Pinet et al., 2007).

Enfin, nous proposons la mise en oeuvre automatique d'une telle méthode au sein d'une architecture SOLAP relationnelle classique.

Le papier est structuré comme suit : la section 2 introduit les principaux concepts liés aux EDS, ainsi que les travaux sur les contraintes d'intégrité. Dans la section 3 nous proposons une nouvelle classification des problèmes de qualité dans les systèmes SOLAP. La section 4 décrit notre méthode conceptuelle, puis à la section 5, nous détaillons sa mise en oeuvre automatique dans un système SOLAP.

2 Etat de l'art

La nécessité croissante d'intégrer l'information spatiale dans l'analyse multidimensionnelle a conduit aux concepts d'EDS et du SOLAP. Ces systèmes se basent sur le modèle spatio-multidimensionnel, qui étend le modèle multidimensionnel classique, et permet de représenter l'information spatiale en axes d'analyse (dimension spatiale) et/ou en mesures (mesure spatiale). La représentation des données spatiales dans les niveaux de dimension permet de visualiser les résultats des requêtes décisionnelles à l'aide de cartes, et d'utiliser des prédicats

spatiaux pour la sélection et le regroupement des données lors de l'application des opérateurs OLAP comme le RollUp et le Slice (Bédard et al., 2006). Pour illustrer nos propositions tout au long de ce papier nous allons utiliser un même exemple d'EDS. Il s'agit d'une application qui permet l'analyse des températures selon le temps (dimension temporelle) et l'espace (dimension spatiale). La dimension temporelle est classiquement organisée en trois niveaux : jour, mois, et année. La dimension spatiale représente les villes, leurs régions et leurs pays. Les mesures sont des valeurs de température. Cet EDS permet de répondre à des requêtes SOLAP telles que : "Quelles sont les températures moyennes par mois et par pays ?". Pour répondre à cette requête, les systèmes SOLAP doivent agréger les valeurs de température en utilisant la fonction "moyenne". Il est important de noter que l'utilisation de certaines fonctions d'agrégation sur les valeurs de température n'as pas de sens (par exemple, sommer les températures n'a pas de sens puisque la température est une mesure non additive). L'architecture d'un SOLAP Relationnel classique est composée de trois niveaux. Le niveau EDS intègre des données à partir de sources multiples et les gère en utilisant un SGBD Relationnel, ce qui permet une meilleure évolutivité et des bonnes performances. Le serveur SOLAP implémente les opérateurs SOLAP pour le calcul et manipulation des cubes de données spatiales. Enfin, le client SOLAP met à disposition des décideurs une série d'interfaces graphiques simples d'utilisation qui déclenchent des opérateurs SOLAP et qui permettent une exploration interactive et une visualisation sous différents formats des données entreposées.

La gestion de la qualité dans les ED(S) est une question de recherche importante. La qualité des données spatiales dépend de plusieurs facteurs : la précision, la complétude, et la cohérence logique. La précision dans les EDS a été traitée par (Siqueira et al., 2011) et (Perez et al., 2007) qui proposent des modèles logiques et une technique d'indexation pour le stockage et l'interrogation des données spatiales vagues. La complétude a été seulement considérée dans les ED classiques, comme par exemple dans le travail de (Dyreson et al., 2003) qui propose différentes techniques pour la prise en compte des valeurs manquantes dans les hiérarchies. La cohérence logique fait référence à l'existence de contradictions logiques dans les EDS. Ces inconsistances sont classiquement contrôlées pas les CI. Comme indiqué dans (Boulil et al., 2011), l'expression des CI sur des modèles conceptuels est essentiel pour la prise en compte de toutes les règles de qualité et leur validation par les utilisateurs. Dans ce contexte, (Carpani et Ruggia, 2001) et (Ghozzi et al., 2003) proposent des modèles conceptuels multidimensionnels ad-hoc permettant l'expression de certaines CI de données au moyen de prédicats logiques. (Malinowski et Zimanyi, 2008) proposent une extension du modèle ER pour la conception d'entrepôts de données spatio-temporelles. Il définit un ensemble de pictogrammes ad hoc pour exprimer des CI sur les données spatiales (des relations topologiques entre les membres spatiaux). (Glorio et Trujillo, 2008) proposent un profil UML pour les EDS, mais ils ne considèrent qu'un très petit nombre de CI de données. Un état de l'art sur les problèmes d'agrégation est présenté dans (Mazon et al., 2009). Ces auteurs définissent des contraintes de schéma simples avec les multiplicités d'UML (e.g., les faits doivent être liés aux dimensions avec des associations un-à-plusieurs). Dans (Pinet et Schneider, 2009), des contraintes d'agrégation structurelles complexes sont exprimées avec OCL. En se basant sur un modèle UML, (Boulil et al., 2011) montre que le langage Spatial OCL permet la définition d'un très grand nombre de CI sur des données spatiales entreposées. Spatial OCL est une extension d'OCL pour les données spatiales (Pinet et al., 2007). (Boulil et al., 2011) proposent également une mise en oeuvre automatique dans le SGBD Oracle 11g. Enfin, (Levesque et al., 2007) proposent un cadre pour

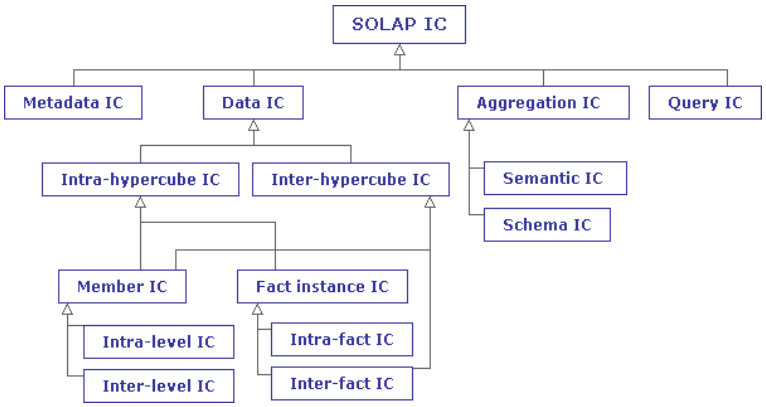

FIG. 1 – *classification des CI SOLAP*

l'identification des risques de qualité dans les ETL et les systèmes SOLAP. Ils définissent 3 points impactant la qualité : les sources de données, les cubes de données et les fonctionnalités des Systèmes d'Information Géographiques (SIG). Ils proposent un ensemble de formulaires papier pour définir ces problèmes de qualité qu'ils mettent en oeuvre dans le système JMAP-SOLAP. Enfin, au meilleur de notre connaissance, aucun travail ne propose une méthode basée sur des standards pour exprimer au niveau conceptuel les CI pour contrôler la cohérence : des données entreposées ; des agrégations ; des requêtes spatio-multidimensionnelles. Aussi, il n'existe aucune proposition permettant l'implémentation automatique de ces CI dans une architecture SOLAP relationnelle classique.

3 Contraintes d'Intégrité pour le Spatial OLAP

Dans cette section, nous présentons une extension de notre précédente classification des CI proposée dans (Boulil et al., 2011). Nous introduisons une nouvelle classe, la classe des CI de requêtes (Query IC) qui a pour but d'éviter de mauvaises interprétations des résultats de requêtes insensées ou impossibles (Levesque et al., 2007). Cette classification (figure 1) sert de guide de référence pour le processus de traitement des trois principaux types de problèmes de qualité dans les systèmes SOLAP. Nous fournissons ici l'explication de cette classification et donnons des exemples utilisant l'étude de cas décrite précédemment. Comme le montre (Boulil et al., 2011), les CI peuvent être utilisées pour vérifier la cohérence des méta-données (Metadata IC) des différentes sources d'information intégrées dans l'entrepôt ; par exemple, les membres et mesures spatiales doivent être définis à la même échelle géographique.

Les CI sur les données (IC Data) assurent la cohérence logique et la complétude des données spatiales entreposées :

Exemple 1 : "La géométrie d'une ville doit être incluse dans la géométrie de sa région"

Exemple 2 : "Aucun fait (par exemple, une valeur de température) ne doit exister pour l'URSS après le 26 Décembre 1991".

Ces contraintes peuvent concerner tous les éléments de l'EDS (les faits, les membres, etc.)

Les CI d'agrégation (Aggregation IC) permettent de garantir une agrégation correcte et sensée des mesures. En particulier, les contraintes d'agrégation sémantiques vérifient l'applicabilité des fonctions d'agrégation aux mesures, en considérant les types et natures sémantiques des fonctions d'agrégation, des mesures et des dimensions :

Exemple 3 : "La somme des températures n'a pas de sens"

Exemple 4 : "Il n'est pas possible d'agréger des températures (mesure numérique) en utilisant la fonction d'agrégation spatiale Union Spatiale ".

Les contraintes de schéma sont des conditions qui doivent être satisfaites par les hiérarchies de dimension et relations fait-dimension, afin d'éviter les problèmes du comptage en double de mesures et d'agrégats incomplets. Par exemple, les dimensions et les faits doivent être liés par des relations un-à-plusieurs (Mazon et al., 2009) (par exemple pour une date et une ville, il n'est possible d'enregistrer qu'une valeur de température).

Les CI sur les requêtes (Query IC) définissent des conditions sur les requêtes SOLAP pour éviter les problèmes d'interprétation des résultats de requêtes insensées ou impossibles (requêtes dont le résultat est toujours vide). Par exemple, la requête SOLAP "Quelles sont les températures moyennes en URSS en 2010 ?" renvoie un résultat vide puisqu'aucune valeur de température n'est stockée pour l'URSS après le 26 Décembre 1991. Même si aucune valeur n'existe dans l'EDS à partir de cette date (26 Décembre 1991) pour ce pays (CI de données de l'exemple 2), les outils SOLAP existants n'empêchent pas les décideurs de formuler cette requête en combinant les deux membres "URSS" et "2010" ; ce qui retourne un résultat vide. Ceci conduit à un problème d'interprétation : le décideur pourrait percevoir ce résultat vide comme une absence de valeurs de température enregistrées pour l'URSS en 2010, au lieu de constater que sa requête est définie par une combinaison impossible de membres (URSS et 2010). Par conséquent, pour éviter cette interprétation erronée, nous pouvons définir la contrainte de requête suivante :

Exemple 5 : "Il est incorrect de combiner l'URSS avec des jours ultérieurs au 26 Décembre 1991".

A noter que cette contrainte pourrait être résolue en utilisant d'autres techniques comme par exemple le versioning des ED (Arigon et al., 2007) ou les ED actifs.

4 Méthode proposée

Dans cette section, nous présentons et illustrons avec des exemples, notre méthode pour la modélisation conceptuelle des CI SOLAP (Section 4.2) après avoir introduit les principaux concepts sur lesquels se base cette méthode : les profils UML et Spatial OCL (Section 4.1).

4.1 Rappel sur Spatial OCL et profils UML

Les profils UML constituent un moyen pour adapter UML à des domaines ou des plateformes particuliers. Ils permettent l'extension des méta-classes d'UML (classe, propriété, etc.) grâce à trois mécanismes : les stéréotypes, les valeurs marquées et les contraintes. Un stéréotype est une extension d'une méta-classe UML. Il est représenté par la notation «nom du stéréotype» et/ou une icône. Par exemple, il est possible de créer un stéréotype «Geographic-Class» qui étend la méta-classe "classe" d'UML. Les valeurs marquées sont des méta-attributs. Elles sont définies comme des propriétés de stéréotypes. Enfin, un ensemble de contraintes peut

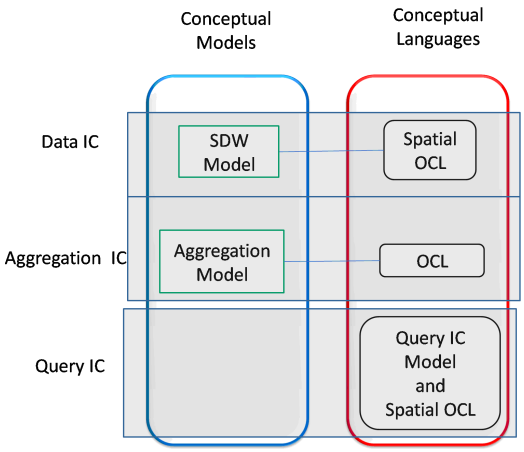

FIG. 2 – *Méthode basée sur UML et Spatial OCL pour la définition conceptuelle des CI SOLAP*

être associé à chaque stéréotype, afin de préciser sa sémantique et de définir ces règles d'utilisation dans les modèles - par exemple pour définir que dans les diagrammes UML, une classe ayant un stéréotype «GeographicClass» doit avoir un attribut géométrique appelé "geometry". Ces contraintes sont souvent définies en utilisant OCL. OCL fournit une méthode générique et indépendante de la plateforme pour exprimer les contraintes au niveau conceptuel. Il peut être interprété par des générateurs de code pour générer du code automatiquement. En effet, certains outils permettent la production de mécanismes de contrôle d'intégrité dans différents langages (Java, SQL, etc.), à partir des spécifications de contraintes exprimées en OCL. Les contraintes OCL peuvent être vérifiées au niveau instance (i.e., sur les objets et ou les données) mais également au niveau modèle (i.e., lors de la définition du modèle de données ou l'instance du profil UML). Pour exprimer des contraintes d'intégrité spatiale sur des données spatiales au niveau conceptuel, Spatial OCL a été proposé. Cette extension d'OCL permet d'exprimer des relations spatiales topologiques (e.g., à l'intérieur, disjoint de, etc.) entre géométries simples et complexes (Pinet et al., 2007). Voici un exemple de contrainte définie en Spatial OCL indiquant que les géométries des villes (instances de la classe City stéréotypée «GeographicClass») ne doivent pas s'intersecter :

context City inv :
City.allInstances()->forAll(c\| c<>self implies c.geo.isDisjoint(self.geo))

4.2 Cadre conceptuel pour la modélisation des CI SOLAP

Afin de définir au niveau conceptuel, des CI sur les données, les agrégations et les requêtes dans les systèmes SOLAP, nous proposons une méthode basée sur un profil UML et Spatial OCL (figure 2). L'idée principale est d'avoir un seul profil UML qui permet de définir 3 métamodèles interconnectés pour spécifier : a) les structures du modèle de données et les contraintes sur les données (le métamodèle de l'EDS) ; b) comment les mesures sont agré-

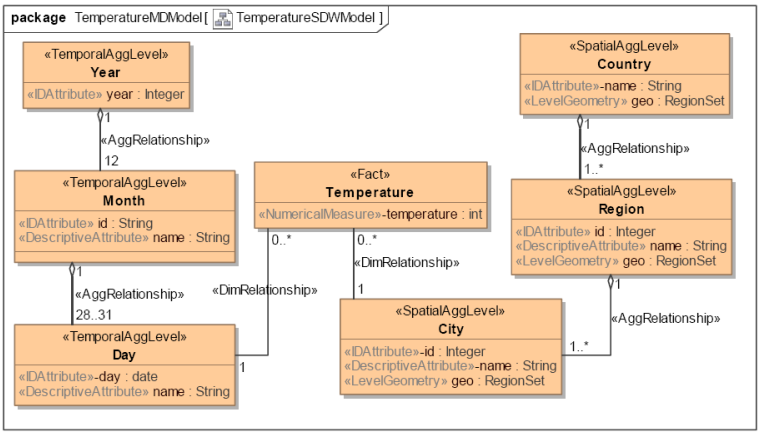

FIG. 3 – *Un exemple de modèle d'EDS - Instance du métamodèle EDS du profil*

gées pour répondre aux besoins d'analyse des décideurs et les CI d'agrégation (le métamodèle d'agrégation) ; et c) les CI sur les requêtes SOLAP (métamodèle de CI de requêtes). Dans notre approche, les CI sur les données sont définies au niveau modèle par les concepteurs. Elles sont exprimées sur le modèle de données en utilisant Spatial OCL. Les contraintes d'agrégation sont spécifiées au niveau métamodèle (i.e., dans la définition du profil UML). Elles sont exprimées en OCL sur les stéréotypes du métamodèle d'agrégation. Les CI sur les requêtes sont définies au niveau modèle par les concepteurs. Elles sont exprimées en utilisant les stéréotypes du métamodèle de CI sur les requêtes SOLAP et Spatial OCL.

Pour des questions de place, nous ne détaillons pas tout le profil que nous proposons, néanmoins nous présentons quelques exemples d'utilisation. Plus de détails sur notre profil peuvent être trouvés dans (Boulil et al., 2011). Dans cet article, nous nous focalisons sur nos nouvelles contributions, à savoir la définition des CI sur les requêtes et les techniques de génération automatique de code utilisées pour l'implémentation des trois familles de contraintes dans différents niveaux de l'architecture SOLAP.

4.2.1 Les CI sur les données

Le métamodèle EDS permet (i) la définition des structures de données de l'EDS et (ii) l'expression des CI sur les données en utilisant Spatial OCL (Boulil et al., 2011). La figure 3 montre le modèle de données (modèle EDS) de l'étude de cas que nous considérons, représenté en utilisant la partie métamodèle EDS de notre profil UML. Ce modèle contient deux dimensions : (i) une dimension spatiale composée de 3 niveaux spatiaux (stéréotypés «SpatialAggLevel»), City, Region et Country, et (ii) une dimension temporelle composée de trois niveaux temporels (stéréotypés «TemporalAggLevel»), Day, Month et Year. La mesure numérique temperature (stéréotypée «NumericalMeasure») est définie comme un attribut de la classe de fait Temperature (stéréotypée «Fact»). Le modèle de données de l'EDS étant défini par une extension d'UML (partie métamodèle EDS du profil UML), nous pouvons maintenant exprimer assez facilement les CI de données sur ce modèle en utilisant Spatial OCL. Consi-

FIG. 4 – *Un exemple de modèle d'agrégation - Instance du métamodèle d'agrégation du profil le d'EDS*

dérons par exemple la contrainte de l'exemple 1. Cette CI de données peut être exprimée en Spatial OCL de la manière suivante :

context Region inv DataIC1 :
self.geo.isInside(country.geo) or self.geo.coveredBy(country.geo)

La CI de données de l'exemple 2 est exprimée en utilisant OCL comme suit :

context Temperature inv DataIC2 :
not (self.day.day >= '1991-12-26' or self.city.region.country.name = 'URSS')

4.2.2 Les CI d'agrégation

Le modèle d'agrégation définit la façon dont les décideurs souhaitent agréger les mesures le long des hiérarchies de dimension, pour calculer les différents indicateurs de l'analyse décisionnelle. Dans notre approche, ce modèle est spécifié au niveau conceptuel en utilisant la partie métamodèle d'agrégation du profil UML.

Le modèle d'agrégation du cas d'étude est représenté en figure 4. Ce modèle définit un seul indicateur d'analyse, Avg_Temperature (moyenne des températures) ; qui est composé d'une seule règle d'agrégation (stéréotype «AggRule»). Cette règle indique que cet indicateur est calculé en agrégeant la mesure temperature (valeur marquée *aggregatedAttribute*) à l'aide de la fonction moyenne (valeur marquée *aggregator*) le long de toutes les dimensions.

Dans (Boulil et al., 2011), nous avons identifié un ensemble de contraintes d'agrégation sémantiques (voir leur définition en Section 3) qui sont valables pour toutes les applications SOLAP. Ces CI sont spécifiées au niveau métamodèle et intégrées à la définition du profil UML : elles sont définies en OCL et associées aux définitions des stéréotypes de la partie métamodèle d'agrégation du profil. De cette façon, elles peuvent être contrôlées au niveau conceptuel dans l'AGL lorsque le concepteur valide le modèle conceptuel. Ceci permet d'éviter l'implémentation de modèles d'agrégation incorrects.

Ci-dessous un exemple de contrainte d'agrégation sémantique, qui permet d'éviter l'agrégation des mesures non-additives (ou mesures de type valeurs par unité) comme la température en utilisant la fonction d'agrégation Sum (voir exemple 3). Cette contrainte est définie en OCL et associée au stéréotype «AggRule» du métamodèle d'agrégation du profil.

context AggRule inv notSumValuePerUnitMeasure :
if (baseIndicator.aggregatedAttribute.OclIsKindOf(Measure)
and baseIndicator.aggregatedAttribute.addType = 'ValuePerUnit')
then aggregator.name <> 'Sum'

Voici un autre exemple de contrainte d'agrégation sémantique, qui vérifie, cette fois, que la fonction d'agrégation est applicable à la mesure (voir exemple 4).

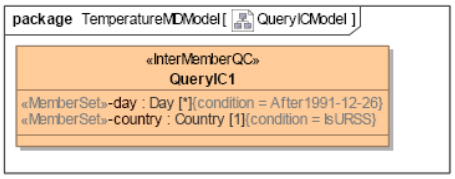

FIG. 5 – *Un exemple de CI de requêtes SOLAP - Instance du métamodèle de CI de requêtes SOLAP du profil*

```
context Day inv After1991-12-26:
self.day >= '1991-12-26'
```

FIG. 6 – *Expression OCL qui sélectionne les jours après le 26/12/1991*

context AggRule inv AggregatorMeasureTypesConformity :
aggregator.applicableTo->exists(dt|
baseIndicator.aggregatedAttribute.oclIsTypeOf(dt))

Cette contrainte OCL vérifie que le type de données de la mesure est inclus dans l'ensemble des types de données auxquels la fonction d'agrégation (aggregator) peut être appliquée.

4.2.3 Les CI de requêtes SOLAP

Les concepteurs peuvent aussi exprimer des CI sur les requêtes SOLAP en utilisant la partie métamodèle des CI sur les requêtes SOLAP du profil. En règle générale, une requête SOLAP est une combinaison de mesures et de membres de différentes dimensions. Ainsi, notre métamodèle de CI sur les requêtes peut être utilisé par exemple pour définir des combinaisons non valides d'ensembles de membres. Dans les contraintes, ces ensembles de membres sont spécifiés par des attributs stéréotypés «MemberSet». Le domaine de valeurs d'un attribut stéréotypé «MemberSet» est un sous-ensemble des membres d'un niveau de dimension. Ce domaine peut être précisé en utilisant une expression OCL qui sélectionne les membres du niveau impliqué dans la contrainte de requêtes SOLAP (valeur marquée *condition*). Un exemple de notre représentation UML des CI sur les requêtes est montré à la figure 5. Dans cette contrainte, on précise que cela n'a pas de sens de combiner dans une requête SOLAP, les jours après le 26 Décembre 1991 (*condition* = *After1991-12-26* - l'expression en OCL correspondante à cette condition est indiquée à la figure 6) avec l'URSS (*condition* = *IsURSS*).

5 Implémentation

Dans cette section, nous présentons l'architecture que nous proposons pour mettre en oeuvre les 3 types de contrôle de qualité dans les systèmes SOLAP (contrôle de données, d'agrégation et des requêtes SOLAP), voir figure 7. L'idée principale est d'offrir un système qui à partir de la représentation conceptuelle, implémente automatiquement chaque type de contraintes dans un niveau différent de l'architecture SOLAP. Selon son type, chaque CI

conceptuelle est automatiquement traduite dans le format supporté par le niveau de l'architecture SOLAP dans lequel elle peut être implémentée. Les CI de données sont converties en utilisant le générateur de code "Spatial OCL2SQL" (Pinet et al., 2007) et implémentées au niveau de l'EDS (i.e., SGBD Spatial). Les CI sur les requêtes sont traduites par un générateur de code MDX (UML2MDX) et sont implémentées au niveau du Serveur et Client SOLAP. Enfin, les CI d'agrégation sont implémentées dans le profil et contrôlées lorsque le concepteur valide le modèle conceptuel de l'EDS dans l'AGL.

L'architecture SOLAP (figure 7) que nous avons considérée pour l'implémentation de ces solutions est basée sur : le SGBD Oracle Spatial 11g, le serveur OLAP Mondrian et le client SOLAP JRubik. Oracle Spatial 11g est un SGBD relationnel qui fournit un support natif pour le stockage et l'interrogation des données spatiales. Mondrian est un logiciel Open Source conçu pour fournir des fonctionnalités OLAP avec une base de données relationnelle. Mondrian comprend une couche logicielle de traitement, qui valide et exécute des requêtes multidimensionnelles MDX, et une couche logicielle pour le calcul des agrégations. MDX (MultiDimensional eXpressions) est un langage de requête "standard" pour les bases de données multidimensionnelles. JRubik est un client SOLAP conçu pour fournir une couche de représentation graphique au-dessus de Mondrian.

5.1 Les CI de données

Afin d'implémenter automatiquement les CI de données dans Oracle 11g Spatial, nous avons utilisé le générateur de code Spatial OCL2SQL (figure 7). Spatial OCL2SQL est un outil Open Source développé en Java et intégrant l'extension spatiale d'OCL appelée OCL 9IM (Pinet et al., 2007). Il génère automatiquement des scripts SQL pour Oracle Spatial à partir de contraintes écrites en Spatial OCL. Afin de réaliser la génération de code, les principaux inputs nécessaires sont : (i) un modèle conceptuel (UML) de données au format XMI, (ii) un fichier de contraintes Spatial OCL et (iii) un fichier de méta-données géométriques (incluant entre autres le système de coordonnées des données spatiales). L'outil génère (a) des scripts pour créer le schéma physique de l'EDS (pour le SGBD Oracle Spatial) et (b) un ensemble de requêtes et de triggers SQL pour implémenter les CI. Des détails concernant les règles de génération de code peuvent être trouvés dans (Pinet et al., 2007).

Par exemple, en utilisant Spatial OCL2SQL, l'expression Spatial OCL de la CI de données de l'exemple 1 est traduite vers la requête Spatial SQL pour Oracle suivante :

*select * from CITIES SELF where not (*
mdsys.sdo_relate(SELF.COUNTRY_GEO, SELF.REGION_GEO,
'mask=contains querytype=window') = 'true' or
mdsys.sdo_relate(SELF.COUNTRY_GEO, SELF.REGION_GEO,
'mask=covers querytype=window') = 'true');

Cette requête sélectionne les villes (les tuples de la table Cities) qui ne satisfont pas la contrainte.

5.2 Les CI d'agrégation

MagicDraw est un AGL, basé sur UML, qui prend en charge OCL au niveau du méta-modèle (c'est-à-dire au niveau du profil UML). Plus précisément, MagicDraw est capable de

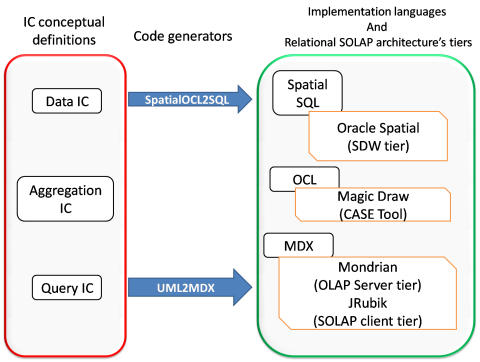

FIG. 7 – *Méthode pour une implémentation automatique des CI SOLAP dans un système Spatial ROLAP*

vérifier des contraintes OCL définies dans le profil UML (métamodèle) au niveau des instances du profil (modèles). Comme dit précédemment, nous avons implémentées un ensemble de contraintes OCL dans notre profil UML, en particulier des CI d'agrégation, afin de les vérifier dés la phase de conception lorsque les concepteurs valident leurs modèles conceptuels dans l'AGL ; ceci permet d'éviter l'implémentation de modèles (d'agrégation et d'EDS) incorrects sémantiquement et structurellement. Il faut noter que cette approche nouvelle et rapide de vérification des contraintes d'agrégation sémantiques est indépendante de tout type d'architecture SOLAP. Par exemple, supposons qu'avec MagicDraw, le concepteur spécifie dans son modèle d'agrégation qu'il souhaite utiliser la fonction Sum pour agréger la mesure temperature (voir figure 8). Une contrainte qui interdit ce type d'agrégation a été définie dans le profil (i.e., interdiction de sommer des mesures non-additives comme la température - voir exemple 3 en Section 4.2). Dans ce cas, MagicDraw informe le concepteur que la contrainte n'est pas satisfaite en mettant en rouge l'indicateur concerné (ici Sum_Temperature) et en affichant un message d'erreur dans la console explicitant le type d'erreur.

5.3 Les CI sur les requêtes

Nous utilisons MDX avec le serveur OLAP Mondrian comme langage et plateforme d'implémentation cibles des CI sur les requêtes SOLAP. A noter que MDX est un standard indépendant de Mondrian. Ainsi notre proposition pourrait être utilisée avec d'autres serveurs OLAP. L'idée principale est de traduire chaque CI de requêtes conceptuelle en une formule MDX qui sera stockée sur le serveur OLAP et visualisée au niveau du client SOLAP avec différentes politiques de visualisation. Cette formule, lorsqu'elle est exécutée, informera l'utilisateur sur la qualité des résultats des requêtes. Pour chaque type de CI de requêtes, nous avons défini un template MDX particulier. Les templates sont remplis à l'aide d'une application Java (UML2MDX) qui analyse les fichiers XMI des CI de requêtes conceptuelles.

La formule MDX correspondant à la CI de requêtes de l'exemple 5 (figure 5 et figure 6) est montrée dans la figure 9.

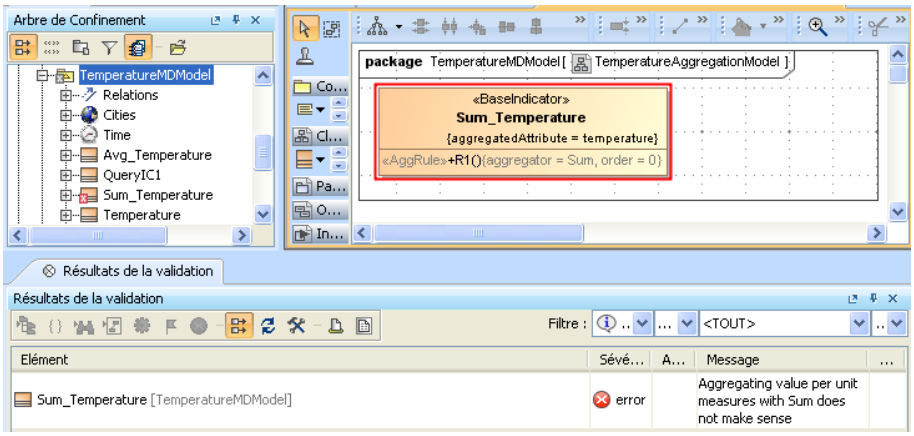

FIG. 8 – *Un exemple de vérification d'une CI d'agrégation sémantique - modèle d'agrégation incorrect*

Avant d'expliquer cette formule MDX, il est important de noter qu'un résultat de requête incorrect peut induire facilement d'autres résultats incorrects. Par exemple, si l'utilisateur effectue un RollUp du niveau Day vers le niveau Month après avoir exécuté une requête correcte (e.g., les températures de l'URSS pour tous les jours antérieurs à 26/12/1991), les agrégats mensuels des températures obtenus ne seront pas tous corrects (l'agrégat du mois de Décembre 1991 inclut des valeurs non valides avec des dates postérieures à 26/12/1991). Pour éviter ces problèmes d'exploration multidimensionnelle, nous prenons en compte dans les formules MDX les liens hiérarchiques des membres en considérant les membres ancêtres et descendants des membres impliqués dans les CI de requêtes.

Le template que nous avons défini pour ce type de CI de requêtes (i.e., combinaisons incorrectes de membres de dimension - exemple 5) est composé d'un ensemble de déclarations MDX. D'abord, pour chaque ensemble de membres de la CI conceptuelle (modélisé par le stéréotype «MemberSet»), trois ensembles calculés sont définis en utilisant l'opérateur MDX SET, et ce afin de sélectionner : (i) les membres du niveau d'agrégation impliqué qui satisfont la condition (valeur marquée "condition" dans la figure 5), (ii) leurs descendants dans la même dimension, et (iii) leurs ascendants. Par exemple, pour l'ensemble de membres («MemberSet») day dans la CI de la figure 5, les ensembles suivants sont calculés : (i) les jours qui sont postérieurs au 26/12/1991, (ii) l'ensemble des descendants de ces jours dans la dimension temporelle, (iii) les ascendants de ces jours (les mois et les années qui comportent des dates postérieures au 26/12/1991).

Pour définir ces différents ensembles, nous utilisons principalement les opérateurs MDX suivants : FILTER (retourne un sous-ensemble de membres de dimension qui satisfont une condition donnée), DESCENDANTS (retourne les descendants d'un membre à un niveau d'agrégation donné), ANCESTOR (retourne l'ancêtre d'un membre à un niveau d'agrégation donné).

Enfin, dans notre implémentation, des modes d'affichage spécifiques sont associés aux différentes combinaisons des membres de ces ensembles (au niveau du client SOLAP). Trois

```
with
 //Day members that satisfy the MemberSet day's condition (days after 1991-12-26)
 set C11MemberSet as 'Filter([Time].[Day].Members, ([Time].[Day].CurrentMember.Name >= "1991-12-26"))'
// Time members that lead to yellow cells, ascendants of Day members that satisfy the day's condition
set C13MemberSet as 'Filter([Time].Members, (Intersect(Descendants([Time].CurrentMember, [Time].[Day], SELF), [C11MemberSet]).Count >0.0))'
//Time members that may lead to invalid cells,members that satisfy the day's condition with their descendants and certain of their
ascendants
  set C12MemberSet as 'Union(Filter([Time].Members, ((Ancestor([Time].CurrentMember, [Time].[Day]).Name >= "1991-12-26") OR
(([Time].CurrentMember IN [C13MemberSet]) AND (Intersect(Descendants([Time].CurrentMember, [Time].[Day], SELF), [C11MemberSet]).Count =
Descendants([Time].CurrentMember, [Time].[Day], SELF).Count)))), [C11MemberSet])'

 //Country members that satisfy the MemberSet country's condition (name = URSS)
  set C21MemberSet as 'Filter([Cities].[Country].Members, ([Cities].[Country].CurrentMember.Name = "URSS"))'
// Cities members that lead to yellow cells, ascendants of Country members that satisfy the country's condition
  set C22MemberSet as 'Union(IIf(([Cities].CurrentMember.Level.Name = "Country"), {}, Filter([Cities].Members,
((Ancestor([Cities].CurrentMember, [Cities].[Country]).Name = "URSS") OR (Intersect(Descendants([Cities].CurrentMember,
[Cities].[Country], SELF), [C21MemberSet]).Count = Descendants([Cities].CurrentMember, [Cities].[Country], SELF).Count)))),
[C21MemberSet])'
//Cities members that may lead to invalid cells, members that satisfy the country's condition with their descendants and certain of their
ascendants
  set C23MemberSet as 'Filter([Cities].Members, (Intersect(Descendants([Cities].CurrentMember, [Cities].[Country], SELF),
[C21MemberSet]).Count > 0.0))'
//visual policies
  member [Measures].[DisplayedMeasure] as '([Measures].CurrentMember + 0.0)', FORMAT_STRING = IIf((([Time].CurrentMember IN
[C12MemberSet]) AND ([Cities].CurrentMember IN [C22MemberSet])), "|#|style=red", IIf((([Time].CurrentMember IN [C12MemberSet]) OR
([Time].CurrentMember IN [C13MemberSet])) AND (([Cities].CurrentMember IN [C22MemberSet]) OR ([Cities].CurrentMember IN [C23MemberSet]))),
"|#|style= yellow", "|#|style=green"))

< MDX Query Select Clause >
```

FIG. 9 – *Formule MDX correspondant à la CI de requêtes de l'exemple 5*

modes de visualisation ont été définis, permettant au décideur d'avoir une indication sur la qualité des résultats de requêtes affichés au niveau du client SOLAP : la couleur verte pour les cellules valides ; la couleur jaune pour les cellules agrégées qui comprennent à la fois des cellules valides et non valides ; la couleur rouge pour les cellules non valides.

Dans la figure 10, nous montrons le résultat d'une requête OLAP où ces politiques de visualisation sont appliquées : les cellules valides sont en vert - les cellules concernant un pays autre que l'URSS, ou bien encore les cellules où l'URSS est combiné avec des dates antérieures au 26/12/1991 (e.g., le 01/12/1991) ; les cellules non valides sont en rouge - les cellules combinant l'URSS et des dates postérieures au 26/12/1991 (e.g., le 27/12/1991) ; d'autres cellules sont affichées avec la couleur jaune - par exemple, la cellule impliquant à la fois le mois de décembre 1991 et l'URSS, car celle-ci est le résultat de l'agrégation de cellules valides (par exemple, l'URSS avec le 01/12/1991) et de cellules non valides (e.g., l'URSS avec le 27/12/1991).

6 Conclusions et perspectives

La qualité de l'analyse SOLAP dépend de la qualité des données entreposées et la qualité des agrégations effectuées et la façon dont les données spatio-multidimensionnelles sont explorées. Ainsi, dans cet article, nous étendons et utilisons les contraintes d'intégrité (CI) dans le contexte des systèmes SOLAP pour effectuer ces trois types de contrôles de qualité. Nous proposons une méthode basée sur un profil UML et Spatial OCL pour exprimer les CI SOLAP au niveau conceptuel. Nous montrons aussi les mécanismes d'implémentation automatique de ces contraintes dans une architecture ROLAP classique, en utilisant les standards SQL et MDX. Notre travail actuel porte sur la définition plus précise de l'outil de génération

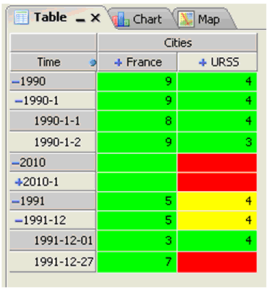

FIG. 10 – *Visualisation des CI sur les requêtes de l'exemple 5*

de code UML2MDX et sur l'intégration de notre approche avec le serveur SOLAP GeoMondrian. Nos travaux futurs concernent l'étude de la complétude de la classification proposée et l'expressivité des formalismes (UML profil et Spatial OCL) proposés pour les CI d'agrégation et de requêtes, ainsi que la satisfiabilité et les dépendances de l'ensemble des CI.

Références

Aguila, P. S. R. D., R. d. N. Fidalgo, et A. Mota (2011). Towards a more straightforward and more expressive metamodel for sdw modeling. In *Proceedings of the ACM 14th international workshop on Data Warehousing and OLAP*, pp. 31–36. ACM.

Arigon, A.-M., M. Miquel, et A. Tchounikine (2007). Multimedia data warehouses : a multiversion model and a medical application. *Multimedia Tools and Applications 35*(1), 91–108.

Bédard, Y., S. Rivest, et M.-J. Proulx (2006). Spatial on-line analytical processing (solap) : Concepts, architectures, and solutions from a geomatics engineering perspective. In R. Wrembel et C. Koncilia (Eds.), *Data Warehouses and OLAP : Concepts, Architecture, and Solutions*, pp. 298–319. Idea Group Publishing.

Boulil, K., S. Bimonte, et F. Pinet (2011). Un modèle uml et des contraintes ocl pour les entrepôts de données spatiales. de la représentation conceptuelle à l'implémentation. *Ingénierie des Systèmes d'Information 16*(6), 11–39.

Carpani, F. et R. Ruggia (2001). An integrity constraints language for a conceptual multidimensional data model. In *SEKE 2001*.

Dyreson, C. E., T. B. Pedersen, et C. S. Jensen (2003). Incomplete information in multidimensional databases. In *Multidimensional databases*, pp. 282–309. IGI Publishing.

Ghozzi, F., F. Ravat, O. Teste, et G. Zurfluh (2003). Modèle multidimensionnel à contraintes. *Revue d'Intelligence Artificielle 17*(1-3), 43–55.

Glorio, O. et J. Trujillo (2008). An mda approach for the development of spatial data warehouses. In I.-Y. Song, J. Eder, et T. Nguyen (Eds.), *Data Warehousing and Knowledge Discovery*, Volume 5182 of *Lecture Notes in Computer Science*, pp. 23–32. Springer.

Lenz, H.-J. et A. Shoshani (1997). Summarizability in olap and statistical data bases. In *Proceedings of the International Conference on Scientific and Statistical Database Management*, pp. 132–143. IEEE.

Levesque, M., Y. Bédard, M. Gervais, et R. Devillers (2007). Towards managing the risks of data misuse for spatial datacubes. In *ISSDQ 2007*.

Malinowski, E. et E. Zimanyi (2008). Advanced data warehouse design. *Advanced Data Warehouse Design : From Conventional to Spatial and Temporal Applications, Data-Centric Systems and Applications, Volume. ISBN 978-3-540-74404-7. Springer, 2008*.

Mazon, J.-N., J. Lechtenborger, et J. Trujillo (2009). A survey on summarizability issues in multidimensional modeling. *Data & Knowledge Engineering 68*(12), 1452–1469.

Perez, D., M. J. Somodevilla, et I. H. Pineda (2007). Fuzzy spatial data warehouse : A multidimensional model. In *Decision Support Systems Advances*, pp. 3–9. IEEE Computer Society.

Pinet, F., M. Duboisset, et V. Soulignac (2007). Using uml and ocl to maintain the consistency of spatial data in environmental information systems. *Environmental modelling & software 22*(8), 1217–1220.

Pinet, F. et M. Schneider (2009). A unified object constraint model for designing and implementing multidimensional systems. In *Journal on Data Semantics XIII*, Volume 5530 of *Lecture Notes in Computer Science*, pp. 37–71. Springer.

Ribeiro, L. d. S., R. R. Goldschmidt, M. Claudia, et M. C. Cavalcanti (2011). Complementing data in the etl process. In *13th International Conference on Data Warehousing and Knowledge Discovery DaWaK 2011*, pp. 112–123. Springer.

Salehi, M. (2009). *Developing a Model and a Language to Identify and Specify the Integrity Constraints in Spatial Datacubes*. Phd.

Siqueira, T. L. L., R. C. Mateus, R. R. Ciferri, V. C. Times, et C. D. A. Ciferri (2011). Querying vague spatial information in geographic data warehouses advancing geoinformation science for a changing world. Volume 1 of *Lecture Notes in Geoinformation and Cartography*, pp. 379–397. Springer.

Stefanovic, N., J. Han, et K. Koperski (2000). Object-based selective materialization for efficient implementation of spatial data cubes. *IEEE Transactions on Knowledge and Data Engineering 12*(6), 938–958.

Torlone, R. (2003). Conceptual multidimensional models. In *Multidimensional databases*, pp. 69–90. IGI Publishing.

Summary

Spatial Data Warehouses and Spatial OLAP systems are decision-support technologies allowing on-line multidimensional analysis of huge volumes of spatial data. In such a kind of systems the goodness of analysis depends on: the warehoused data quality, how aggregations are performed, and how warehoused data are explored. In this paper we propose a framework based on a UML profile and Spatial OCL-defined integrity constraints to grant quality in the whole SOLAP system. We also propose an automatic implementation in a classical ROLAP architecture to validate our proposal.