Vers la définition des contraintes d'intégrité d'entrepôts de données spatiales avec OCL

Kamal Boulil, Sandro Bimonte, Hadj Maboubi, François Pinet

Cemagref, UR TSCF, 24 Avenue des Landais,
63172 Aubière, France
kamal.boulil@cemagref.fr

Résumé. Les Entrepôts de Données Spatiales (EDS) et les systèmes SOLAP représentent une solution efficace pour l'analyse spatiale de phénomènes géographiques. Cependant, la qualité de cette analyse dépend fortement de la qualité des données stockées. Dans ce contexte, quelques travaux se sont intéressés à la définition et la spécification des contraintes d'intégrité spécifiques aux EDS. Dans cet article, motivé par le manque d'implémentation basée sur une approche MDA (Model Driven Approach), nous proposons deux classifications de contraintes d'EDS ainsi que leurs spécifications et implémentations en utilisant le standard OMG OCL (Object Constraint Language).

1 Introduction

Les entrepôts de données associés aux outils d'analyse OLAP représentent une solution efficace pour l'informatique décisionnelle (Inmon, 1996). Par ailleurs, les nouvelles technologies de l'information et de la communication permettent de récolter de très gros volumes de données spatiales. Ces données peuvent par exemple provenir de systèmes de télédétection. Des solutions, connues sous le terme d'OLAP Spatial (Yvan Bédard, 1997) visent à intégrer la donnée spatiale dans les systèmes OLAP. Le SOLAP enrichit les capacités d'analyse des systèmes OLAP classiques, en combinant des analyses multidimensionnelles avec des visualisations cartographiques. Ceci permet par exemple de comprendre la distribution géographique d'un phénomène et de comparer divers phénomènes à diverses échelles géographiques.

L'intégration des données (alphanumériques et notamment géographiques) dans l'analyse OLAP constitue un enjeu majeur car la qualité de cet analyse dépend de la qualité des données stockées dans l'Entrepôt de Données Spatiales (EDS). En effet, contrairement aux Bases de Données (BD), un Entrepôt de Données (ED) permet d'intégrer et d'historiser des données provenant de multiple sources. Cette intégration peut facilement poser des problèmes liés à l'imprécision, l'incertitude ou la sémantique des données. Dans ce contexte, quelques travaux ont été menés concernant la définition des Contraintes d' Intégrité (CI) dans les EDS (Salehi, 2009; Malinowsky et Zimànyi, 2008). Les CI sont des assertions qui identifient les données erronées et incohérentes et/ou empêchent leurs insertions. Dans le contexte spatial, les CI peuvent indiquées les relations spatiales (topologiques, d'ordre, métriques) autorisées entre objets géométriques.

Ces dernières années, la modélisation conceptuelle UML des ED et EDS, a été étudiée dans de nombreux travaux (Abello et al.2006; Glorio et Trujillo, 2008; Bédard et al., 2002). En effet, UML (Unified Modeling Language) permet de modéliser un système indépendamment de toute démarche ou plate-forme (OMG, 2007). Le langage OCL (Object Constraint Language) permet l'expression de contraintes sur des diagrammes UML (OMG, 2006; Pinet et Schneider, 2009). OCL 9IM (Duboisset, 2007) est une extension spatiale d'OCL permettant d'exprimer les relations spatiales topologiques du Modèle 9IM (Egenhofer & Herring, 1992), entre géométries simples et composites.

Ces deux standards, OCL et UML, sont souvent utilisés ensemble dans l'architecture MDA (OMG, 2006) pour développer des systèmes d'information. La MDA permet de séparer les spécifications fonctionnelles d'un système des spécifications liées à son implémentation sur une plate-forme donnée. En effet, MDA encourage l'utilisation de modèles à différents phases du cycle de vie de développement d'applications logicielles. En premier, des modèles indépendants des plateformes technologiques PIM (Platform Independent Model) sont spécifiés au niveau conceptuel. Ensuite, ces PIM sont transformés, en modèles PSM (Platform Specific Model) incluant des informations spécifiques aux plateformes cibles. Enfin, des outils automatisent la transformation de ces PSMs en code exécutable. Les PIMs et PSMs sont typiquement spécifiés en utilisant des standards tels qu'UML (Glorio et Trujillo, 2008).

Dans le contexte des EDS et à notre connaissance, Salehi (2009) est le seul à avoir proposé une classification très détaillée des CI. Cependant, l'auteur ne considère pas les contraintes de méta-données spatiales, de schéma, d'exhaustivité de données, et entre hypercubes spatiaux. Il ne donne pas également l'implémentation des contraintes qu'il spécifie au niveau conceptuel, avec un langage naturel hybride avec pictogrammes[1]. Les autres travaux ne s'intéressent qu'à des classes très spécifiques de CI (Bimonte et al., 2009; Malinowsky et Zimànyi, 2008). De plus, aucun travail ne se base sur des langages standards de spécification de contraintes, tel qu'OCL. L'utilisation de standards tels qu'OCL et UML faciliterait l'emploi d'une approche MDA pour un développement rapide et facile des EDS.

Dans cet article, nous proposons deux nouvelles classifications des contraintes d'intégrité pour les entrepôts de données spatiales. La première regroupe les contraintes selon les niveaux d'implémentation possibles. La seconde se base sur la nature des éléments sur lesquels portent les CI. Nous montrons comment spécifier avec les langages OCL et OCL Spatial/9IM (Pinet et al., 2009) les contraintes qui peuvent être implémentées au niveau de l'EDS. Ces propositions ont pour objectif de faciliter la mise en place d'une architecture MDA pour la spécification et l'implémentation rapides et faciles des CI d'EDS.

Le reste de cet article est organisé comme suit. La section 2 présente et discute les limites des travaux existants sur les contraintes d'intégrité dans les ED et les EDS. Un exemple illustratif de notre travail est introduit en section 3. La section 4 détaille nos deux nouvelles classifications, et la section 5 montre la spécification des CI avec OCL et OCL Spatial. En section 6, nous montrons comment ces CI sont automatiquement implémentées sous forme

[1] Langage naturel contrôlé enrichi par des symboles visuels intuitifs pour représenter la spatialité et temporalité des objets.

de triggers et vues SQL dans le SGBD Spatial Oracle. Finalement, les conclusions et perspectives de recherche sont présentées en section 7.

2 Etat de l'art

Stefanovic et al. (2000) définissent un entrepôt de données spatiales comme *une collection de données spatiales et thématiques, intégrées, non volatiles et historiées pour la prise de décisions spatiales*.

Les entrepôts de données spatiales sont modélisés selon le modèle spatio-multidimensionnel qui définit les concepts de mesure spatiale et de dimension spatiale pour prendre en compte la composante spatiale de l'information géographique. La dimension spatiale désigne l'introduction de l'information spatiale dans une application décisionnelle en tant qu'axe d'analyse (Rivest et al., 2001; Malinowski et Zimányi, 2008). La mesure spatiale y est parfois vue comme une collection de pointeurs vers des objets spatiaux et/ou comme le résultat d'opérateurs métriques ou topologiques spatiaux, par exemple la distance entre deux régions (Stefanovic et al., 2000; Rivest et al., 2001; Malinowski et Zimányi, 2008).

L'architecture typique d'un système SOLAP (figure 1) est constituée de trois tiers : Entrepôt de données spatiales, Serveur SOLAP et Client SOLAP (Rivest et al., 2001). L'EDS est souvent implémenté en utilisant un SGBD Spatial. Ce dernier permet de gérer et d'interroger les données spatiales tout en garantissant le passage à l'échelle et de bonnes performances. Le serveur SOLAP définit les hypercubes spatiaux en définissant les mesures, les dimensions spatiales et les operateurs d'agrégation SOLAP. Enfin, le client SOLAP permet des analyses et des explorations interactives pertinentes du contenu de l'EDS, en exploitant divers types d'affichage: histogrammes, tableaux croisés dynamiques et des cartes interactives.

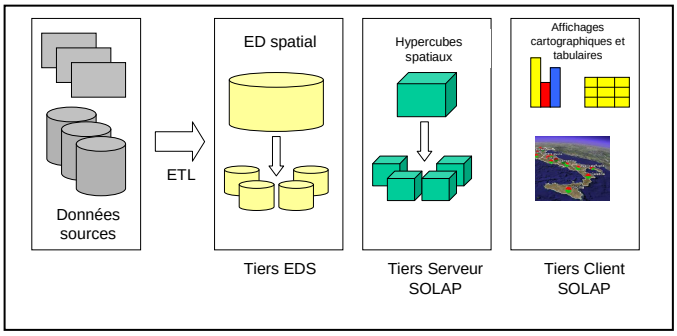

FIG. 1 - *Architecture SOLAP*.

Dans le reste de cette section, nous présentons et discutons les travaux existants sur les contraintes d'intégrité dans les ED et les EDS.

Dans le contexte des entrepôts de données classiques, Carpani et Ruggia (2001) sont les premiers à avoir proposé un modèle conceptuel multidimensionnel formel supportant des contraintes d'intégrité. Les auteurs utilisent des macros et des prédicats logiques pour spécifier des contraintes intra et inter membres, inter niveaux et inter dimensions, au niveau

conceptuel sans proposer d'implémentation. Hurtado et Mendelzon (2002) s'intéressent à l'agrégeabilité dans les dimensions hétérogènes[2]. Ils introduisent de nouvelles contraintes, nommées contraintes de dimension, qui spécifient les chemins d'agrégation possibles des membres, le long des hiérarchies de dimension. Ces contraintes sont spécifiées sous formes d'expressions booléennes combinant des atomes d'égalité et de chemin. Ghozzi et al., (2003) proposent un modèle multidimensionnel en constellation pour les ED. Ils introduisent les CI intra et inter dimensions, qui définissent des inclusions et exclusions entre hiérarchies d'une ou plusieurs dimensions. Les auteurs utilisent un langage naturel hybride avec symboles pour spécifier ces contraintes au niveau conceptuel.

Dans les entrepôts de données spatiales, Malinowski et Zimányi (2008) présentent un modèle conceptuel spatio-multidimensionnel, nommé *Spatial MultiDim*. En utilisant des pictogrammes, les auteurs spécifient des CI spatiales (relations spatiales topologiques du modèle 9IM, types géométriques, etc.). Ils proposent également un mapping de ce modèle vers le modèle logique Objet-Relationnel et une implémentation de ces contraintes sous forme de triggers, vues et fonctions dans le SGBD Oracle 10g. Cependant la représentation graphique proposée ne permet pas de spécifier toutes les CI possibles. Glorio et Trujillo (2008) proposent un profil UML pour modéliser les EDS selon une approche MDA. Ils spécifient aussi avec des pictogrammes des CI spatiales de domaine (attributs et mesures spatiaux) mais ne considèrent pas d'autres contraintes. Bimonte et al., (2009) défissent un modèle logique pour les EDS qui prend seulement en compte les dépendances sémantiques entre fonctions d'agrégation spatiales et alphanumériques mais ne présentent pas d'implémentation. Enfin, Salehi (2009) propose un modèle logique pour les hypercubes spatiaux ainsi qu'une classification très détaillée mais non exhaustive des CI. Il propose également les spécifications d'un langage de modélisation de CI spatiales et temporelles (naturel hybride avec pictogrammes et naturel contrôlé). L'auteur ne présente pas d'implémentation.

Le tableau 1 synthétise ces travaux en prenant en compte différents aspects fondamentaux pour la spécification et l'implémentation des CI selon une approche MDA :

(a) Utilisation de standards pour la définition des CI et du modèle conceptuel de l'EDS ; ceci facilitera l'implémentation dans des outils existants de type Atelier de Génie Logiciel supportant UML et la MDA.

(b) Les règles de transformation PIM vers PSM facilitent l'implémentation des CI et du modèle de données en automatisant le passage entre les niveaux conceptuel et physique.

(c) Utilisation de notations visuelles pour exprimer certaines CI spatiales dans les modèles de données (par exemple des pictogrammes – voir par exemple (Malinowski et Zimányi 2008)) : les pictogrammes sont des symboles visuels intuitifs qui améliorent la lisibilité des modèles conceptuels en facilitant l'interaction avec les décideurs (Papajorgj et al., 2010).

(d) Nombre de Classes de CI considérées. La qualité de l'analyse SOLAP dépend forcément de toutes les CI pouvant être définies à différents niveaux de l'architecture d'EDS (figure 1).

[2] Une dimension hétérogène est dimension où deux membres d'un niveau donné ont des ancêtres dans deux niveaux différents.

	Utilisation de standards		Nombre de Classes de CI considérées	Transformations PIM vers PSM	Notations visuelles pour les CI
	CI	Modèle de données			
(Carpani et Ruggia 2001)	Non	Non	Moyen	Non	Non
(Hurtado et Mendelzon 2002)	Non	Non	Réduit	Non	Non
(Ghozzi et al 2003)	Non	Non	Réduit	Non	Non
(Glorio et Trujillo 2008)	Non	Oui	Réduit	Oui	Oui
(Malinowski et Zimányi 2008)	Non	Non	Réduit	Non	Oui
(Mehrdad 2009)	Non	Non	Elevé	Non	Non
(Bimonte et al., 2009)	Non	Non	Réduit	Non	Non

TAB. 1 - *Comparaison des travaux sur les CI dans les ED et les EDS*

Finalement, aucun travail ne présente encore toutes les caractéristiques nécessaires à la définition des CI d'EDS, suivant une approche MDA. De plus, nous pensons que ces travaux peuvent être étendus pour prendre en compte d'autres classes de contraintes.

3 Exemple illustratif d'EDS

Dans cet article, pour illustrer les classes de CI spatiales que nous proposons ainsi que leur spécification en OCL, nous présenterons plusieurs exemples tous basés sur la même étude de cas. Nous considérons le modèle multidimensionnel représenté par le diagramme de classes UML de la figure 2. Cet exemple est issu de (Salehi, 2009). Il permet l'analyse multi-dimensionnelle des incendies selon les dimensions temps, localisations, et types de feux.

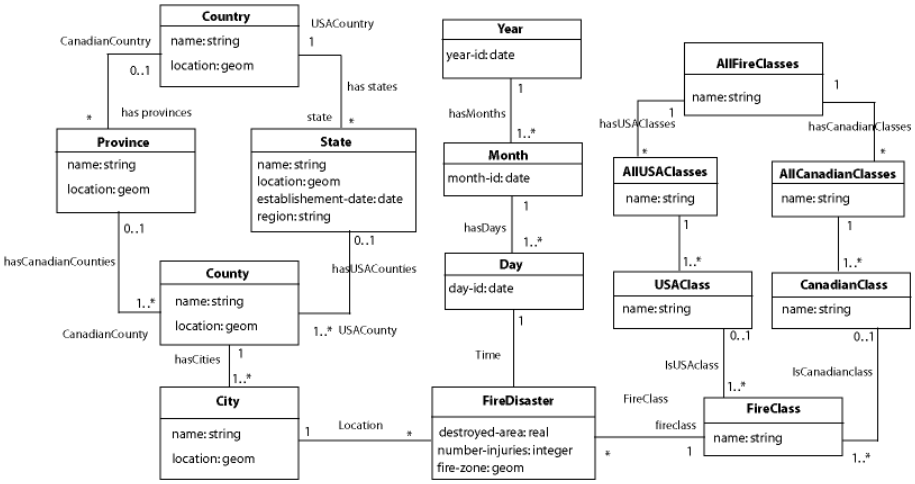

FIG. 2 - *Modèle multidimensionnel dl'analyse des feux (issu de Salehi (2009)).*

La dimension temporelle "temps" contient les niveaux de dimension (classes) : "Day", "Month" et "Year". La dimension spatiale "localisations" comprend deux hiérarchies exclu-

sives : "localisations USA" et "localisations Canada". Les niveaux spatiaux "City", "County", "State" et "Country" forment la hiérarchie "localisations USA". Les niveaux spatiaux "City", "County", "Province" et "Country" forment la hiérarchie "localisations Canada". Pareillement, la dimension "classes de feux" comprend les hiérarchies exclusives: "Classes de feux américaines" et "Classes de feux canadiennes". Les attributs *name* dans la dimension "classes de feux", indiquent les noms des classes de feux.

Les feux sont analysés suivant les mesures (attributs de la classe *FireDisaster*) : (a) *destroyed–area*, représentant la surface des résidences ravagées par les feux; (b) *number-injuries*, représentant le nombre de blessés ; (c) *fire-zone*, représentant les géométries des zones de feux (mesure spatiale).

Enfin, ce modèle multidimensionnel permet de répondre à des requêtes telles que : "*Quelle est la surface (destroyed –area) des résidences ravagées par les feux de classe A dans tous les états américains, entre les années 2000 et 2003?*" ou encore "*Quelle est la distribution géographique des feux de classe A entre les années 1998 et 2008?* ".

4 Nouvelles classifications de contraintes d'intégrité d'EDS

Comme montré à la section 2, la classification la plus complète a été proposée par Salehi (2009). Cependant, cette classification ne considère pas les contraintes sur les métadonnées spatiales, inter-hypercubes, de schéma, d'exhaustivité de données et de dépendances entres fonctions d'agrégation. Nous proposons donc dans cet article deux nouvelles classifications :

1. une classification orientée implémentation,
2. et une autre basée sur les concepts principaux d'ED (membres, faits, agrégation, etc.) en considérant les points de vue structurel et d'instance.

Ces deux classifications peuvent constituer un cadre formelle pour spécifier et implémenter des CI d'EDS, selon une approche MDA. En effet, suivant le type de CI et son niveau d'implémentation (le niveau dans l'architecture SOLAP), des spécifications et des implémentations différentes peuvent être envisagées. Par exemples les CI du tiers EDS peuvent être implémentées en utilisant des triggers et des vues. Au niveau du tiers SOLAP, d'autres techniques peuvent être employées pour interdire par exemple, l'exécution de certaines requêtes « incohérentes ». De plus, des extensions des modèles conceptuels d'EDS, des langages de spécification de contraintes tels qu'OCL, seront aussi nécessaires pour inclure les concepts d'agrégation, de requête multidimensionnelle, etc.

4.1 Classification orientée implémentation

Nous proposons ici une classification inédite basée sur les niveaux d'implémentation possibles des CI. Nous distinguons donc les contraintes d'EDS, selon ce critère en (Figure 3) :

(a) Contraintes d'ETL : ce sont des contraintes implémentées au niveau de l'outil ETL (Liu et al., 2009; Muñoz et al., 2009). Elles permettent de filtrer les données erronées avant leur chargement dans l'entrepôt.

(b) Contraintes d'EDS: ces contraintes identifient les données (faits et membres) erronées et empêchent leur stockage dans l'EDS. Ces contraintes peuvent être vérifiées soit au cours du chargement des données dans l'entrepôt, i.e. à chaque nouvelle insertion, en utilisant des triggers sur les opérations SQL LMD[3] INSERT, UPDATE et DELETE; ou bien une fois que toutes les données sont chargées dans l'entrepôt, par exemple pour vérifier l'exhaustivité des données i.e. vérifier que toutes les données nécessaires à l'analyse ont bien été insérées.

(c) Contraintes SOLAP : ces contraintes sont utilisées pour garantir une agrégation correcte des mesures, et une exploration sémantiquement correcte et cohérente des hypercubes spatiaux. Ces contraintes peuvent être implémentées soit dans le schéma de l'hypercube du Serveur SOLAP, soit dans le parseur des requêtes du serveur SOLAP, ou dans le client SOLAP.

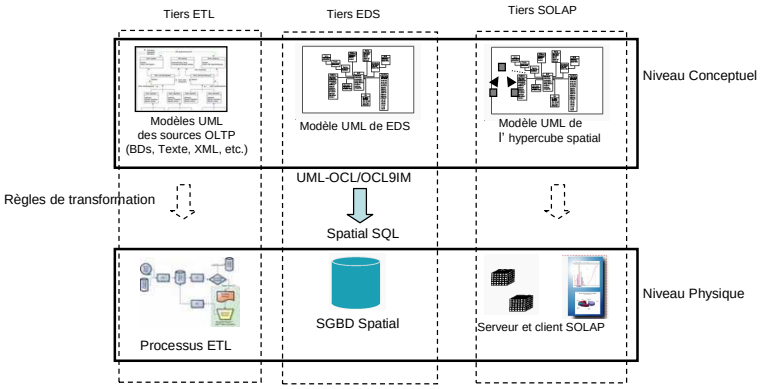

FIG. 3 - *Types d'implémentation des contraintes d'EDS.*

Il est important de noter qu'il est possible qu'une même contrainte puisse être implémentée et vérifiée dans les trois niveaux.

4.2 Classification orientée concepts d'EDS

A présent, nous allons décrire la deuxième classification, en donnant des exemples de contraintes spatiales à implémenter au niveau du tiers EDS (Figure 3). Cette classification distingue les contraintes suivant leurs portées sur les concepts d'EDS (méta-données, membres, niveaux, attributs, faits, mesures, et agrégation). Le positionnement de notre classification par rapport à d'autres travaux ainsi que les niveaux d'implémentation possibles pour chaque classe de contraintes (indiqués par un jeu de couleurs) sont présentés en figure 4. Notons que notre classification est expressive, car elle couvre et étend toutes les classifications existantes.

[3] LMD: Langage de Manipulation de Données

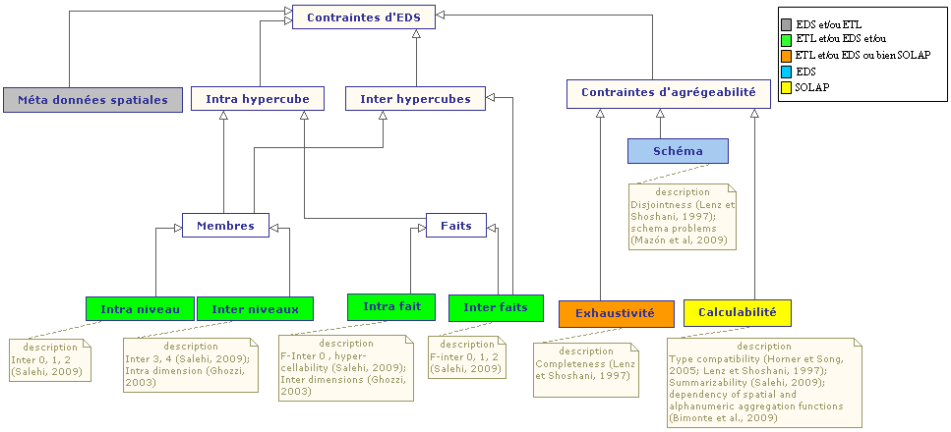

FIG. 4 - *Classification des Contraintes d'EDS selon les concepts fondamentaux d'ED.*

4.2.1 Les contraintes sur les méta-données spatiales

Elles définissent les méta-données des types géométriques (le système de référence spatiale, l'échelle ou plage de valeurs, le degré de précision, etc.). Ces contraintes peuvent être implémentées au niveau du SGBD lors de la création du schéma de l'EDS et/ou au niveau de l'ETL en exploitant les sources de données transactionnelles qui peuvent se présenter sous différentes formes (BDs, fichiers texte, fichiers XML, etc.).

4.2.2 Les contraintes intra hypercube spatial

Elles portent sur des membres et des faits d'un seul hypercube spatial. Nous distinguons les contraintes sur les membres et les contraintes sur les faits.

(a) Les contraintes sur les membres: ces contraintes peuvent être utilisées pour identifier les membres erronés, et relations erronées (spatiales, temporelles, etc.) entre membres. Selon qu'elles portent sur les attributs d'un ou plusieurs niveaux de dimension, nous distinguons les contraintes *intra niveau* ou les contraintes *inter niveaux*.

Contraintes intra niveau : elles portent sur les membres et relations entre membres d'un seul niveau de dimension.

Contrainte 1: les états américains situés dans la région "West" doivent être disjoints des états américains situés dans la région "Northeast". Cette contrainte définit une relation spatiale topologique entre les membres d'un seul niveau.

Contraintes inter niveaux : elles portent sur les membres et relations entre membres de plusieurs niveaux de dimensions.

Contrainte 2 : seuls les états situés dans les régions "West", "Midwest" ou "Northeast" peuvent avoir des frontières avec des provinces. Cette contrainte porte sur les attributs *region* et *location* du niveau *State* et l'attribut *location* du niveau *Province*.

(b) Les contraintes sur les faits : ces contraintes impliquent un ou plusieurs faits d'un seul hypercube spatial et peuvent être donc *intra fait* ou *inter faits*. Elles peuvent être utilisées soit pour identifier les faits incorrects et relations entre faits incorrectes, ou bien pour interdire l'exécution de requêtes spatio-multidimensionnelles sémantiquement incorrectes, portant sur un ou plusieurs faits.

Contraintes intra fait : elles portent sur une ou plusieurs mesures d'un seul fait.

Contrainte 3 : les feux qui se produisent aux USA ne peuvent pas être liés à des classes de feux canadiennes. Inversement, les feux se produisant au Canada, ne peuvent pas être liés à des classes de feux américaines.

Contraintes inter faits: elles portent sur plusieurs faits d'un seul hypercube spatial.

Contrainte 4 : aux USA, la somme des surfaces détruites par les feux de classe "A" est supérieure à la somme des surfaces détruites par les feux de classe E" Cette contrainte porte sur plusieurs faits.

4.2.3 Les contraintes inter hypercubes spatiaux

Elles portent sur des membres et des faits appartenant à plusieurs hypercubes spatiaux. Elles peuvent être: (a) *intra niveau*, i-e portent sur les membres d'un niveau partagé entre plusieurs hyperubes spatiaux (schéma en constellation); (b) *inter niveaux*, portent sur des membres de plusieurs niveaux appartenant à des hypercubes spatiaux différents, par exemple pour définir des hiérarchies compatibles (Cabibbo et Torlone, 2004) lors d'une opération de drill-across; et (c) *inter faits*, impliquent plusieurs faits de plusieurs hypercubes spatiaux.

Contrainte 5 : considérons un deuxième hypercube pour l'analyse des inondations selon les dimensions temps, localisations et types d'inondation. Cet hypercube partage les dimensions temps et localisations avec l'hypercube "incendies". Il a aussi deux mesures zone-inondée (spatiale) et niveau de précipitations (numérique). Un exemple simple de contrainte inter hypercubes et inter faits est: les zones de feux (fire-zone) et les zones inondées (zone-inondée) des incendies et inondations, se produisant à la même date, doivent être disjointes.

4.2.4 Les contraintes d'agrégeabilité

Elles garantissent des agrégations correctes des mesures le long des différentes dimensions. Pour cette classe de contraintes, nous distinguons trois sous classes.

(a) Les contraintes d'exhaustivité de données : elles vérifient la présence de toutes les données (membres et faits) dont l'analyse décisionnelle a besoin. Elles correspondent en partie à la condition de complétude de (Lenz et Shoshani, 1997).

(b) Les contraintes de schéma : elles sont des conditions qui doivent être respectées lors de la définition des hiérarchies des dimensions et du choix de la granularité des faits. Elles portent sur les multiplicités des relations entre niveaux de dimension et celles des relations entre faits et dimensions.

(c) Les contraintes de calculabilité: elles garantissent une utilisation correcte des fonctions d'agrégation pour agréger les mesures le long des dimensions. Plus précisément, elles définissent les combinaisons (dimension, niveau à partir duquel on agrège, niveau vers lequel on agrège, mesure, fonction d'agrégation) correctes et incorrectes. Elles peuvent aussi définir les relations de dépendance entre plusieurs de ces combinaisons, en considérant par exemple, les dépendances entre les fonctions d'agrégation spatiales et les fonctions d'agrégation numériques (Bimonte et al., 2009).

Dans cet article, nous nous focalisons sur les contraintes de données intra hypercube spatial, qui peuvent être implémentées au niveau du tiers EDS (Figure 3). Dans les sections suivantes nous montrons comment ces contraintes peuvent être exprimées en OCL (Section 5) et comment elles peuvent être implémentées sous le SGBD Oracle 11g en utilisant l'extension spatial de l'outil OCL2SQL (Duboisset, 2007) (section 6).

5 Spécification des contraintes d'intégrité intra hypercube

Dans cette section, nous présentons la spécification en OCL et OCL 9IM des exemples de contraintes intra hypercube spatial, décrits dans la section 4.2.2. Notre contribution est de montrer que ces contraintes peuvent facilement s'exprimer en OCL et OCL 9IM.

5.1 Les contraintes sur les membres

Contrainte 1: c'est une contrainte OCL de type invariant (mot clef *inv*) qui s'applique à une instance "State" (*Context State*). Un invariant OCL exprime une contrainte sur un objet ou un groupe d'objets qui doit être respectée en permanence. En utilisant la syntaxe d'OCL 9IM, (geoA).opération_topo (geoB)[4], elle définit une relation spatiale topologique (areDisjoint) entre l'instance courante de "State" (self) et chaque instance "State" (s) de la collection des instances "State" retournée par l'expression "State.allInstances". Le mot clef self est utilisé dans OCL pour référencer l'objet courant de la classe spécifiée dans le contexte.

```
Context State inv:
State.allInstances -> forAll(s |
(self.region = 'West' and  s.region = 'Northeast') implies
self.location.areDisjoint (s.location))
```

Contrainte 2: cet invariant vérifie pour chaque objet courant "State" (self), s'il existe un objet "Province" p, dans la collection des objets "Province", tel que les géométries des deux

[4] L'expression (geoA).opération_topo (geoB) retourne vrai si la relation spatiale topologique "opération-topo" est respectée entre les deux géométries geoA et geoB

objets (self.location et p.location) soient adjacentes. Dans le cas où cette condition est vérifiée, la valeur de l'attribut "region" de l'objet courant "self "doit être égale à l'une des valeurs indiquées par la contrainte.

```
context State inv:
Province.allInstances -> exists(p|
self.location.areAdjacent(p.location))) implies
((self.region = 'West') or (self.region = 'Midwest') or
(self.region = 'Northeast'))
```

5.2 Les contraintes sur les faits

Contrainte 3: cette contrainte porte sur un fait (instance de la classe FireDisatster). Elle vérifie que le fait courant (self) n'est pas, au même temps, lié à une ville américaine et une classe de feux canadienne et inversement. "State.allInstances.county.city.firedisaster" retourne la collection des faits (instances de la classe FireDisatster) liés à la hiérarchie "localisations USA". De même, "CanadianClass.allInstances.fireclass.firedisaster" retourne la collection des faits liés à la hiérarchie des classes de feux canadiennes. L'expression "collection - >excludes(self)" retourne vrai si l'objet "self" n'appartient pas à collection.

```
context FireDisaster inv :
(State.allInstances.county.city.firedisaster ->
excludes(self) xor
CanadianClass.allInstances.fireclass.firedisaster ->
excludes(self) ) and
(Province.allInstances.county.city.firedisaster ->
excludes(self) xor
USAClass.allInstances.fireclass.firedisaster ->
excludes(self) )
```

Contrainte 4: cette contrainte est spécifiée par deux invariants définis dans le même contexte (classe Year). Le premier invariant définit une variable "surfaceA" et lui affecte la somme des surfaces détruites par les feux de classe "A" (fireclass.name ='A') dans des états américains (counrty.name ='USA'). Le deuxième calcule la somme des surfaces détruites par les feux de classe E dans des états américains et vérifie que la valeur de la variable "surfaceA" est bien supérieure à celle de la variable "surfaceE". En général, l'opération sur collections *select* retourne la sous-collection d'éléments qui vérifient une certaine condition. Dans l'exemple, l'opération *select (fd|cond)* retourne la sous collection de faits (instances de FireDisaster) qui respectent la condition *cond*. L'opération *collect* construit une nouvelle collection : les éléments de cette collection sont les valeurs des attributs "destroyed-air " des faits sélectionnés par l'opération *select*.

```
context Year def:
surfaceA: Real = self.month.day.firedisaster -> select(fd|
(fd.city.county.state.country.name = 'USA') and
(fd.fireclass.name= 'A'))-> collect(destroyed-air)-> sum()

context Year inv :
let surfaceE: Real = self.month.day.firedisaster ->
collect(fd| fd.destroyed-air) -> sum() in
((fd.city.county.state.country.name = 'USA') and
(fd.fireclass.name= 'E')) implies surfaceA > surfaceE
```

6 Implémentation des contraintes au niveau du tiers EDS

Pour implémenter le modèle conceptuel UML d'EDS (figure 2) et les contraintes de données intra hypercube spatial spécifiées en OCL et OCL 9IM (sections 5.1 et 5.2) sous Oracle spatial, nous avons utilisé l'extension spatiale d'OCL2SQL. OCL2SQL (Duboisset, 2007) est un outil open source, entièrement écrit en java. Il permet de générer à partir d'une contrainte OCL *c*, une requête SQL qui sélectionne toutes les données qui ne satisfont pas *c*. Le principe de génération de code est basé sur des patrons de conversion d'OCL vers SQL pour Oracle.

L'outil OCL2SQL étendu au spatial intègre les extensions spatiales d'OCL appelées OCL 9IM et OCL ADV[5]. Il implémente les règles de mappings automatiques de ces langages vers le SQL d'Oracle spatial. Les inputs principales de l'outil sont le modèle conceptuel sous forme de schéma XMI 1.0, le fichier de contraintes OCL, le fichier de méta-données géométriques. En sortie, l'outil produit les scripts SQL pour la création de la base, les requêtes et les triggers pour la vérification automatique de contraintes. Pour chaque contrainte OCL, il crée une vue et un ou plusieurs triggers oracle. La vue sélectionne les tuples de table (la table qui correspond au contexte de la contrainte OCL) qui ne satisfont pas la contrainte. Le trigger renvois un message d'erreur si le nombre de tuples sélectionnés par la vue est supérieur à zéro. Plus de détails sur ce mode d'implémentation peuvent être trouvés dans (Pinet et al., 2009).

Pour implémenter les contraintes et l'entrepôt de données spatiales, nous avons procédé en 5 étapes :

(a) Définition du schéma XMI 1.0 correspondant au modèle conceptuel UML d'EDS de la Figure 2.

(b) Spécification et validation des contraintes OCL, sachant la syntaxe supportée par l'outil OCL2SQL et les éléments définis dans le schéma XMI. La Figure 5 montre la spécification en OCL 9IM de la contrainte 2 (Section 4.2.2) avec l'éditeur intégré à l'outil OCL2SQL. Notons l'utilisation du prédicat spatial "areAdjacent" entre les attributs spatiaux "location" de deux membres : un état (self) et une province (p).

---

[5] OCL ADV: Extension d'OCL pour définir des relations spatiales topologiques qualifiées par des ADVerbes entre régions spatiales composites (Pinet et al , 2009).

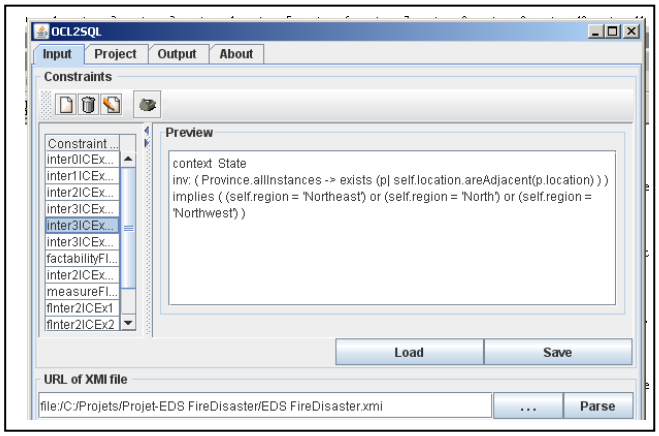

FIG. 5 - *Spécification de la contrainte 2 avec l'éditeur intégré d'OCL2SQL.*

(c) *Définition du fichier des méta-données spatiales Oracle.* Ce fichier indique pour chaque attribut géométrique, le système de référence spatiale et les plages de valeurs possibles en cordonnées X et Y. Il est utilisé pour mettre à jour la table des méta-données spatiales d'oracle, USER_SDO_GEOM_METADATA.

(d) *Génération du code SQL pour Oracle spatial.* Le SQL généré pour la contrainte 2 est présenté à la figure 6. Il est important noter que le prédicat spatial OCL 9IM "areAdjacent" est traduit en une relation spatiale définie par l'opérateur spatial d'Oracle "MDSYS.SDO_RELATE" et le maque "TOUCH".

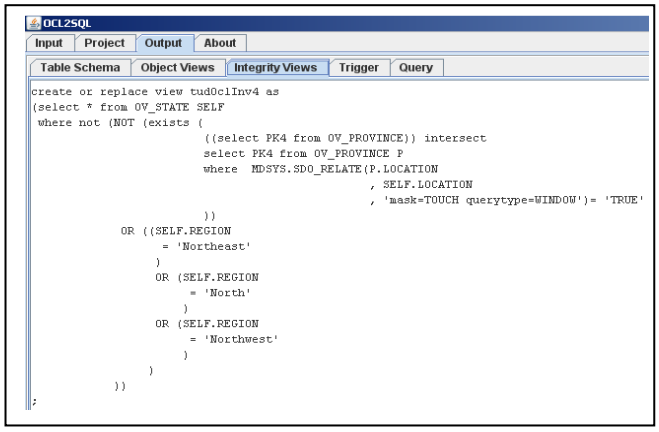

FIG. 6 - *Vue SQL qui implémente la contrainte 2 sous Oracle spatial.*

(e) Implémentation des scripts générés (contraintes et modèle de données) sous Oracle.

Enfin, nous avons vérifié le bon fonctionnement des mécanismes de contrôle d'intégrité (triggers et vues) générés en insérant des données de test dans l'entrepôt. Nous avons testé les

contraintes une par une. Pour une contrainte donnée, à chaque fois que nous tentons d'insérer des données erronées (qui violent la contrainte), le système (trigger) empêche l'insertion et revoit un message d'erreur.

7 Conclusions et Perspectives.

L'introduction de l'information spatiale dans les ED et les systèmes OLAP a mené à la définition des EDS et des systèmes SOLAP. La qualité des analyses OLAP et SOLAP dépend fortement de la qualité des données. Dans ce contexte, différents travaux se sont intéressés à la définition de contraintes d'intégrité dans les ED et EDS. Les travaux les plus complets en termes de classes de contraintes considérées sont actuellement ceux de Salehi et al. (2007). Cependant et comme montré dans le présent papier, de nouvelles classes de contraintes pourraient encore être considérées. Aussi, aucune étude ne proposait de moyen pour générer du code à partir des CI modélisées. Comme les méthodes existantes n'exploitent pas des standards existants tels qu'OCL pour spécifier les contraintes au niveau conceptuel, la mise en place d'une architecture MDA est rendue difficile.

Dans cet article, nous avons tout d'abord proposé deux nouvelles classifications de contraintes d'EDS : une classification orientée implémentation et une classification basée sur les concepts fondamentaux d'ED. Ensuite, nous avons donné la spécification OCL et OCL 9IM des contraintes de données intra hybercube spatial. Enfin, nous avons montré qu'il était possible d'implémenter automatiquement ces contraintes écrites en OCL et OCL 9IM sous Oracle 11g avec des vues et des triggers SQL. Pour cela, l'extension spatiale de l'outil OCL2SQL a été exploitée.

Actuellement nous travaillons sur la définition d'un profil UML pour la modélisation des entrepôts de données spatiales (Glorio et Trujillo, 2008; Bédard et al., 2002) supportant des patrons OCL pour la définition de contraintes de données intra hypecube. Par la suite, nous envisageons d'introduire des extensions des modèles conceptuels d'EDS et d'OCL pour prendre en compte les concepts d'agrégation et de requête multidimensionnelle SOLAP (Mazon et al., 2009), etc. Ceci sera dans le but de spécifier toutes les contraintes d'agrégeabilité et de requêtes au niveau conceptuel. Nous pourrons envisager des techniques adaptées pour leurs implémentations. Une possibilité serait de passer des technologies de type « bases de données actives » utilisant des triggers sur des BD, à une technologie de type « SOLAP actif » contrôlant les requêtes possibles au niveau tiers SOLAP. Finalement, nous envisageons d'investiguer les contraintes inter hypercubes dans le but d'introduire l'opérateur de "drill-across" (Cabibbo et Torlone, 2004) dans l'analyse SOLAP.

Références

Abello A., J. Samos et F. Saltor (2006). *YAM2: a multidimensional conceptual model extending UML*. Information Systems, vol. 31(6), p. 541-567.

Bédard, Y. (1997). *Spatial OLAP*. 2ème Forum annuel sur la R-D, Géomatique VI: Un monde accessible, Montréal, Canada, p. 13-14.

Bédard, Y., M. Proulx, S. Larrivée et E. Bernier (2002). *Modeling Multiple Representations into Spatial Datawarehouses: A UML-based Approach*. Joint International Symposium ISPRS Commission IV (SDH), 95th Annual CIG Conference: Geospatial Theory, Processing and Applications.

Bimonte, S., M. Villanova-Oliver et J. Gensel (2009). *A Multidimensional Model for Correct Aggregation of Geographic Measures*. Evolving Application Domains of Data Warehousing and Mining: Trends and Solutions, IGI Group.

Cabibbo, L. et R. Torlone (2004). *On the integration of autonomous data marts*. 16th International Conference on Scientific and Statistical Database Management, Santorini Island, Greece, p. 223-234.

Carpani, F. and R. Ruggia (2001). *An Integrity Constraints Language for a Conceptual Multidimensional Data Model*. 13th International Conference on Software Engineering and Knowledge Engineering, Bones Aires, Argentina, p. 220-227.

Duboisset, M. (2007). *Un Système de Contraintes d'Intégrité OCL pour les Bases de Données Spatiales - Application à un Système d'Information pour l'Epandage Agricole*. Thèse de doctorat, Université Blaise Pascal, Clermont-Ferrand, France.

Egenhofer, M. et J. Herring(1992). *Categorizing Binary Topological Relations between Regions, Lines, and Points in Geographic Databases*. Technical report, University of Maine, Orono, USA.

Ghozzi, F., F. Ravat, O. Teste, et G. Zurfluh (2003). *Constraints and Multidimensional Databases*. 5th International Conference on Enterprise Information Systems, Angers, France, p. 104-111.

Glorio, O. et J. Trujillo (2008). *An MDA Approach for the Development of Spatial Data Warehouses*. 10th International Conference on Data Warehousing and Knowledge Discovery, Turin, Italy, p. 23-32.

Horner, J. et I.-Y. Song (2005). *A taxonomy of inaccurate summaries and their management in OLAP systems.* 24th International Conference on Conceptual Modeling, Klagenfurt, Austria. p. 433-448.

Hurtado, C.A. et A.O. Mendelzon (2002). *OLAP Dimension Constraints.* 21st ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, Wisconsin, USA, p. 169-179.

Inmon, W.H. (1996). *Building the Data Warehouse.* 2nd Ed. New York: Wiley.

Lenz, H.-J. et A. Shoshani (1997). *Summarizability in OLAP and statistical data bases.* 9th International Conference on Scientific and Statistical Database Management, Washington, USA, p. 132-143.

Liu, J., S. Liang, D. Ye, J. Wei et T. Huang (2009). *ETL Workflow Analysis and Verification Using Backwards Constraint Propagation*. 21st International Conference in Advanced Information Systems Engineering, Amsterdam, The Netherlands, p. 455-469.

Malinowski, E. et E. Zimányi (2008). *Advanced Data Warehouse Design: From Conventional to Spatial and Temporal Applications*. Berlin: Springer-Verlag.

Mazón , J.-N., J. Lechtenbörger et J. Trujillo (2009). *A survey on summarizability issues in multidimensional modelling*. Data & Knowledge Engineering, vol. 68(12), p. 1452-1469.

Muñoz, L., J.-N. Mazón et J. Trujillo (2009). *Automatic generation of ETL processes from conceptual models*. 12th International Workshop on Data Warehousing and OLAP, Hong Kong, China, p. 33-40.

OMG (2006). *Object Constraint Language, Version 2.0*. http://www.omg.org/spec/OCL/2.0/PDF.

OMG (2007). *Unified Modeling Language (UML), Version 2.1.2*. http://www.omg.org/spec/UML/2.1.2/Infrastructure/PDF.

Papajorgji, P., F. Pinet, A. Miralles, E. Jallas et P. Pardalos (2010). *Modeling: a central activity for flexible information systems development in agriculture and environment*. International Journal of Agricultural and Environmental Information Systems, vol. 1(1), p. 1-25.

Pinet, F., M. Duboisset et M. Schneider (2009). *Modélisation de contraintes d'intégrité spatiales avec OCL*. Revue Internationale de Géomatique, vol.19(1), p. 93-122.

Pinet, F. et M. Schneider (2009). *A unified object constraint model for designing and implementing multidimensional systems.* Journal on Data Semantics, vol. 13, p. 37-71.

Salehi, M., Y. Bédard, M. A. Mostafavi et J. Brodeur (2007). *On Languages for the Specification of Integrity Constraints in Spatial Conceptual Models.* Advances in Conceptual Modeling – Foundations and Applications, Auckland, New Zealand, p. 388-397.

Salehi, M. (2009). *Developing a Model and a Language to Identify and Specify the Integrity Constraints in Spatial Datacubes*. Thèse de doctorat, Faculté des études supérieures de l'Université Laval, Canada.

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, vol. 12(6), p. 938-958.

Summary

Spatial Data Warehouses (SDW) and OLAP systems represent an effective solution to perform spatial analysis on geographical phenomena. However, the quality of such analysis heavily depends on the quality of the stored data. Due to this, a number of researches have been attempted to address the issues of data quality by introducing SDW-specific Integrity Constraints (SDW IC). In this paper, motivated by the lack of a Model Driven Approach-Based implementation, we present two classifications for SDW IC and a MDA-Based specification and implementation of these constraints, using the Object Constraint Language (OCL).