AVERTISSEMENT

Ce document est le fruit d'un long travail approuvé par le jury de soutenance et mis à disposition de l'ensemble de la communauté universitaire élargie.

Il est soumis à la propriété intellectuelle de l'auteur : ceci implique une obligation de citation et de référencement lors de l'utilisation de ce document.

D'autre part, toute contrefaçon, plagiat, reproduction illicite de ce travail expose à des poursuites pénales.

Contact : portail-publi@ut-capitole.fr

LIENS

Code la Propriété Intellectuelle – Articles L. 122-4 et L. 335-1 à L. 335-10
Loi n°92-597 du 1er juillet 1992, publiée au *Journal Officiel* du 2 juillet 1992
http://www.cfcopies.com/V2/leg/leg-droi.php
http://www.culture.gouv.fr/culture/infos-pratiques/droits/protection.htm

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par l'université Toulouse1 Capitole (UT1 Capitole)
Discipline : Informatique

Présentée et soutenue par Fatma ABDELHÉDI
Le 03 avril 2014

Conception assistée d'entrepôts de données et de documents XML pour l'analyse OLAP

JURY

Philippe ANIORTE	Professeur à l'université de Pau	Examinateur
Omar BOUSSAID	Professeur à l'université Lyon 2	Rapporteur
Corine CAUVET	Professeur à l'université Aix-Marseille 3	Rapporteur
Claude CHRISMENT	Professeur à l'université Toulouse 3	Président du jury
Faïez GARGOURI	Professeur à l'université de Sfax	Examinateur
Gilles ZURFLUH	Professeur à l'université Toulouse 1	Directeur de thèse

---

Ecole doctorale : EDMITT
Unité de recherche : Institut de Recherche en Informatique de Toulouse (UMR 5505)
Directeur de Thèse : Gilles ZURFLUH

Remerciements

Je tiens à remercier très sincèrement Monsieur Claude CHRISMENT Professeur à l'université Toulouse 3, Madame Josiane MOTHE Professeur à l'ESPE de Toulouse et Monsieur Gilles ZURFLUH Professeur à l'université Toulouse 1, qui ont tous trois dirigé l'équipe « Système d'Informations Généralisées » (SIG) de l'IRIT pendant le déroulement de ma thèse.

J'adresse mes remerciements les plus sincères à Monsieur Gilles ZURFLUH, pour avoir dirigé et encadré cette thèse, pour sa rigueur scientifique, pour ses critiques constructives ainsi que pour ses précieux conseils et encouragements. Je lui suis très reconnaissante de m'avoir permis de faire mes premiers pas dans le monde de la recherche. Je le remercie aussi pour la grande liberté d'action et la confiance qu'il m'a accordées. Merci également pour les excellentes conditions de travail qu'il m'a offertes, ainsi que pour ses qualités d'écoute et sa bonne humeur. Qu'il soit assuré de ma profonde reconnaissance et de mon très grand respect.

Je tiens à mentionner le plaisir et l'honneur que m'ont fait Monsieur Omar BOUSSAID Professeur à l'Université Lyon 2 et Madame Corine CAUVET Professeur à l'Université d'Aix-Marseille 3, qui ont accepté d'être rapporteurs de mon travail. Je les remercie pour les remarques pertinentes qui ont permis d'améliorer mon mémoire et pour l'honneur qu'ils me font en participant au jury.

Je tiens à mentionner le plaisir et l'honneur que m'ont fait Monsieur Claude CHRISMENT, Monsieur Faïez GARGOURI Professeur à l'Université de Sfax et Monsieur Philippe ANIORTE Professeur à l'IUT de Bayonne pour leurs remarques et pour l'honneur qu'ils me font en participant au jury de ma thèse.

Résumé

Aujourd'hui, les entrepôts de données constituent un enjeu majeur pour les applications décisionnelles au sein des entreprises. Les sources d'un entrepôt, c'est à dire l'origine des données qui l'alimentent, sont diverses et hétérogènes : fichiers séquentiels, feuilles de tableur, bases de données relationnelles, documents du Web. La complexité est telle que les logiciels du marché ne répondent que partiellement aux attentes des décideurs lorsque ceux-ci souhaitent analyser les données.

Nos travaux s'inscrivent donc dans le contexte des systèmes décisionnels qui intègrent tous types de données (principalement extraites de bases de données relationnelles et de bases de documents XML) et qui sont destinés à des décideurs. Ils visent à proposer des modèles, des méthodes et des outils logiciels pour élaborer et manipuler des entrepôts de données. Nos travaux ont plus précisément porté sur deux problématiques complémentaires : l'élaboration assistée d'un entrepôt de données ainsi que la modélisation et l'analyse OLAP de documents XML.

L'élaboration assistée d'un entrepôt de données

Les processus de description et d'alimentation d'un entrepôt exigent un niveau de compétence tel qu'ils sont généralement confiés à un informaticien (administrateur des données). Ceci impose aux décideurs, dans un premier temps, d'exprimer leurs besoins à l'informaticien puis, dans un second temps, de valider l'entrepôt de données élaboré par cet informaticien. Nous sommes convaincus que la généralisation de l'accès à l'information, tant pour le grand public que pour les professionnels, nécessite des systèmes d'entreposage que les utilisateurs pourront manipuler eux-mêmes de bout en bout. Nous proposons donc un processus d'assistance qui permet à un décideur d'élaborer lui-même son entrepôt de données à partir d'une source identifiée, ceci sans avoir recours à un informaticien. Après avoir modélisé les sources et l'entrepôt multidimensionnel avec le langage UML, nous formalisons un processus selon lequel le décideur définit les faits et les dimensions de l'entrepôt de manière incrémentale. Un prototype logiciel (le système SelfStar) a été développé et a permis de valider notre approche.

La modélisation et l'analyse OLAP d'un entrepôt de documents

L'objectif à long terme de nos travaux est d'étendre le processus d'assistance ci-dessus aux données multimédia. Mais une telle extension se heurte à la complexité de certaines données ; c'est notamment le cas des documents XML dont la structure et le volume ne sont pas compatibles avec les solutions d'entreposage classiques.

Les travaux présentés dans cette thèse ont porté sur la modélisation d'un entrepôt de documents et son analyse OLAP. Nous proposons un modèle multidimensionnel et un processus permettant d'élaborer automatiquement un entrepôt. La source de l'entrepôt est une collection thématique de documents XML identifiée par le décideur. Le schéma conceptuel de l'entrepôt obtenu permet d'exprimer des requêtes OLAP en termes de faits et de dimensions ; ces requêtes sont ensuite traduites dans le langage standard XQuery. Le langage d'analyse proposé est entièrement spécifié. Un prototype de traducteur a été développé pour transformer toute requête d'analyse en requête XQuery.

Mots-clés : entrepôt de données, entrepôt de documents XML, base de données décisionnelle, modèle multidimensionnel, analyse OLAP, Extraction Transformation Chargement.

Abstract

Today, data warehouses are a major issue for business intelligence applications within companies. Sources of a warehouse, i.e. the origin of data that feed, are diverse and heterogeneous sequential files, spreadsheets, relational databases, Web documents. The complexity is such that the software on the market only partially meets the needs of decision makers when they want to analyze the data.

Therefore, our work is within the decision support systems context that integrate all data types (mainly extracted from relational databases and XML documents databases) for decision makers. They aim to provide models, methods and software tools to elaborate and manipulate data warehouses. Our work has specifically focused on two complementary issues: aided data warehouse and modeling and OLAP analysis of XML documents.

Aided data warehouse design

Description and loading process for a data warehouse require a level of skill as they are usually assigned to a computer specialist (Data Administrator). This requires decision makers, at first time, to express their needs to the computer specialist and then, in a second time, to validate the data warehouse designed by the computer specialist. We are convinced that the generalized access to information, both the general public and professionals, requires storage systems that users can manipulate themselves from start to finish. We propose a support process that allows a decision maker to develop his own data warehouse from an identified source, this without recourse to a computer specialist. After modeling sources and the multidimensional warehouse with the UML language, we formalize a process by which the decision maker defines incrementally facts and dimensions of the warehouse. A software tools (SelfStar system) was developed and allowed to validate our approach.

Modeling and OLAP analysis of XML documents

The long-term goal of our work is to extend the above support process with other multimedia data. But such an extension is hampered by the complexity of some data,

this in the case of XML documents whose structure and volume are not compatible with conventional warehousing solutions.

The work presented in this thesis focused on the modeling of a document warehouse and OLAP analysis. We propose a multidimensional model and a process allowing automatically design a warehouse. The source of the warehouse is a thematic collection of XML documents identified by the decision maker. The schema of the virtual warehouse obtained allows expressing OLAP queries in terms of facts and dimensions; these queries are then translated into the standard XQuery language. The proposed analysis language is fully specified. A translator tools was developed to convert any query analysis in XQuery.

Key-words: Data Warehouse, XML document warehouse, Decisional data, multidimensional model, OLAP analysis, Extract Transform Load.

Sommaire

Table des figures

CHAPITRE 1 INTRODUCTION GÉNÉRALE

Aujourd'hui, les entrepôts de données constituent un support essentiel pour les applications décisionnelles au sein des entreprises. Les sources d'un entrepôt, c'est à dire l'origine des données qui l'alimentent, sont diverses et hétérogènes : fichiers séquentiels, feuilles de tableur, bases de données relationnelles, documents du Web. La complexité est telle que les logiciels du marché ne répondent que partiellement aux attentes des décideurs.

Les travaux que nous avons menés s'inscrivent donc dans le contexte des systèmes décisionnels qui intègrent tous types de données (principalement extraites de bases de données relationnelles et de bases de documents XML) et qui sont destinés à des décideurs pour effectuer des analyses OLAP. Ce type d'acteurs humains correspond à des utilisateurs des systèmes d'information qui :

- ne maîtrisent pas les outils informatiques (non informaticiens),
- ont des besoins ponctuels (utilisateurs occasionnels), ce qui justifie de ne pas apprendre des langages d'analyse complexes,
- souhaitent un haut niveau de service dans le cadre de l'analyse des données, notamment en mixant des données structurées avec des données faiblement structurées.

Nos travaux visent à proposer des modèles, des méthodes et des outils logiciels pour permettre à des décideurs, n'ayant aucune formation informatique spécifique, d'élaborer un entrepôt de données en toute autonomie. Cette approche est originale lorsqu'elle s'applique aux systèmes décisionnels ; mais elle reprend le principe des langages de requêtes déclaratifs (SQL et QBE notamment) permettant à des non informaticiens d'élaborer des requêtes de façon autonome pour manipuler des bases de données relationnelles.

Notre thèse porte plus précisément sur deux problématiques complémentaires :

- l'élaboration assistée d'un entrepôt de données,
- la modélisation et l'analyse OLAP de documents XML.

Cette dichotomie des problèmes abordés est liée au niveau de maturité de l'état de l'art. En effet, l'objectif général de nos travaux est bien de définir un système intégré permettant à un décideur d'élaborer lui-même des entrepôts mixant données et documents. Mais les modèles et langages actuels dédiés aux documents XML ne sont pas compatibles avec notre objectif ; la spécification d'un nouveau modèle de documents a donc été un préalable à la prise en compte des documents dans le processus d'élaboration assisté d'un entrepôt.

Partie I : L'élaboration assistée d'un entrepôt de données

Actuellement, les systèmes commerciaux proposent des outils informatiques capables :

- de décrire et créer un entrepôt de données à partir de sources principalement constituées de bases de données relationnelles ;
- d'alimenter périodiquement cet entrepôt à partir des sources ;
- d'analyser les données de l'entrepôt à l'aide d'un langage de requête textuel ou graphique.

Les deux premières opérations, très techniques, exigent un niveau de compétence tel qu'elles sont généralement confiées à un informaticien (administrateur des données). Ceci impose aux décideurs, dans un premier temps, d'exprimer leurs besoins à l'informaticien puis, dans un second temps, de valider l'entrepôt de données élaboré par cet informaticien.

Or ce processus, basé sur l'intervention d'un informaticien, nous semble lourd et long au regard de la réactivité exigée par les décideurs en matière de données à analyser (Annoni 2007). Nous proposons donc un processus d'assistance qui permet à un décideur d'élaborer lui-même son entrepôt de données à partir d'une source identifiée, ceci sans avoir recours à un informaticien.

Partie II : La modélisation et l'analyse OLAP d'un entrepôt de documents

L'objectif à long terme de nos travaux est d'étendre le processus d'assistance ci-dessus à d'autres types de données (Cauvet and Guzelian, 2008). Mais une telle extension se heurte à la complexité de certaines données ; c'est le cas des documents XML dont la structure et le volume ne sont pas compatibles avec les solutions d'entreposage classiques.

Les travaux présentés dans cette thèse correspondent à une première étape avant le mixage de plusieurs types de données ; ils ont donc porté sur la modélisation d'un entrepôt de documents et son analyse OLAP. Nous proposons un modèle et un processus permettant d'élaborer automatiquement un entrepôt. La source de l'entrepôt est une collection thématique de documents XML identifiée par le décideur. Le schéma de l'entrepôt obtenu permet d'exprimer des requêtes OLAP en termes de faits et de dimensions ; ces requêtes seront ensuite traduites dans le langage standard XQuery et appliquées à la source de documents XML.

L'organisation du mémoire

Nous avons organisé notre mémoire de thèse en deux parties ; chaque partie se déclinant en plusieurs chapitres. Cette dichotomie est liée à la nature des problématiques abordées dans chaque partie. En effet la première partie traite des entrepôts de données ; or les travaux de recherche menés dans ce domaine nous ayant permis d'utiliser des concepts stables, nous avons pu traiter le processus d'élaboration des entrepôts et développer des outils associés. Par contre la seconde partie concerne les entrepôts de documents où les modèles proposés ne font pas l'objet de consensus ; nous avons donc proposé un nouveau modèle multidimensionnel et un nouveau langage d'analyse.

Le CHAPITRE 2 présente le contexte de nos travaux ainsi que la première partie de notre problématique : l'élaboration assistée d'un entrepôt de données.

Dans le CHAPITRE 3, nous présentons les principaux travaux liés à cette problématique. Trois aspects sont notamment abordés : les processus d'élaboration d'entrepôts, les mécanismes d'alimentation et les techniques de personnalisation.

Le CHAPITRE 4 est consacré à nos propositions en matière d'élaboration d'un entrepôt. Le processus incrémental de spécification d'un schéma multidimensionnel est détaillé. La démarche d'élaboration qui associe l'utilisateur non informaticien est décrite.

Le CHAPITRE 5 présente le mécanisme de personnalisation associé au processus d'élaboration de l'entrepôt. L'objectif visé est d'assister l'utilisateur non spécialiste pour lui éviter de se perdre dans l'appréhension des objets à analyser.

Dans le CHAPITRE 6, le prototype développé pour mettre en œuvre notre processus d'élaboration est détaillé. Ce système, dénommé SelfStar, prend en compte l'ensemble des mécanismes proposés précédemment. Il a été testé sur un cas réel.

Dans le CHAPITRE 7, nous évoquons le contexte, la problématique et les travaux associés de notre deuxième partie : Modélisation et analyse OLAP de documents.

Le CHAPITRE 8 présente notre démarche d'élaboration d'un schéma d'entrepôt de documents XML. Nous avons proposé un nouveau modèle multidimensionnel StarCD permettant des analyses OLAP des documents. Nous détaillons la formalisation du modèle proposé ainsi que la démarche de mise en œuvre en partant d'une source thématique de documents XML.

Le CHAPITRE 9 est consacré à la définition d'un nouveau langage d'analyse OLAP qui porte sur notre modèle multidimensionnel présenté dans le chapitre précédent.

Finalement, le CHAPITRE 10 conclut sur les deux parties de notre thèse. D'une part, nous rappelons les caractéristiques de nos contributions, leurs originalités mais aussi leurs limites. D'autre part, nous abordons les principales perspectives de recherche envisagées pour atteindre l'objectif général des travaux dans lesquels s'inscrit cette thèse.

PARTIE I

L'ÉLABORATION ASSISTÉE D'UN ENTREPOT DE DONNÉES

CHAPITRE 2

CONTEXTE ET PROBLÉMATIQUE DE L'ÉLABORATION D'UN ENTREPÔT DE DONNÉES

Dans cette section, nous définissons le contexte de l'étude en précisant le vocabulaire et les enjeux sociétaux liés au développement de l'informatique décisionnelle. Nous présentons ensuite notre problématique concernant l'entreposage des données.

2.1. Les systèmes décisionnels

De nos jours, les entreprises ont recours à des systèmes d'entreposage qui offrent aux décideurs une vision synthétique et globale des informations circulant dans leurs organisations, ceci pour les aider à prendre les décisions. Ces systèmes reposent sur des entrepôts de données qui correspondent à des bases de données dédiées à la prise de décision. Généralement, ces entrepôts n'utilisent pas les modèles de données classiques (Entité-Association et Relationnel), mais des modèles multidimensionnels où les données sont organisées en termes de faits et de dimensions. Les modèles conceptuels multidimensionnels (Etoile, Constellation et Flocon) sont adaptés aux analyses OLAP effectuées par le décideur. Ceux-ci sont généralement des non informaticiens et experts d'un domaine de l'entreprise (commercial, production, personnel) ; ils sont chargés d'analyser les données décisionnelles pour assurer le pilotage de l'entreprise.

2.1.1. Le système décisionnel dans l'organisation

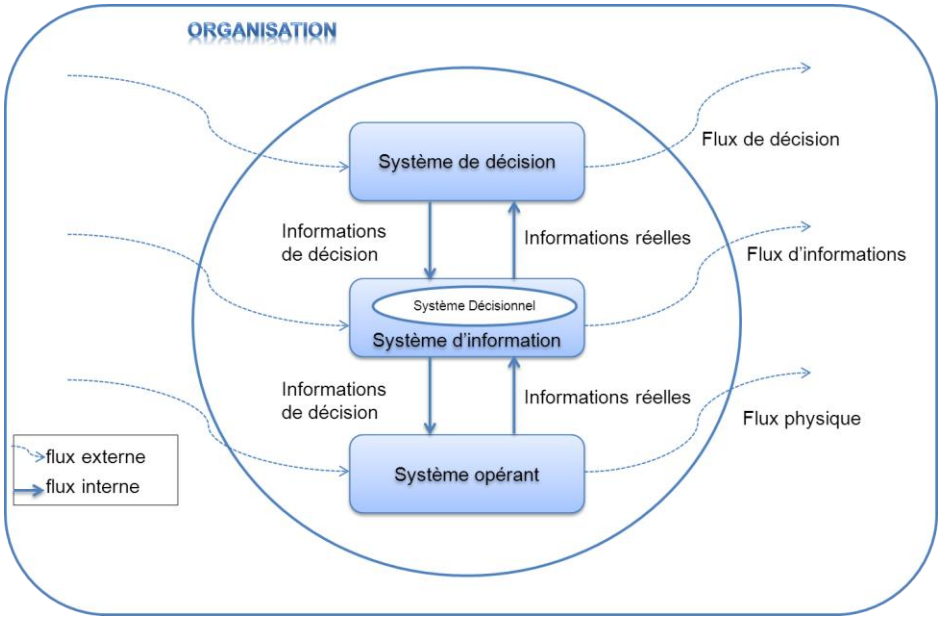

Figure 1. Positionnement d'un système décisionnel dans l'organisation

Toute organisation peut être décomposée en trois systèmes (Le Moigne, 1990) :

- Le système opérant qui correspond à l'activité de production de l'organisation en transformant les flux primaires pour répondre aux besoins des clients,
- Le système de décision correspondant à l'ensemble des traitements et du personnel dirigeant qui contrôle, régule, pilote et adapte l'organisation par leurs décisions,
- Le système d'information permettant de collecter, conserver, traiter et restituer les données produites dans l'organisation ; il joue le rôle d'interfaces entre les deux systèmes précédents.

La Figure 1 présente les trois systèmes et leurs interactions.

Le système d'information comporte deux types de traitements : les traitements transactionnels destinés au système opérant et les traitements décisionnels qui mettent les informations prétraitées à la disposition du système de décision pour assurer un pilotage optimal.

2.1.2. Les traitements décisionnels

Ces traitements consistent à extraire, prétraiter et stocker des données utiles à la prise de décision. Les données peuvent provenir des sources internes à l'entreprise (base de données de production dédiés aux traitements transactionnels) ou de sources de données externes (base de données de partenaires, sites web). Une base décisionnelle contient un ensemble de données destiné au système de décision et organisé selon un modèle adapté à l'usage qui en est fait. Les outils ETL (pour Extract – Transform - Load) permettent d'alimenter régulièrement une base décisionnelle à partir des sources.

Définition 1. Une base décisionnelle est l'ensemble des outils informatiques (matériels et logiciels) permettant l'analyse des données issues du système d'information des entreprises. Ces données sont représentées dans la base décisionnelle en une vision orientée décideur puis analysées au moyen des outils de restitution et d'analyse.

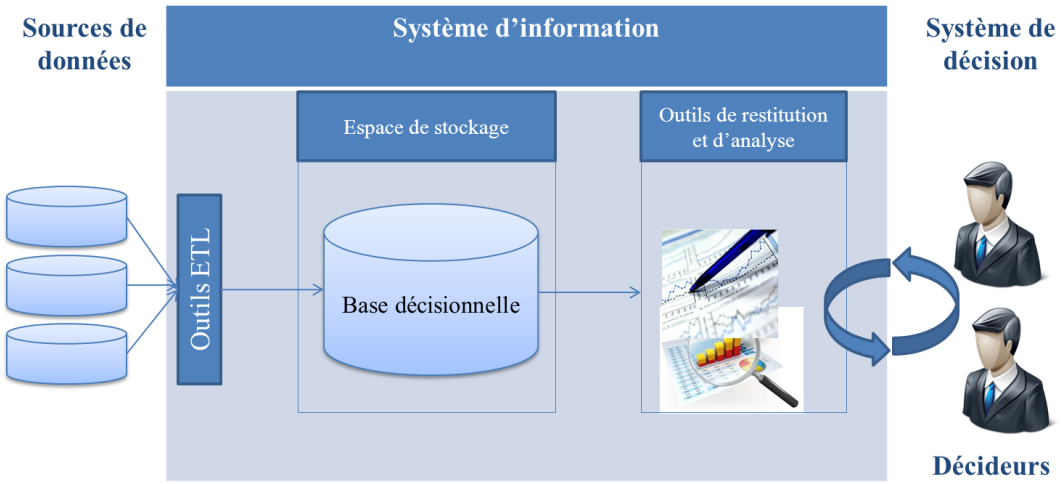

Figure 2. Le système d'aide à la décision

Les données décisionnelles peuvent être organisées selon des modèles différents en fonction de l'usage qui en est fait. On distingue l'entrepôt du magasin (cf. paragraphe 2.1.3).

2.1.3. Entrepôt et magasin

Bill Inmon définit l'entrepôt de données comme « une collection de données intégrées, orientées sujet, non volatiles et historisées, résumées et disponibles pour l'interrogation et l'analyse » (Inmon, 1996).

On parle généralement d'architecture à niveaux en raison des différents espaces de stockage considérés ; on distingue principalement l'architecture à 2 niveaux et l'architecture à 3 niveaux. Par exemple si l'on prend une architecture à 3 niveaux, on considère :

- les sources de données, souvent hétérogènes et réparties, associées aux outils ETL permettent l'intégration et l'alimentation de l'entrepôt ;
- l'entrepôt contenant les données pour la prise de décision ;
- le magasin extrait de l'entrepôt et dédié à une classe de décideurs ; il est organisé suivant un modèle multidimensionnel.

Nous présentons une architecture à 3 niveaux dans la Figure 3.

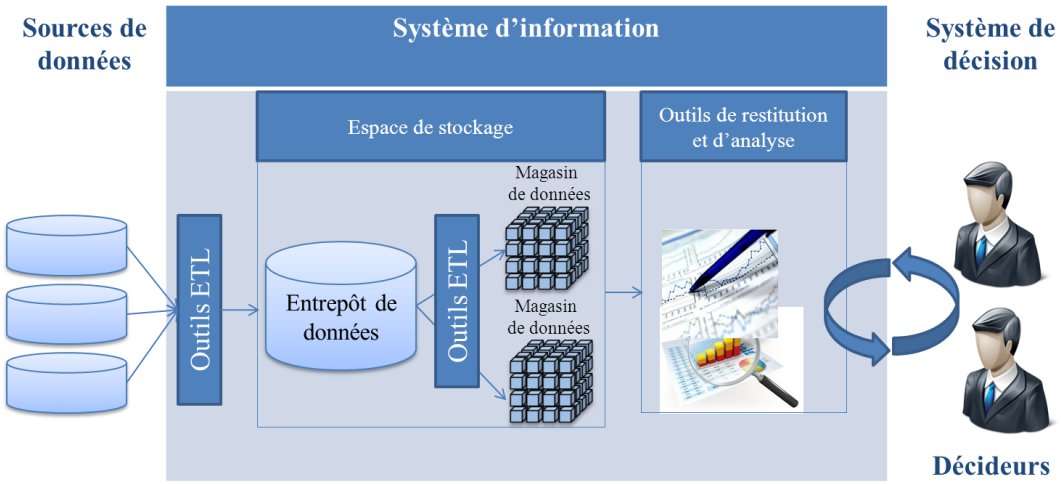

Figure 3. Architecture décisionnelle à 3 niveaux

Cette architecture sépare clairement les deux espaces de stockage ; l'« entrepôt » où les données sont représentées selon un modèle informatique et le « magasin » où les données sont décrites dans un modèle multidimensionnel.

Dans nos travaux, nous considérons l'architecture à deux niveaux distinguant les sources de l'entrepôt. Celui-ci contient des données organisées selon un modèle multidimensionnel. Je décris les données entreposées en termes de fait et de dimension et offre une vue conceptuelle de l'entrepôt aux décideurs.

Nous représentons l'entrepôt par une constellation (Kimball, 1996), c'est-à-dire un ensemble de sujets d'analyse, les faits, et d'axes d'études, les dimensions (Teste, 2009).

Définition 2. Une *constellation* C est définie par le triplet (F ; D ; $Star$) où :

- $F = \{F_1, \dots, F_n\}$ est un ensemble de faits,
- $D = \{D_1, \dots, D_n\}$ est un ensemble de dimensions,
- $Star : F \rightarrow 2^D$ est une fonction qui associe un fait à l'ensemble des dimensions qui permet de l'analyser.

Définition 3. un fait F_i, $\forall\, i \in [1..n]$est défini par ($\mathcal{N}F_i$; M_i) où

- $\mathcal{N}F_i$est le nom identifiant le fait dans la constellation,
- $M_i = \{m_1, \dots, m_{xi}\}$ est l'ensemble de mesures.

Définition 4. $\forall\, i \in [1..m]$, une dimension $D_i \in D$ est définie par le triplet (ND_i; A_i; H_i) où

- ND_i est le nom identifiant la dimension dans la constellation,
- $A_i = \{Id_i, All_i\} \cup P_i \cup W_i$ est l'ensemble *des attributs de la dimension*. On distingue les paramètres $P_i \subseteq P$ représentant les graduations possibles, des attributs faibles $W_i \subseteq W$ représentant des informations additionnelles associées aux paramètres.
- $H_i = \{H_1, \dots, H_{pi}\} \subseteq H$ est l'ensemble des *hiérarchies*.

Propriétés. Les attributs de la dimension D_i respectent les propriétés suivantes :

- Recouvrement des attributs de la dimension : $A = \bigcup_{i=1}^{m} A_i$
- Recouvrement des paramètres : $P = \bigcup_{i=1}^{m} P_i$
- Recouvrement des attributs faibles : $W = \bigcup_{i=1}^{m} W_i$
- Disjonction des attributs de dimension : $\forall\, (j_1, j_2) \in [1..m]^2$, si $j_1 \neq j_2$ alors $A_{j1} \neq A_{j2}$.

Définition 5. $\forall\, H_j \in H_i$ une hiérarchie H_j est définie par NH_j; P_{H_j}; $\prec_{H_j}$; $Weak_{H_j}$ où

- NH_j est le nom identifiant la hiérarchie dans la constellation,
- $P_{H_j} = \{p_i, \dots, p_y\} \subseteq P$ est l'ensemble *des paramètres de la hiérarchie,*
- $\prec_{H_j}$ est une relation d'ordre sur P_{H_j} telle que

- l'ordonnancement des paramètres suit un ordre total $\forall p_{k1} \in P_{H_j}, p_{k2} \in P_{H_j}, k_1 \neq k_2, p_{k1} \prec H_j p_{k2} \vee p_{k2} \prec H_j p_{k1}$
- il existe un paramètre racine $\forall p_{k1} \in P_{H_j}, Id_i \prec H_j p_{k1}$
- il existe un paramètre extrémité $\forall p_{k1} \in P_{H_j}, p_{k1} \prec H_j All_i$ $Weak_{H_j}: P_{H_j} \rightarrow 2^{W_{H_j}}$ associé les paramètres à un ensemble d'attributs faibles.

Propriétés :

Les hiérarchies respectent les propriétés suivantes :

- Recouvrement des hiérarchies : $H = \bigcup_{i=1}^{m} H_i$
- Disjonction : $\forall i_1 \in [1..m], \forall i_2 \in [1..m]$, si $i_1 \neq i_{i2}$ alors$H_{i1} \neq H_{i2}$.

2.1.4. Restitution et analyse OLAP

Selon l'architecture à deux niveaux, un entrepôt de données est structuré suivant une modélisation multidimensionnelle. Ceci permet de représenter l'extension d'un entrepôt sous la forme de points dans un espace à plusieurs dimensions avec la métaphore du cube ou de l'hyper-cube de données.

La Figure 4 présente un exemple de cube qui permet l'analyse des ventes de matériels informatiques. L'analyse des montants de ventes s'effectue en fonction de trois dimensions : les magasins où ont été effectuées les ventes, les dates de ventes et les produits vendus. Chacune de ces dimensions est associée à des paramètres de granularité différente (pour la dimension Magasin : ville, pays et continent). Ces niveaux hiérarchiques permettent d'obtenir des visions plus ou moins synthétiques lors des analyses OLAP.

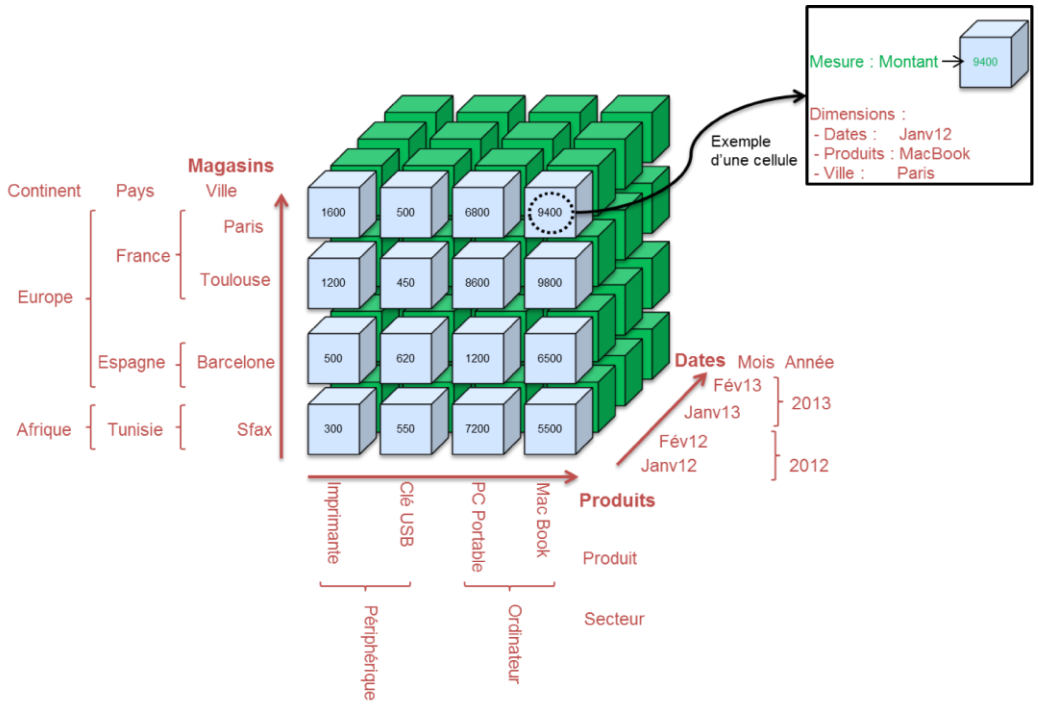

Figure 4. Exemple d'un cube représentant les ventes de matériels informatiques

La modélisation d'un entrepôt sous la forme d'un cube s'avère très limitée puisqu'elle se limite à trois dimensions (Torlone, 2003). Pour concevoir des schémas multidimensionnels plus élaborés, des structures plus avancées ont été définies ; elles permettent la modélisation de sujets d'analyse appelés faits, et d'axes d'analyse appelés dimensions (Kimball, 1996), (Abelló et al., 2001a) et (Abelló et al., 2001b). Les faits sont des regroupements d'indicateurs d'analyse appelés mesures. Les dimensions sont composées d'attributs, appelés paramètres, agencés de manière hiérarchique et qui modélisent les différents niveaux de détails des axes d'analyse. Un fait et ses dimensions associées composent un schéma en étoile (Kimball, 1996). Les données des mesures sont appelées données factuelles car elles représentent un événement. Elles correspondent aux données des cellules du cube qui seront analysées en fonction des axes d'analyse. Le schéma multidimensionnel, associé à l'exemple de la Figure 4, est présenté en Figure 5.

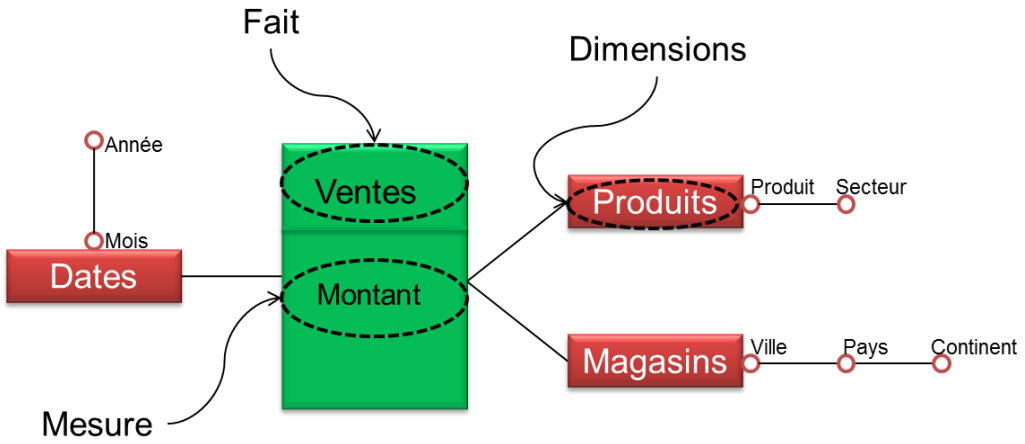

Figure 5. Exemple d'un schéma multidimensionnel

Une analyse multidimensionnelle est une requête partant sur les données d'un entrepôt. Généralement, le résultat d'une requête OLAP est représenté sous la forme d'une table à deux dimensions. La table multidimensionnelle de la Figure 6 représente le résultat d'une requête OLAP. Dans cet exemple, la table contient les analyses des montants des ventes en fonction des pays auxquels appartiennent les magasins. La vente est restreinte aux ventes effectuées en janvier 2012.

Ventes SUM(Montant)			Magasins			
			Continent	Europe		Afrique
			Pays	France	Espagne	Tunisie
Produits	Secteur	Produit				
	Ordinateur	MacBook		19200	6500	5500
		PCPortable		15400	1200	7200
	Périphériques	CléUSB		950	620	550
		Imprimante		2800	500	300
Dates = Janv12						

Valeurs cumulées à partir des magasins situés à Paris et à Toulouse

Figure 6. Une table multidimensionnelle

2.2. Les démarches d'élaboration des entrepôts

La construction d'un entrepôt est un processus qui comporte plusieurs étapes successives : la conception d'un schéma multidimensionnel, la création de l'entrepôt conforme à ce schéma et le chargement de l'entrepôt depuis les sources. La conception d'un schéma multidimensionnel peut être effectuée selon l'une des 3 démarches suivantes.

- La démarche ascendante utilise uniquement le schéma des sources pour générer des schémas multidimensionnels candidats sans prendre en compte, dans un premier temps, les besoins des décideurs. Ceux-ci choisissent ensuite le schéma le plus adapté à leurs besoins.
- La démarche descendante prend uniquement en compte les besoins des décideurs. Elle se base sur la spécification de ces besoins pour définir les sujets et les axes d'analyse. A l'issue du processus d'élaboration du schéma multidimensionnel, la correspondance entre le schéma résultat et la source de données est établie.
- La démarche mixte combine les deux démarches précédentes. En effet, cette démarche construit d'une part des schémas candidats à partir des sources de données (démarche ascendante) et d'autre part des schémas multidimensionnels à partir des besoins d'analyse (démarche descendante). L'informaticien doit confronter ces deux types de schémas pour obtenir un schéma multidimensionnel cohérent et répondant aux besoins des décideurs.

2.3. Le décideur face à ses besoins

Les décideurs effectuent sur les données des analyses OLAP (pour « On Line Analytical Processing ») et de la fouille d'information (Inmon, 2002) et (Missaoui et al., 2007). Dans notre contexte, nous limitons aux traitements de type OLAP qui consistent à réaliser des opérations statistiques simples (somme, moyenne, variance, …) sur des données numériques.

Ces analyses se heurtent généralement à la complexité des bases de données de production qui sont exploitées par les applications transactionnelles au sein des entreprises ; il s'agit principalement de bases de données relationnelles. Ces bases, qui

comportent généralement plusieurs dizaines de tables, sont modélisées par un schéma conceptuel ou logique qui se révèle abscons pour des décideurs dès que le nombre de tables dépasse la dizaine. C'est ce constat qui a conduit les informaticiens à créer des bases décisionnelles distinctes des bases de production et dont l'usage est réservé exclusivement aux décideurs.

2.4. Justification de nos travaux

2.4.1. Des utilisateurs occasionnels

Dans les entreprises, on distingue généralement deux catégories d'utilisateurs d'un système informatique : les opérateurs et les décideurs.

Les opérateurs utilisent des logiciels fermés et répétitifs qui sont élaborés par des informaticiens après une étude minutieuse des besoins des applications transactionnelles ; par exemple des magasiniers utilisent quotidiennement un logiciel pour saisir les mouvements de stock.

Les décideurs ont des besoins généralement pas ou peu prévisibles. Ils paramètrent eux-mêmes des logiciels pour réaliser des traitements spécifiques. Ils sont souvent dénommés utilisateurs occasionnels (Codd, 1982) parce ce qu'ils font un usage ponctuel des logiciels (contrairement aux opérateurs). C'est le cas d'un responsable commercial utilisant un tableur pour élaborer des scénarios commerciaux lors de la négociation d'un contrat. Les entrepôts de données et les logiciels décisionnels sont bien destinés à cette catégorie d'utilisateurs qui n'a généralement pas la maîtrise des techniques informatiques.

2.4.2. Un processus guidé par le décideur

Les données décisionnelles sont issues d'une part des bases de production stockées dans les entreprises et d'autre part des bases de données publiques ou bien détenues par des partenaires. L'entreposage est une réponse à cette diversité des sources ; il permet d'intégrer, prétraiter et organiser les données destinées à l'analyse OLAP. Mais l'accroissement incessant des volumes de données décisionnelles manipulées par les décideurs ne permet pas aux informaticiens de construire des entrepôts qui pourraient anticiper les besoins d'analyse, besoins par nature difficiles à prévoir. Nous sommes donc convaincus de la nécessité de disposer d'outils logiciels permettant aux décideurs d'élaborer eux-mêmes leurs entrepôts.

Notre thèse tente de répondre à cette évolution. En effet, elle vise à permettre à des décideurs, non spécialistes de l'informatique, d'élaborer eux-mêmes des entrepôts de données selon leurs besoins, par essence évolutifs. Il s'agit donc de rendre les décideurs totalement autonomes dans la mise en œuvre d'un système informatique décisionnel.

Cette démarche, qui consiste à donner une certaine autonomie aux décideurs, n'est pas nouvelle en informatique puisqu'elle a donné naissance, dès les années 70, à des langages de requêtes tels que SQL et QBE (Codd et al., 1978) dans le domaine des bases de données ainsi qu'à des logiciels de type tableur ou générateur d'applications dans le monde de la bureautique.

2.5. Notre problématique

2.5.1. Contexte

Comme nous l'avons présenté en section 2.1.3, une base décisionnelle suit une des deux architectures suivantes :

- L'architecture à deux niveaux où l'on considère uniquement la source de données et l'entrepôt. Dans ce cas, le terme entrepôt est défini selon son acception générique ; il s'agit d'une base de données décrite par un schéma multidimensionnel : étoile, flocon, constellation.
- L'architecture à trois niveaux où apparaissent la source, l'entrepôt et le magasin ; ici seul le magasin, extrait de l'entrepôt, est décrit par un modèle multidimensionnel. L'entrepôt, quant à lui, est décrit par un modèle classique (relationnel ou objet) ; il contient les données utiles à la prise de décision (données agrégées et historisées) et il est administré par des informaticiens.

Nos travaux s'inscrivent dans l'architecture à deux niveaux, ceci dans la mesure où l'intégralité du processus d'élaboration de l'entrepôt est confié à un décideur. L'entrepôt est donc décrit par un modèle multidimensionnel. Le décideur élabore directement l'entrepôt à partir de la source de données.

Cependant et hors du champ d'application de SelfStar, le décideur peut extraire d'un entrepôt un ou plusieurs magasins matérialisés correspondants à des vues multidimensionnelles particulières.

2.5.2. Objectif de nos travaux

Jusqu'à présent, les logiciels d'entreposage sont mis en œuvre par des informaticiens pour le compte des décideurs ; ceux-ci expriment leurs besoins d'analyse et les informaticiens construisent un entrepôt : élaboration du schéma multidimensionnel, création de l'entrepôt, paramétrage d'un progiciel pour le chargement des données. L'intervention des informaticiens, qui nécessite une communication longue et parfois minutieuse avec les décideurs pour comprendre et formaliser les besoins d'analyse, s'avère indispensable avec les systèmes logiciels actuels. Et cette intervention représente une forme d'entrave dans l'activité d'analyse de données souvent ponctuelle, imprévisible et limitée dans le temps. Les logiciels commerciaux tels que Oracle-Warehouse, Tableau software (TableauSoftware, 2014) et Business-Object offrent les fonctionnalités permettant à des informaticiens de construire des entrepôts et magasins. Mais le processus actuel de construction d'un entrepôt présente à notre avis deux inconvénients :

- la nécessité pour le décideur d'exprimer explicitement son besoin-métier auprès d'un informaticien ; celui-ci peut mal appréhender ou mal interpréter les spécifications qui lui sont fournies et ainsi déformer le besoin réel ;
- la nécessité de passer par un intermédiaire (ici l'informaticien) entraine un allongement du temps d'élaboration de l'entrepôt.

Nos travaux ont pour objectif d'étudier le processus d'élaboration d'un entrepôt de données et de proposer une démarche, des techniques et des outils pour permettre à des décideurs de concevoir eux-mêmes leur entrepôt de bout en bout.

En effet, il est généralement admis (Mélèse, 1982) que les décideurs doivent être « proactifs » face à l'évolution de leur environnement décisionnel. Dans ce cadre, l'autonomie des décideurs apparaît donc un enjeu moyen pour accéder rapidement aux informaticiens utiles et permette des analyses pertinentes.

Notre thèse repose donc sur la conviction que les décideurs ont la capacité :

- de définir des entrepôts à partir de la connaissance conceptuelle qu'ils ont des sources et des besoins d'analyse ;

- de paramétrer un logiciel pour construire et alimenter les entrepôts, ceci sans l'aide directe des informaticiens.

2.6. Les verrous technologiques

Nos travaux doivent résoudre plusieurs problèmes de nature conceptuelle ou technique. Ces verrous sont au nombre de trois :

1. La complexité du schéma de la source tout d'abord. Pour être autonome, un décideur doit pouvoir appréhender la sémantique des bases de données source au travers d'un schéma. Les principaux modèles utilisés pour schématiser les sources internes à une entreprise sont les modèles Relationnel, Entité-Association et Objet. Même si ces modèles sont utilisés depuis plus de vingt ans par des informaticiens, la sémantique qu'ils renferment peut s'avérer absconse pour des décideurs.

2. Un décideur connaît bien ses besoins métier en matière d'analyse OLAP ; mais il doit être capable de les formuler clairement, si possible avec un langage simple et précis pour les intégrer lui-même dans le système décisionnel.

3. Le système informatique mis à la disposition d'un décideur doit, tout en présentant une interface-utilisateur simple à mettre en œuvre, être suffisamment puissant pour assurer toutes les fonctionnalités des logiciels d'entreposage classiques et ceci sans nécessiter l'intervention d'un spécialiste.

CHAPITRE 3

ÉTAT DE L'ART

Notre étude des travaux existants porte sur trois thématiques : les démarches d'élaboration des entrepôts de données, le chargement de données et la personnalisation. L'élaboration des entrepôts est réalisée en suivant une démarche particulière (ascendante, descendante ou mixte) ; bien que nos travaux reposent sur la démarche mixte, nous présentons les études qui ont été réalisées dans le cadre de chaque démarche.

3.1. Les démarches d'élaboration des entrepôts de données

Une étude synthétique des démarches de modélisation multidimensionnelle a été proposée dans (Romero and Abelló, 2010).

3.1.1. La démarche ascendante

Cette démarche est guidée par les données source (Golfarelli and Rizzi, 1998) et (Moody and Kortink, 2000) ; elle utilise les sources pour générer des schémas multidimensionnels candidats sans prendre en compte, dans un premier temps, les besoins des décideurs. Les auteurs de (Moody and Kortink, 2000) indiquent que les besoins des utilisateurs sont très évolutifs et difficiles à définir, d'où le risque d'obtenir un schéma instable.

Dans l'article de (Golfarelli et al., 1998), les auteurs présentent un processus pour la conception des schémas multidimensionnels. Ils définissent une méthode semi-automatique pour générer un ensemble de schémas candidats à partir d'un schéma source Entité/Association ou à partir du schéma relationnel. Cette méthode se compose des étapes suivantes (1) la définition du fait, (2) pour chaque fait, 5 sous étapes sont représentées (a) construire un arbre d'attributs, (b) élaguer et greffer l'arbre d'attributs, (c) définir les dimensions, (d) définir les mesures, (e) définir les hiérarchies. Le schéma conceptuel est par la suite transformé en un schéma logique (ROLAP). Au niveau physique, la méthode propose un ensemble d'optimisation basées sur l'analyse des indexes et des vues matérialisées. Cette méthode a été mis en œuvre dans un outils logiciel qui vise à assister les concepteurs (Golfarelli et al., 2002b).

Dans (Song et al., 2008), les auteurs proposent de générer des schémas candidats suivant une démarche ascendante. Ils proposent une nouvelle approche semi-automatique en se basant sur des heuristiques structurelles pour identifier automatiquement les faits candidats à partir d'un schéma Entité/Association en analysant le nombre de liens multivalués (de type 1-N) émanant de chaque classe. Ainsi des dimensions sont générées pour chaque fait candidat en se basant sur des liens

directement liés à la classe du fait. Ils utilisent aussi les patrons et une ressource linguistique telle que WordNet (Miller et al., 1990) et (Miller, 1995) pour étendre le nombre de dimensions. Cette approche vise à simplifier le travail du concepteur pour élaborer un sous ensemble de schémas multidimensionnels candidats. Ces schémas multidimensionnels se limitent à la génération des faits et des dimensions et omettent les hiérarchies de dimensions.

Dans (Pinet and Schneider, 2009), les auteurs proposent un nouveau modèle multidimensionnel construit à partir d'un schéma conceptuel UML représentant la source. Ce modèle représente les classes de la source sous forme d'un graphe acyclique orienté. Le décideur choisit un nœud de ce modèle pour représenter le fait. Tous les nœuds reliés au fait choisi représentent les dimensions potentielles de ce fait. Cependant, cette représentation du schéma multidimensionnel s'avère, à notre avis, complexe pour le décideur en raison du nombre de nœuds générés pour représenter les hiérarchies des dimensions.

Dans (Carmè et al., 2010), les auteurs proposent une approche dirigée par les modèles pour assister les concepteurs dans le processus d'élaboration d'entrepôts de données suivant une démarche mixte. A partir des sources de données relationnelles, le système détecte automatiquement les tables pouvant jouer le rôle de fait, ceci en respectant trois heuristiques. Une table peut être un fait si (1) elle contient plus d'instances que les autres tables, (2) elle dispose d'un large ratio (calculé en fonction du nombre d'attributs numériques par rapport aux attributs non numériques) et (3) elle est peu ou pas référencée par des clés étrangères situées dans d'autres tables. Le résultat est un ensemble de schémas multidimensionnels candidats élaborés pour les faits détectés.

Les approches ascendantes ignorent donc les besoins d'analyse à priori. Elles visent principalement à concevoir les schémas multidimensionnels à partir des schémas sources et supposent que les schémas résultants pourront répondre aux besoins des décideurs.

3.1.2. La démarche descendante

Par opposition à la démarche précédente, la démarche descendante est guidée par les besoins des décideurs (Trujillo et al., 2004) et (Prat et al., 2006b). À l'issue du processus d'élaboration, la correspondance entre le schéma résultat et le schéma de la source est établie. Dans (Jovanovic et al., 2012), les auteurs définissent une méthode semi-automatique pour la conception de schémas multidimensionnels basée sur les besoins des décideurs. La démarche proposée est itérative et interactive. Pour chaque requête, un concepteur élabore un schéma multidimensionnel. En entrée du système, on trouve l'ensemble de ces schémas ainsi qu'une ontologie de domaine. En sortie, le système

génère un nombre limité de schémas multidimensionnels unifiés. Le processus comporte plusieurs étapes successives ; l'utilisateur intervient à la fin de chaque étape pour valider et affiner le résultat de l'itération. L'intérêt de ce système est de proposer un processus interactif pour élaborer des schémas multidimensionnels selon une démarche descendante. D'autre part, cette étude se limite au niveau conceptuel et ne prend pas en compte le chargement des données dans l'entrepôt.

Les travaux de (Prat et al., 2006a) présentent une méthode pour générer des schémas multidimensionnels suivant une démarche descendante. Le concepteur collecte les besoins informels des décideurs pour les représenter avec un diagramme de classes UML. Ce schéma est d'abord traduit en un schéma multidimensionnel en simplifiant le diagramme de classes. La traduction du niveau conceptuel au niveau logique est faite d'une façon semi-automatique par le concepteur et en sollicitant le décideur. Le modèle logique est ensuite traduit au niveau physique par un ROLAP ou MOLAP. Un outil logiciel a été développé et qui vise à assister le concepteur à l'élaboration de l'entrepôt de données au niveau physique en utilisant un SGBD Oracle.

L'approche descendante propose d'élaborer le schéma de l'entrepôt en fonction des besoins des décideurs. Elle suppose que la confrontation du schéma multidimensionnel avec le schéma des sources est toujours possible et sera réalisée dans un second temps.

3.1.3. La démarche mixte

Cette démarche combine les deux processus précédents. Aussi appelée démarche hybride, elle consiste à élaborer un schéma multidimensionnel à partir des besoins des décideurs (généralement formalisés par des requêtes d'analyse) en établissant une correspondance avec le schéma de la source de données. La confrontation entre les deux types de schémas est assurée par le concepteur.

Dans (Romero and Abelló, 2010), les auteurs proposent de partir d'une part des besoins des décideurs exprimés sous la forme de requêtes SQL et d'autre part du schéma d'une base de données relationnelle. L'interrogation des sources est assurée par des requêtes SQL et une connaissance de la sémantique du schéma relationnel de la source. Les auteurs font l'hypothèse que les attributs figurant dans la clause SELECT de chaque requête SQL, appartiennent à une table dans la base de données source et qu'ils correspondent au fait analysé. L'ensemble des schémas produits peut être affiné par l'utilisateur. Bien qu'il soit important de pouvoir faire valider les schémas multidimensionnels par l'utilisateur, cette démarche reste toujours dépendante d'un expert (un informaticien) pour formuler les requêtes SQL en entrée du système.

Les auteurs de (Zepeda et al., 2008) proposent une démarche MDA en partant du modèle conceptuel source et des besoins sources suivant trois étapes. La première étape consiste à transformer le schéma Entité/Association de la source en un schéma multidimensionnel grâce à des métarègles. La deuxième étape collecte les besoins des décideurs en les formalisant avec un modèle de but. La troisième étape consiste à confronter les résultats des deux étapes précédentes pour fournir un schéma multidimensionnel ; l'utilisateur peut alors affiner le schéma résultant. Ces travaux se limitent au niveau conceptuel et ne proposent pas d'outil logiciel pour valider la démarche proposée.

Plusieurs travaux utilisent l'approche MDA (Model Driven Approach) pour formaliser les schémas multidimensionnels ainsi que les processus de transformation de schémas, ceci du niveau conceptuel au niveau physique. Dans (Mazón and Trujillo, 2008) et (Mazón and Trujillo, 2009), les auteurs proposent une méthode basée sur MDA en suivant une démarche mixte. Les besoins des décideurs sont représentés sous forme d'un diagramme de buts au niveau CIM. Ensuite, ce modèle de buts est fusionné avec les modèles sources pour obtenir un modèle conceptuel hybride (PIM hybride). Ce dernier est transformé en un modèle logique (PSM). Enfin, le modèle logique est traduit selon le formalisme spécifique d'une plateforme physique. Dans (Atigui et al., 2012), les auteurs proposent une approche MDA pour formaliser les transformations de schémas de l'entrepôt et de son chargement à partir des sources. Cette méthode permet de formaliser le schéma multidimensionnel et les opérations de chargement de manière conjointe. Elle fournit un ensemble de méta-modèles et de règles de transformation pour générer les modèles logique et physique décrivant à la fois les données et les opérations.

La conception d'un schéma multidimensionnel selon la démarche mixte prend en compte simultanément le schéma des sources et les besoins des décideurs. En raison de son caractère progressif, cette démarche nous a parus mieux adaptée à notre processus qui est destiné à des utilisateurs non informaticiens.

3.2. Le chargement des données

L'alimentation des données dans l'entrepôt est assurée par un processus ETL (acronyme anglais pour Extract, Transform, Load) pour extraire, traiter et enregistrer les données dans l'entrepôt. Ce processus automatique assure non seulement le premier chargement de l'entrepôt depuis les sources mais aussi les rafraîchissements périodiques au fil de l'exploitation. Des logiciels ETL standards et paramétrables ont été développés (Vassiliadis, 2009); par exemple les systèmes commerciaux Oracle Warehouse Builder, SAS/Warehouse Administrator, SAS BusinessObject, ou « open

source » tel que Talend Open Studio. Cette étape qui fait suite à la création de l'entrepôt, a fait l'objet de travaux de recherche sur la modélisation et l'automatisation des processus impliqués (Pinet and Schneider, 2009), (Jovanovic et al., 2012) et (Atigui et al., 2012).

La modélisation conceptuelle des processus ETL consiste à décrire la correspondance entre les métadonnées sources et les métadonnées cibles.

Dans (Vassiliadis, 2009), les auteurs proposent un modèle conceptuel décrivant les concepts ETL utilisés et l'architecture des processus correspondants. Ce modèle permet d'assurer la correspondance entre les métadonnées des sources de données et ceux de l'entrepôt. Il décrit notamment la correspondance avec un métamodèle formalisé en UML.

Dans (Pinet and Schneider, 2009), les auteurs décrivent le processus de chargement de données entre la source et l'entrepôt. Le chargement est assuré grâce à une table de correspondance entre les métadonnées des sources et les métadonnées des tables multidimensionnelles. Cette table de correspondance peut aussi être utilisée pour la génération du code SQL pour le chargement des attributs calculables dans les tables multidimensionnels.

Dans (Atigui et al., 2012), les auteurs proposent une approche MDA permettant de formaliser les transformations entre modèles et automatiser les processus ETL. L'approche fournit une extension du langage OCL spécifiant les opérations de transformation entre métadonnées sources et métadonnées cibles au niveau conceptuel. Ensuite, un ensemble de règles de transformation est appliqué pour générer les modèles logique et physique ainsi que les scripts d'alimentation automatique.

3.3. La personnalisation

Dans les années récentes, des recherches sur la personnalisation ont été menées dans les domaines des systèmes d'information, de la recherche d'information et des bases de données. Néanmoins, peu de travaux ont abordé la personnalisation dans les entrepôts de données (Ahmed et al., 2011). Généralement, la personnalisation d'un système consiste à définir, modéliser puis exploiter un profil utilisateur qui permet d'adapter le comportement du système à un utilisateur particulier. Un tel profil regroupe généralement un ensemble d'attributs correspondant aux caractéristiques de l'utilisateur.

Selon la Figure 7 (Bentayeb et al., 2009), un profil peut exiger une interaction directe de l'utilisateur pour choisir une recommandation ou bien exécuter une transformation automatique ; il s'agit du mode configuration. Par contre dans le mode adaptation, le système déduit automatiquement un profil utilisateur à partir de l'observation de son fonctionnement. Par exemple dans le cadre de la recherche d'information nous trouvons les travaux de (Bradley et al., 2000) où les auteurs proposent des techniques de personnalisation en définissant les profils selon un système adaptatif. Les profils sont mis à jour par rapport au comportement des utilisateurs à travers l'interaction entre le système et l'usager.

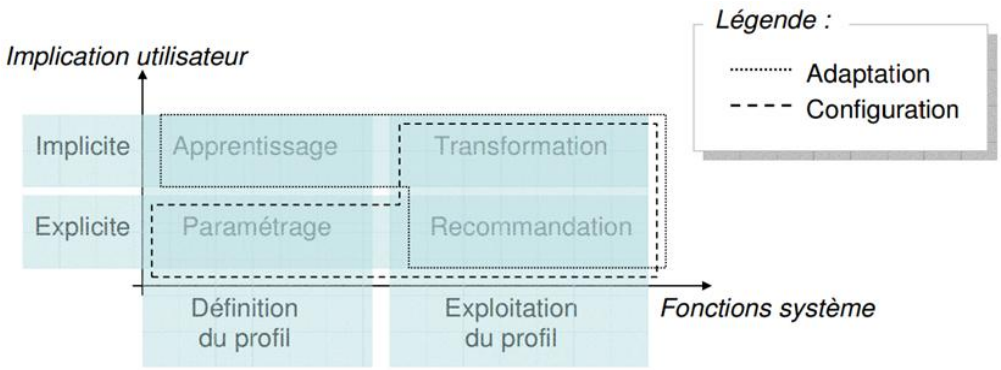

Figure 7. Principe de la personnalisation (Bentayeb et al., 2009)

D'autre part, on considère généralement deux types d'approches pour la personnalisation, les approches qualitative et quantitative. L'approche qualitative utilise des relations binaires ; par exemple « je préfère plutôt travailler dans un laboratoire que dans une entreprise ». Par contre, l'approche quantitative permet d'exprimer un degré de personnalisation avec des poids (exprimés entre 0 et 1) ; par exemple « mon intérêt pour travailler dans un laboratoire est de 0.8 alors qu'il est de 0.6 pour une entreprise ». Nous remarquons que l'approche qualitative présente une expression moins restrictive que l'approche quantitative dans la mesure où elle permet d'affecter des scores aux différents concepts.

Dans le cadre des entrepôts de données et selon les travaux de (Hurtado et al., 1999), (Favre et al., 2007) et (Garrigós et al., 2009), les besoins des décideurs peuvent évoluer dans le temps et nécessiteraient des mises à jour différenciées du schéma de l'entrepôt. Les mécanismes de personnalisation permettent donc d'adapter le schéma d'un entrepôt aux besoins évolutifs de chaque décideur dans le temps.

Les travaux sur la personnalisation des schémas d'un entrepôt se répartissent selon deux axes : l'évolution du schéma et la gestion de vues. Le premier axe concerne la mise à jour des hiérarchies de dimensions : Notamment l'ajout ou la suppression d'un niveau de granularité ou bien l'ajout ou la suppression d'une instance de dimension. Ainsi dans (Hurtado et al., 1999), les auteurs proposent des opérateurs permettant l'évolution des hiérarchies de dimensions. Tandis que plus récemment, (Favre et al., 2007) ont proposé une approche à base de règles pour ajouter de nouvelles granularités dans la hiérarchie des dimensions existantes.

Dans le deuxième axe concernant les vues, nous trouvons les travaux développés dans (Garrigós et al., 2009) où les auteurs définissent la personnalisation d'un schéma multidimensionnel basé sur la génération de vues particulières pour chaque décideur. Cette approche s'inscrit au niveau conceptuel en collectant les préférences spécifiques des différents décideurs et en leurs proposant un schéma multidimensionnel personnalisé (ou vue personnalisée). Pour spécifier le profil d'un décideur, une base de règles ECA (pour Event, Condition, Action) répertorie les conditions exprimées manuellement par le décideur. Dans (Ravat and Teste, 2008), les auteurs proposent la personnalisation des requêtes OLAP. Une requête étendue est générée à partir d'une requête de l'utilisateur, du schéma de l'entrepôt et du profil de l'utilisateur comportant plusieurs règles. Ces règles de personnalisation consistent à associer un poids aux différents éléments multidimensionnels d'une constellation (mesures, paramètres et attributs faibles). Ce poids modélise l'importance que l'utilisateur souhaite associer à chaque élément. Un poids w doit être compris entre 0 et 1. Lors de l'exploitation du profil, le système affiche uniquement les données relatives aux éléments ayant un poids supérieur à un seuil. Cependant, les travaux cités ci-dessus dans le cadre des entrepôts de données nécessitent une intervention explicite du décideur pour intégrer ses préférences et configurer son profil dans le système.

3.4. Intérêts et limites des travaux actuels

Dans cette section, nous dressons un tableau comparatif des travaux étudiés précédemment. Ce tableau montre qu'aucun travail ne couvre l'ensemble des critères définis ci-dessous.

Nous comparons les travaux que nous venons de présenter, en nous basant sur les critères suivants :

- La démarche utilisée : ascendante, descendante ou mixte.
- L'automatisation : le processus proposé peut être automatique, semi-automatique (avec intervention humaine) ou manuel.

- La source : le(s) type(s) de bases de données source acceptée(s) : Relationnel, Entité-association, Objet.
- Le formalisme de représentation des besoins d'analyse : tableaux de bord, requêtes SQL, etc.
- La formalisation de la démarche proposée.
- Les niveaux d'abstraction des modèles utilisés : conceptuel, logique ou physique.
- La personnalisation de la conception par décideur.
- L'implication du décideur dans l'élaboration du schéma multidimensionnel.
- Le développement d'outil logiciel validant les mécanismes proposés.

Tableau 1 : Synthèse des travaux de la modélisation des entrepôts de données

		(Golfarelli et al., 1998)	(Mazón and Trujillo, 2008)	(Romero and Abelló, 2010)	(Song et al., 2008)	(Zepeda et al., 2008)	(Pinet and Schneider, 2009)	(Carmè et al., 2010)	(Jovanovic et al., 2012)
Démarche		Ascendante	Mixte	Ascendante	Ascendante	Mixte	Ascendante	Ascendante	Descendante
Automatisation		Semi-Automatique	Automatique	Automatique	Semi-automatique	Automatique	Semi-Automatique	Automatique	Semi-automatique
Sources		Schéma E/A		Schéma Relationnel	Schéma E/A, WordNet, Patrons	Schéma E/A	DCL ou Relationnel	Schéma Relationnel	
Représentation des besoins			Modèle de but	Non	UML étendu	Modèle de but	Non		Requêtes
Formalisation			MDA		Algorithmes	MDA	Algorithmes	QVT	
Niveaux d'abstraction	**Conceptuel**	X	X	X	X	X	X		X
	Logique	X	X		X		X	X	
	Physique	X	X						
Personnalisation de la conception		Non	Non	Non	Non	Non	Non	Non	Non
ETL		Oui	Oui	Non	Non	Non	Oui	Non	Non
Implication du décideur		Non	Non	Non	Non	Non	Oui	Non	Non
Outil logiciel		Oui	Oui	Oui	Oui	Oui	Oui	Oui	Non
Modèle multidimensionnel		Etoile	Méta modèle CWM	Constellation	UML étendu	MétaModèle CWM	UML étendu	Métamodèle CWM	Etoile

Nous venons de présenter un panorama des principaux travaux qui ont été réalisés dans le domaine des entrepôts de données. Nous avons pu constater que les processus d'élaboration proposés exigent un niveau d'expertise élevé ; ils font donc intervenir des spécialistes (administrateurs de données, concepteurs, développeurs) tout au long des processus. Autrement dit, la complexité des démarches et des outils logiciels actuels ne permet pas aux décideurs d'être autonomes pour concevoir un schéma multidimensionnel, créer et charger l'entrepôt puis le personnaliser au fil de l'exploitation.

Nos travaux ont donc pour objet de proposer une démarche complète et un environnement logiciel pour permettre à des décideurs de construire, de manière interactive, un entrepôt à partir de bases de données source.

CHAPITRE 4

ÉLABORATION INCRÉMENTALE ET ASSISTÉE D'UN ENTREPÔT

La modélisation conceptuelle d'un entrepôt vise à représenter les données à analyser sous la forme d'un schéma multidimensionnel. Cette modélisation, indépendante de toute contrainte d'implantation logique et physique, permet d'obtenir une vision orientée décideur (Golfarelli et al., 2002a) et facilite la compréhension de l'ensemble des données mises à disposition de l'analyste (Rizzi et al., 2006). L'objectif de nos travaux est de permettre à un décideur, par nature non informaticien, d'élaborer lui-même un schéma multidimensionnel.

Le projet SelfStar que nous allons présenter dans cette thèse, propose une démarche complète et un environnement logiciel permettant d'élaborer des entrepôts de données à partir de sources de données à analyser ; les besoins d'analyse du décideur sont intégrés progressivement tout au long du processus.

4.1. Notre approche

Le système SelfStar assiste le décideur pour élaborer un schéma multidimensionnel (schéma en étoile ou en constellation). Le processus d'élaboration du schéma repose sur une démarche mixte : il part du schéma de la base source et intègre progressivement les besoins d'analyse dans 3 schémas successifs (dits intermédiaires) comme le montre la Figure 8. Des métadonnées sont stockées tout au long de ce processus ; elles permettent d'une part de spécifier des contraintes de chargement de l'entrepôt et d'autre part de personnaliser l'élaboration de schémas ultérieurs (cf. section 4 et 5).

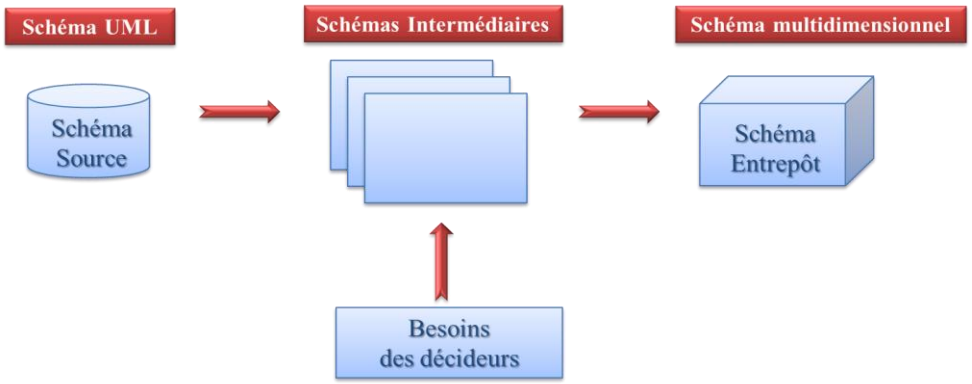

Figure 8. Le système SelfStar : élaboration du schéma de l'entrepôt

Une fois le schéma de l'entrepôt établi, le système extrait automatiquement les données source pour alimenter l'entrepôt en utilisant les métadonnées. L'exploitation de l'entrepôt, c'est-à-dire les requêtes d'analyse ROLAP, s'effectue avec les langages et techniques classiques appliqués à l'entrepôt. L'analyse des données sort donc du champ de nos travaux.

Grâce au système SelfStar, le décideur peut créer son propre entrepôt de données et ceci en toute autonomie. Compte tenu de la simplicité du processus mis en œuvre par le décideur, nous avons écarté la possibilité de mise à jour du schéma de l'entrepôt ; dans le cas où le schéma n'est plus adapté aux besoins évolutifs du décideur, celui-ci supprimera l'entrepôt correspondant et utilisera SelfStar pour en créer un nouveau.

Afin d'illustrer les entrées/sorties du processus d'élaboration de SelfStar, nous définissons les sources, les besoins d'analyse et l'entrepôt.

4.2. La source de données

Un entrepôt peut être défini sur une source de données composée d'une ou plusieurs bases de données homogènes ou hétérogènes (Golfarelli and Rizzi, 1998).

Dans SelfStar, nous avons limité la source à une base de données unique. En effet, la prise en compte de multiples bases posent des problèmes de sémantique que nous n'avons pas traités dans notre thèse. Ces problèmes sont liés à la compatibilité des données : synonymie, polysémie des attributs, liens interbases, etc. Des travaux ont été effectués dans ce domaine de recherche (Cabanac et al., 2010) ; ils utilisent généralement des ontologies pour assurer la correspondance sémantique entre les bases et produisent une base unifiée.

Un processus de fusion de plusieurs bases de données génère une seule base homogène. Par conséquent, nous considérons que la source de SelfStar peut être limitée à une base unique et que ceci ne constitue pas une limite en soi ; cette base de données pouvant résulter d'une fusion préalable de plusieurs bases.

La source de données que nous considérons corresponde à des bases de données de production, c'est-à-dire des bases de données dédiées aux traitements transactionnels. Le modèle Relationnel est principalement utilisé pour décrire ces bases de données aux niveaux logique et physique ; il s'est généralisé dans les entreprises dès les années 1980. Deux autres modèles sont utilisés mais de manière plus confidentielle ; le modèle Entité-Association tout d'abord, mais son application se limite au niveau conceptuel ; le modèle Objet ensuite que l'on trouve aux niveaux conceptuel, logique et physique. Dans SelfStar, nous avons opté pour le modèle Objet qui apparaît comme le modèle le plus général utilisé par les praticiens, ceci pour deux raisons. D'abord, la capacité descriptive du modèle Objet est plus riche que celle des deux autres modèles ; il permet donc de mieux prendre en compte la sémantique des données comme le montre la Définition 6. D'autre part, la correspondance entre le modèle Objet et les deux autres modèles a été établie; un schéma Objet peut donc facilement être traduit en un schéma Relationnel ou un schéma Entité-Association.

Définition 6. une source contient un ensemble de classes d'objets et se définit comme suit :

$S : \{C_1, C_2, \dots, C_n\}$

où $\forall C_i \in S$ avec $i \in \{1,2, \dots, n\}$:

1. si C_i est définie sur un type de données atomiques, généralement prédéfini, tel que Integer, Real, String ou Date, alors C_i est une classe d'atomes ;
2. si les classes $C_1, C_2, \dots, C_q \in S$ avec $q \in \{1, \dots, n\}$, si les symboles $A_1, A_2, \dots, A_q \in \mathcal{N}$ représentent des noms d'attributs et si le constructeur de n-uplet est noté [ ] alors

   $C_i : \{[A_1: C_1, A_2: C_2, \dots, A_q: C_q]\}$ est une classe de n-uplets ;
3. si $C_1, C_2, \dots, C_r$ sont des classes $\in S$ avec $r \in \{1, \dots, n\}$ et si le constructeur d'ensemble est noté { } (et le constructeur de liste est noté ⟨ ⟩), alors $C_i : \{C_1, C_2, \dots, C_r\}$ est un ensemble de classes (respectivement une liste de classes).

Une relation entre classes est exprimée via un attribut dont le nom traduit la sémantique du lien ; elle peut être typée : Héritage (IsA), Composition (C), Association (A).

Toute classe dispose d'un identifiant interne permettant de distinguer les différents objets contenus dans la classe.

Prenons l'exemple d'une source contenant les classes Magasins et Clients ; la classe Produits étant imbriquée dans Magasins.

Magasins : {[NoMag : Integer, Adresse : String, Pro-Vendus (C) : {[NoPro : Integer, Intitulé : String, Poids : Real]}]}

Clients : {[NoCli : Integer, Nom : String, PointVente (A) : Magasins]}

La relation de composition entre les classes Magasins et Produits est exprimée par une imbrication via l'attribut Pro-Vendus. La relation d'association entre Magasins et Clients est traduite par l'attribut PointVente.

Dans les entreprises, les sources sont majoritairement constituées de bases de données Relationnelles. Un schéma Relationnel est un cas particulier de schéma Objet ; autrement dit, le modèle Relationnel correspond à une restriction de la définition du modèle Objet. En reprenant, la Définition 6 d'une source, on obtient la

définition d'une source Relationnelle en modifiant les termes de la définition pour les cas 2 et 3 comme suit :

2- est une classe de n-uplets ; chaque n-uplet est constitué uniquement d'atomes ;

3- est un ensemble de classes ; chacune d'elles contenant exclusivement des n-uplets.

Le cas 3 permet donc d'élaborer une classe de n-uplets, c'est-à-dire une table Relationnelle comme le montre la Définition 7.

Toute table est donc un ensemble de n-uplets constitués à partir des valeurs prises dans les domaines : $T_i \subseteq D_1 \text{x } D_2 \text{ x} \dots \text{x } D_q$.

Définition 7. Une source Relationnelle contient un ensemble de tables et se définit comme suit :

$$SR : \{T_1 , T_2 , \dots , T_n\}$$

Notons D un domaine contenant un ensemble de valeurs atomiques, généralement prédéfini, tel que Integer, String ou Date ;

Notons $\mathcal{N}$ un ensemble de symboles représentant des noms d'attributs ;

Notons [ ] le constructeur de n-uplet et { } le constructeur d'ensemble ;

Le schéma d'une table définie sur un ensemble de q domaines, est noté comme suit :

$\forall T_i \in SR$ avec $i \in \{1,2, \dots , n\}$, $T_i : \{[A_1 : D_1, A_2 : D_2, \dots , A_q : D_q]\}$

Considérons le schéma suivant qui décrit une source Relationnelle contenant les tables Magasins et Produits.

Magasins : {[NoMag : Integer, Adresse : String]}

Produits : {[NoPro : Integer, Intitulé : String, Poids : Real, LieuStockage : Integer]}

Dans cet exemple, l'attribut LieuStockage (clé étrangère) établit une relation entre les tables Magasins et Produits

4.3. La source vue par le décideur

Le décideur n'est pas un informaticien ; il doit donc voir l'organisation des données d'une source au travers d'un modèle simple et lisible. S'il est communément admis qu'un non informaticien peut difficilement élaborer un schéma de données, on peut admettre que, compte tenu du nombre limité de concepts utilisés, la lecture d'un tel schéma est à sa portée.

Le modèle Relationnel peut paraître simple de prime abord. Mais dès que le nombre de tables dépasse la dizaine, un schéma devient abscons principalement en raison de l'absence de dénomination de certaines associations. Le modèle Objet offre une capacité descriptive plus riche mais en contrepartie nécessite de connaître des concepts plus nombreux. Dans le projet SelfStar, nous avons choisi le modèle Objet comme formalisme conceptuel pivot de description d'une source. Mais, chaque fois que ce sera possible, nous masquerons à la vue des décideurs les aspects sémantiques inutiles au processus décisionnel.

4.3.1. Le modèle objet comme modèle unique

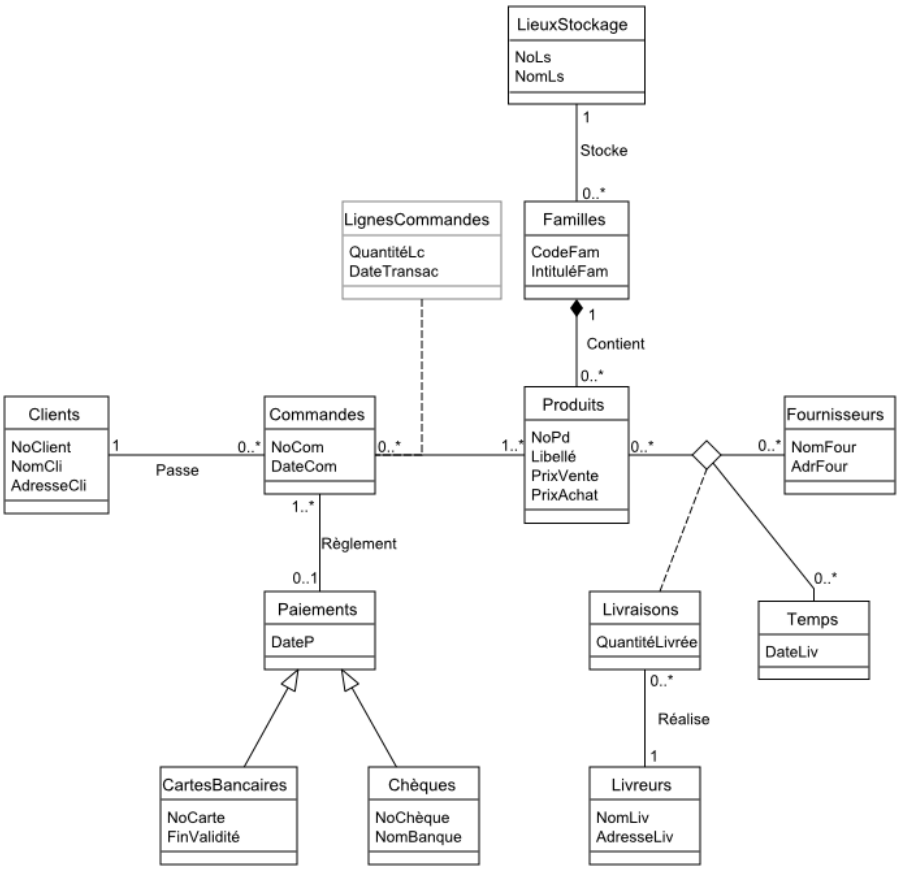

Figure 9. Un DCL pour la gestion des ventes

Toute source de données est décrite avec un diagramme des classes d'UML. Le choix de ce formalisme est justifié par la richesse sémantique du modèle Objet (Prat et al., 2006a) et aussi par la correspondance entre ce langage et les modèles Entité-Association et Relationnel.

La Figure 9 donne l'exemple d'un diagramme de classes (DCL) décrivant une source qui se situe dans le domaine de gestion des ventes de produits.

4.3.2. La simplification du schéma Objet

Le DCL se situe à la source du processus d'élaboration du schéma multidimensionnel. Comme ce schéma comporte des informations sans intérêt pour SelfStar et que l'expression de son traitement algorithmique peut s'avérer complexe, il fait l'objet d'un traitement de réduction. Ce principe a déjà été proposé dans (Song et al., 2007) pour le modèle Entité-Association et dans (Pinet and Schneider, 2009) pour le modèle des classes UML. Selon ce principe, le schéma conceptuel est transformé en un schéma simplifié (le schéma exploitable) comportant uniquement des classes d'objets et leurs attributs ainsi que les relations binaires de type 1..N entre ces classes. La transformation d'un DCL en un schéma exploitable s'effectue comme suit :

- une classe d'objets ou d'associations de la source devient une classe dans le schéma exploitable,
- les domaines des attributs deviennent des classes
- un lien d'association binaire de type 1..N est reporté en l'état,
- un lien d'association binaire de type M..N est transformé en une classe liée par 2 liens 1..N,
- un lien d'agrégation ou de composition est traité comme un lien d'association (ces types de liens ne sont pas significatifs dans un schéma multidimensionnels),
- un lien d'héritage disparaît ; la sous-classe de la source devient une classe et se retrouve au même niveau que la super-classe en héritant de ses attributs et liens (on préserve ainsi la sémantique des données).

Définition 8. Un schéma exploitable est une restriction du schéma source ; il contient un ensemble de classes d'objets :

$$SE : \{C_1 , C_2 , \dots , C_n\}$$

où $\forall C_i \in SE$ avec $i \in \{1,2,\ldots,n\}$, C_i est une classe d'atomes, de n_uplets, d'ensembles ou de listes conforme à la Définition 6.

Les relations entre classes sont limitées aux seuls liens d'association (type A) de cardinalité 1..N.

Notons que dans le cas où la source est une base Relationnelle, le schéma exploitable contient uniquement des tables, c'est-à-dire des classes de n-uplets.

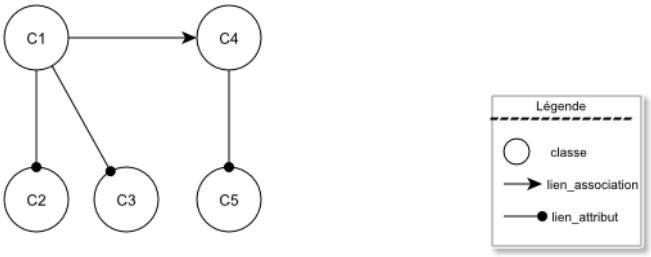

Figure 10. Le schéma exploitable : simplification du schéma conceptuel

Le schéma exploitable ainsi défini est un graphe acyclique orienté comme le montre la Figure 10. Les nœuds correspondent aux classes de la source. On considère deux types de liens entre classes : les liens-attributs et les liens-associations. Un lien-attribut correspond à une fonction qui retourne, pour chaque objet de la classe source, au plus un objet de la classe cible (la valeur de l'attribut). Un lien d'association correspond à une relation sémantique de type 1..N entre classes.

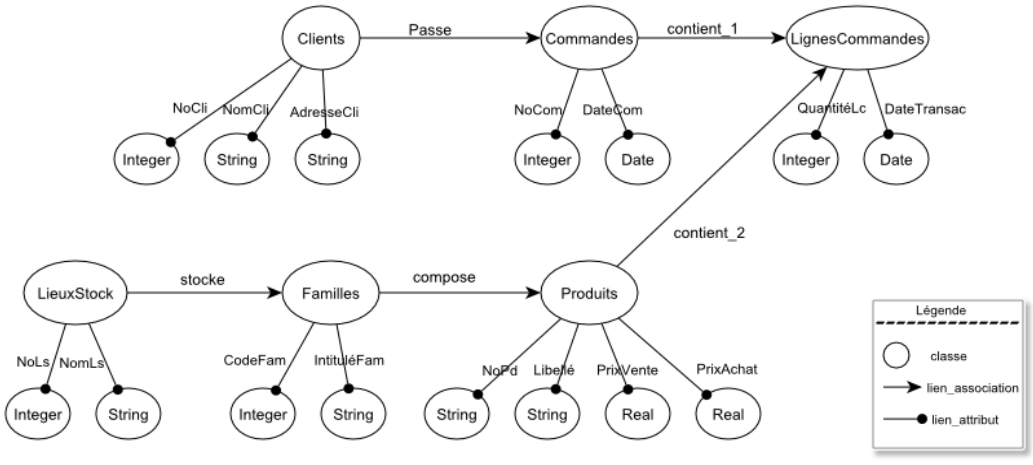

Figure 11. Le schéma exploitable partiel de notre exemple

La Figure 11 présente le schéma exploitable partiel correspondant à l'exemple du DCL de la Figure 9.

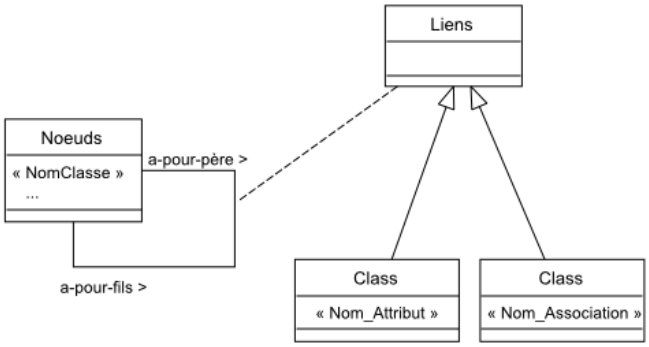

Figure 12. Le méta schéma UML du schéma exploitable

La Figure 12 représente le méta schéma UML permettant d'instancier n'importe quel schéma exploitable.

4.4. Le schéma multidimensionnel

L'un des objectifs de SelfStar est d'assister les décideurs dans le processus d'élaboration d'un schéma multidimensionnel intégrant leurs besoins d'analyse. Les modèles de données classiques Relationnel, Entité-Association et Objet ne sont pas les plus adaptés pour permettre aux décideurs de comprendre un tel schéma (Kimball, 1996). C'est la raison pour laquelle nous reprenons le formalisme proposé dans (Golfarelli et al., 1998) pour présenter les schémas multidimensionnels aux décideurs.

Un schéma multidimensionnel (SM) comporte un ou plusieurs faits autour desquels gravitent des dimensions. Il est constitué d'un ensemble de graphes n-aires. Chaque graphe orienté acyclique correspond à une étoile et a pour racine un fait ; les nœuds correspondent à des classes d'objets et sont reliés à la classe-fait par des liens de type mesure et des liens de type dimension.

Définition 9. Un schéma multidimensionnel est composé d'une ou plusieurs étoiles : $SM : \{E_1, E_2, \dots, E_n\}$

L'étoile E_i est définie par un graphe $[F_i, X_i]$ où F_i est la racine (représentant le fait) et X_i est un ensemble de sous-arbres n-aires fils de F_i (représentant les mesures et les dimensions). Le lien qui relie F_i et chacun de ses sous-graphes

appartient à l'un des deux types suivants :

- un lien-mesure représente une fonction (d'agrégation) entre la classe-fait et une classe d'objets atomiques (Integer, Real, String, etc) ; cette fonction d'agrégation retourne une mesure d'analyse ;
- un lien-dimension exprime une relation entre la classe-fait et une classe d'objets atomique ; cette relation exprime un axe d'analyse du fait ;

Les dimensions sont organisées en hiérarchie où chaque niveau correspond à un niveau de granularité de l'axe d'analyse.

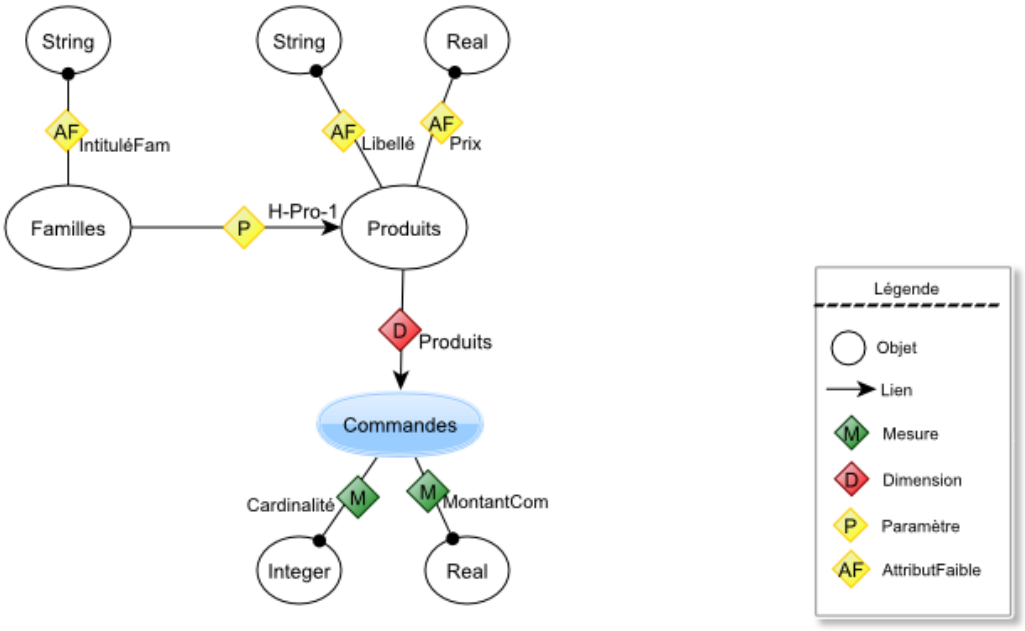

Figure 13. Un schéma multidimensionnel

La Figure 13 présente le graphe correspondant à un extrait du schéma multidimensionnel élaboré par SelfStar à partir du schéma source de la Figure 9.

4.5. Phases du processus

Le processus proposé dans SelfStar a pour point de départ le schéma de la source de données (DCL) et les besoins (informels) du décideur. Contrairement à certaines

propositions (Romero and Abelló, 2010), il n'est pas utile de formaliser les besoins des décideurs sous forme de requêtes SQL.

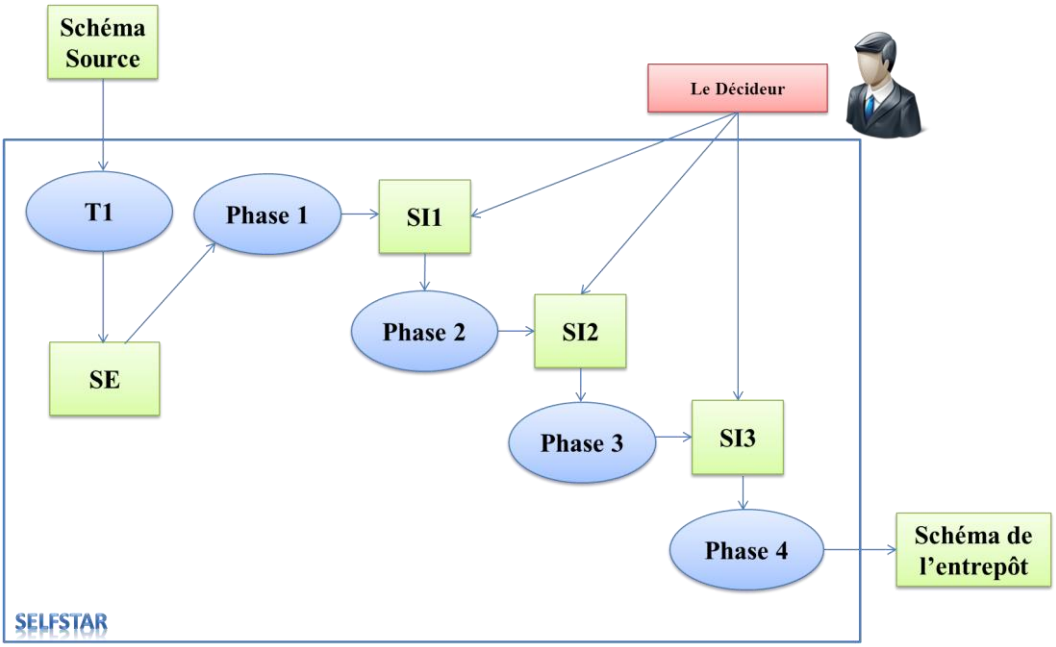

Figure 14. Architecture synthétique de SelfStar

4.5.1. L'élaboration des schémas intermédiaires

Comme la montre la Figure 14, le processus comporte quatre phases successives dans lesquelles le décideur va interagir avec le système pour intégrer progressivement ses besoins. Chaque phase produit un nouveau schéma multidimensionnel plus complet que celui de la phase précédente. Le 4ème et dernier schéma correspond à celui de l'entrepôt multidimensionnel, c'est-à-dire au résultat escompté Figure 29. L'élaboration du schéma est donc effectuée de manière incrémentale. Tout au long du processus, des métadonnées sont mémorisées pour affiner les profils des utilisateurs et pour préparer l'alimentation de l'entrepôt.

La *première phase* consiste à extraire du schéma de la source (plus précisément du schéma exploitable) l'ensemble des faits candidats (susceptibles d'être choisis par les décideurs) et à les présenter dans le schéma intermédiaire numéro 1 (noté SI1). Sur le SI1, le décideur choisit le fait ou les faits qu'il souhaite analyser parmi tous les faits candidats proposés avec les mesures et leurs fonctions d'agrégations.

Dans une *deuxième phase*, le système élabore automatiquement le schéma intermédiaire numéro 2 (SI2) ; il propose toutes les dimensions possibles associées au(x) fait(s) choisi(s). Sur le SI2, le décideur va pouvoir désigner les dimensions, c'est-à-dire les axes selon lesquels il souhaite analyser le fait

La *troisième phase* génère le schéma intermédiaire numéro 3 (SI3) contenant le schéma en constellation (fait(s) + dimensions) avec toutes les hiérarchies possibles. Sur le SI3, le décideur va choisir chacune des hiérarchies correspondante à ses besoins.

La *quatrième phase* du processus va permettre au système d'élaborer le schéma multidimensionnel de l'entrepôt.

Ce processus incrémental va produire plusieurs catégories de métadonnées, notamment :

- des *métadonnées de base* qui assurent la relation entre l'entrepôt et la source,
- des *métadonnées de personnalisation* qui vont assister le décideur dans l'élaboration du schéma multidimensionnel (cf. CHAPITRE 5).

4.5.2. L'interaction du décideur

A la fin de chaque phase, le décideur interagit avec le système comme le montre la Figure 15. Chaque phase produit un schéma intermédiaire contenant des éléments candidats (faits, dimensions ou hiérarchies). Le décideur doit désigner les seuls éléments qu'il souhaite voir figurer dans le schéma multidimensionnel final.

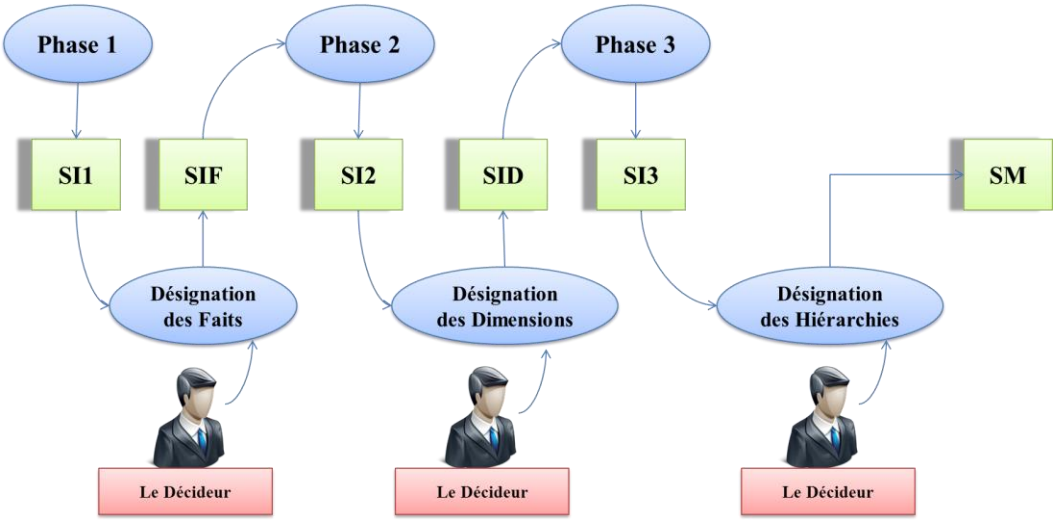

Figure 15. Architecture de SelfStar avec les interactions du décideur

4.5.3. Le Schéma Intermédiaire n°1 : faits et mesures

L'expérience industrielle de notre équipe (Annoni et al., 2006) a mis en évidence que (1) une source de données (BD commerciale ou de production) comporte fréquemment de 30 à 60 classes d'objets et autant de liens et (2) les analyses effectuées par un même décideur sur un entrepôt sont très proches en termes de faits et dimensions. Le Schéma Intermédiaire n°1 (noté SI1) doit être visualisé au décideur afin que celui-ci choisisse les faits à analyser parmi les classes du schéma exploitable (source réduite).

Le constat (1) nous a conduits à proposer d'assister le décideur en ne lui montrant, dans le SI1 les faits candidats ordonnés selon leur poids (le calcul des poids est défini dans la section 4).

Le constat (2) nous a conduits à utiliser des profils de personnalisation pour aider le décideur dans ses choix en lui faisant des propositions de faits potentiels ordonnés. Ceci repose sur des techniques de personnalisation qui ont été développées dans notre équipe (Ravat and Teste, 2009). Le système SelfStar enregistre des métadonnées de personnalisation chaque fois qu'un décideur élabore un schéma décisionnel. Ces métadonnées permettent d'ordonner les classes dans le SI1 selon leur intérêt d'analyse et ceci pour chaque décideur (ces mécanismes de personnalisation sont détaillés dans la section 4).

Comme le montre la Figure 16, SelfStar génère le SI1 à partir du schéma exploitable décrivant la source à analyser ainsi que les métadonnées préenregistrées.

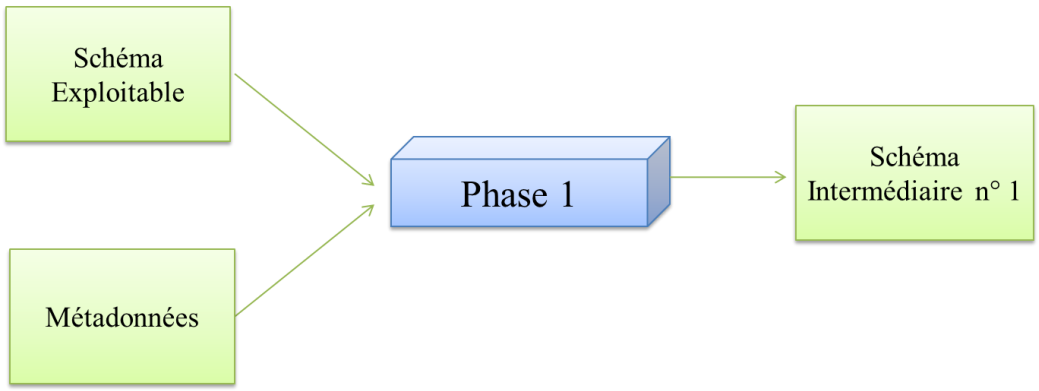

Figure 16. La phase 1 du processus

Le SI1 est une restriction du schéma exploitable. Seuls les attributs numériques ayant des propriétés additives apparaissent dans le SI1 en tant que mesures potentielles. Le SI1 est une suite d'arbres représentant des classes et des attributs pouvant jouer le rôle de faits et de mesures. Une relation d'ordre est définie sur l'ensemble des arbres du SI1. Les arbres sont ordonnés selon leurs poids traduisant

leurs importance à être un fait pour les décideurs. La Figure 17 présente un extrait du schéma intermédiaire n°1 correspondant à notre exemple.

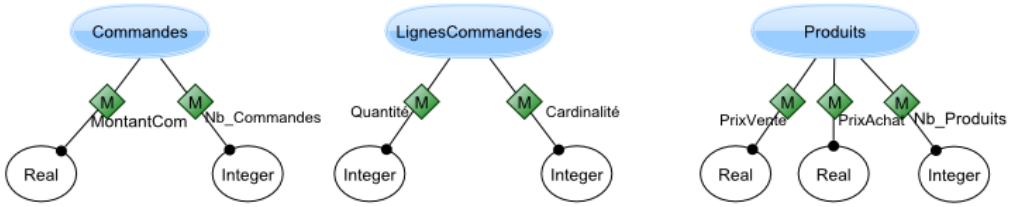

Figure 17. Un schéma multidimensionnel partiel (SI1)

La phase 1 du processus a généré le SI1 qui est un schéma multidimensionnel en construction ; il est composé à ce stade d'un ensemble de faits candidats :

$$SI1 : \{F_1, F_2, \dots, F_n\}$$

Le décideur désigne alors les faits et les mesures qu'il souhaite analyser parmi les faits candidats du SI1. Lors de la désignation d'une mesure, le décideur précise la fonction d'agrégation ; celle-ci permettra d'évaluer la mesure correspondante lors de l'alimentation et de l'interrogation de l'entrepôt. Le décideur peut aussi créer une mesure portant sur plusieurs attributs du SE. Par exemple dans le SI1 de la Figure 17, on considère que le décideur choisit le fait LignesCommandes. Pour définir les mesures du fait, le décideur doit désigner au moins un attribut de la classe et l'associer à une fonction d'agrégation. Par exemple la mesure Cardinalité associe à la fonction count ; de même, la mesure ChiffreAffaires associe les attributs Quantité et PrixVente (de Produits) avec la formule (simplifiée) suivante : SUM (Quantité x PrixVente).

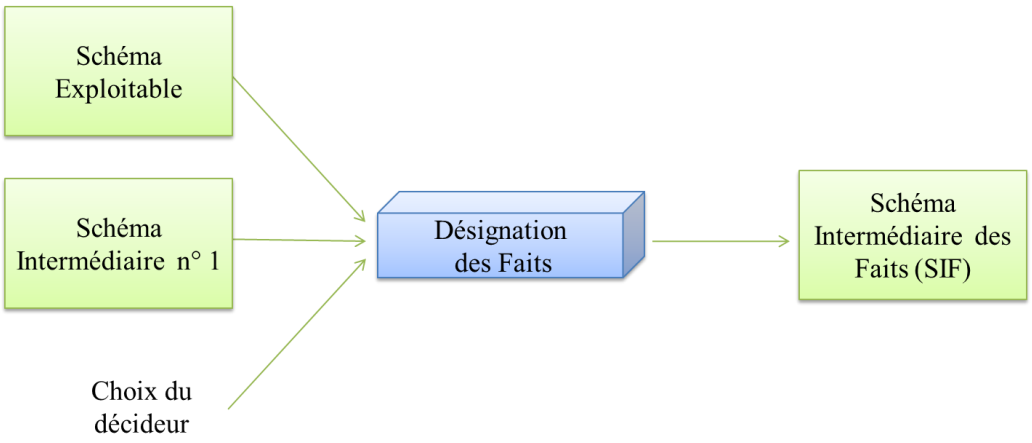

Figure 18. Le processus de désignation des faits par le décideur

La Figure 18 représente les entrées et la sortie du processus de désignation des faits avec l'intervention du décideur.

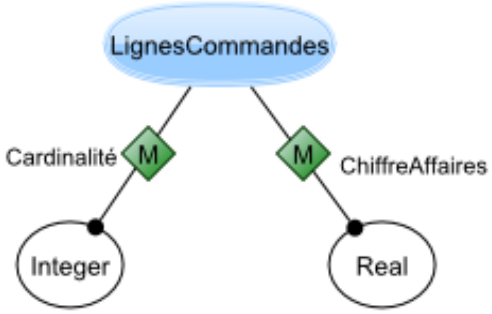

Figure 19. Un Schéma Intermédiaire des Faits (SIF)

Ce processus produit un SIF (Schéma Intermédiaire des Faits) dont un exemple est présenté dans la Figure 19. Le SIF est un schéma multidimensionnel restreint aux faits et mesures choisis par le décideur.

4.5.4. Le Schéma Intermédiaire n°2 : les dimensions

La phase 1 et le processus de désignation des faits terminés, le rôle de la phase suivante est de générer le Schéma Intermédiaire n°2 (SI2) en déterminant l'ensemble des dimensions candidates. Le principe d'élaboration du SI2, qui consiste à déduire les dimensions grâce à l'analyse de la source, n'est pas nouveau et des algorithmes ont été proposés pour les démarches ascendantes (Song et al., 2007) et (Pinet and Schneider, 2009). En entrée de la phase 2 (Figure 20), nous disposons de la source décrite par son schéma exploitable et du SIF contenant le ou les faits choisis par le décideur.

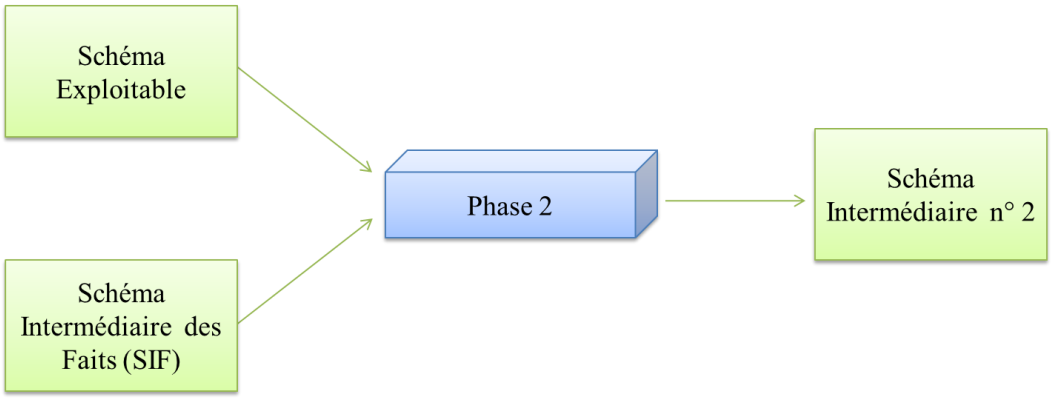

Figure 20. La phase 2

Prenons un fait quelconque dans le Schéma Intermédiaire n°1 : F ∈ SI1. Ce fait correspond à une classe d'objets dans le schéma exploitable : C ∈ SE . La détermination des dimensions candidates associées à F consiste à analyser les relations qui émanent de C dans le schéma exploitable.

Est dimension candidate de F :

1. Une classe D liée directement à C par un lien de type 1..N ; dans ce cas, le nom de la dimension est D, le paramètre est l'identifiant de D et les attributs faibles sont les attributs de D ;

2. un attribut A de la classe C , à l'exclusion (1) des attributs choisis en tant que mesure et (2) des attributs dont les valeurs sont distinctes ; dans ce cas, le nom de la dimension est A et le paramètre est l'identifiant de la classe correspondante au domaine de A.

La Figure 21 illustre le principe de détermination des dimensions.

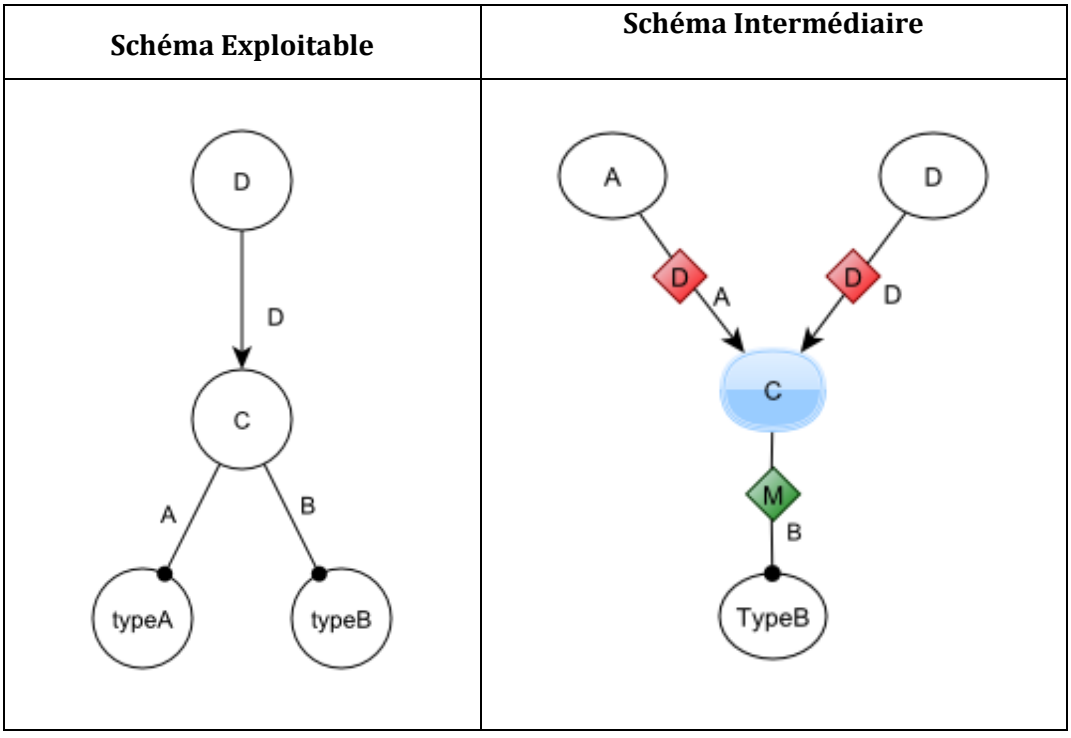

Figure 21. Le principe de détermination des dimensions

Dans la Figure 22, les dimensions candidates du fait LignesCommandes ont été déterminées à partir du schéma exploitable. Sont ainsi devenues dimensions dans le SI2 :

- les classes Commandes et Produits,
- l'attribut DateTransac.

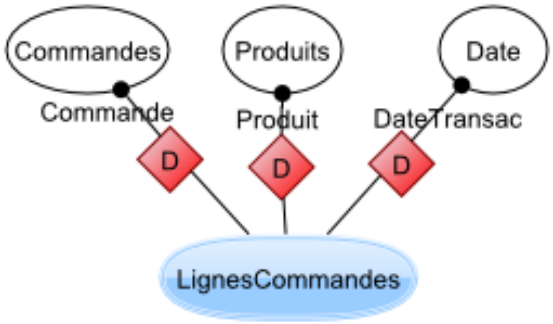

Figure 22. Un schéma multidimensionnel partiel (SI2)

La Figure 23 représente les entrées et sortie du processus de désignation des dimensions avec l'intervention du décideur.

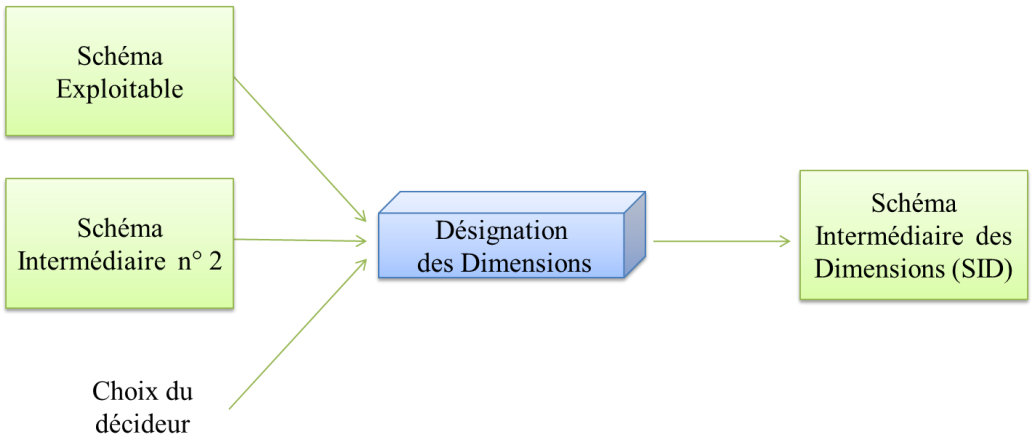

Figure 23. Le processus de désignation des dimensions

Ce processus produit un SID (Schéma Intermédiaire des Dimensions) dont un exemple partiel est présenté dans la Figure 24 : le décideur a choisi les dimensions Produit et DateTransac.

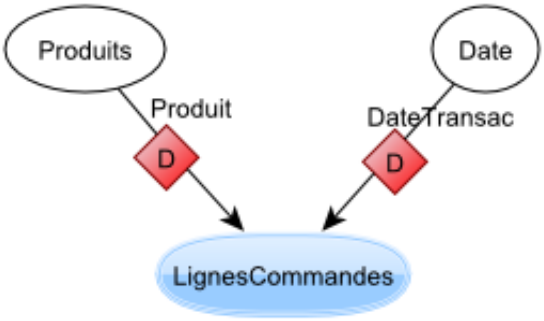

Figure 24. Un schéma intermédiaire des Dimensions (SID)

4.5.5. Les hiérarchies

La phase 3 consiste à générer les hiérarchies candidates pour chaque dimension choisie ; ceci est effectué à partir du SID obtenu dans la phase 2 et du Schéma Exploitable comme le montre la Figure 25.

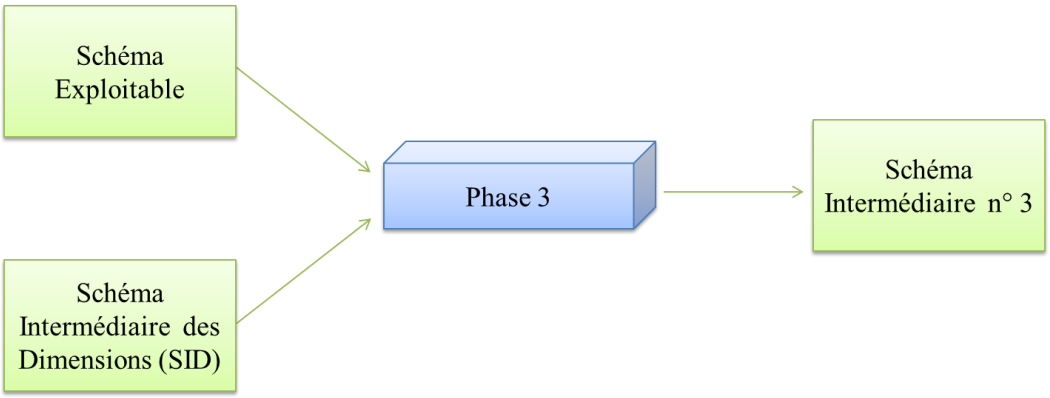

Figure 25. La phase 3

Une hiérarchie représente une possibilité d'analyse précisant les niveaux de granularités (paramètres) auxquels peuvent être manipulés les mesures. Ces paramètres sont organisés de la granularité la plus fine à la granularité la plus générale. Différents types de hiérarchies existent : hiérarchie stricte, hiérarchie couvrante, hiérarchie multiple. Dans la mesure où le décideur doit appréhender la sémantique associée aux hiérarchies afin de choisir les paramètres puis les manipuler, nous considérons uniquement les hiérarchies strictes (Malinowski and Zimányi, 2006).

La phase 2 a généré un SID de la forme (F, D) ; notons $d_i \in D$ une dimension appartenant au SID. La détermination des hiérarchies d'une dimension d_i consiste à rechercher des classes dans le Schéma Exploitable comme suit.

Si la dimension d_i correspond à une classe du Schéma Exploitable $c_j \in SE$ alors toute classe $c_k \in SE$, telle que c_k est reliée transitivement à c_j par un lien 1-N, est un niveau de granularité de d_i. Une hiérarchie est définie par une suite de niveaux appelés paramètres et reliés consécutivement ; le premier niveau correspond à la dimension elle-même ; et le dernier niveau est dénommé « All ».

La Figure 26 montre le principe de détermination des paramètres d'une hiérarchie stricte en utilisant le Schéma Exploitable.

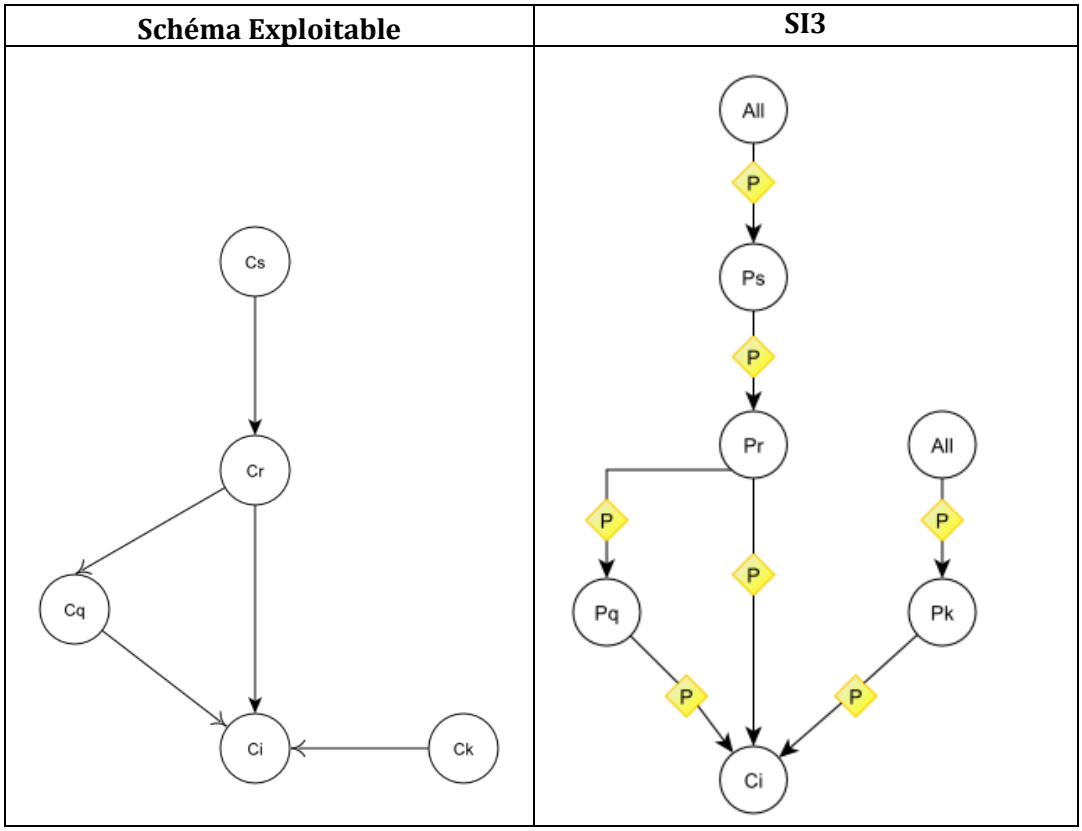

Figure 26. Le principe de détermination des hiérarchies

La dimension temporelle (Temps) est associée à un graphe enraciné prédéfini comportant les paramètres : date, semaine, mois, trimestre, semestre et année ; ce graphe comporte deux hiérarchies. Dans notre exemple, le système génère dans SI3 les hiérarchies candidates des dimensions Produits et Dates. Chaque paramètre d'une hiérarchie est associé à un ensemble d'attributs faibles pouvant être utilisés par le décideur lors de l'affichage des résultats d'analyse. Les

attributs faibles d'un paramètre sont extraits de la classe qui est associée à ce paramètre dans le schéma exploitable.

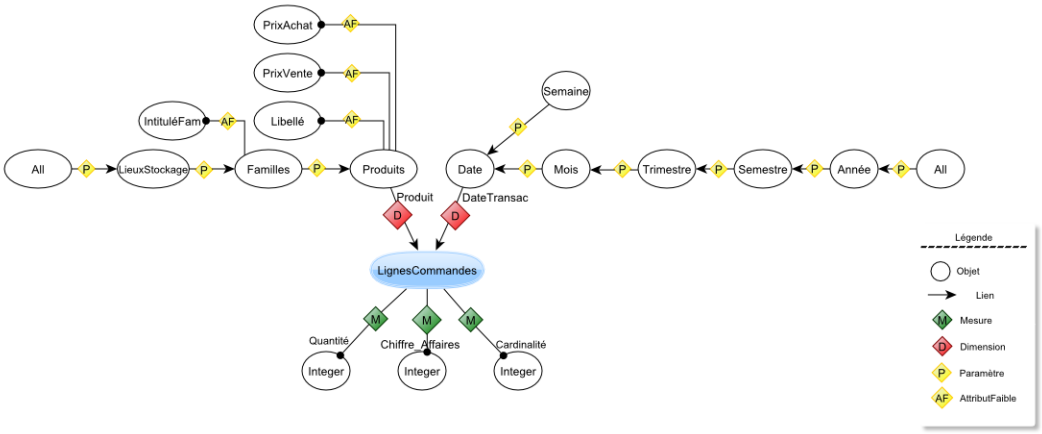

Figure 27. Un schéma multidimensionnel partiel (SI3)

La Figure 27 donne un exemple de représentation des attributs faibles dans un schéma multidimensionnel.

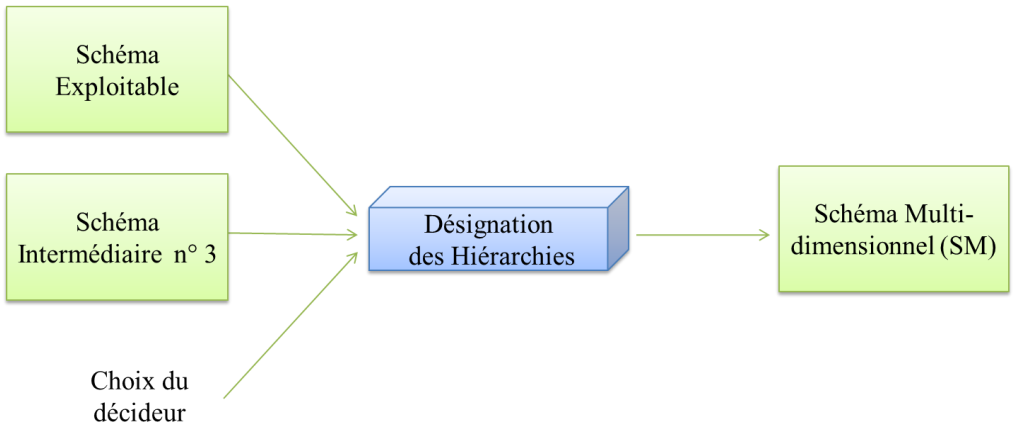

Figure 28. Le processus de détermination des hiérarchies

Le décideur peut alors désigner, dans le SI3, les paramètres utiles dans chaque hiérarchie proposée comme le montre la Figure 28.

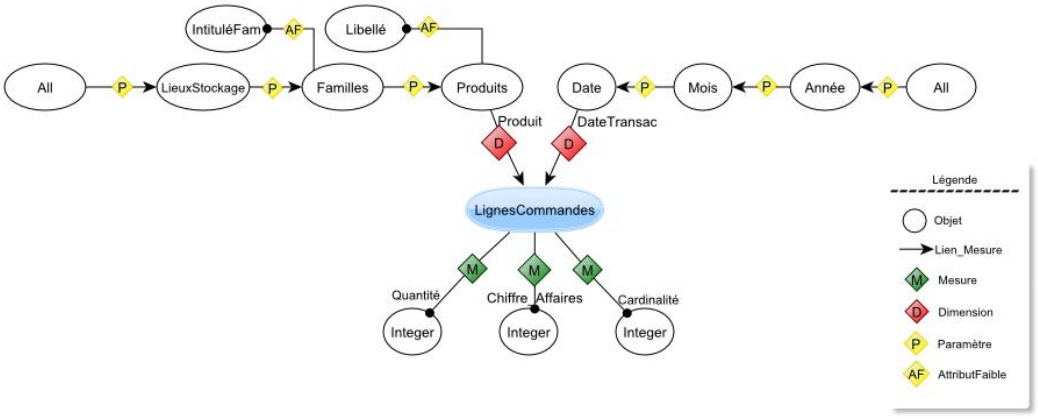

Figure 29. Le Schéma Multidimensionnel (SM)

Comme le montre la Figure 29, un schéma multidimensionnel final est produit avec un ou plusieurs faits et des dimensions associées, ce schéma traduisant les besoins d'analyse du décideur.

4.5.6. La validation du schéma multidimensionnel

Le résultat de la phase 3 correspond à la prise en compte des choix successifs du décideur en matière de faits et de dimensions. Dans la dernière phase, l'utilisateur valide l'ensemble de ces choix ; ceci entraine l'enregistrement de métadonnées de personnalisation. Ces métadonnées associent, pour chaque décideur ayant élaboré une base décisionnelle, des poids aux classes de la source. Elles seront utilisées pour assister ce même décideur lors de nouvelles constructions de schémas. Dans la section suivante, les mécanismes de personnalisation sont détaillés.

Les contributions de ce chapitre ont été présentées dans (Abdelhédi et al., 2011a), (Abdelhédi et al., 2011c) et (Abdelhédi and Zurfluh, 2013).

CHAPITRE 5

L'ENRICHISSEMENT DES MÉTADONNÉES

Dans la mesure où le système SelfStar est destiné à des utilisateurs non informaticiens, notre objectif est d'assister ces utilisateurs pour faciliter leurs activités métier. Un des problèmes auquel est confronté un décideur est la difficulté d'appréhender la complexité de la base de données source. Dans l'industrie, celle-ci peut en effet être constituée d'une cinquantaine de classes d'objets (généralement des tables relationnelles) et autant de liens.

Une des premières étapes d'élaboration d'un entrepôt consiste à extraire un ensemble de faits potentiels de la source ; or le nombre des faits proposés au décideur peut être important (Song et al. 2008). Face à cette multiplicité d'objets, le décideur est souvent démuni pour déterminer les faits qu'il souhaite analyser.

Notre objectif est de proposer aux décideurs les faits les plus pertinents afin qu'ils ne soient pas contraints de naviguer dans un grand nombre de faits potentiels.

Pour répondre à cet objectif dans SelfStar, nous proposons de mémoriser des métadonnées qui vont s'enrichir au fur et à mesure de la création de nouveaux entrepôts. Nous apportons deux solutions complémentaires à ce problème selon la situation du décideur dans le processus d'élaboration d'un entrepôt.

D'une part, lorsque le décideur élabore pour la première fois un entrepôt de données sur une source, SelfStar propose d'ordonner l'ensemble des faits potentiels à partir de données structurelles (métadonnées de base).

D'autre part, dans le cas où le décideur a déjà construit un ou plusieurs entrepôts sur une même source, SelfStar ordonne l'ensemble des faits potentiels à partir de données de personnalisation (métadonnées de personnalisation).

5.1. La justification de la personnalisation

Les études effectuées dans l'industrie par notre laboratoire (Annoni et al., 2006) ont montré, au travers d'applications décisionnelles, qu'un décideur peut faire évoluer ses analyses dans le temps. Ainsi, le décideur peut dans un premier temps analyser une source selon différents sujets d'analyse et axes d'étude ; et dans un second temps modifier ou compléter ses analyses. On constate que sujets et axes d'études sont généralement voisins : les dimensions et faits sont fréquemment interchangés. Ceci s'explique par le fait que les analyses du décideur sont liées à son métier (généralement spécialisé). Considérons par exemple un contrôleur appartenant à un organisme de gestion des cotisations de Sécurité Sociale ; il est chargé de vérifier des déclarations émanant des employeurs d'une région donnée. Ses analyses les plus courantes portent sur les montants de cotisation déclarés selon les effectifs de salariés, les secteurs d'activité et les types juridiques des entreprises. Suite à une

volonté politique ponctuelle du gouvernement de détecter le travail clandestin, les contrôles vont à présent porter aussi sur les effectifs salariés (nouveau sujet d'analyse) selon les secteurs d'activité et les chiffres d'affaires.

Ceci montre qu'un décideur peut analyser un entrepôt puis exprimer le besoin d'analyser un nouvel entrepôt extrait de la même source de données, ceci tout en poursuivant l'exploitation du premier. Ainsi dans le cadre de son métier, le décideur fait évoluer ses analyses. A partir de ce constat, nous proposons un mécanisme de personnalisation qui assiste un décideur (ou une classe de décideurs) dans la construction d'entrepôts successifs à partir d'une même source de données. Le décideur dispose alors de deux solutions :

- Soit modifier le schéma d'un entrepôt qu'il a précédemment élaboré (solution non implantée dans notre prototype),

- Soit construire un nouvel entrepôt en étant guidé par le logiciel SelfStar.

5.2. Les métadonnées de base

Les processus et algorithmes proposés dans cette section concernent la toute première élaboration d'un entrepôt sur une source.

Pour assister et guider le décideur dans la construction d'un schéma multidimensionnel, le système SelfStar alimente et utilise une métabase. Faits et dimensions sont extraits du schéma de la source de données selon un algorithme utilisé dans les démarches ascendantes. Dans la phase1, SelfStar propose au décideur un ensemble de faits candidats qu'il présente dans le SI1 (Schéma Intermédiaire n°1). Ces faits correspondent à des classes d'objets extraites du schéma de la source. Dans un second temps, le décideur désignera le ou les faits à analyser parmi ces faits candidats.

Une source contenant un nombre important de classes, SelfStar peut générer un grand nombre de faits candidats (Song et al. 2008). Pour faciliter le choix du décideur, SelfStar ordonne les faits candidats selon leur faculté à être analysés.

Lors de la première élaboration d'un entrepôt sur une source, SelfStar ne dispose pas de métadonnées de personnalisation. Le tri des faits candidats repose donc sur des métadonnées structurelles obtenues après examen du schéma de la source.

Nous allons présenter les principes et les mécanismes utilisés par SelfStar pour ordonner les faits candidats.

5.2.1. La relation d'ordre

Dans (Song et al. 2008), chaque classe d'un schéma source Entité-Association est associée à un poids calculé en fonction du nombre de liens de cette classe avec les autres classes de la source. Un lien 1..N entre deux classes A et B, noté B↠A, indique que la classe B est une dimension potentielle pour le fait A. Par conséquent, plus le nombre de dimensions potentielles associées à une classe est important, plus cette classe est susceptible d'intéresser le décideur pour devenir un fait à analyser.

Nous reprenons ce mécanisme de comptabilisation des liens en l'appliquant à un diagramme de classes UML (DCL). Nous complétons ce mécanisme en comptabilisant également le nombre d'attributs numériques contenus dans la classe. En effet, seuls ces attributs peuvent être agrégés dans une analyse multicritère ; par conséquent, plus une classe possède d'attributs numériques, plus forte est sa probabilité d'être un fait désigné par un décideur.

Le poids de base d'une classe (notée c_i) de la source est calculé par une fonction récursive qui comptabilise les attributs et les relations de cette classe.

5.2.2. Le calcul des poids de base

Dans un schéma multidimensionnel, on distingue :

- le fait qui contient des mesures,
- les dimensions et leurs hiérarchies liées au fait ; ces dernières permettent de regrouper les données contenues dans le fait lors des analyses.

Une classe dans une source a la faculté de devenir un fait ou une dimension en fonction de ses caractéristiques : nombre d'attributs et nombre de relations.

Les attributs d'une classe

Au sein de chaque classe, SelfStar comptabilise les attributs qui présentent les caractéristiques pour devenir des mesures et ceux pouvant devenir des dimensions.

Dans un schéma multidimensionnel, toute mesure d'un fait est de type numérique et associée à une fonction d'agrégation.

Ainsi, toute classe du schéma source est un fait candidat si elle possède au moins un attribut numérique. Son poids sera d'autant plus élevé qu'elle contient un grand nombre d'attributs numériques. Pour associer un poids à une classe, SelfStar comptabilise le nombre d'attributs numériques de cette classe et multiplie ce nombre par un coefficient de pondération ; il détermine ainsi le poids brut de la classe. Il convient de noter que SelfStar comptabilise indistinctement tous les attributs numériques (attributs additifs, semi-additifs et non additifs) alors que

certains d'entre eux (non additifs) ne pourront pas devenir des mesures ; c'est notamment le cas des valeurs de l'attribut NuméroPasseport d'une classe EMPLOYES qui, bien que numériques, ne peuvent être agrégées. Nous avons considéré que la mise en place d'un filtre pour éliminer les attributs non additifs n'apporterait pas un gain significatif.

D'autre part, tout attribut non numérique (chaîne de caractères, date, booléen principalement) d'une classe de la source est une dimension potentielle si ses valeurs ne sont pas distinctes. Par exemple, l'attribut NuméroINSEE de la classe EMPLOYES contenant des numéros distincts, n'est donc pas une dimension potentielle.

Ainsi tout attribut non numérique et non associé à la contrainte « valeurs distinctes », est comptabilisé et leur nombre est multiplié par un coefficient.

Soit $c_i \in SE$ une classe du schéma exploitable.

$PoidsAtt(c_i) = (NbAn(c_i) * CoefAn) + (NbNAn(c_i) * CoefNAn)$ où :

- $PoidsAtt(c_i)$ est le poids des attributs de la classe c_n,
- $NbAn(c_i)$ est le nombre d'attributs numériques,
- CoefAn est le coefficient des attributs numériques,
- $NbNAn(c_i)$ est le nombre d'attributs non numériques,
- CoefNAn est le coefficient des attributs non numériques.

Les relations d'une classe

Dans un schéma multidimensionnel, toute relation entre une dimension et un fait est une relation de type 1..N ; c'est à dire qu'une valeur de la dimension est liée à une ou plusieurs valeurs du fait. Il en est de même de la relation entre un paramètre d'une hiérarchie et un fait.

Dans la source et abstraction faite des attributs qu'elle contient, le poids d'une classe c_i est d'autant plus élevé que le nombre de liens 1..N dans lequel c_i intervient est important. Un poids élevé signifie donc que la classe c_i peut être analysée selon différents axes d'étude. Dans (Song et al. 2008), il a été montré que les relations indirectes transitives entre c_i et les autres classes de la source permettent de constituer les hiérarchies des dimensions. SelfStar comptabilise donc le nombre de relations directes et indirectes d'une classe.

Soit $c_i \in SE$ une classe du schéma exploitable.

$PoidsRel(c_i) = PoidsAtt(c_i) + (NbLd(c_i) * CoefLd) + (NbNLd(c_i) * CoefNLD)$ où :

- $PoidsRel(c_i)$ est le poids des relations de la classe c_i
- $NbLd(c_i)$ est le nombre de liens directs émanant de c_i
- CoefLd est le coefficient des liens directs
- $NbNLd(c_i)$ est le nombre de liens indirects transitifs émanant de c_i
- CoefNLd est le coefficient des liens indirects

Le poids d'une classe correspond au cumul du poids des attributs et du poids des relations.

$Poids(c_i) = PoidsAtt(c_i) + PoidsRel(c_i)$.

5.2.3. La métabase

SelfStar calcule et enregistre les métadonnées de base (structurelles) dès qu'une base de données de production est déclarée comme une source. Ces métadonnées seront utilisées lorsqu'un décideur élabore son premier schéma multidimensionnel sur la source. La Figure 30 présente le schéma de la base contenant les métadonnées de base.

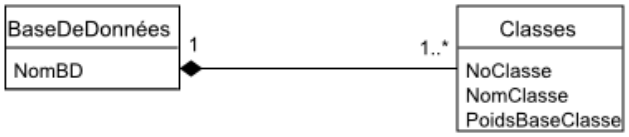

Figure 30. Les métadonnées de base

5.2.4. Les algorithmes

La production des poids associés aux classes du schéma exploitable s'effectue selon Algorithme 1 :.

Notations

Le schéma exploitable, noté SE, est en entrée de l'algorithme 1 ; Il est composé d'un ensemble de n classes et de p liens de type 1..N entre ces classes :

$SE = (\{C_1, C_2, ..., C_n\}, \{L_1, L_2, ..., L_p\})$.

Le résultat de l'algorithme est un ensemble de n poids correspondant aux n classes du SE : $P = \{P_1, P_2, ..., P_n\}$.

```
Algorithm PoidsDeBase
-- Calcul des poids de base pour les classes d'une source
Input : SE                          -- Schéma exploitable
Output : P                          -- poids des n classes du SE
Constants CoefAn, CoefNAn, CoefLd, CoefNLd   -- coefficients de pondération paramétrables
begin
 for i←1 to n do
   ANList ← nil
   NANList ← nil
   for each A_k in C_i do
    if (isnumeric(A_k) ^ ¬isdistinct(A_k))  then
      ANList ← A_k
      else  if ¬distinct(A_k) then
              NANList ← A_k
            end if
    end if
   end for
   P_i ← Length(ANList) * CoefAn
   P_i ← P_i + Length(NANList) * CoefNAn
   P_i ← P_i  + PoidsLiens (C_i,0)
 end for
end
```

Algorithme 1 : Calcul des poids de base pour chaque classe du SE

La fonction isnumeric(A) retourne vrai si l'attribut A est numérique et la fonction isdistinct(A) retourne vrai si ses valeurs sont distinctes. La fonction Liens(C) retourne l'ensemble des liens directs d'une classe C. La fonction PoidsLiens(C, P) calcule le poids P d'une classe C selon ses liens. Les variables ANList et NANList contiennent respectivement les attributs numériques et les attributs non numériques ; la fonction Length retourne la longueur d'une liste c'est-à-dire son nombre d'éléments.

```
Function PoidsLiens(C, PL) return integer
-- fonction récursive de calcul du poids PL d'une classe C selon le nombre de liens directs et indirects
begin
  if Liens(C) <> set()  then
    if (PL= 0) then
      PL ← Length(Liens(C)) *  CoefLd
      Else  PL ← PL + Length(Liens(C))  * CoefNLd
    end if
    for each  x  in  Liens(C) do
      PoidsLiens(x,PL)
    end for
  end if
end
```

Algorithme 2 : Détermination des poids des liens

5.3. Les métadonnées de personnalisation

Dans le processus d'élaboration d'un schéma multidimensionnel à partir d'une source, SelfStar propose au décideur un ensemble des faits candidats ordonnés dans le SI1. Comme nous venons de le voir, lorsque le décideur élabore son premier schéma sur cette source, SelfStar utilise les métadonnées de base calculées à partir de la structure de la source. Ensuite, pour chaque nouveau schéma élaboré par ce même décideur, SelfStar utilisera des données personnalisées qui affinent le poids de chaque classe de la source.

Dans les expériences industrielles menées par notre équipe (Annoni et al. 2006), nous avons observé, de manière empirique, le comportement de décideurs analysant régulièrement un même entrepôt. Nous avons fréquemment constaté qu'une même classe de décideurs, effectuait des analyses proches en termes de faits et de dimensions. Ceci semble lié, à l'évidence, à la nature des activités métier menées par telle ou telle classe de décideurs.

Ce constat nous a conduits à proposer au décideur, à chaque nouvelle analyse, un ensemble de faits ordonné selon des données personnalisées, c'est-à-dire des données propre à ce décideur. Les données personnalisées sont cumulées aux données structurelles présentées dans la section précédente.

5.3.1. Le principe

Un décideur analysant fréquemment une même source, utilise des faits et des dimensions présentant une forte proximité (Annoni 2007). Ainsi, dans tel schéma multidimensionnel, une classe est le fait analysé et dans tel autre, cette même classe est devenue une dimension ou un paramètre d'une hiérarchie. Ce constat nous a conduits à intégrer des mécanismes de personnalisation dans SelfStar.

Alors que les métadonnées de base sont obtenues à partir de la source, les métadonnées de personnalisation sont calculées à partir d'un schéma multidimensionnel. Le schéma multidimensionnel élaboré par le décideur est analysé par SelfStar en termes de faits, mesures, dimensions et paramètres. Les éléments issus de cette analyse sont comptabilisés, pondérés et cumulés aux poids de chaque classe. Mais, contrairement aux poids de base, ces poids sont personnalisés, c'est à dire associés à chaque décideur qui a élaboré un nouveau schéma.

5.3.2. Le calcul des poids

SelfStar analyse le schéma multidimensionnel et comptabilise les concepts mesures et dimensions. Ces concepts correspondent à des classes de la source. Ils sont dénombrés par classe, affectés d'un coefficient de pondération et cumulés aux poids des classes correspondantes. La fonction récursive suivante est utilisée.

Soit $c_i \in SE$ une classe du schéma exploitable.

$Poids(c_i) = Poids(c_i) + (NbMe(c_i) * CoefMe) + (NbDi(c_i) * CoefDi)$ où :

- $Poids(c_i)$ est le poids de la classe c dans la source,
- $NbMe(c_i)$ est le nombre de faits correspondants à c_i,
- CoefMe est le coefficient des faits,
- $NbDi(c_i)$ est le nombre de dimensions correspondantes à c_i ou à ses attributs,
- CoefDi est le coefficient des dimensions.

5.3.3. La métabase

Les poids de base sont partagés par l'ensemble des décideurs qui souhaitent analyser la même source. Par contre, les poids de personnalisation sont associés aux décideurs (les auteurs de schémas multidimensionnels). La Figure 31 présente le schéma de la base permettant d'enregistrer à la fois les métadonnées de base et les métadonnées de personnalisation.

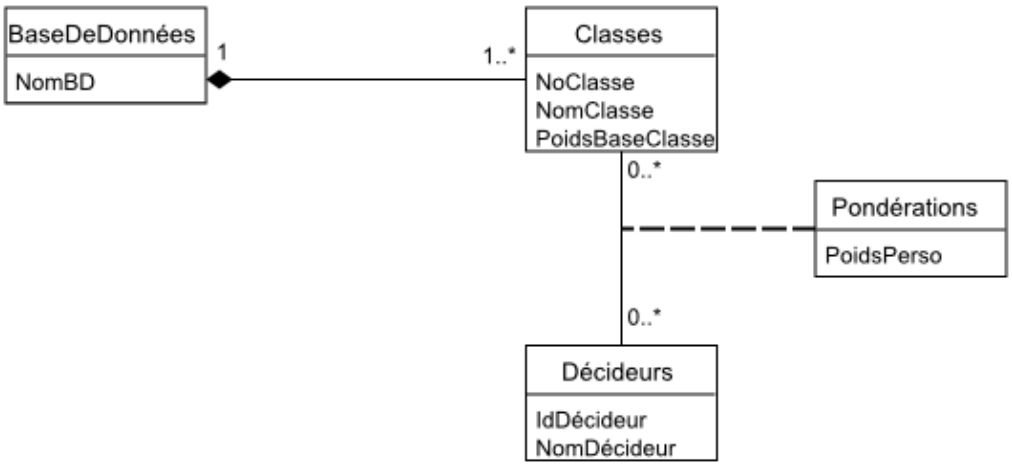

Figure 31. Le schéma de la métabase

5.3.4. Les algorithmes

Chaque fois qu'un décideur élabore un nouveau schéma multidimensionnel, l'algorithme suivant analyse ce schéma et génère des poids. Ces poids sont propres à chaque auteur du schéma (décideur) et viennent s'ajouter aux poids associés aux classes de la source. Pour chaque fait, le poids de la classe-source correspondante est majoré par les coefficients CoefMe et CoefDi (respectivement coefficient de mesures et de dimensions). Ces coefficients traduisent généralement l'importance supérieure que l'on accorde à un fait par rapport à une dimension. Leurs valeurs peuvent être modifiées au cours du temps pour affiner les résultats.

Notations

Le schéma multidimensionnel, noté SM, se définit par un ensemble de p faits et un ensemble de q dimensions liées aux faits ; les hiérarchies ne sont pas prises en compte :

$$SM = (\{F_1, F_2, ..., F_p\}, \{D_1, D_2, ..., D_q\})$$

Les poids personnalisés, notés P, correspondent aux métadonnées mises à jour par l'algorithme ; il s'agit d'un ensemble de triplets (U, C_i, P) avec P le poids associé à la classe-source Ci de SE et à l'identificateur d'un décideur U.

La fonction Liens(Classe) retourne l'ensemble des liens directs d'une classe. La fonction source(X) retourne l'identificateur de la classe-source dont le fait ou la dimension X est extrait.

```
Algorithm PoidsPerso
-- Calcul des poids personnalisés
Input : SE, P , SM, U          -- schéma exploitable, poids des classes, schéma
--  multidimensionnel et utilisateur
Output : P                     -- nouvelles valeurs des poids associées à un utilisateur
begin
 for i←1  to p do              -- pour chaque fait du SM
  s ←  source(Fi )
  Ps ←   Ps + NbMe x CoefMe    -- majore le poids de la classe-source
  for  x  in Dim(Fi) do        -- pour chaque dimension associée au fait
   s   ← source (x)
   Ps ←  Ps  +  CoefDi         -- majore le poids de la classe-source
  end for
 end for
end
```

Algorithme 3 : Le calcul des poids personnalisés

L'Algorithme 3 :produit la variable P contenant un ensemble de couples (poids, utilisateur) ; ces couples sont associés à chacune des classes de la source qui ont permis de générer les faits et dimensions.

5.4. Mise en œuvre

Dans cette section, nous présentons un exemple d'élaboration des métadonnées de base ainsi que de métadonnées de personnalisation. Cet exemple est basé sur le schéma d'une source présenté Figure 9 et du schéma multidimensionnel de la Figure 33.

5.4.1. Calcul des métadonnées de base

Ce calcul est déclenché chaque fois qu'une nouvelle source est introduite dans SelfStar. Il s'agit de calculer le poids de chaque classe appartenant au schéma exploitable. Un poids détermine la capacité de la classe à devenir un fait : plus le poids est élevé et plus la classe est un sujet d'analyse potentiel. Ces poids permettent donc d'ordonner l'ensemble des classes du DCL selon leur intérêt pour le décideur.

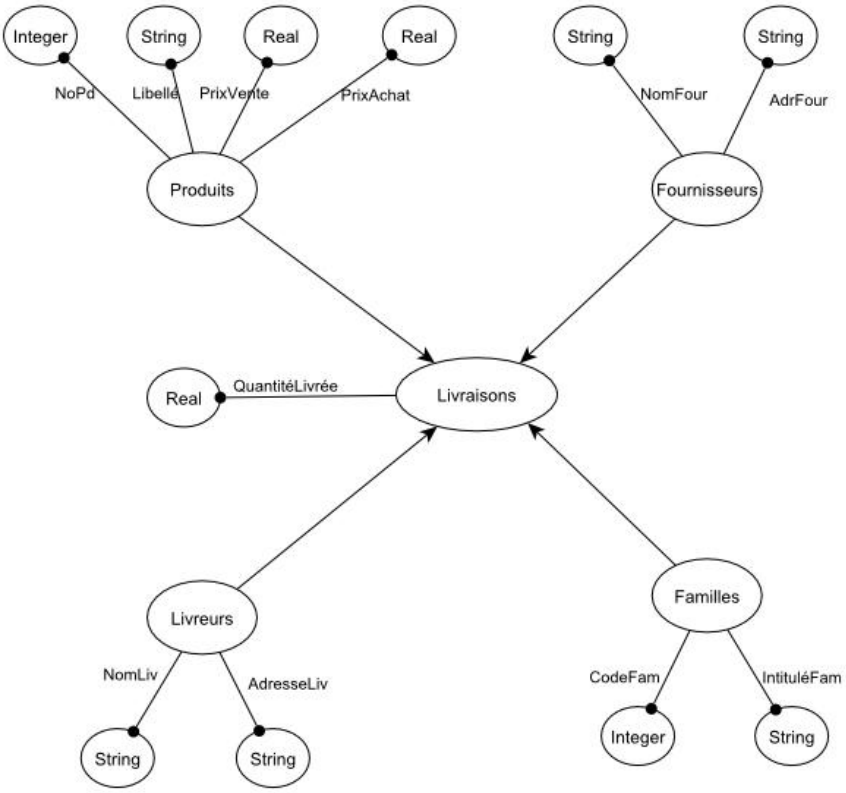

Figure 32. Le schéma exploitable (partiel) de la source « Gestion des ventes » (Figure 9)

Nous fixons les coefficients en privilégiant les attributs numériques et les liens directs (ici du simple au double). Des ajustements de ces coefficients peuvent être réalisés après expérimentation afin d'obtenir un ordre des classes plus approprié.

Les coefficients sont fixés comme suit :

- CoefAn et CoefLd = 2,
- CoefNAn et CoefNLd = 1.

Nous appliquons l' Algorithme 1 :pour calculer les métadonnées de base ; nous nous limitons ici aux classes Livraisons, Produits, et Temps du schéma exploitable de la Figure 11.

Classe Livraisons :

Prise en compte des attributs :

Length(ANList) x CoefAn + Length(NANList) x CoefNAn

1 x 2 + 0 x 1 = 2

Prise en compte des relations :

(NbLd(Livraisons)*CoefLd)+(NbNLd(Livraisons)*CoefNLd)

4 x 2 + 2 x 1 = 10.

Le poids de la classe Livraison est ainsi obtenu :

Poids (Livraisons) = 2 + 10 = 12.

De la même façon, on obtient les poids des classes Produits et Dates.

Classe Produits :

Attributs :

2 x 2 + 1 x 1 = 5

Relations :

1 x 2 + 1 x 1 = 3

Poids (Produits) = 5 + 3 = 8

Classe Temps :

Attributs :

0 x 2 + 1 x 1 = 1

Relations :

0 x 2 + 0 x 1 = 0

Poids (Temps) = 1 + 0 = 1

Le poids associé à chaque classe détermine son ordre dans l'ensemble des faits candidats.

5.4.2. Calcul des métadonnées de personnalisation

Ces métadonnées sont calculées après élaboration d'un schéma multidimensionnel sur une source et sont affectées au décideur qui a élaboré le schéma. Pour le premier schéma élaboré par un décideur sur une source, elles sont ajoutées aux

métadonnées de base ; pour les schémas suivants, elles sont cumulées aux métadonnées précédentes associées au décideur.

Considérons l'exemple du schéma multidimensionnel de la Figure 33 élaboré à partir du schéma exploitable de la Figure 32.

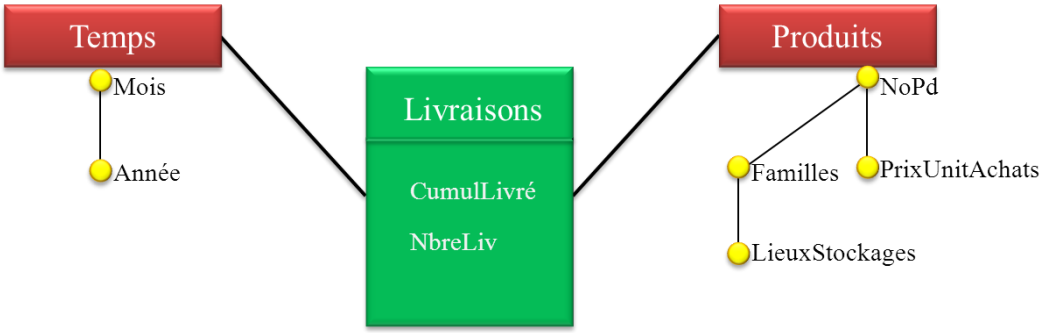

Figure 33. Schéma multidimensionnel d'une base décisionnelle

La validation de ce nouveau schéma par le décideur X va entraîner l'enregistrement des métadonnées de personnalisation associées à X. (Remarque : pour chaque auteur X, chacun des poids de personnalisation d'une classe (PoidsPerso dans la métabase de la Figure 31, est initialisé avec le poids de base de cette classe).

Les trois classes de la source Figure 32, correspondantes aux fait et dimensions du schéma en étoile, vont être enrichies des poids de personnalisation calculés ci-dessous. Les coefficients ont été fixés comme suit : CoefMe = 4 et CoefDi =2.

Poids (X, Livraisons) = Poids (X, Livraisons) + (NbMe x CoefMe)

= 12 + 2 x 4

= 20

Poids (X, Produits) = Poids (X, Produits) + CoefDi

= 8 + 2

= 10

Poids (X, Temps) = Poids (X, Temps) + CoefDi

= 1 + 2

= 3

Lors de l'élaboration d'un prochain schéma multidimensionnel sur la source « Gestion des ventes » et compte tenu de ces métadonnées, les classes du Schéma Intermédiaire n°1 (SI1) présentés au décideur X seront ordonnées comme suit :

Livraisons > Produits > Temps

Les techniques présentées dans cette section permettent d'assister un décideur qui souhaite créer un nouvel entrepôt de données. Les faits potentiels lui sont présentés selon un ordre déterminé par :

- la structure de la source (métadonnées de base),
- les faits et les dimensions utilisés dans les analyses précédentes (métadonnées de personnalisation).

Ce processus évite donc à l'utilisateur une exploration aléatoire d'une multiplicité de faits candidats pour rechercher les sujets qu'il souhaite analyser. Seuls les faits sont concernés par ce processus ; en effet les dimensions et hiérarchies, liées aux faits, sont proposées aux décideurs en nombre réduit par SelfStar.

Les contributions de ce chapitre ont été présentées dans (Abdelhédi et al., 2011b).

CHAPITRE 6

EXPÉRIMENTATION ET VALIDATION

DE NOS PROPOSITIONS

L'objectif de nos travaux est de permettre à des décideurs d'élaborer, par leurs propres moyens, des entrepôts de données à partir de sources qu'ils souhaitent analyser. Cet objectif est original dans le sens où il constitue une rupture avec le mode de fonctionnement actuel. En effet, aussi bien au niveau industriel que dans les travaux de recherche sur les systèmes décisionnels, les entrepôts et magasins de données sont conçus et créés par des informaticiens pour le compte des décideurs.

Afin de montrer la pertinence de nos propositions, nous avons développé le système SelfStar, un logiciel mettant en œuvre les principes et techniques proposés dans notre thèse.

Ce chapitre présente le logiciel SelfStar, notamment son architecture et les interfaces destinées aux décideurs. Il permet d'élaborer des entrepôts en constellation à partir de sources de données relationnelles. Dans cette présentation, nous avons volontairement pris une source différente de celle utilisée dans le début du mémoire.

6.1. L'architecture fonctionnelle

Un utilisateur du logiciel SelfStar est un décideur qui souhaite concevoir et construire un nouvel entrepôt de données. Pour ce faire, le logiciel offre trois fonctions :

- la désignation et la traduction d'une source à analyser,
- la construction incrémentale du schéma d'un entrepôt,
- le chargement de l'entrepôt.

La Figure 34 montre les trois modules de SelfStar correspondants à chacune de ces fonctions.

Le traducteur

Ce module requiert en entrée le nom d'une base de données source qui n'a jamais été exploitée par SelfStar. Pour le prototype que nous avons développé, nous nous sommes limités à des sources relationnelles ; en effet le modèle relationnel est largement utilisé dans l'industrie puisque l'on estime que 90% des bases de données de production repose sur ce modèle. Le rôle du traducteur est double :

- il transforme le schéma source en un schéma exploitable conformément aux principes énoncés dans le CHAPITRE 4 ;
- il génère les métadonnées de base à partir du schéma exploitable.

Le schéma exploitable et les métadonnées sont mémorisés dans le système ; ils seront utilisés chaque fois qu'un utilisateur souhaitera construire un nouvel entrepôt sur la source correspondante.

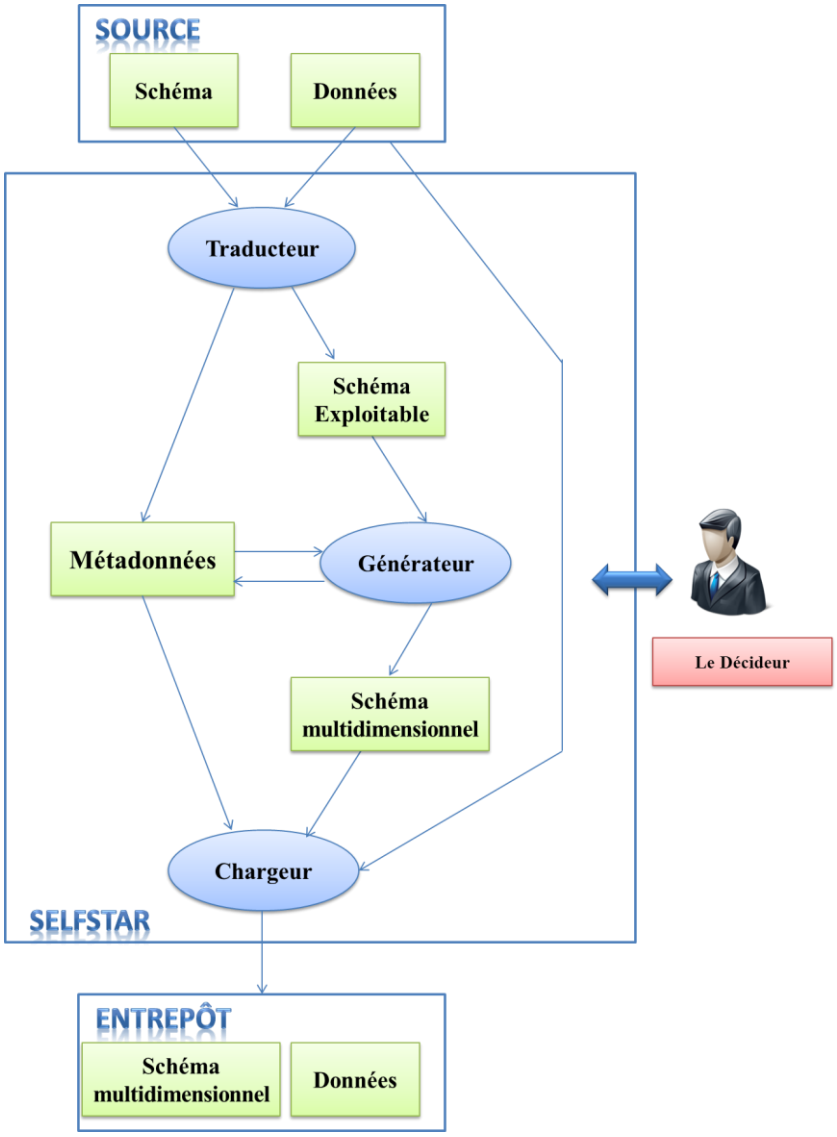

Figure 34. Architecture de SelfStar

Le chargeur

Ce module assure la fonction d'ETL, c'est à dire la création de l'entrepôt et son chargement à partir de la source. La source est relationnelle et l'entrepôt est structuré selon le modèle ROLAP (Golfarelli et al., 2002b).

Le générateur

Il s'agit du module qui élabore un nouveau schéma multidimensionnel à partir d'un schéma exploitable. Il requiert des interactions avec l'utilisateur pour déterminer successivement faits, mesures, dimensions et hiérarchies. En plus du schéma de l'entrepôt, il génère des métadonnées de personnalisation et de chargement. La Figure 35 présente la structure interne de ce module logiciel ; on y retrouve l'ensemble des fonctions étudiées dans les CHAPITRE 4 et CHAPITRE 5 du mémoire : les trois phases de spécification du schéma multidimensionnel, les interactions avec l'utilisateur et l'enrichissement des métadonnées.

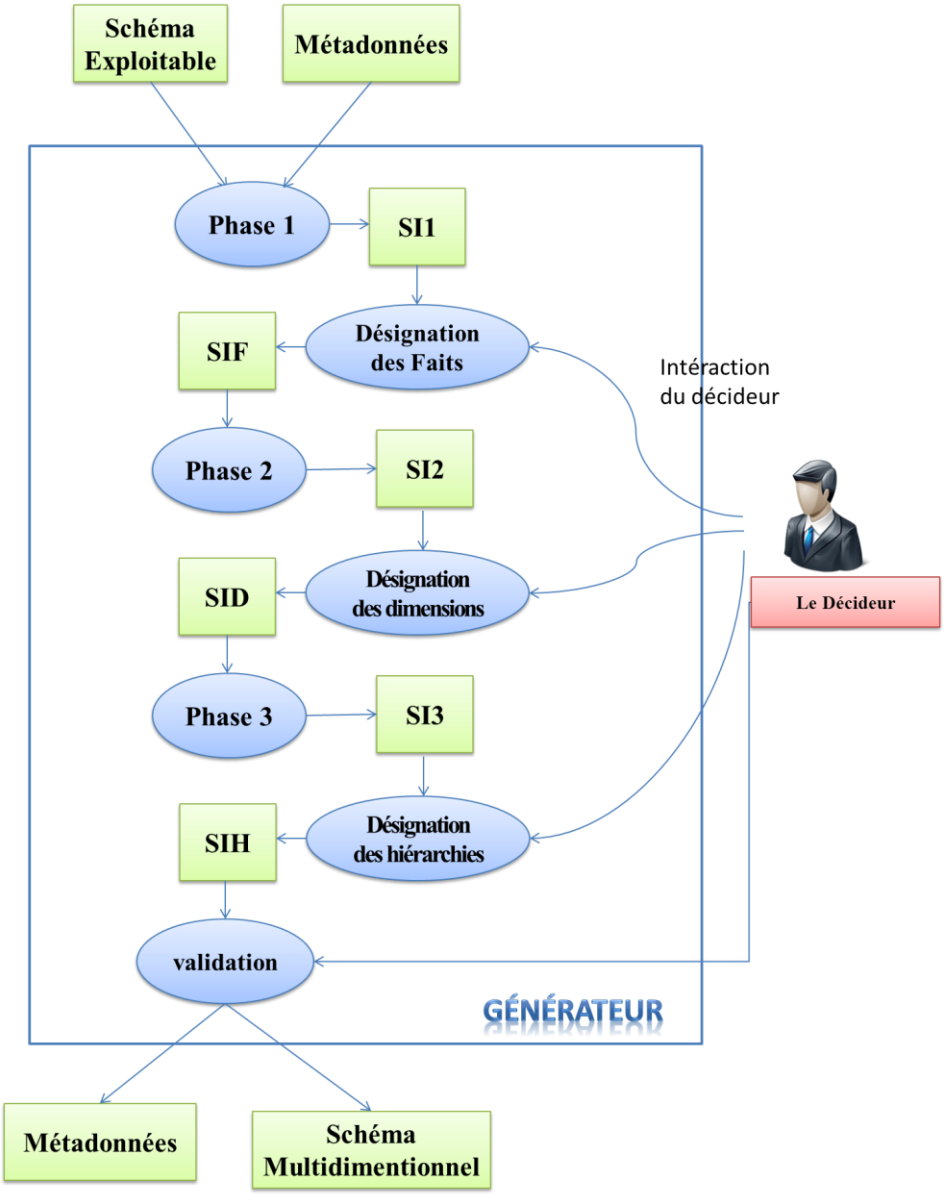

Figure 35. L'architecture du module Générateur

6.2. L'architecture technique

La Figure 36 présente les modules techniques tels qu'ils ont été mis en œuvre dans SelfStar, ainsi que leurs interactions.

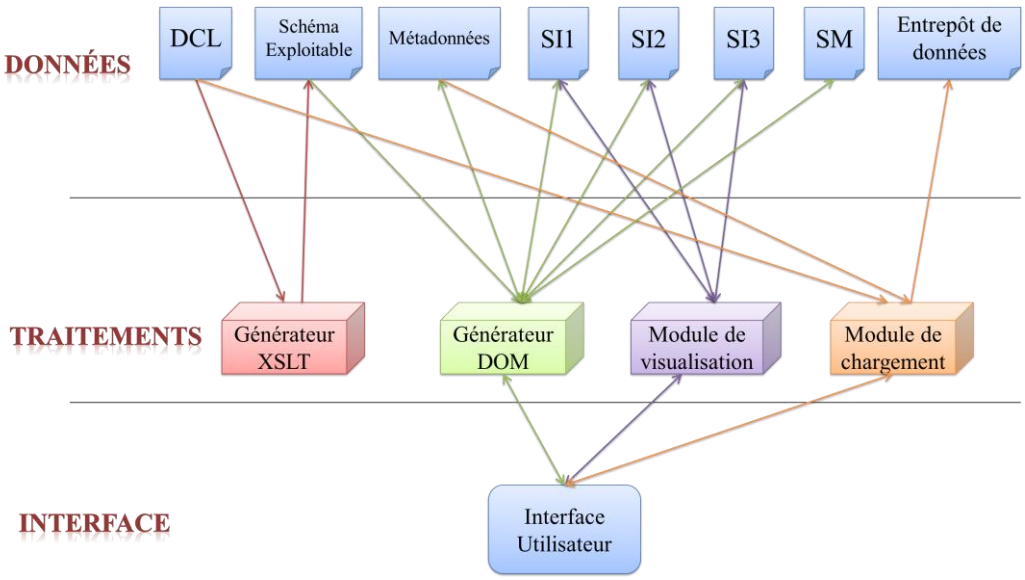

Figure 36. Architecture technique du prototype SelfStar

SelfStar a été développé en java et comporte 3 niveaux :

- Le niveau données qui correspond à l'ensemble des données en entrée et en sortie des différents modules ; toutes les données sont codées en XML pour assurer leur portabilité. Ce niveau décrit les structures de la source (DCL), celles de l'entrepôt de données, les différents schémas intermédiaires et les métadonnées. L'ensemble des métadonnées servira au module Chargeur pour alimenter l'entrepôt.
- Le niveau traitements utilise le système JAXP[1] qui est un ensemble d'API incluant les modules SAX, DOM, XSLT et XPATH. Il se décompose en quatre modules : (1) le générateur XSLT transformant le schéma source DCL en un schéma exploitable ; (2) le générateur DOM générant les métadonnées, les schémas intermédiaires, le schéma multidimensionnel ; (3) le module de visualisation et d'interaction permettant à l'utilisateur d'interagir avec le

[1]Java API for XML Processing

système via les différents schémas intermédiaires, il utilise la bibliothèque JGraph ; (4) le module de chargement qui alimente l'entrepôt depuis la source.

- Le niveau Interface qui met les différents schémas à la disposition du décideur.

6.3. Les sources de données

Pour illustrer le fonctionnement de SelfStar, nous prenons l'exemple d'une source de données utilisée par un logiciel de gestion d'étudiants. Le diagramme de classes partiel de la source est présenté dans la Figure 37.

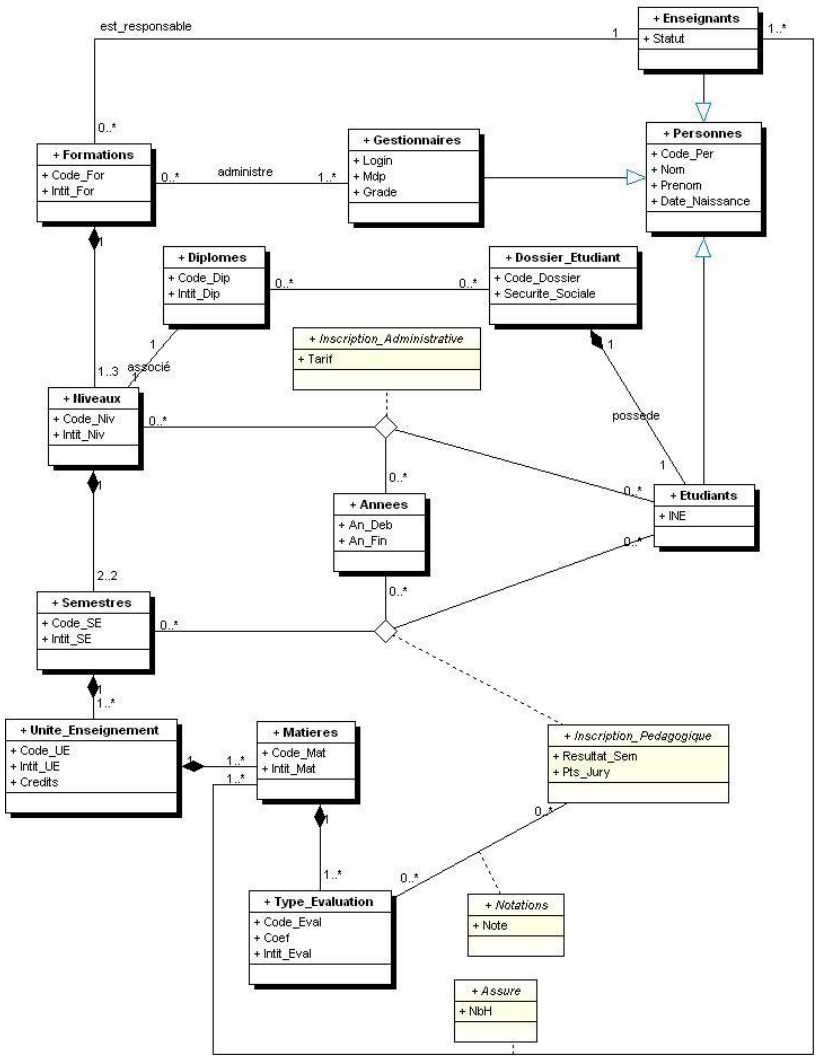

Figure 37. Un DCL pour la gestion des étudiants

Cette application gère des dossiers d'étudiants, des inscriptions à des formations et les résultats de contrôles de connaissances. Une formation est composée de niveaux regroupant des unités d'enseignements. Une unité d'enseignement, dispensée pendant un semestre, est composée d'une ou plusieurs matières et est associée à un nombre de crédits. L'étudiant obtient ces crédits lorsqu'il valide l'unité d'enseignement. Les étudiants s'inscrivent, pour une année donnée, dans un ou plusieurs niveaux de formation.

Lors de l'activation du logiciel SelfStar, l'utilisateur s'identifie et indique la source qu'il souhaite analyser. Dans la version actuelle de SelfStar, seules les sources relationnelles sont considérées. Dans le cas où la source identifiée n'a jamais été exploitée, le Traducteur transforme le schéma source en un schéma exploitable grâce à l'API XSLT (Générateur XSLT).

Le générateur DOM crée ensuite le schéma intermédiaire numéro 1 (SI1) en analysant le schéma exploitable et en utilisant des métadonnées.

6.4. Les faits

Grâce à l'utilisation des métadonnées, les faits candidats ont été ordonnés et sont présentés à l'utilisateur par ordre d'importance comme le montre la Figure 38.

Figure 38. Génération automatique du SI1

L'utilisateur peut alors désigner directement le ou les faits à analyser. Chaque fait contient l'ensemble des attributs additifs et semi-additifs extraits de la table source correspondante ; chaque attribut peut, en lui appliquant une ou plusieurs fonctions

d'agrégation, participer à l'élaboration de mesures. Considérons que l'utilisateur désigne les faits « Inscription_Administrative » et « Inscription_Pédagogique ». Lorsque l'utilisateur désigne un attribut dans un fait, un éditeur d'expression lui permet de spécifier le calcul d'une mesure comme le montre la Figure 39.

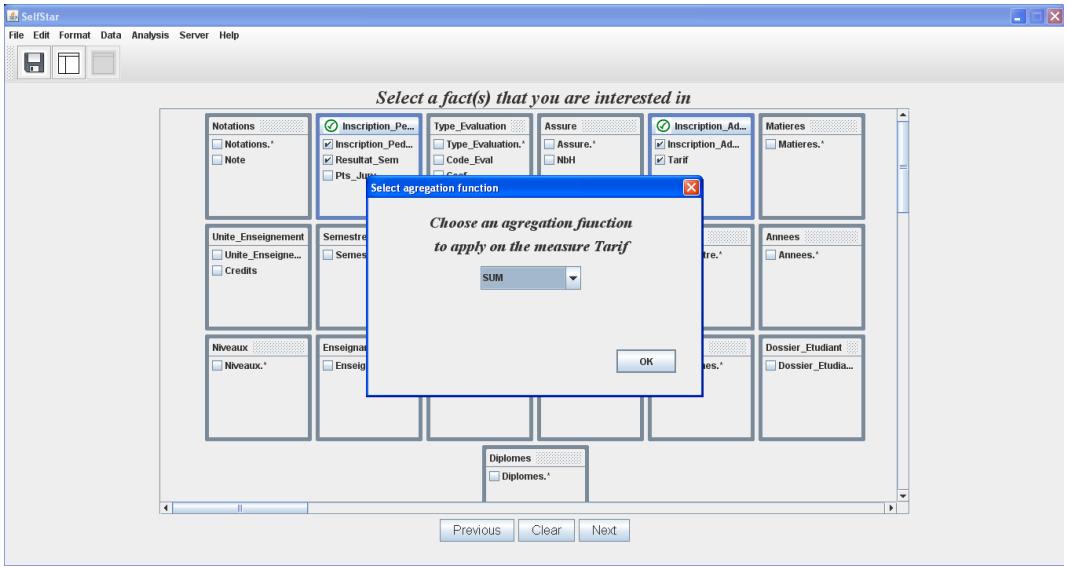

Figure 39. Choix d'une fonction d'agrégation pour la mesure RésultatAdmission

Après que l'utilisateur a choisi les faits et a défini les mesures, SelfStar va proposer les dimensions candidates dans le SI2, c'est-à-dire les axes d'étude possibles pour chaque fait choisi.

6.5. Les dimensions

Les dimensions potentielles d'un fait sont déterminées à partir de la source de données, plus précisément à partir de la classe correspondante au fait dans le schéma exploitable. Le processus de détermination des dimensions a été étudié dans le CHAPITRE 4 du mémoire. La Figure 40 présente à l'utilisateur le schéma intermédiaire n°2 (SI2) généré par SelfStar ; Le SI2 contient l'ensemble des dimensions candidates associé aux faits « Inscription_Administrative » et « Inscription_Pédagogique ».

Dans le SI2, le décideur désigne les dimensions utiles à son analyse ; supposons qu'il désigne les dimensions « Etudiants » et « Années » pour le fait « Inscription_Pédagogique » et les dimensions « Niveaux », « Etudiants » et « Année » pour le fait « Inscription_Administrative ».

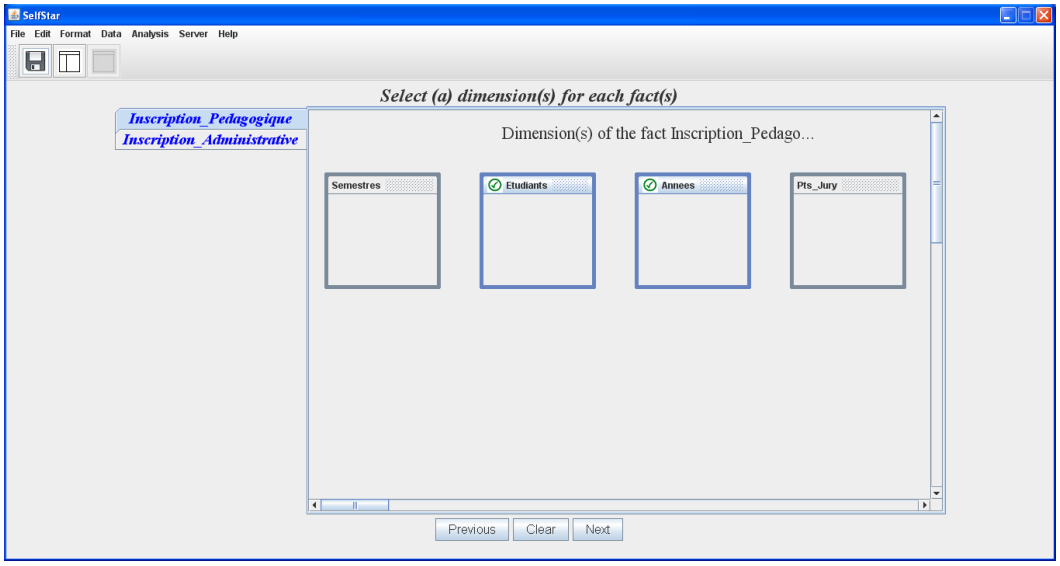

Figure 40. Génération automatique du SI2

6.6. Les hiérarchies de dimensions

Pour déterminer les hiérarchies potentielles de chaque dimension du SI2, SelfStar s'appuie de nouveau sur le schéma source. Ainsi, à partir de la classe C du schéma exploitable correspondante à une dimension, le logiciel recherche toutes les classes liées transitivement à C par un lien 1-N ; il détermine ainsi les hiérarchies de la dimension comme ceci a été présenté dans le CHAPITRE 4 du mémoire. Dans l'exemple de la Figure 41, le système génère dans le SI3 les hiérarchies potentielles pour chaque dimension.

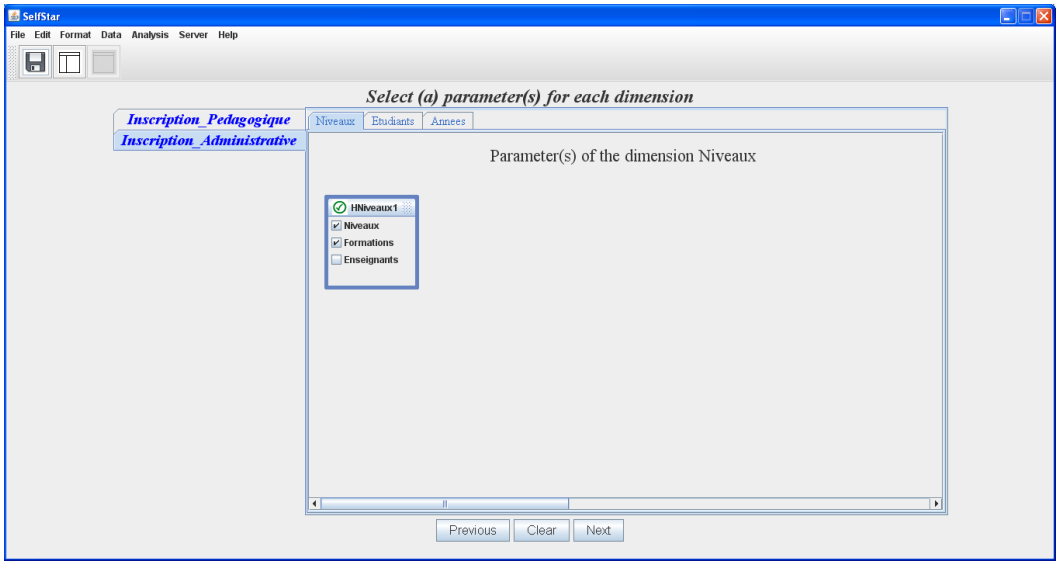

Figure 41. Génération automatique du SI3

Le décideur ayant spécifié faits, mesures, dimensions et hiérarchies, le schéma multidimensionnel final est visualisé comme le montre la Figure 42.

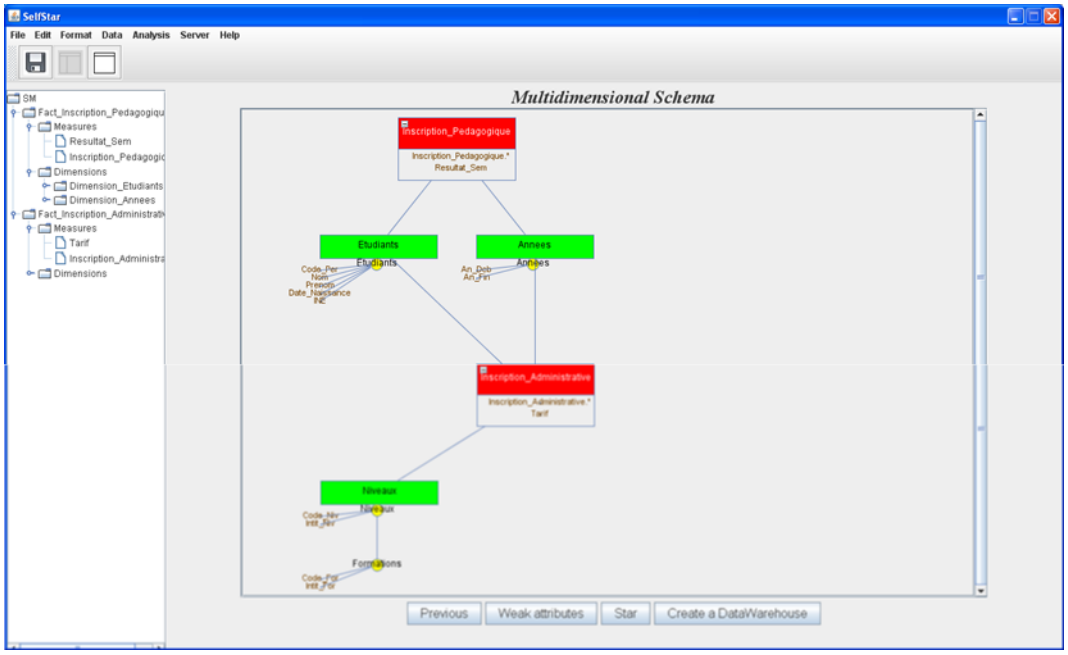

Figure 42. Génération du schéma multidimensionnel de l'entrepôt

Les métadonnées de personnalisation sont automatiquement enregistrées. Elles seront utilisées ultérieurement pour guider le décideur lors de l'élaboration de nouveaux schémas multidimensionnels.

6.7. Le chargement des données

Pendant le déroulement du processus d'élaboration du schéma (désignation des faits, mesures, dimensions er hiérarchies), SelfStar enregistre la correspondance entre le schéma de la source et le schéma de l'entrepôt dans un référentiel de données. Après que le décideur a validé son schéma multidimensionnel (Figure 43), l'entrepôt est créé puis alimenté automatiquement.

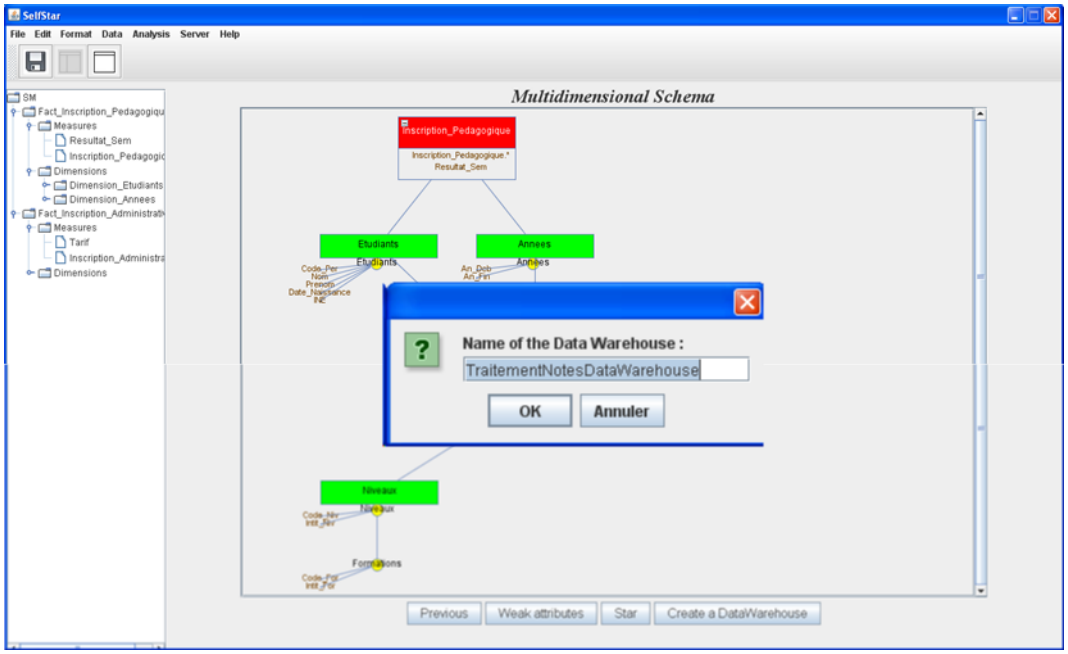

Figure 43. Création de l'entrepôt pour la gestion des inscriptions administrative

Le schéma multidimensionnel est donc traduit automatiquement en ROLAP et implanté en mémoire. SelfStar procède ensuite au chargement de données grâce aux données stockées dans le référentiel. Nous avons développé nos propres scripts dans un module intitulé « ETL » qui contient :

- Un script pour générer un schéma ROLAP dénormalisé de l'entrepôt à partir d'un schéma multidimensionnel.
- Un script pour l'alimentation de l'entrepôt consiste à extraire les données à partir des sources et à les enregistrer dans l'entrepôt. Il se base sur le référentiel dans lequel nous avons stocké les métadonnées de correspondance entre le schéma source et le schéma multidimensionnel d'une part et entre le schéma multidimensionnel et le schéma ROLAP d'autre part. Le code généré contient des requêtes SQL qui alimentent l'entrepôt automatiquement.

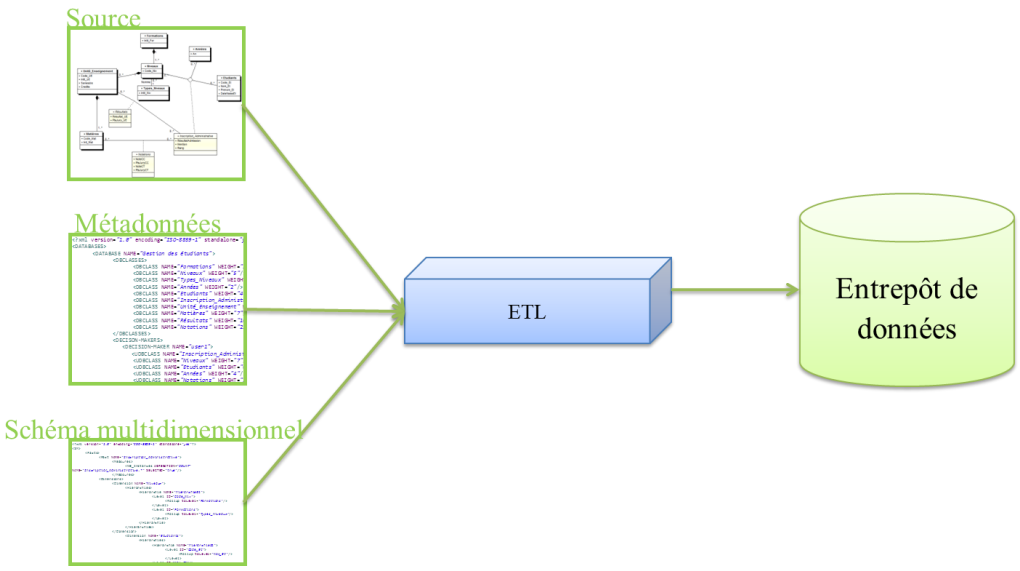

Figure 44. Le processus ETL

Ce module ETL, présenté dans la Figure 44, prend en entrée le schéma multidimensionnel, les métadonnées et la source de données et produit en sortie l'entrepôt de données.

Figure 45. Exemple d'un entrepôt généré par SelfStar

La Figure 45 montre un exemple de schéma ROLAP dénormalisé d'entrepôt de données.

6.8. L'enrichissement des métadonnées

Une source avec un nombre important de classes pourrait générer un grand nombre de faits candidats. Pour faciliter le choix du décideur, SelfStar ordonne les faits candidats.

6.8.1. Les métadonnées de base

Dès qu'une source de données est répertoriée dans SelfStar, celui-ci calcule les métadonnées en déterminant le poids de chaque classe grâce à l'algorithme présenté dans le CHAPITRE 5. Le calcul des poids s'effectue par parcours du graphe correspondant au schéma source codé en XML. Ce parcours est effectué par l'intermédiaire de l'API DOM. Les métadonnées générées sont introduites dans un fichier XML comme le montre la Figure 46.

```
<DATABASES>
      <DATABASE NAME="Gestion des étudiants">
            <DBCLASSES>
                  <DBCLASS NAME="Formations" WEIGHT="1"/>
                  <DBCLASS NAME="Niveaux" WEIGHT="5"/>
                  <DBCLASS NAME="Types_Niveaux" WEIGHT="1"/>
                  <DBCLASS NAME="Années" WEIGHT="2"/>
                  <DBCLASS NAME="Etudiants" WEIGHT="4"/>
                  <DBCLASS NAME="Inscription_Administrative" WEIGHT="13"/>
                  <DBCLASS NAME="Unité_Enseignement" WEIGHT="9"/>
                  <DBCLASS NAME="Matières" WEIGHT="7"/>
                  <DBCLASS NAME="Résultats" WEIGHT="16"/>
                  <DBCLASS NAME="Notations" WEIGHT="21"/>
            </DBCLASSES>
```

Figure 46. Un extrait des métadonnées de base pour la Gestion de formations

Nous fixons les coefficients en privilégiant les attributs numériques et les liens directs (ici du simple au double). Des ajustements de ces coefficients peuvent être réalisés après expérimentation afin d'obtenir un ordre des classes plus adapté à un décideur :

CoefAn et CoefLd = 2, CoefNAn et CoefNLd =1.

Le logiciel génère les poids de base suivants :

Poids (Inscription_Administrative) = 13

Poids (Niveaux) = 5

Poids (Etudiants) = 4

Poids (Années) = 2.

6.8.2. Les métadonnées de personnalisation

Après élaboration par le décideur, le schéma multidimensionnel est enregistré sous format XML. SelfStar calcule les métadonnées de personnalisation et les stocke dans la métabase. L'algorithme de calcul a été programmé en Java et utilise l'API DOM.

Un poids de base est affecté à une classe. Par contre, chaque poids de personnalisation dépend d'une classe et de l'auteur d'un schéma multidimensionnel comme le montre la Figure 47.

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<DATABASES>
      <DATABASE NAME="Gestion des étudiants">
            <DBCLASSES>
                  <DBCLASS NAME="Formations" WEIGHT="1"/>
                  <DBCLASS NAME="Niveaux" WEIGHT="5"/>
                  <DBCLASS NAME="Types_Niveaux" WEIGHT="1"/>
                  <DBCLASS NAME="Années" WEIGHT="2"/>
                  <DBCLASS NAME="Etudiants" WEIGHT="4"/>
                  <DBCLASS NAME="Inscription_Administrative" WEIGHT="13"/>
                  <DBCLASS NAME="Unité_Enseignement" WEIGHT="9"/>
                  <DBCLASS NAME="Matières" WEIGHT="7"/>
                  <DBCLASS NAME="Résultats" WEIGHT="16"/>
                  <DBCLASS NAME="Notations" WEIGHT="21"/>
            </DBCLASSES>
            <DECISON-MAKERS>
               <DECISION-MAKER NAME="user1">
                  <UDBCLASS NAME="Inscription_Administrative" WEIGHT="15"/>
                  <UDBCLASS NAME="Niveaux" WEIGHT="7"/>
                  <UDBCLASS NAME="Etudiants" WEIGHT="6"/>
                  <UDBCLASS NAME="Années" WEIGHT="4"/>
                  <UDBCLASS NAME="Notations" WEIGHT="25"/>
                  <UDBCLASS NAME="Matières" WEIGHT="9"/>
               </DECISION-MAKER>
               <DECISION-MAKER NAME="user2">
                  <UDBCLASS NAME="Inscription_Administrative" WEIGHT="17"/>
                  <UDBCLASS NAME="Niveaux" WEIGHT="7"/>
                  <UDBCLASS NAME="Etudiants" WEIGHT="6"/>
                  <UDBCLASS NAME="Années" WEIGHT="4"/>
               </DECISION-MAKER>
            </DECISON-MAKERS>
      </DATABASE>
```

Figure 47. Un extrait des métadonnées de personnalisation pour la Gestion de formations

6.9. Expérimentation et validation

Le prototype SelfStar que nous avons présenté dans les sections précédentes, permet à un décideur d'élaborer seul un schéma multidimensionnel. A partir de là, SelfStar crée automatiquement l'entrepôt que des décideurs peuvent utiliser avec les logiciels du marché tels que Business-Object et Oracle Warehouse.

L'utilisateur, c'est-à-dire un décideur, est seul à être impliqué tout au long du processus d'élaboration. Il s'agit d'une démarche originale puisque les autres processus équivalents, commerciaux ou issus du domaine de la recherche, font intervenir un informaticien à un moment ou à un autre (Jovanovic et al., 2012). Cette

autonomie, qui va permettre au décideur de construire son entrepôt rapidement (sans intermédiaire) et au moment qu'il juge opportun, devrait contribuer à améliorer l'efficience de la prise de décision dans les entreprises.

Nous avons présenté le logiciel SelfStar à cinq universitaires, responsables de formations au sein d'une université, ayant un statut de décideur (trois enseignants et deux administratifs). Nous leurs avons demandé ensuite de tester SelfStar pour élaborer des schémas décisionnels en vue d'analyser les résultats obtenus par des groupes d'étudiants dans leurs formations. Le logiciel commercial Business Object a été utilisé pour effectuer les analyses sur les entrepôts générés par SelfStar.

Jusqu'alors, trois de ces responsables réalisaient des analyses manuellement et deux effectuaient des analyses sommaires avec un tableur. Un des responsables connaissait les concepts utilisés dans les entrepôts (fait, dimension, hiérarchie) ; ceci a largement facilité la prise en main du logiciel et les interactions tout au long du processus.

Cette expérience a montré l'efficacité des deux principes fondamentaux sur lesquels est basé le système SelfStar : l'autonomie de l'utilisateur et l'élaboration incrémentale d'un schéma décisionnel.

La formation de ces responsables aux concepts décisionnels (faits et dimensions) n'a pas été négligeable. On peut cependant supposer que le même investissement eût été nécessaire pour former des utilisateurs qui analyseraient un entrepôt créé par un informaticien.

Le principe d'ordonnancement des faits candidats grâce aux métadonnées, n'a pas pu être validé de façon convaincante compte tenu du nombre limité de classes dans la source (18 classes).

6.10. Extension de SelfStar

L'objectif de nos travaux est de concevoir et développer des outils informatiques pour faciliter l'accès aux données à des décideurs. Le système SelfStar a apporté une contribution significative pour les entrepôts classiques dont les données sont extraites des sources Relationnelles. Or, de plus en plus, les décideurs utilisent à la fois des données structurées et des données semi-structurées issues du Web ou des réseaux sociaux.

Ce mixage de données de types différents pose de nouveaux problèmes dans les entrepôts de données. En effet, les modèles et langages utilisés actuellement en entreposage, s'avèrent inadaptés à la modélisation multidimensionnelle et à l'analyse OLAP de ces données. Par conséquent, l'extension de SelfStar pour la prise

en compte de nouveaux types de données passe nécessairement par la définition de nouveaux modèles

Les chapitres suivants sont consacrés à la spécification d'un modèle multidimensionnel et d'un langage pour analyser des documents XML entreposés.

PARTIE II

MODELISATION ET ANALYSE OLAP DE

DOCUMENTS

CHAPITRE 7 CONTEXTE ET PROBLÉMATIQUE

Dans les entreprises, la prise des décisions est effectuée à partir d'informations issues de sources diverses (Inmon and Inmon, 1996). Ces sources de données sont constituées non seulement des bases de données gérées dans l'entreprise mais aussi d'une multitude de textes, documents, dossiers et formulaires issus du Web. Le format XML est devenu un standard pour la représentation et l'échange des documents. Ainsi l'analyse des documents XML est devenue un enjeu majeur en informatique décisionnelle.

Les modèles et outils décisionnels actuels permettent d'entreposer des données principalement extraites de bases relationnelles. Les logiciels d'entreposage classiques travaillent donc sur des tableaux de valeurs atomiques (nombres, dates, chaînes de caractères, …) mais n'intègrent pas les documents XML pour effectuer des analyses OLAP. Il convient cependant de noter que certains logiciels commerciaux (Inmon and Inmon, 1996) permettent de stocker les documents XML dans une base de données XML native. Ces bases supportent un langage d'interrogation comme XPATH ou XQuery. Mais la complexité du langage de représentation des données et du langage de requête rend leur utilisation inappropriée à des décideurs non informaticiens.

Cet article propose un nouveau modèle conceptuel pour décrire simplement un entrepôt de documents XML. Après avoir décrit le contexte et donné la justification de nos travaux dans la section 2, la section 3 présente l'état de l'art sur la modélisation des entrepôts de documents. La section 4 décrit notre démarche pour élaborer un schéma conceptuel multidimensionnel**.** La section 5 montre l'utilisation du modèle objet pour formaliser la source des documents. La section 6 décrit ensuite le processus de construction du schéma conceptuel de l'entrepôt. La section 7 présente les principales opérations de manipulation des données d'un entrepôt. La section 8 : discussion.

7.1. Contexte et problématique

Les documents auxquels nous nous intéressons sont des objets multimédias représentés selon le format XML et contenant des données peu ou pas structurées, principalement du texte. Le standard d'échange de données XML permet de représenter le contenu d'un document et de l'associer à sa structure appelée DTD ou XSchema. La structure décrit les composants d'un document sous la forme d'éléments juxtaposés ou imbriqués ; chaque élément étant caractérisé par son type de données et encadré par des balises d'identification. L'entreposage et l'analyse OLAP de tels documents représentent un élément fondamental pour l'informatique décisionnelle.

Les travaux dans le domaine de la recherche d'information (Perez et al., 2008) ont permis de développer des mécanismes permettant la recherche de documents XML

sur le contenu (mots-clés) en utilisant des techniques basées sur des thésaurus ou des ontologies. Mais ces travaux exploitent peu la structure des documents et n'ont pas recours à des langages de requêtes OLAP pour réaliser des analyses. D'autres types de travaux ont proposé des processus capables de créer un entrepôt à partir d'une source de documents XML. Généralement, ces processus visent à élaborer un entrepôt de données classiques en offrant des techniques d'analyse ROLAP ; mais les possibilités d'analyses des documents s'en trouvent limitées.

Un exemple qui illustre bien notre problématique est celle d'une bibliothèque d'articles scientifiques produits dans un laboratoire et accessibles via le Web. Ces articles sont des objets « orientés documents », disponibles sur le Web et stockés dans le format XML. Ils sont accessibles à des publics de toutes origines (étudiants, enseignants, chercheurs, …) mais aussi aux décideurs du laboratoire. Ceux-ci interrogent les articles pour effectuer des analyses, notamment sur la typologie des publications, les référencements, les indicateurs portant sur les auteurs, etc.

Actuellement, on constate que les laboratoires suivent généralement leur propre stratégie en matière de gestion de bibliothèques d'articles de recherche. Prenons le cas du laboratoire Lamda où chaque article est stocké dans un fichier au format PDF. On demande ensuite aux auteurs de saisir et d'enregistrer dans une base de données relationnelle des caractéristiques prédéfinies de l'article : titre, coordonnées des auteurs, mots-clés, nom de la conférence, date de publication, etc. Ces descripteurs, extraits par les auteurs du contenu des articles, permettront des recherches ou des analyses ultérieures. On voit bien dans cet exemple la lourdeur du processus liée à la saisie des descripteurs (qui peut cependant être partiellement automatisée) ainsi que les limites de l'analyse réduite aux seuls descripteurs prédéfinis.

Nos travaux de recherche s'inscrivent dans ce contexte et traitent l'entreposage des documents XML. Ils doivent permettre à des décideurs d'analyser des collections de documents sans avoir à prédéfinir les modes d'accès et les critères d'analyse qui pourront être mis en œuvre.

Si nous reprenons l'exemple de la bibliothèque ci-dessus, les auteurs auront uniquement à stocker leurs articles dans le format XML ; l'analyse OLAP par les décideurs pourra ensuite se dérouler sans être précédée d'une intervention humaine visant à préparer l'analyse. D'autre part, nous faisons l'hypothèse que la multiplicité des collections de documents sur le Web et l'évolutivité des besoins des décideurs sont une justification de la recherche de nouveaux modes d'analyse des documents, plus simples et plus performants que ceux qui existent aujourd'hui. Ainsi, nous considérons que les décideurs doivent être autonomes pour élaborer un entrepôt (Abdelhédi and Zurfluh, 2013) ceci induit que l'on propose des mécanismes accessibles à des non informaticiens.

Les documents XML que nous étudions sont multimédias dans le sens où ils peuvent contenir à la fois des données structurées (titre, noms des auteurs, mots-clés, etc.), du texte ainsi que des figures et illustrations (graphiques et images). Mais il est à noter que le processus d'élaboration d'entrepôts de documents que nous proposons, fait appel principalement aux données structurées contenues dans ces documents pour jouer le rôle de fait et dimensions. Les données de type texte, image, son, ne sont pas prises en compte comme c'est le cas dans (Tournier, 2007) pour le texte.

7.2. Travaux associés

7.2.1. Modélisation multidimensionnelle des documents XML

Dès les années 2000, des chercheurs se sont intéressés à la modélisation des entrepôts de documents XML et ont apporté des contributions significatives.

Dans (Golfarelli et al., 2001), les auteurs définissent une démarche pour élaborer un schéma multidimensionnel classique à partir d'une source de documents XML qui partagent la même DTD. Cette démarche semi-automatique se compose de 4 étapes : (1) simplification de la DTD, (2) création d'un graphe de la structure, (3) choix des faits par le concepteur et (4) dérivation des mesures et dimensions. En suivant ce même principe, les auteurs de (Vrdoljak et al., 2006) proposent de travailler, non pas sur des DTD, mais sur des XSchemas qui apportent une sémantique plus riche. Bien que ces travaux portent sur des documents XML, ils visent à élaborer un schéma en étoile classique (Kimball et al., 2002) destinés à des documents « orientés données ». Le schéma est donc identique à celui qui serait construit pour une source relationnelle.

L'article de (Li and An, 2005) propose de représenter des sources XML « orientées données » par un diagramme de classes UML puis, comme dans les travaux précédents, de convertir manuellement ce diagramme en un schéma multidimensionnel classique.

Les travaux de (Tseng and Chou, 2006) portent sur la construction d'entrepôt de documents « orientés documents ». Ils proposent d'élaborer un schéma multidimensionnel en étoile classique (cf. Figure 48). Le fait représente les documents à analyser et ses mesures sont limitées à des comptages d'occurrences. Trois types de dimensions sont considérés. Les dimensions ordinaires correspondent aux caractéristiques d'indexation des documents. Les dimensions méta-données représentent des critères extraits du modèle de descriptions de documents Dublin Core (Dublin Core, 2012). Enfin les dimensions catégorie sont définies par les utilisateurs selon leurs points de vue pour l'accès aux documents. Dans ces travaux, les mesures et les dimensions ne sont pas directement extraites de

la structure des documents. Comme le montre la Figure 48, la seule agrégation possible est d'appliquer un comptage simple.

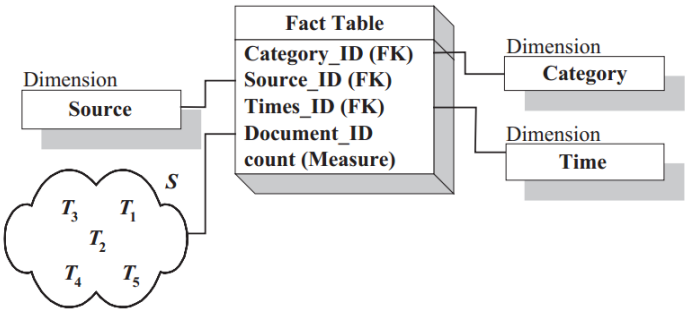

Figure 48. Un schéma en étoile d'un entrepôt de journaux scientifiques

Dans (Boussaid et al., 2008) ainsi que (Boukraâ et al., 2013), les auteurs proposent un modèle multidimensionnel décrit par un méta-modèle formalisé en langage UML. Les faits et les dimensions ne sont pas spécifiés au départ ; ils sont définis sur le schéma en appliquant un opérateur de « projection cubique » lors des interrogations. Ce schéma est ensuite traduit en un schéma logique XML et stocké soit dans une base de données native XML, soit dans une base de données relationnelle via un mécanisme de correspondance. Un ensemble d'opérations basées sur la fouille de données ont été définies.

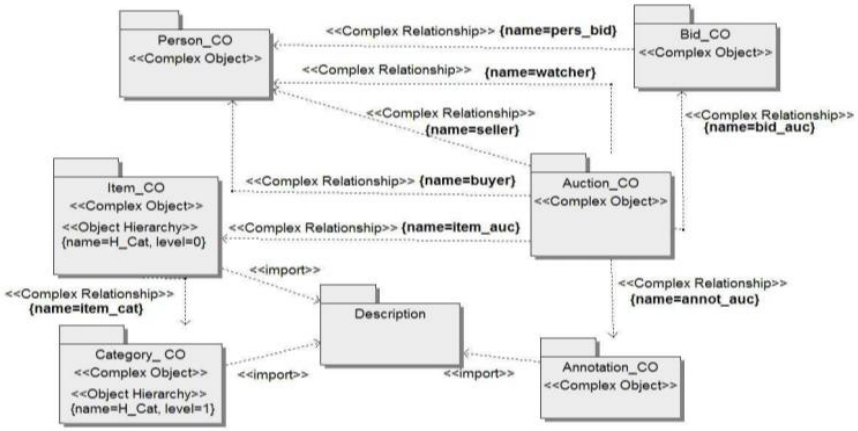

Figure 49. Modèle multidimensionnel des objets complexes (Boukraâ et al., 2013)

Les articles de (Nassis et al., 2004) et proposent d'utiliser le modèle objet associé au formalisme UML pour décrire un schéma conceptuel multidimensionnel nommé XFact. Le fait est décrit par un schéma UML distinct qui est obtenu après analyse du

contexte d'étude. Ce schéma contient les descriptions de toutes les données susceptibles d'être analysées (sans se limiter aux attributs agrégatifs). D'autre part, les dimensions sont élaborées à partir des besoins d'analyse des décideurs et validées en les confrontant à la description du fait. Ces schémas de dimensions peuvent être regroupés en paquetages lorsqu'ils sont liés et forment ainsi une hiérarchie de paquetages. Ces dimensions sont représentées par des vues XML virtuelles. Ces travaux utilisent UML pour formaliser fait et dimensions mais ils reposent sur une démarche descendante pour concevoir un schéma multidimensionnel ; ceci les éloigne de notre contexte d'étude dont le point de départ est une collection de documents à analyser. Les auteurs ne présentent pas les possibilités d'analyse de ce modèle. Vue l'absence de la matérialisation de l'entrepôt, les requêtes multidimensionnelles doivent interroger directement sur les sources ; ceci peut mener à des traitements un peu complexes.

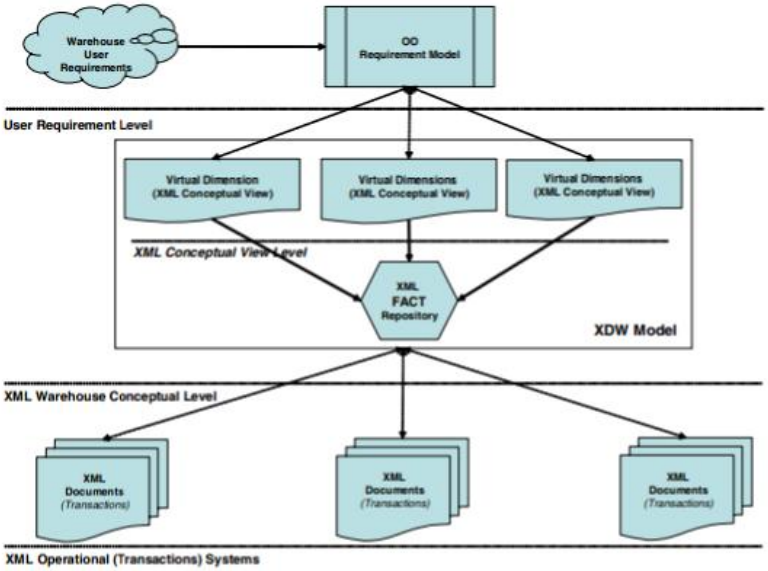

Figure 50. Modèle de l'entrepôt de documents XML (Nassis et al., 2004)

Dans (Pujolle et al., 2011), les auteurs proposent un nouveau modèle multidimensionnel pour l'analyse des documents XML « orientés documents ». Ce modèle dit « en galaxie » permet d'élaborer un schéma sous la forme d'un graphe qui utilise l'unique concept de « dimension ». Les dimensions de la galaxie sont liées entre elles par un ou plusieurs nœuds exprimant la compatibilité des concepts et c'est au moment de l'interrogation de l'entrepôt que les faits sont désignés parmi les dimensions. Ainsi, une dimension dans un schéma en galaxie représente à la fois un axe et un sujet d'analyse. Le schéma en galaxie est élaborés selon une démarche ascendante à partir d'une source XML ; les documents sont déstructurés, c'est-à-dire décomposés en dimensions (distinctes et liées par des liens d'usage) afin de faciliter

les analyses. Mais ceci induit la disparition des liens sémantiques entre les dimensions et tous les types d'analyse ne sont pas permis : par exemple, il n'est pas possible d'effectuer une agrégation sur les auteurs d'articles et une agrégation sur les auteurs cités en bibliographie.

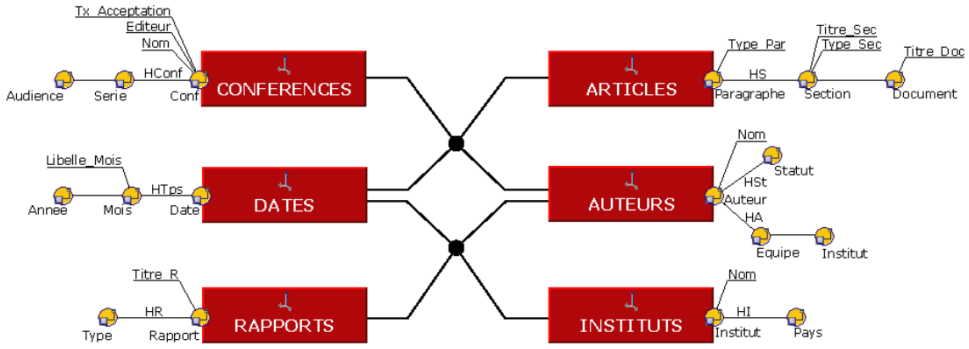

Figure 51. Modèle en Galaxie (Tournier, 2007)

Les travaux de recherche, que nous venons de présenter dans cet état de l'art, apportent une contribution significative à la problématique de la modélisation des entrepôts de documents XML. Mais les solutions qu'ils proposent ne permettent pas à des décideurs, par essence non informaticiens, d'appréhender aisément le schéma de l'entrepôt et de le mettre en relation avec la structure des documents qu'ils souhaitent analyser.

7.2.2. Analyse OLAP des documents XML

Des travaux de recherche ont été menés pour spécifier de nouveaux langages capables d'analyser des entrepôts. Nous présentons dans cette section les travaux marquants concernant la manipulation OLAP des entrepôts de données et de documents XML.

De nombreux travaux ont repris et étendu les opérateurs de l'algèbre relationnelle à la manipulation de cubes (Gray et al., 1997), (Agrawal et al., 1997). Des opérateurs spécifiques ont été proposés (Cabibbo and Torlone, 1998), (Abelló et al., 2003) et (Pedersen et al., 2001) et (Yin and Pedersen, 2004).

Les travaux présentés dans (Ravat et al., 2007b) ont défini un langage algébrique de manipulation OLAP complet. De nouveaux opérateurs ont été aussi définis sur les entrepôts de documents. Ainsi, dans (Ravat et al., 2007a) et (Ravat et al., 2008), les auteurs ont proposé deux fonctions d'agrégation qui s'appliquent sur le texte ; TOP_KW_k permet d'agréger un ensemble de documents en ses k termes les plus

représentatifs et AVG_KW permet de résumer un ensemble de mots-clés issus d'un vocabulaire contrôlé par un ensemble limité en termes plus généraux.

Malgré l'absence de consensus sur un modèle de données unique, la plupart des travaux s'accordent sur un noyau minimal d'opérations. On retrouve notamment les trois opérateurs suivants.

L'opérateur de restitution : il permet de spécifier des analyses multidimensionnelles en termes de faits et de dimension. L'utilisateur focalise l'analyse sur un sujet et projette les mesures sur plusieurs axes d'analyse. Les mesures projetées sont agrégées par une fonction d'agrégation.

L'opérateur de forage qui permet de naviguer sur les hiérarchies de chaque dimension, ceci afin de permettre l'analyse des mesures avec plus ou moins de précisions. Le forage vers le haut (roll-up) consiste à analyser les données en fonction d'un niveau de granularité moins détaillé. L'inverse, le forage vers le bas (drill-down) permet d'analyser les données avec un niveau plus fin.

L'opérateur de sélection qui donne la possibilité à un utilisateur de restreindre l'ensemble des données analysées. La spécification d'une « tranche de cube » (slice) consiste à exprimer une restriction sur une des données de l'un des axes d'analyse. La spécification d'un « souscube » (dice) consiste à exprimer une restriction sur les données d'un indicateur d'analyse.

Il convient de souligner la distinction entre les entrepôts de documents « orientées données » et « orientés documents ». Les premiers correspondent à des entrepôts de données issus de sources relationnelles et implantés dans un format XML ; dans ce cas les opérations OLAP classiques s'appliquent. Les seconds concernent des entrepôts contenant des données faiblement structurés : les documents XML ; les opérations OLAP doivent être redéfinies pour s'appliquer à de nouvelles structures de données.

A moyen terme, l'objectif de développement du projet SelfStar est de mixer tous types de données dans un entrepôt. Ainsi un décideur pourra faire appel à des bases de données relationnelles, à des données publiques (Open Data) issues du web, à des documents multimédia sous des formats divers (XML, PDF et autres), pour les analyser. Actuellement, le système SelfStar accepte comme source de données uniquement des bases de données structurées (relationnelles ou objet). Les bases de documents XML ont été modélisées mais n'ont non encore été implantées dans le système d'entreposage SelfStar.

CHAPITRE 8

ÉLABORATION D'UN SCHÉMA D'ENTREPÔT

8.1. Notre démarche

Notre objectif est de permettre l'analyse multidimensionnelle de documents thématiques du Web en reprenant les principes d'entreposage basés sur les faits et les dimensions. Il s'agit donc d'élaborer un entrepôt de documents décrit par un schéma multidimensionnel en étoile et accessible aux décideurs pour effectuer des opérations OLAP. Les faits et les dimensions du schéma sont définis selon une démarche mixte, en intégrant la connaissance des sources et les besoins des décideurs.

La matérialisation de l'entrepôt, c'est à dire sa création et son chargement, est effectuée après élaboration de son schéma. Le volume de la source peut être important dans la mesure où chaque document à analyser, qui mixe textes et images, peut représenter à lui seul plusieurs Giga-octets. Mais l'entrepôt ne contiendra que les éléments-clés utiles aux analyses : titres des sections, mots-clés, noms des auteurs, etc.

Notre démarche comporte les étapes suivantes :

1. Nettoyer et unifier les XSchemas afin de créer des classes de documents,
2. Pour chaque XSchema unifié obtenu, simplifier et réorganiser la structure pour obtenir un diagramme de classes UML unique (SourceCD),
3. Analyser un SourceCD pour identifier les mesures et les dimensions potentielles puis générer une version brute d'un schéma en étoile (StarCD),
4. Intégrer les besoins des décideurs dans le StarCD,
5. Créer et alimenter l'entrepôt,
6. Appliquer des requêtes d'analyse OLAP sur l'entrepôt à partir du StarCD.

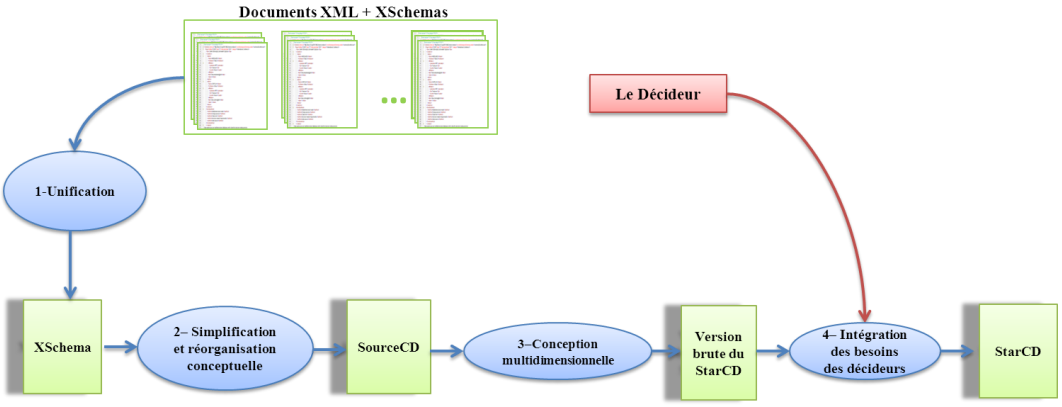

Figure 52. Les étapes d'élaboration du schéma multidimensionnel StarCD

La Figure 52 présente les étapes 1 à 4 permettant d'élaborer un schéma d'entrepôt (StarCD) à partir d'une source XML.

La source XML correspond à une collection thématique de documents que les décideurs souhaitent analyser ; nous prendrons l'exemple des articles scientifiques d'un laboratoire publiés sur le Web. On constate fréquemment que les documents du Web, même lorsqu'ils appartiennent à un même thème, ne sont pas toujours regroupés en classes dont les éléments partagent le même schéma. En effet, en reprenant l'exemple des articles de recherche, un auteur conçoit généralement un document en l'associant à un schéma unique, ceci sans se préoccuper de "réutiliser" un schéma existant. Ainsi les articles sont souvent stockés en format PDF et la transformation de ces documents en format XML génère des schémas spécifiques (autant de schémas que de documents). De nombreux travaux se sont attachés à unifier des schémas différents afin d'élaborer un nombre limité de schémas décrivant des documents dont les structures sont proches (Alqarni and Pardede, 2012) et (Yoo et al., 2005).

Le présent chapitre traite des étapes 2, 3 et 4 ; nous partons d'un schéma XML unifié décrivant une collection de documents et nous élaborons un diagramme en étoile.

Nous avons choisi d'utiliser le langage UML pour décrire le schéma de la source (SourceCD) et celui de l'entrepôt (StarCD) ; nous avons considéré que le formalisme UML, largement utilisé dans la communauté informatique, produit des schémas faciles à comprendre par des non informaticiens. Cette propriété est d'autant plus importante que les décideurs vont devoir compléter le StarCD puis formuler leurs requêtes d'analyse à partir de ce schéma.

8.2. Modélisation de la source

Un décideur, qui souhaite analyser des documents XML, désigne une source de données, généralement sous la forme d'une URL référençant un site Web.

8.2.1. Les documents XML

Un document XML est un objet multimédia contenant des constituants de types différents tels que du texte, des graphiques, des nombres, des chaînes de caractères ; ces constituants pouvant être organisés séquentiellement ou par imbrication. Pour être valide, un document XML doit être conforme à une structure (ou schéma) qui décrit ces imbrications, c'est à dire les relations entre les constituants. Cette structure arborescente est appelée DTD (Document Type Definition) ou XSchema (XML Schema). La Figure 53 présente un exemple de document XML conforme à un XSchéma.

Figure 53. Un document XML (Paper) et son XSchema

Un XSchema décrit les documents XML avec plus de précision qu'une DTD. Dans cet article, nous nous limiterons à l'usage de XSchema qui est actuellement le formalisme le plus évolué (bien que moins utilisé que les DTD). Il convient de souligner que nos travaux sur les documents ne cherchent pas à exploiter la sémantique des contenus, notamment celle du texte (Martin-Bautista et al., 2013).

8.2.2. Le schéma unifié (SourceCD)

Le présent chapitre ne porte pas sur le processus d'unification des schémas représenté par l'étape 1 de notre démarche dans la Figure 52 ; nous supposons être capable, grâce aux algorithmes existants (Janga and Davis, 2013), d'élaborer de façon automatique ou semi-automatique, un ensemble de schémas unifiés à partir d'une collection de documents. Chaque schéma unifié se présente sous la forme d'un XSchema dont la racine de l'arbre est un type document (objet de l'analyse).

Après unification, la source contient des documents regroupés en classe. Tous les documents appartenant à une même classe sont décrits par un XSchéma unique ; le type de document est situé au premier niveau du XSchéma comme le montre la Figure 53 (element name = « Paper »). Un document XML peut être défini comme un objet complexe composé d'objets atomiques ou complexes.

Définition 10. Le principe de définir un document sous la forme d'un objet n'est pas nouveau (Boukraâ et al., 2013) ; il permet de bénéficier des résultats de la recherche dans ce domaine, notamment d'une algèbre pour objets complexes. La source se définit donc comme une base de données (notée BDO) contenant un ensemble d'objets regroupés en classes. Une classe peut contenir des objets atomiques ou des objets complexes :

$$\text{BDO} = \{C_1 , C_2 , \dots , C_n\} \cup \{C_{n+1} , C_{n+2} , \dots , C_p\}$$

où $\forall C_i \in \text{BDO}$ avec $i \in [1, 2, \dots , n]$, C_i est une classe d'objets atomiques abstraite correspondant à un type de données classique : Integer, String, Date, etc

et où $\forall C_j \in \text{BDO}$ avec $j \in [n + 1, n + 2, \dots , p]$, C_j est une classe d'objets complexes.

Une classe d'objets complexes est définie sur un ensemble de classes d'objets atomiques ou/et complexes : $C_k \subseteq C_1 \times C_2 \times \dots \times C_q$ où $\forall k \in [1, 2, \dots , q]$, $C_k \in \text{BDO}$. Tout objet complexe C_k est un tuple qui est constitué de valeurs atomiques et/ou complexes.

Dans le schéma d'une classe d'objets, le rôle de chaque constituant est donné par un attribut. Soit $\mathcal{N}$ un ensemble de noms d'attributs. $\forall k \in [1, 2, \dots , q]$, $A_k \in \mathcal{N}$ est un attribut de C_k. Le schéma de C_k se définit alors comme suit :

$$C_k : \{[A_1 : C_1, A_2 : C_2, \dots , A_q : C_q]\}$$

Cette définition permet de construire des d'objets complexes par imbrication de classes existantes ; ceci correspond bien à la définition d'un document XML composé d'éléments imbriqués. En reprenant l'exemple de la bibliothèque, nous considérons ci-après la définition partielle de la collection de documents Paper conforme au XSchema de la Figure 53.

La BDO, ici limitée à une classe de documents nommée Paper, contient des classes atomiques et complexes.

$$\text{BDO} = \{\text{Integer} , \text{String}, \text{Date}, \dots\} \cup \{\text{CoA} , \text{Paper} , \dots\}$$

Le schéma de Paper utilise la définition de la classe CoA correspondante aux CoAuthors et définie comme suit :

CoA: {[Author: [FirstName: String, LastName: String, Affiliation: String,
H − Index: Integer], AffiliationGroup: String]}

Le schéma (partiel) de la classe Paper comporte les définitions de ses constituants.

Paper : {[ConfAcronym : String, ConfName : String, ConfDate : Date, Title : String, CoAuthors : CoA , KWord : <String> …]}

Pour représenter les schémas des documents XML, nous adoptons le formalisme graphique d'UML. Cette notation est largement reconnue dans la communauté des bases de données pour décrire les aspects sémantiques des objets complexes. Elle s'avère également adaptée à un usage professionnel tel que la prise de décision (Nassis et al., 2005). En effet, si l'on s'accorde sur le fait qu'un décideur n'est pas toujours capable d'élaborer un tel schéma de bout en bout, on admet généralement qu'un non informaticien puisse comprendre la signification d'un diagramme de classes UML. Un tel diagramme permet notamment de représenter des hiérarchies (relation de composition), des sous-classes (relation de généralisation) et diverses contraintes d'intégrité. Il permet aussi d'utiliser des stéréotypes pour particulariser certaines structures de données et de distinguer les concepts d'attribut et d'élément dans XML.

L'unification des XSchemas d'une source génère un ou plusieurs XSchemas (étape 1 de la Figure 52), chacun d'eux décrit une classe de documents particulière. Dans l'étape 2, tout XSchema unifié est transformé en un Diagramme de Classes UML (noté SourceCD pour Source Class Diagram) ; il se présente sous la forme d'un graphe arborescent dont la racine correspond à l'élément de plus haut niveau : la classe de documents à analyser. Tout « élément » du XSchema est transformé en une classe dans le SourceCD avec son nom et ses attributs ; les éléments imbriqués sont reliés à l'élément hiérarchiquement supérieur par des relations de composition. Les éléments associés au terme Choice sont traduits par des liens d'héritage. La Figure 54 montre la simplicité du modèle utilisé pour formaliser un XSchema.

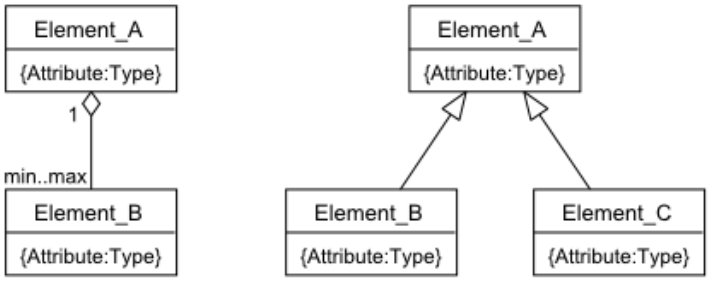

Figure 54. Le modèle

Notons que les liens Key et KeyRef entre éléments imbriqués sont traduits par des relations de référence ; ils ne sont pas utilisés dans nos travaux. La Figure 55 présente le diagramme de classes (SourceCD) décrivant une classe d'articles scientifiques et correspondant à l'extrait du XSchema de la Figure 53.

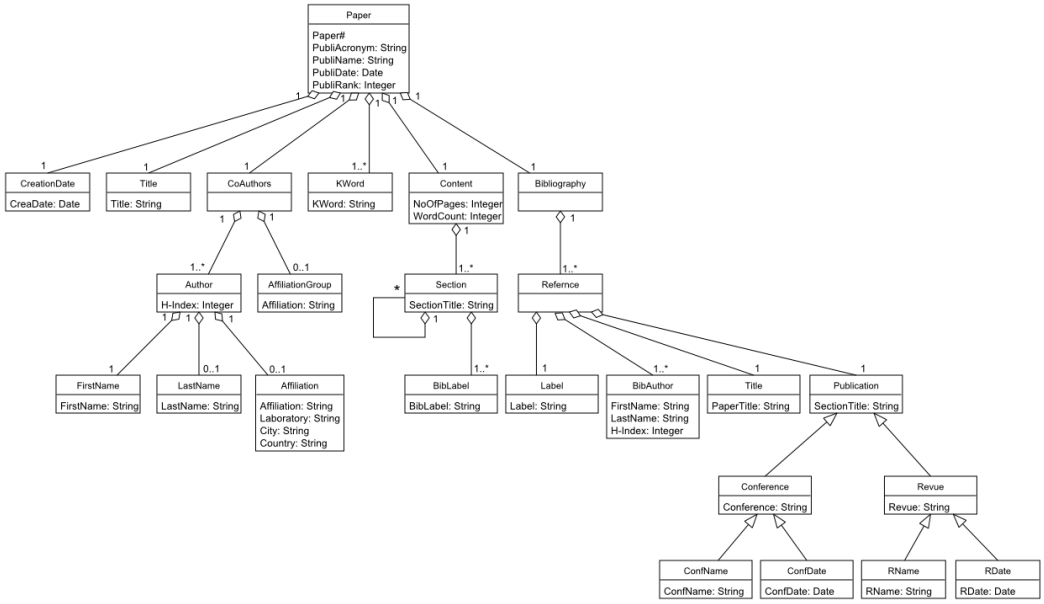

Figure 55. SourceCD décrivant des articles scientifiques

La transformation d'un XSchema en un diagramme de classe (SourceCD) a été automatisée. Elle fait correspondre les différents constituants du XSchema avec les concepts d'un diagramme de classe. La Figure 56 présente les règles de correspondance. Un XSchema et son SourceCD sont tous deux des arbres n-aires enracinés. Chaque élément du XSchema est analysé, décomposé le cas échéant et transformé en une classe dans le SourceCD.

XSchema	SourceCD
Elément (atomique ou complexe)	Classe
Lien d'imbrication entre 2 éléments	Relation de composition
Lien de spécialisation	Lien d'héritage
Lien de référence	Relation d'association
Attribut d'un élément	Attribut d'une classe

Figure 56. Correspondances entre XSchema et SourceCD

L'élaboration du schéma d'un entrepôt est un processus semi-automatique qui suit une démarche mixte : nous partons du schéma de la source pour générer automatiquement un schéma multidimensionnel brut, puis les décideurs interviennent pour préciser leurs besoins.

8.3. L'élaboration du schéma en étoile StarCD

Notre objectif est de créer un entrepôt pour permettre à des décideurs d'analyser des documents à partir d'un schéma multidimensionnel.

8.3.1. Justification de notre approche

Des travaux présentés dans l'état de l'art (Nassis et al., 2005) et (Pujolle et al., 2011) ont choisi de créer un entrepôt dont le schéma ne respecte pas la représentation initiale des documents de la source. Ainsi, à l'image des modèles de données, les éléments des documents sont positionnés au même niveau : les documents proprement dits, les auteurs qui les ont rédigés, les mots-clés qui les décrivent, etc. Ce type d'approches a l'avantage de la simplicité pour des informaticiens puisqu'il repose sur la représentation classique des bases de données.

Au contraire, notre approche repose sur le constat que les décideurs connaissent la structure des documents qu'ils souhaitent analyser. Par exemple, le responsable d'un laboratoire connaît la structure des articles scientifiques, l'ingénieur de maintenance connaît les normes internationales régissant la documentation avionique.

Les principes que nous avons retenus pour élaborer un StarCD à partir d'un SourceCD sont les suivants :

- le fait correspond à la classe racine du SourceCD,
- les mesures sont calculées à partir des attributs appartenant à la fois à la classe racine et aux classes imbriquées,
- les dimensions et leurs hiérarchies sont extraites des attributs de la classe racine ainsi que des classes imbriquées.

8.3.2. Formalisation du StarCD

Une source qui contient plusieurs types de documents, est représentée par plusieurs SourceCD. Notre processus produira un StarCD comportant plusieurs étoiles, c'est-à-dire un schéma en constellation :

$$\text{StarCD}: \{E_1, E_2, \dots, E_n\}$$

Chaque étoile E_i est représentée par un arbre n-aire où le nœud racine correspond au fait et les nœuds fils définissent les mesures et les dimensions. L'étoile E_i est donc définie par un arbre $[F_i, X_i]$ où F_i est la racine (représentant le fait) et X_i est une forêt de sous-arbres n-aires fils de F_i (représentant les mesures et les

dimensions). Le lien qui relie F_i et chacun de ses sous-arbres de la forêt appartient à l'un des deux types suivants :

- un lien de composition qui associe le fait à des mesures imbriquées dans les documents,
- un lien d'association « By » qui lie le fait à une dimension d'analyse.

Les figures 57 et 58 montrent des sous arbres n-aires représentant respectivement un exemple de mesures et un exemple de dimensions.

La Figure 57 présente un ensemble de sous arbre de mesures. Dans un sous-arbre de mesures, seules les feuilles correspondent à des mesures du fait et sont associées à une fonction d'agrégation ; les nœuds intermédiaires précisent le niveau d'imbrication d'une mesure conformément à la structure des documents (SourceCD). Une des originalités de notre modèle est de permettre la définition de mesures à partir d'attributs situés dans des éléments imbriqués dans le fait.

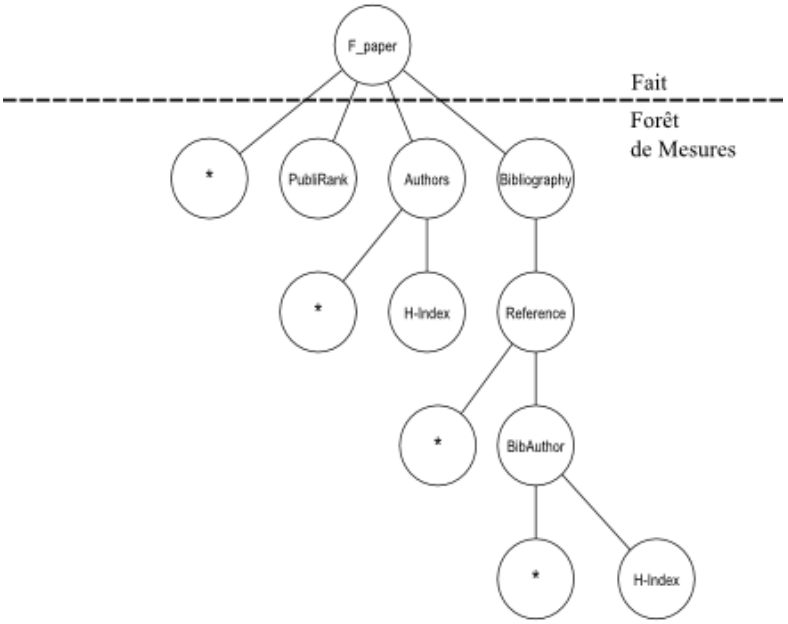

Figure 57. Des sous-arbres de mesures

La Figure 58 présente une forêt de dimensions associée au fait. Dans un sous arbre, chaque nœud correspond à un paramètre dans une hiérarchie.

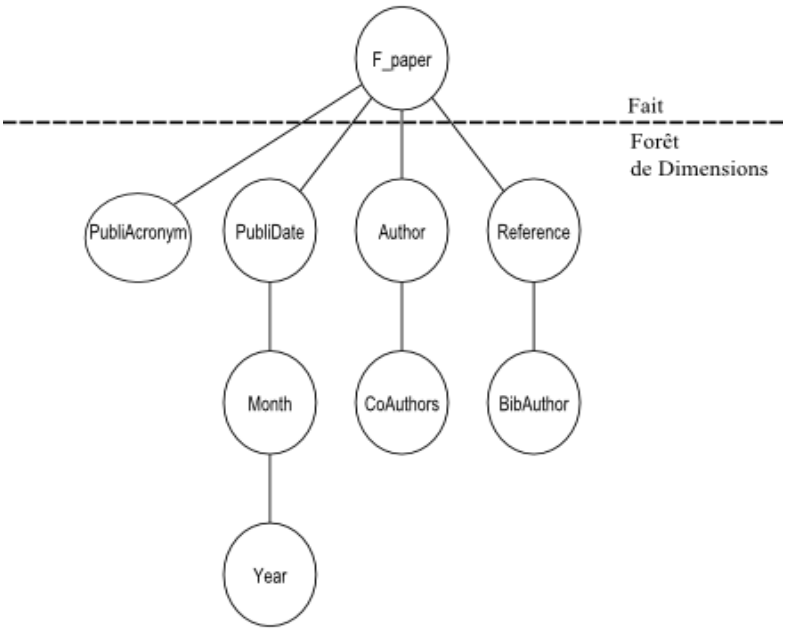

Figure 58. Des sous-arbres de dimensions

Une particularité de notre modèle consiste à reprendre l'organisation des éléments dans le SourceCD. Ceci est un aspect important de notre proposition puisque nous souhaitons préserver la structure des documents dans le schéma multidimensionnel de l'entrepôt. Les décideurs peuvent ainsi appréhender aisément les éléments du schéma de l'entrepôt à partir de leur connaissance structurelle des documents. Une conséquence de cette contrainte est de proposer des hiérarchies inversées pour respecter la structure originelle des documents ; c'est le cas de la dimension CoAuthors dans la Figure 58.

Nous utilisons le langage UML pour représenter faits et dimensions dans le StarCD. L'élaboration du StarCD est un processus semi-automatique qui extrait les faits, mesures et dimensions du SourceCD. Les décideurs interviennent dans un second temps pour préciser leurs besoins.

8.3.3. Les mesures

Chaque SourceCD décrit une classe de documents et le fait à analyser se situe à la racine de l'arborescence ; ainsi dans l'exemple de la Figure 55, le fait correspond à la classe Paper. C'est un processus algorithmique et paramétrable qui va produire un StarCD brut en faisant apparaître les attributs du SourceCD pouvant participer à des mesures. Les attributs ainsi sélectionnés sont additifs ou semi-additifs et sont associés explicitement à une ou plusieurs fonctions d'agrégation. A partir du StarCD brut, les décideurs élaboreront les mesures souhaitées sous forme d'expression.

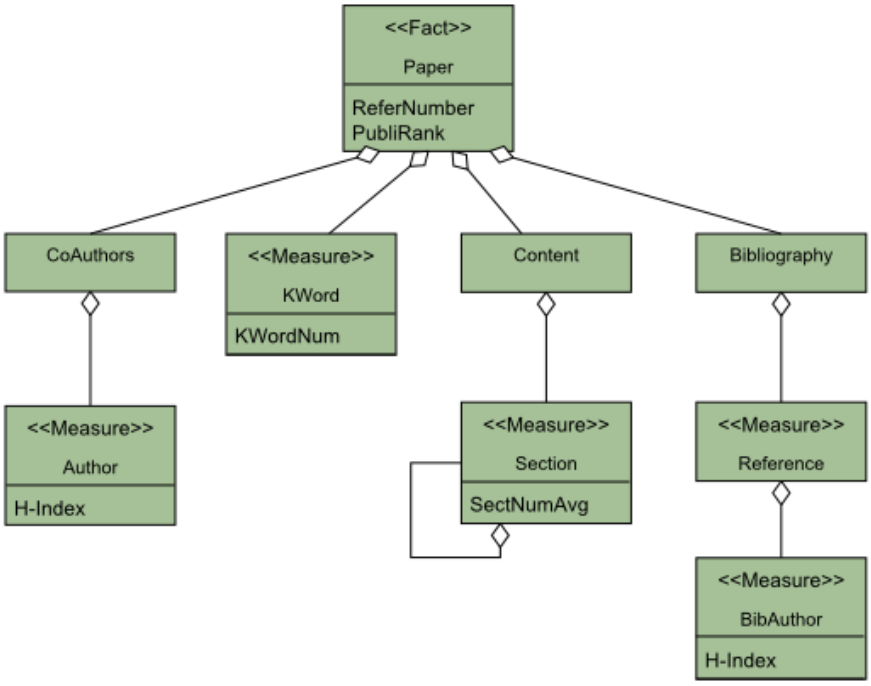

Figure 59. Le fait généré automatiquement à partir du SourceCD de la Figure 55.

La Figure 59 présente le StarCD intermédiaire généré automatiquement à partir du SourceCD de la Figure 55.

Seules les mesures simples (faisant intervenir un seul attribut) sont identifiées de manière algorithmique à partir du SourceCD. La Figure 59 représente la partie « fait » du schéma de l'entrepôt qui a été construite à partir du SourceCD de la Figure 55. On retrouve l'arborescence partielle correspondante à la structure des documents source. Bien que les mesures soient représentées sous forme d'arbres, la valeur d'une mesure est unique pour chaque valeur issue du produit cartésien des dimensions.

Notons l'ensemble des dimensions du schéma $D = \{d_1, d_2, \dots, d_n\}$ et l'ensemble des mesures $F = \{m_1, m_2, \dots, m_q\}$:

$\forall x \in d_1 \times d_2 \times \dots \times d_n$ et $\forall i \in [1, 2, \dots, q], \exists y \in m_i : x \to y$

Ce processus automatique est suivi d'une phase d'interaction avec les décideurs pour définir les mesures souhaitées. Ainsi, dans les classes du StarCD brut, un décideur peut élaborer des mesures sous la forme d'expression utilisant des attributs et des fonctions d'agrégation comme le montre la Figure 60.

Nom de la classe	Nom de la mesure	Expression (simplifiée) donnée par le décideur	Signification (par regroupement d'articles)
Paper	P-Num	= count(Paper.*)	Nombre d'articles
Paper	P-RankAvg	= avg(Paper.PubliRank)	Moyenne des niveaux de publication
KWord	KWordNum	= sum(count(KWord.*))	Nombre de mots-clés
Author	AuthorNum	= sum(count(Author.*))	Nombre d'auteurs
Author	H-IndAvg	= avg(avg(Author.H-Index))	Moyenne des H-index d'auteurs d'articles
Section	SectNumAvg	= avg(count (Section.*))	Moyenne des nombre de sections d'articles
BibAuthor	BibH-IndAvg	= avg(avg(avg(H-Index)))	Moyenne des H-index d'auteurs de la bibliographie

Figure 60. Des exemples de mesures

Selon nos choix d'implantation de l'entrepôt (voir section 9.4), le calcul d'une mesure est réalisé lors des analyses effectuées par les décideurs. Pour chaque mesure, l'entrepôt contient donc l'expression de calcul et les opérandes (attributs de la source). Certains attributs se trouvant imbriqués au sein de la classe « Fait », la fonction de calcul doit appliquer préalablement des opérateurs ensemblistes. Par exemple, le comptage des mots-clés (mesure KWordNb) est effectué sur l'union des ensembles de mots-clés associés à chaque document. L'expression de calcul est complétée automatiquement par le système selon la position des attributs dans l'arbre du SourceCD.

8.3.4. Les dimensions et leurs hiérarchies

Un fait doit pouvoir être analysé selon une ou plusieurs dimensions ; chacune d'elles est un axe d'étude qui permet de partitionner le fait lors des analyses. Dans le StarCD, toute dimension est directement liée au fait par la relation « By ».

Les dimensions sont des objets extraits du SourceCD. Dans une première étape, les dimensions potentielles sont déterminées automatiquement ; dans un second temps, les décideurs désignent les dimensions et les hiérarchies qu'ils souhaitent.

Une dimension est soit un attribut soit une classe dans le SourceCD.

Les attributs

Seuls sont concernés les attributs de la classe racine dans le SourceCD (la classe racine correspond au fait dans le StarCD). Pour identifier un attribut pouvant jouer le rôle d'une dimension, l'étude du type de données associé est déterminante ; on parle de types compatibles. C'est notamment le cas des nombres, des dates, des énumérations, des chaînes courtes. Ainsi un objet de type énumération, dont toutes les valeurs possibles sont spécifiées, est une dimension compatible ; c'est le cas de l'attribut PubliRank de la classe Paper qui donne le rang de la publication pour chaque article : « 1 », « 2 » ou « 3 ». Par contre un attribut dont les valeurs sont distinctes (« key »), n'est pas une dimension candidate ; c'est le cas de l'attribut Paper# dans le SourceCD de la Figure 55. D'autre part, un attribut de type chaîne de caractères dont la longueur ne dépasse pas 20 caractères est une dimension candidate ; au-delà de 20 caractères, nous faisons l'hypothèse que la chaîne correspond à un texte difficilement compatible avec la notion de dimension. Par exemple l'attribut PubliAcronym est une dimension candidate alors que PubliName pouvant dépasser 20 caractères ne l'est pas. Mais la longueur des chaînes n'étant pas toujours précisée dans le XSchema, il est nécessaire d'utiliser une *ontologie de domaine* (Romero and Abelló, 2007), (Nebot et al., 2009) et (Niemi and Niinimäki, 2010) pour lever cette ambiguïté. Par exemple, l'ontologie précise qu'un attribut « FirstName » comporte 15 caractères au plus.

Une classe « Dimension » portant le nom de l'attribut est créée dans le StarCD ; elle comporte un attribut faible de même nom.

Les classes

Les classes directement liées à la classe racine dans le SourceCD, sont des dimensions potentielles si elles respectent les règles suivantes ; dans ce cas elles sont reportées dans le StarCD. Notons que les attributs de ces classes sont considérés comme des attributs faibles et ne peuvent pas jouer le rôle de dimensions.

Notons F la classe du SourceCD située à la racine de l'arborescence.

Règle 1 : toute classe C1 directement liée à la classe F par un lien 1..1 (côté C1) devient une dimension candidate si :

C1 contient au moins un attribut défini sur un type compatible ; dans l'exemple de la Figure 55, c'est le cas de la classe CreationDate ;

C1 se compose de classes située à un niveau quelconque dans l'arbre et contenant au moins un attribut de type compatible ; par exemple la classe CoAuthors est une dimension candidate parce qu'elle est composée de la classe Authors contenant l'attribut H-Index.

Règle 2 : toute classe C2 directement liée à la classe F par un lien 1..N (côté C2) génère une dimension candidate intitulée « C2Group ». Par exemple la classe KWord produit la dimension KWordGroup dans le StarCD : les articles pouvant être analysés par groupe de mots-clés.

Règle 3 : toute classe composante C3 liée à une classe C1 ou C2 par un lien 1..1 génère le transfert de ses attributs dans la classe liée C1 ou C2 ; la classe C3 n'est pas reportée dans le StarCD. Par exemple les attributs de la classe composante Affiliation sont transférés dans la classe Author.

Les hiérarchies

Chaque dimension peut être associée à une ou plusieurs hiérarchies. Une des particularités de notre modèle est de proposer des hiérarchies inversées afin de respecter la structure des documents source.

Notons D une classe du SourceCD qui a généré une dimension candidate. La suite des classes liées consécutivement par des liens de composition 1..N et qui aboutit à D forme une hiérarchie. Chaque classe liée est un paramètre de la hiérarchie (concrètement le paramètre est l'identificateur interne de la classe) ; les attributs de la classe correspondent à des attributs faibles.

Par exemple, dans le SourceCD de la Figure 55, la classe CoAuthors, qui joue le rôle de dimension dans le StarCD, est associée à la hiérarchie (inversée) suivante : CoAuthors > Author. De même la classe Bibliography a pour hiérarchie : Bibliography > Reference > BibAuthor.

Le cas particulier des dimensions temporelle : toute dimension formée à partir d'un attribut de type Date, qui par construction n'est pas associée à une hiérarchie, bénéficie de deux hiérarchies prédéfinies : jour < mois < trimestre < semestre < année ainsi que jour < semaine < année.

Les dimensions générées

L'analyse automatique du SourceCD permet donc de déduire un ensemble de dimensions et de hiérarchies associées. Dans la phase d'interaction avec les décideurs, ces dimensions sont présentées sous la forme d'un StarCD comme le montre la Figure 61 ; les décideurs doivent alors désigner les dimensions utiles à leurs analyses.

Figure 61. Les dimensions générées à partir du SourceCD de la Figure 55

8.3.5. L'élaboration de StarCD par des décideurs

A partir du StarCD brut produit automatiquement, le décideur va exprimer ses mesures et choisir les dimensions qu'il souhaite pour analyser les documents. Nous présentons ci-après deux exemples de StarCD élaborés à partir du même StarCD brut.

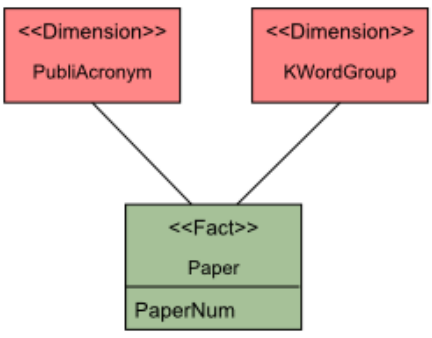

Figure 62. Premier exemple de StarCD

Considérons un premier exemple élaboré à partir du StarCD brut des Figure 59 et Figure 61.

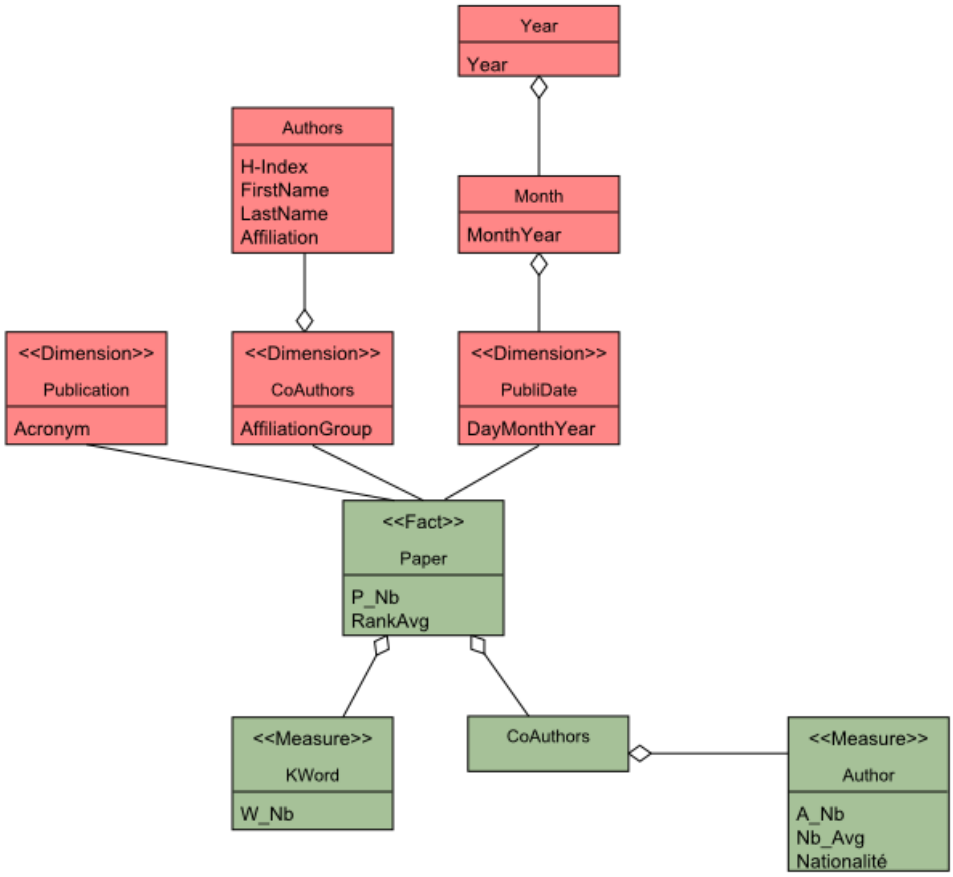

Figure 63. Deuxième exemple de StarCD

Le décideur a d'une part donné l'expression de la mesure PaperNum (nombre d'articles) et d'autre part choisi les dimensions PubliAcronym (acronyme de la publication) et KWordGroup (groupe de mots-clés). Le schéma multidimensionnel correspondant est présenté dans la Figure 62.

Ce schéma d'entrepôt permet donc de regrouper les articles par publication et/ou par groupe de mots-clés. La mesure RankAvg contient le nombre d'articles pour chaque regroupement.

8.3.6. Implantation de l'entrepôt

L'entrepôt, tel qu'il a été défini dans les sections précédentes, a été implanté dans des fichiers XML à partir de l'analyse du StarCD. Le schéma technique de l'entrepôt est obtenu en appliquant deux processus distincts d'implantation :

- le stockage des dimensions,
- le stockage des constituants de chaque mesure : l'expression de calcul et les opérandes.

Le stockage des dimensions

Dans le StarCD, chaque dimension est représentée par une arborescence de classes reliées par des liens de composition. Chaque classe correspond à un paramètre. Tout paramètre est stocké dans une table comportant :

- un identificateur d'objet système (numéro d'incrément automatique) représentant la valeur du paramètre,
- l'ensemble des valeurs des attributs faibles du paramètre,
- le cas échéant, un lien (identificateur d'objet) vers le paramètre père pour traduire le lien de composition.

Pour illustrer le processus de stockage, reprenons l'exemple d'une source d'articles scientifiques et de l'entrepôt correspondant décrit par le StarCD de la Figure 63. Pour la dimension CoAuthors, on considère les paramètres CoAuthors et Author. La lecture d'un article comportant deux co-auteurs entraîne le stockage de :

- deux quadruplets (Id_A, FirstName, LastName, Affiliation) dans la table représentant le paramètre Author,
- d'un triplet (Id_CoA, AffiliationGroup, {Id-A}) dans la table du paramètre CoAuthors.

La Figure 64 illustre le principe de stockage des dimensions.

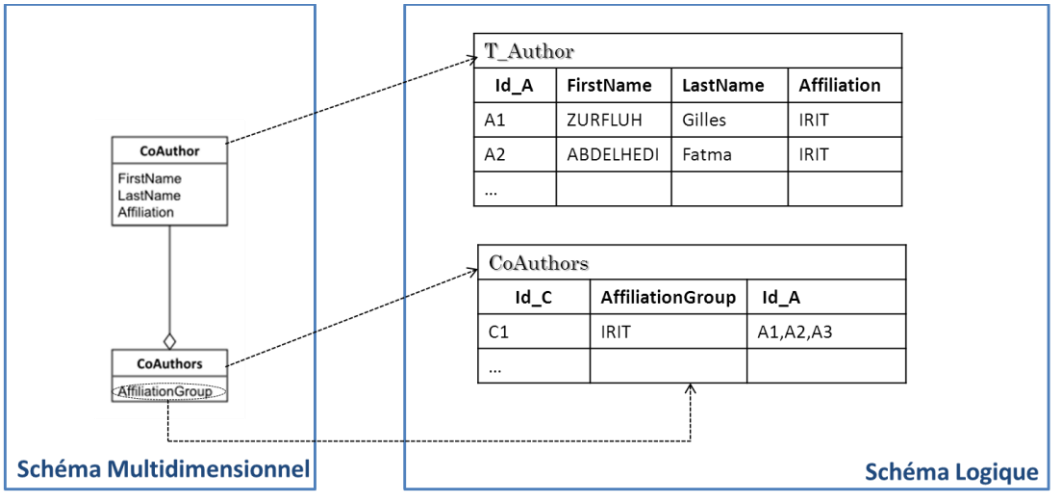

Figure 64. Le stockage des dimensions

Le stockage des mesures

Nous avons choisi de ne pas pré-calculer les mesures pour étendre les possibilités d'analyse (utilisation de filtres notamment). Par conséquent, la classe Paper, qui représente le fait dans le StarCD, contient autant d'instances d'objets qu'il y a de documents dans la source. Les attributs imbriqués dans le fait et représentées par des classes dans le StarCD, sont stockés dans des tables imbriquées comme le montre la Figure 65. Les valeurs stockées correspondent à des identificateurs d'objets situés dans les tables dimensions.

Les expressions de calcul sont stockées dans un fichier distinct.

Le schéma technique de la partie Mesures de l'entrepôt correspond à une table complexe (le fait) dont les attributs sont atomiques ou complexes, c'est-à-dire définis comme des tables (éléments imbriqués).

L'alimentation de l'entrepôt est effectuée en une seule phase qui consiste à parcourir la totalité de la source et à remplir les tables correspondantes aux dimensions et aux mesures. Les tables sont stockées dans des fichiers XML (documents « orientés données ») et alimentées par des requêtes XQuery appliquées à la source XML.

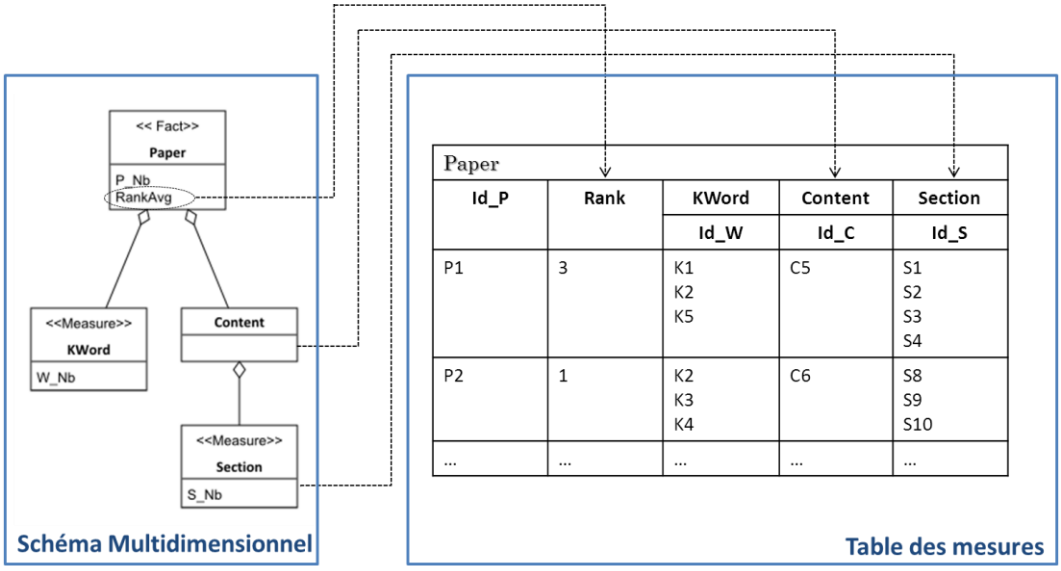

Figure 65. Le stockage du fait

La manipulation des structures de données

Des requêtes d'analyse s'appliquent sur un entrepôt pour analyser les données, c'est-à-dire obtenir les valeurs de mesures selon certaines dimensions. Ces requêtes doivent donc parcourir les structures de données définies par le StarCD et implantées sous la forme que nous venons de présenter.

Nous verrons dans les expressions des requêtes du CHAPITRE 9 que le parcours du graphe des dimensions et du graphe des mesures s'effectue grâce à une opération de jointure naturelle (notée ⋈). Ainsi, pour atteindre un attribut appartenant à une classe C imbriquée dans la classe fait, il convient d'effectuer un nombre de jointures égal au nombre de classes situées entre le fait et la classe C. La Figure 66 présente le mécanisme de la jointure appliqué à une classe C (mesure) imbriquée dans la classe Fait.

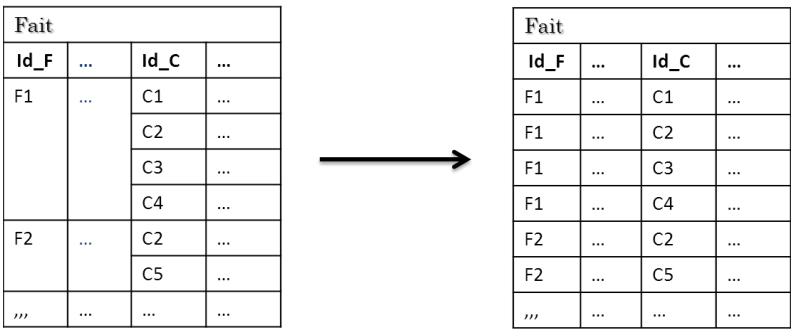

Figure 66. La jointure naturelle

8.3.7. Discussion

Nous avons proposé une méthode originale pour modéliser un entrepôt de documents. La démarche consiste à produire automatiquement un schéma multidimensionnel à partir du schéma (unifié) des documents source, puis à permettre aux décideurs d'affiner ce schéma selon leurs besoins. Notre modèle multidimensionnel, qui repose sur les diagrammes de classes du formalisme UML, décrit l'entrepôt en préservant le schéma initial des documents source ; cette propriété permet aux décideurs de mieux appréhender le schéma de l'entrepôt.

Les travaux les plus proches de notre contexte d'étude, notamment (Nassis et al., 2005) et (Pujolle et al., 2011), ne proposent pas un processus automatique pour générer un schéma multidimensionnel à partir de la source ; ils ne reprennent pas non plus la structure des documents source dans le schéma de l'entrepôt.

Nous avons expérimenté notre approche sur différents types de documents. En plus des documents scientifiques qui ont permis d'illustrer cette section, nous avons testé la méthode :

- sur une collection de messages électroniques échangés entre les services administratifs d'une université (documents à structure de faible profondeur) ;
- sur une collection de documents techniques respectant une norme aéronautique (ATA-100) et utilisés dans l'industrie pour la maintenance de matériels ainsi que pour les manuels de vol (documents à structure hiérarchique complexe).

Dans le cadre général de l'analyse des documents, notre proposition comporte cependant des limites liées à la nature de l'approche. En effet, notre méthode s'inspire des travaux réalisés sur les entrepôts de données classiques où les faits et dimensions sont extraits des schémas de bases de données Relationnelles. Par conséquent le schéma multidimensionnel n'est pas élaboré à partir du contenu des documents (principalement du texte) ; des travaux sont effectués dans ce sens (Martin-Bautista et al., 2013).

CHAPITRE 9

ANALYSE OLAP DE L'ENTREPÔT

L'analyse OLAP, introduite par (Codd et al., 1993), concerne les traitements analytiques effectués par les décideurs sur les entrepôts de données.

9.1. Formalisation

Nous proposons une formalisation d'un entrepôt en étoile. Cette formalisation est par ailleurs généralisable à toute table multidimensionnelle, c'est-à-dire au résultat d'une requête d'analyse OLAP appliquée à l'entrepôt.

Un entrepôt (ou le résultat d'une requête) est un ensemble de couples noté :

$E = (D, F)$ où :

- D représente les dimensions de l'entrepôt et qui est défini par le produit cartésien des dimensions de l'entrepôt,
- F représente l'ensemble des mesures du fait.

Notons l'ensemble des n dimensions $D = \{d_1, d_2, \dots, d_n\}$ et l'ensemble des q mesures $F = \{m_1, m_2, \dots, m_q\}$.

Plus précisément, une dimension quelconque $d_i \in D$ correspond à un paramètre appartenant graphe enraciné des paramètres de D.

De par nos choix d'implantation, les mesures sont calculées lors de l'interrogation grâce aux fonctions d'agrégation associées à chacune d'elles. Un paramètre est associé à un ensemble d'attributs faibles (0 ou plusieurs) qui sont utilisés pour l'affichage des résultats (dans une table multidimensionnelle).

La notation E désigne donc un ensemble de couples de la forme (x, y) où :

- $\forall\ x \in (d_1 X\ d_2 X \dots X\ d_n)$,
- $\exists\ y \in (m_1 X\ m_2 X \dots X\ m_q) / x \rightarrow y$

Cette propriété signifie que les valeurs des mesures (calculées) dépendent fonctionnellement d'une valeur de l'ensemble des dimensions.

9.2. L'algèbre OLAP

Un modèle de données permet de définir d'une part les structures des objets autorisés et d'autre part l'ensemble des opérations applicables à ces objets.

Nous avons donc repris les opérations sur les entrepôts de données classiques (Algèbre OLAP et langage graphique de (Pedersen and Jensen, 1999) et (Ravat et al., 2010)) et nous les avons adaptées aux structures de données que nous avons définies. Dans (Ravat et al., 2007b), les auteurs ont défini un ensemble d'opérations complet pour manipuler les entrepôts de données classiques. Mais notre objectif n'est pas de proposer un langage complet dédié à l'affichage des analyses ; il s'agit de montrer les capacités de notre langage d'analyse présenté dans la section 8.3. Nous avons donc retenu trois opérations OLAP : l'extraction, le forage et la sélection.

Dans la section 8.3, nous avons défini les objets complexes que contient un entrepôt de documents « orientés documents » ; à présent, nous allons spécifier les opérations applicables à un tel entrepôt.

Comme nous l'avons montré dans la section précédente, le résultat de toute opération sur un entrepôt est un nouvel entrepôt. Cette propriété signifie qu'un système d'entrepôts est muni d'un ensemble d'opérateurs fermés sur ce système : toute opération produit un nouvel entrepôt qui peut être à son tour opérande dans une autre opération.

Les opérations présentées sont illustrées avec le StarCD de la Figure 63.

9.2.1. L'extraction

Cette opération permet d'obtenir une ou plusieurs mesures d'un entrepôt E selon le niveau courant des dimensions spécifiées ; les autres dimensions (non précisées dans l'opération) ont pour paramètre All. Pour une dimension de E, le niveau courant de granularité correspond à la racine du graphe des paramètres. Cette opération produit un nouvel entrepôt (noté Rx) contenant les dimensions et les mesures spécifiées.

$Rx \leftarrow$ **Extract** $(< E > ; < \{F.m_i\} > ; < \{d_j . p_k . a_l\} >)$

$\forall i \in [1, 2, \dots, n], \forall j \in [1, 2, \dots, q], \forall k \in [1, 2, \dots, s]$ et $\forall l \in [1, 2, \dots, t]$

Rappelons que l'opérande E peut correspondre à l'entrepôt manipulé ou bien au résultat d'une opération précédente.

Exemples :

L'opération suivante s'applique sur l'entrepôt IRIT et retourne un nouvel entrepôt R1. Seule la dimension Publication est utilisée ; elle est fixée à sa valeur courante et associée à l'attribut faible Acronym. Ainsi, pour chaque publication représentée par son acronyme, le fait contient un nombre d'articles (mesure P_Nb).

R1 ← **Extract** (IRIT ; Paper. P_Nb ; Publication. Acronym)

Dans l'exemple suivant, l'opération **Extract** s'applique toujours sur le fait Paper mais permet d'extraire deux mesures dont la seconde figure dans la classe KWord imbriquée dans la classe-fait Paper. L'expression de calcul associée à la mesure W_Nb est donc appliquée sur le résultat de la jointure des classes Paper et KWord.

R2 ← **Extract** (IRIT ; Paper. P_Nb -- Mesure du fait Paper

(

Extract (Paper ; KWord. W_Nb) -- Mesure imbriquée

) ;

CoAuthors. AffiliationGroup) -- Dimension

Dans l'opération R3, l'extraction du nombre d'articles est effectuée selon la dimension CoAuthors (dont le paramètre éponyme est implicite) au niveau du paramètre Author ; ceci nécessite une jointure des classes Paper, CoAuthors et Author.

R3 ← Extract (IRIT ; Paper. P_Nb ; (Paper ⋈ CoAuthors ⋈ Author). FirstName))

9.2.2. Le forage

Dans un entrepôt, les valeurs des mesures dépendent fonctionnellement d'une valeur du produit cartésien de l'ensemble des dimensions (chaque dimension étant représentée par un paramètre). L'opération de forage permet de calculer de nouvelles valeurs des mesures en sélectionnant des paramètres sur les hiérarchies des dimensions. En effet toute dimension est associée à un graphe de paramètres.

Pour un StarCD comportant q dimensions, chaque dimension est associée à un nombre quelconque de paramètres noté s et chaque paramètre est lui-même associé à t attributs ; l'opération **Roll** s'exprime alors comme suit.

Ry ← **Roll** (< E > ; < F > ; < $\{d_j [p_k^j , a_l^k]\}$ >)

$$\forall j \in [1, 2, \dots, q], \ \forall k \in [1, 2, \dots, s] \text{ et } \forall l \in [1, 2, \dots, t]$$

Dans un StarCD, les hiérarchies peuvent être inversées (voir section). Par conséquent, le choix d'un paramètre va déterminer soit un forage vers le bas (Drill-down) soit un forage vers le haut (Roll-up) sur une dimension.

Exemple :

L'opération suivante modifie les paramètres courants des dimensions CoAuthors et PubliDate. Les mesures du fait Paper sont recalculées dans R4 selon les nouveaux paramètres ; les autres dimensions restent inchangées.

R4 ← **Roll** (IRIT ; Paper ; (CoAuthors [Authors, FirstName],

PubliDate[Month, MonthYear])

L'opération **Roll** permet de choisir les dimensions, les paramètres et les attributs dans le StarCD. Par conséquent, les opérations de rotation (**F-Rotate**, **D-Rotate** et **H-Rotate**) peuvent être simulées sans introduire de nouvelles opérations.

9.2.3. La sélection

Cette opération consiste à restreindre les valeurs des mesures ou des attributs de paramètres. Dans l'expression suivante, un mesure est notée m, une dimension d et un attribut d'un paramètre a ; nous considérons également des prédicats portant sur des mesures ou ceux portant sur des attributs.

Rz ← **Select** (< E > ; < Fact > ; [({< m_i, Pred/m_i >})];

[({< d_j, Pred/a_l^j >})] >)

$\forall i \in [1, 2, \dots, n], \forall j \in [1, 2, \dots, q]$ et $\forall l \in [1, 2, \dots, t]$

Exemple :

L'opération **Select** suivante restreint l'entrepôt IRIT :

- en limitant les mesures aux seuls articles écrits par des auteurs belges (au moins un Belge par article),
- en restreignant la dimension PubliDate (paramètre Year) aux années supérieures à 2010.

$R5 \leftarrow$ **Select** (IRIT ; Paper ; ((Paper $\bowtie$ CoAuthors $\bowtie$ Author). Nationalité $=$ Belge);

((PubliDate$\bowtie$Month$\bowtie$Year). Year $>$ 2010)

9.3. Le langage d'analyse

Nous décrivons dans cette section un nouveau langage OLAP pour analyser un entrepôt de documents décrit par un StarCD. Ce langage est complet au regard de l'algèbre multidimensionnelle présentée dans la section 9.2. Il s'inspire de la syntaxe du langage OQL qui a été développé pour manipuler des objets complexes et qui a été standardisé par l'ODMG (Date, 1999).

Comme pour le langage SQL qui a été initialement créé pour des utilisateurs occasionnels (Date, 1999), le langage que nous proposons est déclaratif et est destiné à l'usage des décideurs. Sa sémantique formelle peut être facilement décrite. La syntaxe du langage est simple au regard des requêtes de type XQuery nécessaires pour manipuler des documents XML. Cependant on peut penser que la mise au point d'un langage graphique sera une prochaine étape pour obtenir un langage plus convivial et mieux adapté à des non informaticiens (voir Conclusion et perspectives du mémoire).

Toute requête d'analyse s'exprime sous la forme suivante :

Analyse <mesures>

From <classes>

Where <predicat>

By <dimensions>

Une requête comporte trois clauses :

1. La clause Analyse qui contient les mesures à évaluer selon les dimensions de la clause By et les fonctions d'agrégation figurant dans le StarCD.

2. La clause From qui mentionne chaque classe du StarCD utilisée dans la requête (fait et dimensions) en lui associant une variable de désignation (alias) ; celle-ci est utilisée dans les deux autres clauses.

3. La clause Where (optionnelle) mentionne les critères ou les filtres permettant de gérer le résultat

4. La clause By (optionnelle) mentionne les dimensions permettant d'évaluer les mesures.

Une requête est une fonction qui s'applique à un entrepôt de documents et qui renvoie un entrepôt dont la structure est déduite des paramètres contenus dans la requête. Les exemples de requêtes que nous présentons s'appliquent sur le StarCD de la Figure 63 qui décrit l'entrepôt intitulé IRIT.

Nous allons examiner une série de requêtes qui montrent les capacités d'analyse de notre langage.

La requête 1 déclare une variable de désignation (alias) dans la clause from ; cette variable p associée ici au fait Paper, permet de référencer toutes les propriétés de la classe désignée. La requête ne mentionne pas de dimension : les paramètres courants sont retenus. Elle retourne donc un entrepôt intitulé R1 contenant les dimensions de E et une classe de fait Paper avec pour unique mesure PaperRkAvg.

Vue par le décideur, R1 retourne la moyenne des rangs des articles selon les axes courants.

R1 ← **Analyse** p. RankAvg

From p **in** Paper

En langage algébrique, la requête s'écrit comme suit :

R1 ← **Extract** (IRIT ; Paper. RankAvg ;)

La requête 2 calcule le nombre d'articles associés à chaque acronyme des publications. Son résultat R2 est un entrepôt contenant la mesure P_Nb dans la classe Paper et la dimension Publication.

R2 ← **Analyse** p.P_Nb

From p **in** Paper, pu **in** Publication

By pu. Acronym

Afin de concrétiser l'action de la requête R2, nous présentons son résultat dans la Figure 67 sous forme d'une table multidimensionnelle.

Paper **Paper.Nb**			
Publication	**Publi.Acronym**		
	CAISE		1500
	DAWAK		600
	DEXA		808
	ICEIS		1221

Figure 67. Résultat de la requête R2

La requête R2 correspond à une opération d'extraction du langage algébrique :

R2 ← **Extract** (IRIT ; Paper. P_Nb ; Publication. Acronym)

La requête 3 calcule le nombre d'articles et le nombre de mots-clés associés à chaque publication. Le parcours dans le graphe des mesures est effectué grâce aux variables de désignation.

R3 ← **Analyse** p.P_Nb, w.W_Nb

From p **in** Paper, w **in** p.KWord, pu **in** Publication

By pu.Acronym

En langage algébrique, cette requête s'exprime avec l'opération d'extraction comme suit :

R3 ← **Extract** (IRIT ; Paper. P_Nb , (Paper ⋈ KWord). W_Nb ;
Publication. Acronym)

La requête 4 renvoie, pour chaque publication, le nombre moyen des co-auteurs qui ont rédigé les articles.

R4 ← **Analyse** a.Nb.Avg

From p **in** Paper, c **in** p.CoAuthors, a **in** c.Author, pu **in** Publication

By pu.Acronym

En termes algébriques, la requête R4 correspond à une opération d'extraction ; elle chemine dans le graphe des mesures en réalisant deux jointures sur les trois classes Paper, CoAuthors et Author.

R4 ← **Extract** (IRIT ; (Paper ⋈ CoAuthors ⋈ Author). NbAvg ;
Publication. Acronym)

La requête 5 retourne, pour chaque publication, le nombre d'articles rédigés par des co-auteurs de nationalité belge.

R5 ← **Analyse** a.Nb.Avg

From p **in** Paper, c **in** p.CoAuthors, a **in** c.Author, pu **in** Publication

Where a.Nationalité = "Belge"

By pu.Acronym

En langage algébrique, la requête R5 se traduit par deux opérations successives. La première applique une sélection sur la mesure Author pour obtenir les auteurs belges ; la seconde porte sur le résultat de la première opération et calcule la mesure NbAvg selon la dimension Publication.

R5a ← **Select** (IRIT ; Paper ; ((CoAuthors⋈Author). Nationalité = Belge);

R5 ← **Extract** (R5a ; (Paper ⋈ CoAuthors ⋈ Author). NbAvg ;
Publication. Acronym)

La requête 6 calcule le nombre d'articles selon deux axes d'étude : l'affiliation des co-auteurs et la date de publication.

R6 ← **Analyse** p.Nb

From p **in** Paper, c **in** CoAuthors, d **in** PubliDate

By c.AffiliationGroup, d.DateMonthYear

La Figure 68 présente la table multidimensionnelle. Cette dernière permet de visualiser le résultat de la R6.

Paper Paper.Nb		PubliDate				
		DateMonth Year	**05/02/13**	**11/05/13**	**12/07/13**	**11/12/013**
CoAuthors	**Affiliation Group**					
	ERIC		5	11	15	7
	IRIT		4	12	11	7
	LSIS		5	10	14	5
	MIRACLE		4	9	15	8

Figure 68. Résultat de la requête R6

La requête 7 calcule le nombre d'articles pour chaque auteur et pour chaque année de publication. Elle utilise deux dimensions : CoAuthors et PubliDate dont les hiérarchies sont inversées ; les paramètres choisis (Author et Year) sont situés sur les hiérarchies respectives.

R7 ← **Analyse** p.P_Nb

From p **in** Paper,

c **in** CoAuthors, a **in** c.Author, -- dimension 1

d **in** PubliDate, m **in** d.Month, y **in** m.Year -- dimension 2

By a.FirstName, y.Year

La table multidimensionnelle Figure 69 permet de visualiser le résultat de la R7.

Paper Paper.Nb		PubliDate				
		Year	**2010**	**2011**	**2012**	**2013**
CoAuthors	**FirstName**					
	Auht1		7	21	15	7
	Auth2		11	12	11	7
	Auth3		13	10	14	5
	Auth4		11	15	18	8

Figure 69. Résultat de la requête R7

La requête équivalente en langage algébrique s'écrit comme suit :

R7a ← **Roll** (IRIT ; Paper ; (CoAuthors [Authors, FirstName],

PubliDate[Year, Year])

R7 ← **Extract** (R7a ; Paper. P_{Nb})

9.4. Implantation du traducteur

Nous avons développé un traducteur pour transformer les requêtes d'analyse OLAP exprimées par les décideurs en requêtes XQuery.

XQuery est un langage de requête normalisé par le W3C et permet non seulement d'interroger des collections de documents décrits par un XSchéma, mais aussi d'effectuer des calculs. A l'image de SQL, XQuery est un langage déclaratif, mais les requêtes qu'il permet d'exprimer s'avèrent souvent complexes comme le montre la Figure 70 ; ceci rend difficile, voire impossible, son usage direct par un non informaticien tel qu'un décideur.

```
<cube>
{
let $t:= (<all>{
        for $a in collection("Paper")/Paper
        return <CoAuthors>
             { for $i in $a//CoAuthors/Author
              order by $i/LastName
              return <author>{($i/LastName, $i/FirstName)}
                     </author>
             }
              </CoAuthors>
        }</all>
     )
for $g in distinct-values($t//CoAuthors)
let $cpt:=count($t/CoAuthors[.=$g])
return<groupe>
           <auteurs>{$g}</auteurs>
           <nbArticles>{$cpt}</nbArticles>
         </groupe>
}
</cube>
```

Figure 70. Une requête XQuery : obtenir le nombre de publications par groupe de CoAuteurs

Chaque requête XQuery générée par le traducteur est ensuite appliquée sur l'entrepôt de documents implanté au format XML. Le processus mis en œuvre ici est transparent pour le décideur ; autrement dit, toute requête exprimée en langage d'analyse est traduite par le système sans intervention du décideur.

Nous avons décrit la syntaxe de toute requête d'analyse OLAP à l'aide de la grammaire BNF. Celle-ci est un métalangage comportant des méta-symboles, des termes terminaux et non terminaux. La syntaxe de nos requêtes est représentée comme suit.

```
<OLAP_Query> :: = Analyse <Measure>
                  From (<var> in <class> ",")+
                  [Where <predicat()> ]
                  By <var>.<attribute>
<Measure> :: = "(" ["a"-"z"]|"_"|["0"-"9"]|""|["A"-"Z"] ")"
<predicat()> :: = <var> <signe><var>

…
```

Toute requête OLAP fait l'objet de trois opérations successives :

- une analyse lexicale et syntaxique,
- une analyse sémantique en utilisant le schéma multidimensionnel (StarCD) au format XML : vérification du typage et contrôle de conformité avec la grammaire BNF,
- la traduction en XQuery en utilisant le schéma XML de l'entrepôt.

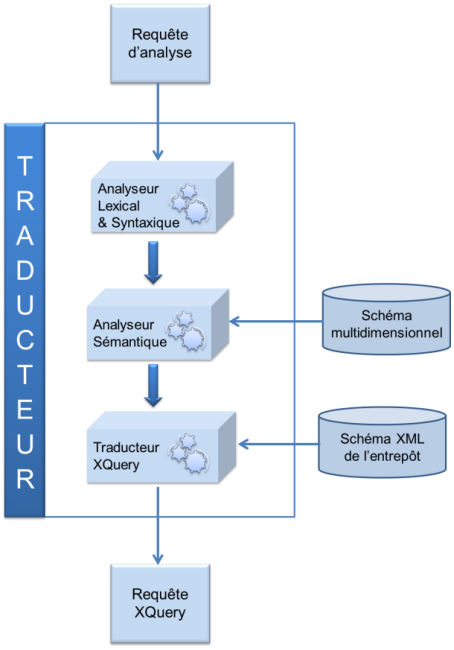

Figure 71. L'architecture du traducteur de requêtes

La Figure 71 montre l'architecture logicielle du traducteur des requêtes d'analyse OLAP.

Dans ce qui suit, nous présentons les caractéristiques du troisième module : le traducteur XQuery.

9.4.1. Les données en entrée

La requête OLAP

Le traducteur XQuery prend en entrée la requête OLAP après vérification lexico-syntaxique (par rapport à la grammaire BNF) et sémantique (par rapport au schéma StarCD). Cette requête est organisée sous la forme d'un arbre afin d'être traitée plus aisément par l'algorithme. La racine de l'arbre représente le fait d'où émanent deux types d'arc : les arcs de mesures et les arcs de dimensions. Comme le montre la figure 7.2, on peut donc distinguer des sous-arbres liés au fait et correspondant aux chemins menant aux mesures et aux dimensions de l'entrepôt.

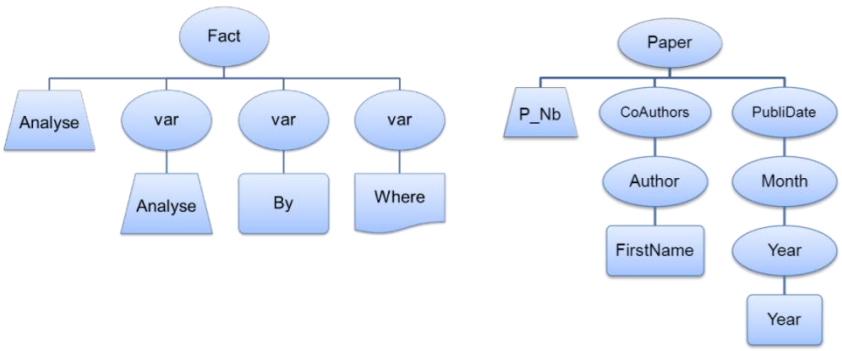

Figure 72. L'arbre générique d'une requête OLAP et une instanciation

L'entrepôt

Pour transformer la requête OLAP en une requête XQuery, le traducteur utilise le schéma de l'entrepôt physique ; celui-ci représente les structures de données XML telles qu'elles sont implantées. L'entrepôt est stocké dans un document XML unique. Les dimensions et leurs paramètres ainsi que le fait et ses mesures sont distingués par des balises spécifiques.

La figure 73 montre la spécificité de l'entrepôt dans lequel les mesures ne sont pas pré-calculées ; il y a donc autant d'instances du fait que de documents dans la source. Bien entendu, le volume de chaque instance est limité à quelques kilo-octets correspondant uniquement au stockage des opérandes de mesures et aux liens. L'intérêt de ce mode de stockage où les chaque instance du fait correspond à un

document, réside dans la possibilité de répondre à certaines requêtes ; notamment, les analyses OLAP faisant intervenir des sélections sur les mesures.

Bien que stocké dans le format XML, les données de l'entrepôt peuvent se décrire par des structures de table d'objets (acceptant des attributs multivalués). Les objets sont associés à des identificateurs d'objet correspondant à des numéros d'ordre dans une table. Comme dans les SGBD objet, ces identificateurs sont utilisés à la fois pour identifier les objets et pour établir des liens. Nous décrivons ci-après la structure de l'entrepôt et son principe d'alimentation à partir de la source de documents. Nous disposons du StarCD et du schéma de la SourceCD

L'entrepôt comporte trois parties (éléments dans XML) distinctes : un répertoire, les dimensions et paramètres ainsi que le fait et ses mesures.

Le « répertoire » contient l'ensemble des données exploitables extraites des sources et qui présentent un intérêt décisionnel : nombres et chaînes de caractères uniquement (à l'exclusion des textes, images, figures, etc.). Ce répertoire contient autant de tables que de classes dans le StarCD (à l'exclusion des hiérarchies liées aux dates). Ces tables sont indépendantes les unes des autres, c'est-à-dire non liées entre elles.

La partie « Dimensions et paramètres » contient les valeurs des paramètres pour chaque dimension. Ces valeurs sont stockées sous la forme de tables ayant pour attributs :

- Un identificateur de paramètres,
- Un lien vers les tables du Répertoire contenant les valeurs de chaque attribut faible,
- Le cas échéant, un lien (multivalué) pointant sur le paramètre de niveau supérieur.

La partie « Fait et mesures » correspond au stockage du fait et de l'ensemble de ses mesures. Les données sont enregistrées dans une table unique contenant une instance par document source. Le schéma de cette table est le suivant :

- Un identificateur du fait,
- Un lien (atomique ou multivalué) vers les tables du « Répertoire » contenant les valeurs de chaque opérande de mesure,
-Un lien atomique vers des tables de la partie « Dimensions et paramètres », c'est-à-dire chaque dimension du StarCD.

« Répertoire »

T_PubliAcronym	
T_Pa	PubliAcronym
1	DAWAK
2	DOLAP
3	EDA
4	ICEIS
5	INFORSID
6	VLDB
…	

T_PubliRank	
T_Pr	PubliRank
1	2
2	2
3	1
4	3
5	1
6	3
…	

T_PubliDate	
T_Pd	PubliDate
1	12/09/2013
2	10/02/2013
3	05/06/2013
4	10/11/2013
5	11/01/2013
6	08/06/2012
…	

T_KWord	
T_K	KWord
1	BigData
2	DataBase
3	Decisional
4	ETL
5	NoSQL
6	Warehouse
…	

T_CoAuthors	
T_Co	AffilitationGroup
1	ERIC
2	IRIT
3	LAAS
4	MIRACL
5	DISI
…	

T_Author			
T_A	FirstName	LastName	Affiliation
1	Gilles	Zurfluh	IRIT
2	Claude	Chrisment	IRIT
3	Fadila	BenTayeb	ERIC
4	Fatma	Abdelhédi	IRIT
5	Matéo	Golfarelli	DISI
…			

« Dimensions et paramètres »

D_PubliAcronym	
D_Pa	T_Pa
1	3
2	4
3	4
4	2
5	5
6	3
…	

D_PubliDate	
D_Pd	T_Pd
1	2
2	4
3	2
4	3
5	1
6	3
…	

D_Author	
D_A	T_A
1	2
2	5
3	4
4	3
5	1
6	5
…	

D_CoAuthors	
D_Co	{D_A}
1	1
	2
2	1
	5
3	3
…	

« Fait et mesures »

F_Paper							
F_P	T_Pr	{T_K}	T_Co	{T_A}	D_Pa	D_Co	D_Pd
1	2	4	3	3	5	3	2
		5					
		7					
2	3	3	1	1	2	1	4
		4		2			
3	3	1	2	1	3	2	1
		6		5			
4	2	5	3	3	4	3	6
		6					
		7					
…							

Figure 73. Les tables de l'entrepôt

La figure 74 montre un exemple de schéma physique de l'entrepôt dans le format XML, tel qu'il est exploité par l'algorithme du traducteur.

Figure 74. Le XSchéma de l'entrepôt

9.4.2. L'algorithme

La figure 75 présente l'algorithme général du traducteur qui a été programmé en Java. L'algorithme se déroule en deux étapes : le parcours de l'arbre d'analyse et l'élaboration de la requête XQuery.

Durant la première étape, nous récupérons l'ensemble de valeurs qui serviront pour la deuxième étape. A partir de la requête OLAP qui lui est soumise, le traducteur parcours l'arbre correspondant à la requête d'analyse ; il extrait les feuilles liées aux mesures, aux dimensions et aux conditions. En partant de la racine et pour chaque feuille, l'algorithme extrait les nœuds intermédiaires qui représentent le chemin de chaque mesure, dimension et condition.

La deuxième étape élabore la requête XQuery. L'expression de chaque clause de la requête (For, Let, Where, OrderBy et Return) est construite à partir des valeurs extraites dans la première étape.

```
Algorithm : Traducteur
Input : TAQ, SourceCD                          -- Requête d'analyse, SourceCD
Output : OLAPXQuery                            -- Requête XQuery

MList ← nil
DList ← nil
DPathList ← nil
CList ← nil
CPathList ← nil

-- première étape
R ← TAQ.root
While (R.filsG ¬ nil)
L ← R.filsG
   for i ← 1 to L.size() Do
      if (h(Ni) ¬ nil) then
         P ← Ni.LeafNode
      else P ← Ni
      end if
      -- IsMeasure est une fonction qui retourne vrai si le nœud courant est une mesure
      if IsMeasure (P) then
         MList ← P
     -- IsDimension est une fonction qui retourne vrai si le nœud courant est une
dimension

         else if IsDimension (P) then
                 DList ← P
              else
                 CList ← P
              end if
      end if
   end for
--  deuxième étape
OLAPXQuery ← nil
 ForClause ← DPathList
 LetClause ← assignation de chaque valeur des dimensions dans une variable
 WhereClause ← CList
 OrderBy ← DList
 Return ← { for rd ← 1 to DList.size()
            <Dimensionrd> DList(rd) </Dimensionrd>
             for me ← 1 to MList.size()
             <Measureme>MList(me)<Measureme>
          }
```

Figure 75. L'algorithme du traducteur XQuery

9.5. Discussion

Les travaux que nous venons de présenter, s'inscrivent dans la problématique de l'entreposage de documents XML et traitent des langages OLAP pour analyser ces documents.

Des langages OLAP ont été proposés pour manipuler des entrepôts XML ; mais la plupart de ces travaux concerne les documents « orientés données » (Pedersen et al., 2001), (Yin and Pedersen, 2004) et (Boussaid et al., 2008). Pour les documents « orientés documents », les langages algébriques proposés (Ravat et al., 2007b) ont été définis en fonction des modèles de données proposés.

Pour notre part, nous avons proposé un nouveau modèle multidimensionnel (CHAPITRE 9) pour décrire un entrepôt de documents XML. L'originalité de notre modèle est de représenter les faits et les dimensions sous la forme d'une forêt (ensemble de graphes) reprenant la structure des documents de la source.

Partant de là, nous avons défini un nouveau langage OLAP permettant d'analyser un entrepôt de documents et présentant les caractéristiques suivantes.

- Ce langage d'analyse est de type SQL, c'est-à-dire déclaratif ; en cela, il est destiné à des non informaticiens puisqu'aucune connaissance algorithmique n'est exigée.
- Nous avons isolé trois opérations majeures de l'algèbre OLAP classique (RAVAT et al), et nous avons montré que notre langage permet d'exprimer simplement ces opérations.
- Le modèle de l'entrepôt de documents reposant sur le concept d'objets complexes, la syntaxe du langage reprend les principes du langage OQL (Object Query Langage).
- Le parcours du graphe des mesures et du graphe des dimensions est effectué grâce à l'utilisation de variables de désignation (alias).

Ce langage OLAP n'est pas une fin en soi. En effet, il nous a permis de vérifier la pertinence de notre modèle multidimensionnel en montrant les possibilités de parcours des mesures et des dimensions par un décideur. A présent, des travaux vont être menés pour mixer données et documents XML dans le système SelfStar et pour spécifier une interface graphique dédiés à l'analyse de ces objets.

Les contributions de ce chapitre ont été présentées dans la publication (Abdelhédi et al., 2014a) et (Abdelhédi et al., 2014b).

CHAPITRE 10

CONCLUSION ET PERSPECTIVES

10.1. Conclusion

Les travaux de recherche présentés dans ce mémoire s'inscrivent dans le contexte des systèmes décisionnels.

> *Notre projet vise à proposer une démarche, des modèles et des outils logiciels permettant à des décideurs de concevoir, mettre en œuvre et analyser un entrepôt de manière autonome.*

Les décideurs doivent pouvoir construire des entrepôts mixant des données extraites de bases de données relationnelles et des documents XML extraits de la Toile ; le mixage de ces deux types d'objets est un enjeu majeur pour les bases décisionnelles dans les années à venir. Jusqu'à présent, l'élaboration d'un entrepôt est confiée à un informaticien et peu de travaux ont pris en compte l'autonomie du décideur dans le processus d'élaboration des entrepôts.

Pour traiter cette problématique, nous avons décomposé nos travaux en deux parties :

- la définition d'une démarche ainsi que la spécification et la mise en œuvre d'un outil permettant à un décideur d'élaborer un entrepôt de données,

- la définition d'un modèle multidimensionnel de documents XML.

Pour atteindre l'objectif général de notre projet, nos travaux devront être poursuivis par l'intégration et l'analyse conjointe des données structurées (relationnelles) et des données semi-structurées (documents XML).

Dans cette thèse, nos contributions ont donc porté sur deux problématiques complémentaires.

Tout d'abord, une démarche originale d'élaboration d'un entrepôt de données a été spécifiée. Au gré de ses besoins, le décideur peut élaborer son propre entrepôt de données, et ceci en toute autonomie. Pour ce faire, nous avons proposé un processus incrémental de spécification de schéma multidimensionnel à partir d'une source de données. Tout au long de ce processus, le décideur intègre progressivement ses besoins d'analyse en termes de faits, dimensions et hiérarchies, sans avoir recours à un informaticien. La connaissance nécessaire des sources par le décideur est réduite grâce à l'usage de techniques de personnalisation. Ce mécanisme ne limite en rien les possibilités du décideur en matière de recherche des faits à analyser ; celui-ci pouvant rechercher un fait en « naviguant » dans le schéma de la base de données.

Tout au long du processus d'élaboration du schéma multidimensionnel, des métadonnées sont stockées. Ces métadonnées sont utilisées pour la personnalisation ainsi que pour l'alimentation de l'entrepôt à partir des sources. Une fois le schéma multidimensionnel élaboré, le chargement des données est assuré automatiquement par le système. D'autre part et bien que, dans la pratique, un entrepôt soit souvent élaboré à partir d'une base de données unique, nous avons formalisé le processus en le généralisant à plusieurs sources de données décrites par un schéma en constellation. Afin de valider nos propositions, nous avons développé un logiciel dénommé SelfStar qui est basé sur une démarche en deux étapes successives. La première étape consiste à élaborer de manière incrémentale un schéma multidimensionnel en interagissant avec le décideur. La seconde étape, entièrement automatique, implante l'entrepôt sous forme relationnelle et effectue son chargement depuis les sources.

Notre seconde contribution a consisté à définir un modèle multidimensionnel de documents XML et un langage d'analyse OLAP associé. Il s'agissait de disposer d'un modèle d'entrepôt adapté à notre problématique. En effet, les modèles actuels ne répondent pas à nos objectifs, notamment celui de disposer d'un modèle de documents « orientés documents » facile à appréhender par des décideurs.

Nous avons donc proposé un nouveau modèle conceptuel d'entrepôt dédié aux analyses OLAP des documents XML. L'originalité de notre modèle est de représenter les faits et les dimensions sous la forme d'une forêt (graphes de mesures et graphes de dimensions) reprenant la structure des documents sources. Nous considérons que cette structure constitue un aspect sémantique majeur pour faciliter le processus d'aide à la décision. Nous avons également proposé une démarche pour la génération automatique d'un schéma multidimensionnel à partir d'une ou plusieurs sources de documents XML. Les schémas sont unifiés à partir d'algorithmes connus (non présentés dans ce mémoire). Nous avons proposé un nouveau langage d'analyse OLAP portant sur notre modèle. Des requêtes OLAP sont exprimées en termes de faits et de dimensions à partir du modèle multidimensionnel. Enfin nous avons développé un traducteur qui assure la transformation des requêtes OLAP en requêtes XQuery ; celles-ci s'appliquent sur l'entrepôt de documents.

10.2. Perspectives

Nous envisageons de poursuivre nos travaux dans le cadre général du projet SelfStar afin d'atteindre son objectif initial. Jusqu'à présent, nous avons apporté une contribution à ce projet à deux niveaux :

- La définition d'une démarche et la réalisation d'un prototype permettant aux décideurs de concevoir et créer un entrepôt de données basées sur des sources classiques (relationnelles notamment),

- La définition d'une démarche et d'un modèle multidimensionnel pour entreposer des documents XML.

Le prolongement de ces travaux consiste à intégrer des documents XML dans le système SelfStar. Ainsi, les décideurs pourront mixer, au sein du même entrepôt, des données structurées extraites de bases relationnelles avec des documents issus du Web. L'intégration et l'analyse conjointe de ces deux types de données posent néanmoins les problèmes suivants.

Le modèle multidimensionnel doit permettre de représenter, au sein d'un schéma d'entrepôt unique, des données extraites de tables ainsi que des données provenant de documents au format XML. Pour représenter le schéma multidimensionnel dans SelfStar d'une part et pour décrire un entrepôt de documents d'autre part, nous avons choisi le modèle des classes d'UML. Ce modèle permet donc de décrire tous types d'objets complexes. Mais la difficulté majeure dans le processus d'élaboration du schéma multidimensionnel intégré, est liée au partage des dimensions entre un fait extrait d'une table et un fait correspondant à un type de documents. Ceci dans le but de faciliter les analyses. Par exemple, dans une base intégrée Labo_Recherche, un fait Projets et un fait Articles-Scientifiques peuvent avoir respectivement une dimension Participants et une dimension Auteurs dont la correspondance est évidente et utile. Un processus d'unification des dimensions est alors nécessaire et sa mise en œuvre devrait être simple pour être effectuée par un utilisateur occasionnel : le décideur.

Un autre problème est lié au langage d'analyse de cet entrepôt intégré. Qu'il s'agisse d'une algèbre ou d'un langage de type SQL, les structures de données présentes dans l'entrepôt exige une capacité d'expression plus grande que les langages spécialisés présentés dans ce mémoire.

Enfin, le langage d'analyse OLAP étant mis en œuvre par des utilisateurs occasionnels ou ne maîtrisant pas les techniques informatiques, une interface graphique faciliterait l'expression de requêtes complexes. Ce type d'interface OLAP a été proposé pour exprimer des requêtes sur des entrepôts de données classiques (interface graphique requête Ravat et all 2006) ; à notre connaissance, aucune proposition n'a été faite pour des entrepôts mixant données structurées et documents.

Un autre prolongement de nos travaux permettrait de généraliser l'approche du projet SelfStar en proposant de nouvelles techniques d'analyse décisionnelle pour les Masse de Données (« Big Data »). Le concept de Masse de Données est apparu avec la prolifération des données échangées et stockées dans les réseaux sociaux et, plus généralement, dans tous les échanges commerciaux, publics et privés dans le monde. Le traitement de ces données multi-types est considéré comme un des défis majeurs des prochaines années dans les domaines de l'informatique et des

télécommunications. Les Masses de Données présentent des caractéristiques assez bien cernées :

- des volumes de données considérables généralement incompatibles avec les techniques de traitement et de stockage actuellement disponibles ;

- une grande hétérogénéité des formats de représentation et une grande variété de la qualité (fiabilité pour le décideur) de l'information selon les sources ;

- des flots continus de données (« tweet », flux RSS, etc.) devant être estampillées, analysées et réduites en temps réel pour une utilisation immédiate ou différée.

Les techniques actuelles de traitement, de stockage et d'analyse doivent être reconsidérées et étendues, voire redéfinies compte tenu des contraintes inhérentes à ce domaine d'étude.

BIBLIOGRAPHIE GÉNÉRALE

Fatma Abdelhédi, Landry Ntsama, and Gilles Zurfluh. 2014a. Analyse OLAP d'un entrepôt de documents XML natif. In *INFORSID*, (à paraître).

Fatma Abdelhédi, Landry Ntsama, and Gilles Zurfluh. 2014b. XML Warehouse Modelling and Querying. In *BADS 2014, Beyond Databases, Architectures and Structures*, (à paraître).

Fatma Abdelhédi, Geneviève Pujolle, Olivier Teste, and Gilles Zurfluh. 2011a. Computer-aided Data-mart Design. In *ICEIS (1)*, pages 239–246.

Fatma Abdelhédi, Franck Ravat, Olivier Teste, and Gilles Zurfluh. 2011b. Métadonnées de personnalisation dans le système SelfStar. In *EDA*, pages 135–153.

Fatma Abdelhédi, Franck Ravat, Olivier Teste, and Gilles Zurfluh. 2011c. SelfStar: un système interactif pour la construction de schémas multidimensionnels. In *INFORSID*, pages 335–350.

Fatma Abdelhédi and Gilles Zurfluh. 2013. User Support System for Designing Decisional Database. In *ACHI 2013, The Sixth International Conference on Advances in Computer-Human Interactions*, pages 377–382.

Alberto Abelló, Jose Samos, and Felix Saltor. 2001a. Understanding facts in a multidimensional object-oriented model. In *Proceedings of the 4th ACM international workshop on Data warehousing and OLAP*, pages 32–39.

Alberto Abelló, José Samos, and Felix Saltor. 2001b. Understanding Analysis Dimensions in a Multidimensional Object-Oriented Model. In *DMDW*, page 4.

Alberto Abelló, José Samos, and Felix Saltor. 2003. Implementing operations to navigate semantic star schemas. In *Proceedings of the 6th ACM international workshop on Data warehousing and OLAP*, pages 56–62.

Rakesh Agrawal, Ashish Gupta, and Sunita Sarawagi. 1997. Modeling multidimensional databases. In *Data Engineering, 1997. Proceedings. 13th International Conference on*, pages 232–243.

Eya Ben Ahmed, Ahlem Nabli, and Faïez Gargouri. 2011. A SURVEY OF USER-CENTRIC DATA WAREHOUSES: FROM PERSONALIZATION TO RECOMMENDATION. *International Journal of Database Management Systems*, 3(2).

Ahmad Abdullah Alqarni and Eric Pardede. 2012. Integration of Data Warehouse and Unstructured Business Documents. In *2012 15th International Conference on Network-Based Information Systems (NBiS)*, pages 32 –37.

Estella Annoni, Franck Ravat, Olivier Teste, and Gilles Zurfluh. 2006. Towards multidimensional requirement design. In *Data Warehousing and Knowledge Discovery*, pages 75–84.

F. Atigui, F. Ravat, O. Teste, and G. Zurfluh. 2012. Using OCL for Automatically Producing Multidimensional Models and ETL Processes. *Data Warehousing and Knowledge Discovery, pages* 42–53.

Fadila Bentayeb, Omar Boussaid, Cécile Favre, Franck Ravat, and Olivier Teste. 2009. Personnalisation dans les entrepôts de données: bilan et perspectives. In *EDA*, pages 7–22.

Doulkifli Boukraâ, Omar Boussaïd, Fadila Bentayeb, and Djamel-Eddine Zegour. 2013. A Layered Multidimensional Model of Complex Objects. In *Advanced Information Systems Engineering*, pages 498–513.

Omar Boussaid, Jérôme Darmont, Fadila Bentayeb, and Sabine Loudcher. 2008. Warehousing complex data from the web. *International Journal of Web Engineering and Technology*, pages 408–433.

Keith Bradley, Rachael Rafter, and Barry Smyth. 2000. Case-based user profiling for content personalisation. In *Adaptive Hypermedia and Adaptive Web-Based Systems*, pages 62–72.

Guillaume Cabanac, Max Chevalier, Claude Chrisment, and Christine Julien. 2010. Social validation of collective annotations: Definition and experiment. *Journal of the American Society for Information Science and Technology*, pages 271–287.

Luca Cabibbo and Riccardo Torlone. 1998. From a procedural to a visual query language for OLAP. In *Scientific and Statistical Database Management, 1998. Proceedings. Tenth International Conference on*, pages 74–83.

Andrea Carmè, Jose-Norberto Mazón, and Stefano Rizzi. 2010. A model-driven heuristic approach for detecting multidimensional facts in relational data sources. In *Data Warehousing and Knowledge Discovery*, pages 13–24.

Corine Cauvet and Gwladys Guzelian. 2008. Business process modeling: A service-oriented approach. In *Hawaii international conference on system sciences, proceedings of the 41st annual*, pages 98–98.

E. F. Codd. 1982. Relational database: a practical foundation for productivity. *Commun. ACM*, pages 109–117.

E. F. Codd, Robert S. Arnold, Jean-Marc Cadiou, Chin-Liang Chang, and Nick Roussopoulos. 1978. *Rendezvous Version 1: An Experimental English-Language: Query Formulation System for Casual Users of Relational: Data Bases*. IBM Thomas J. Watson Research Center.

Edgar F. Codd, S. B. Codd, and C. T. Salley. 1993. *Providing OLAP (On-line Analytical Processing) to User-analysts: An IT Mandate*. Codd & Associates.

C. J. Date. 1999. An Introduction to Database Systems (Introduction to Database Systems.

Dublin Core. 2012. The Dublin Core Metadata Element Set de http://dublincore.org/, Version 1.1.

Cécile Favre, Fadila Bentayeb, and Omar Boussaid. 2007. Evolution of data warehouses' optimization: a workload perspective. In *Data Warehousing and Knowledge Discovery*, pages 13–22. Springer.

Irene Garrigós, Jesús Pardillo, Jose-Norberto Mazón, and Juan Trujillo. 2009. A conceptual modeling approach for olap personalization. In *Conceptual Modeling-ER 2009*, pages 401–414. Springer.

M Golfarelli, S Rizzi, and E Saltarelli. 2002a. Index selection techniques in data warehouse systems. *Proc. International Workshop on Design and Management of DataWarehouses DMDW'02, pages* 33–42.

M. Golfarelli, D. Maio, and S. Rizzi. 1998. Conceptual design of data warehouses from E/R schemes. In *System Sciences, 1998., Proceedings of the Thirty-First Hawaii International Conference on*, volume 7, pages 334–343.

M. Golfarelli, S. Rizzi, and B. Vrdoljak. 2001. Data warehouse design from XML sources. In *Proceedings of the 4th ACM international workshop on Data warehousing and OLAP*, pages 40–47.

Matteo Golfarelli and Stefano Rizzi. 1998. A methodological framework for data warehouse design. In *Proceedings of the 1st ACM international workshop on Data warehousing and OLAP*, pages 3–9.

Matteo Golfarelli, Stefano Rizzi, and Ettore Saltarelli. 2002b. WAND: A CASE tool for workload-based design of a data mart. In *10th National Convention on Systems Evolution for Data Bases*, pages 422–426.

Jim Gray, Surajit Chaudhuri, Adam Bosworth, Andrew Layman, Don Reichart, Murali Venkatrao, Frank Pellow, and Hamid Pirahesh. 1997. Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals. *Data Mining and Knowledge Discovery*, pages 29–53.

Carlos A. Hurtado, Alberto O. Mendelzon, and Alejandro A. Vaisman. 1999. Maintaining data cubes under dimension updates. In *Data Engineering, 1999. Proceedings., 15th International Conference on*, pages 346–355.

W. H. Inmon. 2002. *Building the data warehouse*. J. Wiley.

W. H. Inmon and William H. Inmon. 1996. *Building the Data Warehouse*. Verlag John Wiley & Sons, Inc, 2 Sub edition, April.

Prudhvi Janga and Karen C. Davis. 2013. Schema Extraction and Integration of Heterogeneous XML Document Collections. In Alfredo Cuzzocrea and Sofian Maabout, editors, *Model and Data Engineering*, Lecture Notes in Computer Science, Springer Berlin Heidelberg, pages 176–187.

Petar Jovanovic, Oscar Romero, Alkis Simitsis, and Alberto Abelló. 2012. ORE: an iterative approach to the design and evolution of multi-dimensional schemas. In *Proceedings of the fifteenth international workshop on Data warehousing and OLAP*, pages 1–8.

Ralph Kimball. 1996. The data warehouse toolkit: practical techniques for building dimensional data warehouse. *John Willey & Sons*.

Jean-Louis Le Moigne. 1990. *La modélisation des systèmes complexes*.volume 2. Dunod Paris.

Yu Li and Aijun An. 2005. Representing UML snowflake diagram from integrating XML data using XML schema. In *Data Engineering Issues in E-Commerce, 2005. Proceedings. International Workshop on*, pages 103–111.

Elzbieta Malinowski and Esteban Zimányi. 2006. Hierarchies in a multidimensional model: From conceptual modeling to logical representation. Data & Knowledge Engineering, pages 348–377, November.

Maria J. Martin-Bautista, Carlos Molina, Elizabet Tejeda, and Maria-Amparo Vila. 2013. A new multidimensional model with text dimensions: definition and implementation. *International Journal of Computational Intelligence Systems*, pages 137–155.

Jose-Norberto Mazón and Juan Trujillo. 2008. An MDA approach for the development of data warehouses. *Decision Support Systems*, pages 41–58, April.

Jose-Norberto Mazón and Juan Trujillo. 2009. A hybrid model driven development framework for the multidimensional modeling of data warehouses! *ACM SIGMOD Record*, pages 12–17.

Jacques Mélèse. 1982. *L'analyse modulaire des systèmes de gestion: une méthode efficace pour appliquer la théorie des systèmes au management*. Éditions Hommes et techniques, Boulogne-Billancourt.

George A. Miller. 1995. WordNet: a lexical database for English. *Communications of the ACM*, pages 39–41.

George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J. Miller. 1990. Introduction to wordnet: An on-line lexical database*. *International journal of lexicography*, pages 235–244.

Rokia Missaoui, Ganaël Jatteau, Ameur Boujenoui, and Sami Naouali. 2007. Toward Integrating Data Warehousing with Data Mining Techniques. *Data warehouses and OLAP: concepts, architectures, and solutions*, 253.

Daniel L. Moody and Mark AR Kortink. 2000. From enterprise models to dimensional models: a methodology for data warehouse and data mart design. In *DMDW*, page 5.

Vicky Nassis, Rajugan Rajagopalapillai, Tharam S. Dillon, and Wenny Rahayu. 2005. Conceptual and systematic design approach for XML document warehouses. *International Journal of Data Warehousing and Mining (IJDWM)*, pages 63–87.

Vicky Nassis, Rajagopal Rajugan, Tharam Dillon, and Wenny Rahayu. 2004. Conceptual design of XML document warehouses. *Data Warehousing and Knowledge Discovery*, pages 1–14.

Victoria Nebot, Rafael Berlanga, Juan Manuel Pérez, María José Aramburu, and Torben Bach Pedersen. 2009. Multidimensional integrated ontologies: a framework for designing semantic data warehouses. In *Journal on Data Semantics XIII*, pages 1–36. Springer.

Tapio Niemi and Marko Niinimäki. 2010. Ontologies and summarizability in OLAP. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, pages 1349–1353.

Torben Bach Pedersen and Christian S. Jensen. 1999. Multidimensional data modeling for complex data. In *Data Engineering, 1999. Proceedings., 15th International Conference on*, pages 336–345. IEEE.

Torben Bach Pedersen, Christian S. Jensen, and Curtis E. Dyreson. 2001. A foundation for capturing and querying complex multidimensional data. *Information Systems*, pages 383–423.

J.M. Perez, R. Berlanga, M.J. Aramburu, and T.B. Pedersen. 2008. Integrating Data Warehouses with Web Data: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, pages 940 –955, July.

F. Pinet and M. Schneider. 2009. A unified object constraint model for designing and implementing multidimensional systems. *Journal on Data Semantics XIII*, pages 37–71.

N. Prat, J. Akoka, and I. Comyn-Wattiau. 2006a. A UML-based data warehouse design method. *Decision Support Systems*, pages 1449–1473.

Nicolas Prat, Jacky Akoka, and Isabelle Comyn-Wattiau. 2006b. A UML-based data warehouse design method. *Decision support systems*, pages 1449–1473.

Geneviève Pujolle, Franck Ravat, Olivier Teste, Ronan Tournier, and Gilles Zurfluh. 2011. Multidimensional database design from document-centric XML documents. In *Data Warehousing and Knowledge Discovery*, pages 51–65.

Franck Ravat and Olivier Teste. 2008. Personalization and OLAP databases. *Annals of Information Systems. New Trends in Data Warehousing and Data Analysis*, pages 71–92.

Franck Ravat and Olivier Teste. 2009. Personalization and OLAP databases. In *New Trends in Data Warehousing and Data Analysis*, pages 1–22.

Franck Ravat, Olivier Teste, and Ronan Tournier. 2007a. Olap aggregation function for textual data warehouse. In *ICEIS (1)*, pages 151–156.

Franck Ravat, Olivier Teste, Ronan Tournier, and Gilles Zurfluh. 2007b. Graphical Querying of Multidimensional Databases. In Yannis Ioannidis, Boris Novikov, and Boris Rachev, editors, *Advances in Databases and Information Systems*, Lecture Notes in Computer Science, pages 298–313.

Franck Ravat, Olivier Teste, Ronan Tournier, and Gilles Zurfluh. 2008. Top_Keyword: An Aggregation Function for Textual Document OLAP. In Il-Yeol Song, Johann Eder, and Tho Manh Nguyen, editors, *Data Warehousing and Knowledge Discovery*, Lecture Notes in Computer Science, pages 55–64.

Stefano Rizzi, Alberto Abelló, Jens Lechtenbörger, and Juan Trujillo. 2006. Research in data warehouse modeling and design: dead or alive? In *Proceedings of the 9th ACM international workshop on Data warehousing and OLAP*, pages 3–10.

Oscar Romero and Alberto Abelló. 2007. Automating multidimensional design from ontologies. In *Proceedings of the ACM tenth international workshop on Data warehousing and OLAP*, pages 1–8.

I. Y. Song, R. Khare, Y. An, S. Lee, S. P. Kim, J. Kim, and Y. S. Moon. 2008. SAMSTAR: An automatic tool for generating star schemas from an entity-relationship diagram. *Conceptual Modeling-ER 2008*, pages 522–523.

Il Yeol Song, Ritu Khare, and Bing Dai. 2007. SAMSTAR: a semi-automated lexical method for generating star schemas from an entity-relationship diagram. In *Proceedings of the ACM tenth international workshop on Data warehousing and OLAP*, pages 9–16.

TableauSoftware. 2014. Fast Analytics and Rapid-fire Business Intelligence from Tableau Software - French.

Olivier Teste. 2009. *Modélisation et manipulation des systèmes OLAP: de l'intégration des documents à l'usager*. Ph.D. thesis, Université Paul Sabatier-Toulouse III.

Riccardo Torlone. 2003. Multidimensional Models. *Multidimensional databases: problems and solutions*:69.

Ronan Tournier. 2007. *Analyse en ligne (OLAP) de documents*. Ph.D. thesis, Université Paul Sabatier-Toulouse III.

J. Trujillo, S. Lujan-Mora, and I. Y. Song. 2004. Applying UML and XML for designing and interchanging information for data warehouses and OLAP applications. *Journal of Database Management (JDM)*, pages 41–72.

Frank S.C. Tseng and Annie Y.H. Chou. 2006. The concept of document warehousing for multi-dimensional modeling of textual-based business intelligence. *Decision Support Systems*, pages 727–744.

P. Vassiliadis. 2009. A survey of Extract–transform–Load technology. *International Journal of Data Warehousing and Mining (IJDWM)*, pages 1–27.

Boris Vrdoljak, Marko Banek, and Zoran Skočir. 2006. Integrating XML sources into a data warehouse. *Data Engineering Issues in E-Commerce and Services*:133–142.

Xuepeng Yin and Torben Bach Pedersen. 2004. Evaluating XML-extended OLAP queries based on a physical algebra. In *Proceedings of the 7th ACM international workshop on Data warehousing and OLAP*, pages 73–82.

Chun-Sik Yoo, Seon-Mi Woo, and Yong-Sung Kim. 2005. Unification of XML DTD for XML Documents with Similar Structure. In Osvaldo Gervasi, Marina L. Gavrilova, Vipin Kumar, Antonio Laganà, Heow Pueh Lee, Youngsong Mun, David Taniar, and Chih Jeng Kenneth Tan, editors, *Computational Science and Its Applications – ICCSA 2005*, Lecture Notes in Computer Science, pages 954–963.

Leopoldo Zepeda, Matilde Celma, and Ramon Zatarain. 2008. A mixed approach for data warehouse conceptual design with MDA. In *Computational Science and Its Applications– ICCSA 2008*, pages 1204–1217.