

Intégration de bases de données: Panorama des problèmes et des approches

Christine Parent - Stefano Spaccapietra

École Polytechnique Fédérale de Lausanne, CH 1015 Lausanne

Tél.: +41 21 693.5211 Fax: +41 21 693.5195

(Parent, Spaccapietra)@di.epfl.ch <http://diwww.epfl.ch/w3lbd>

RÉSUMÉ La conception de nouvelles applications de traitement de données ne se fait plus ex nihilo, mais dans un contexte où la plupart des données nécessaires sont déjà stockées dans des bases de données ou dans des fichiers construits de façon autonome pour les besoins des applications existantes. Pour faciliter leur réutilisation, les données à réutiliser peuvent être redéfinies sous forme d'une base de données virtuelle, assurant l'intégration logique des données sous-jacentes. Cet article fournit une vision globale des problèmes soulevés et des approches qui ont été proposées pour réaliser l'intégration de bases de données.

ABSTRACT. New data processing applications are no more built from scratch. They have to re-use existing data, which are already stored in computer files or databases. In fact, they are most likely to use data from several autonomous sources. To facilitate application development, the data to be re-used should preferably be redefined as a virtual database, providing for the logical unification of the underlying data sets. This unification process is called database integration. This article provides a global picture of the issues raised and the approaches which have been proposed to tackle the problem.

MOTS-CLÉS : bases de données, systèmes distribués, fédérations, hétérogénéité, intégration, sémantique, systèmes répartis.

KEY WORDS : databases, distributed systems, federated systems, heterogeneity, integration, semantics.

1. Introduction

De nos jours, le développement d'une nouvelle application de traitement de données fait le plus souvent appel à des données déjà mémorisées dans l'ordinateur, que ce soit dans des fichiers, ou dans une, voire plusieurs bases de données indépendantes. C'est le cas, notamment, des grandes entreprises où l'usage largement répandu de l'informatique se traduit par un développement indépendant de plusieurs bases de données pour les applications spécifiques de chaque service ou filiale. Or, les structures des entreprises évoluent, surtout dans l'environnement économique actuel. Les frontières entre services ou entre sociétés sont alors susceptibles de bouger, créant de nouveaux centres d'intérêts, demandant de nouvelles applications qui devront être construites avec des données prises ici et là, plus qu'avec de nouvelles données spécifiques. Ainsi, le développement des nouveaux systèmes d'informations repose désormais sur la capacité du système informatique de réaliser l'interopérabilité entre bases de données existantes.

Cette interopérabilité peut être réalisée à trois niveaux:

- au niveau le plus interne dans l'architecture des systèmes, à travers des passerelles (gateways), i.e. des programmes spécifiques qui établissent des connections entre systèmes de telle manière qu'un système donné peut accéder aux données d'un autre système. De telles passerelles sont actuellement disponibles entre plusieurs systèmes de gestion de bases de données (SGBD).
- à un niveau intermédiaire, en installant un système dit multi-bases de données [LIT 90], i.e. une couche logicielle permettant à chaque utilisateur de définir sa vue, persistante, sur un ensemble de bases de données. Ces systèmes garantissent les connections appropriées, établies d'après la définition de la vue. Ils permettent donc l'accès aux données réparties, mais ne prennent pas en charge les contraintes de cohérence entre les différentes sources de données.

- au niveau le plus élevé, en développant, au dessus des systèmes existants, un système global pour fournir le niveau désiré d'intégration des sources de données. Dans cette optique les SGBD répartis [CER 87] ont d'abord été proposés. Ces systèmes s'appuient sur une intégration totale et un contrôle centralisé: toutes les données existantes sont intégrées en une base de données logique unique (la BD répartie), et désormais gérées d'une manière cohérente par une seule autorité globale de contrôle. Cette approche a démontré qu'elle ne satisfaisait pas la demande des entreprises dont le besoin d'accéder à plusieurs sources de données doit être satisfait sans interférer avec l'exploitation normale de ces sources par leurs propriétaires respectifs. Afin de fournir des solutions plus flexibles, les chercheurs ont récemment développé les systèmes de bases de données fédérées (BDF) [SET 90]. Ces systèmes fédérés visent une intégration flexible en fonction de la demande, permettant une coexistence harmonieuse entre le besoin d'intégration de données et celui d'autonomie des sites. Ils donnent à chaque administrateur de données la possibilité de définir le sous-ensemble des données locales mis à disposition des utilisateurs du système fédéré. Ces sous-ensembles sont intégrés en une (ou plusieurs) bases de données virtuelles, appelées BDF. L'intégration, l'importation de données dans les BDF et l'exportation vers les BDF sont gérées par le système fédéré, dont le rôle caractéristique est de faire respecter les accords de coopération (en termes de sémantique de données, de règles d'accès, de gestions des copies, etc....) établis par les administrateurs.

Alors que les passerelles et les systèmes multi-bases de données fournissent aux utilisateurs un langage d'accès multi-bases de données (habituellement de type SQL), sans aucune tentative d'unifier les sémantiques des données des différentes sources, les systèmes de bases de données distribuées ou fédérées basent leurs services sur une description intégrée des données qu'ils gèrent. Ceci permet à leurs utilisateurs d'accéder aux données comme sur une base de données centralisée, sans avoir à s'inquiéter de la localisation physique des données accédées ni du type de SGBD qui les gère localement. Cet avantage explique pourquoi chercheurs et entreprises sont très intéressés par l'approche fédérée. Néanmoins, plusieurs problèmes restent à résoudre avant que cette approche devienne commercialisable. Ni les aspects de conception (comment construire une vision commune des données partagées), ni les aspects système (où les techniques doivent être adaptées à un environnement distribué) ne sont complètement résolus. Les recherches actuelles en conception portent notamment sur l'intégration des schémas et des bases, le travail coopératif, l'évolution des schémas et des bases; coté système, elles portent sur les nouveaux types de transaction (transactions longues, transactions emboîtées, ...), le traitement des requêtes, la sécurité des données, etc.

Le coeur du problème de conception est le processus d'intégration des bases de données. Il consiste à prendre en entrée un ensemble de bases de données (schémas et populations), et à produire en sortie une description unifiée des schémas initiaux (le schéma intégré) et les règles de traduction (mapping) qui vont permettre l'accès aux données existantes à partir du schéma intégré.

L'intégration de bases de données est un problème complexe. Un nombre considérable d'articles en ont étudié différents aspects: il en résulte une multitude de contributions techniques, quelques méthodologies et quelques prototypes. Le but de cet article est de donner une image claire de ces approches, de montrer ce que les solutions actuelles peuvent réaliser et de cerner le travail restant à faire. Nous nous focalisons sur les concepts, le principe des solutions et des alternatives, en laissant de coté les considérations techniques détaillées pour lesquelles nous invitons le lecteur à consulter les références. Les arguments sont illustrés sur un exemple académique. La présentation est organisée selon la séquence chronologique des actions qui composent le processus d'intégration de bases de données. Nous identifions trois étapes principales dans ce processus (cf. figure 1):

- pré-intégration, une étape dans laquelle les schémas en entrée sont transformés de différentes manières pour les rendre plus homogènes (sur les plans sémantique et syntaxique);
- recherche des correspondances, une étape consacrée à l'identification des éléments semblables dans les schémas initiaux et à la description précise de ces liens inter-schémas;
- intégration, l'étape finale qui unifie les types en correspondance en un schéma intégré et produit les règles de traduction associées entre le schéma intégré et les schémas initiaux.

L'article se termine par une discussion des aspects méthodologiques et des indications sur les directions de travail à poursuivre.

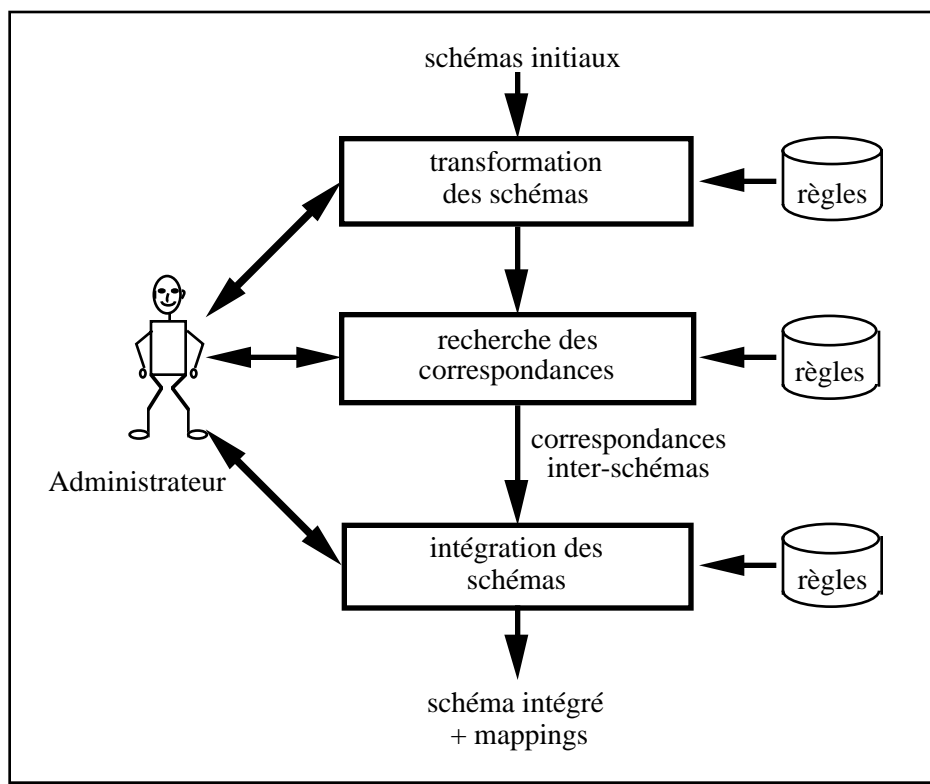


Figure 1. Le processus global d'intégration

2. L'exemple

L'exemple que nous allons utiliser concerne l'intégration des bases de deux bibliothèques d'informatique. La première bibliothèque (schéma S1) utilise un modèle entité-relation étendu. Cette base contient les données sur les articles paru dans des revues ou des conférences. Un objet générique "Publication" représente soit un numéro d'une revue (par exemple, CACM avril 1994) soit les actes d'une conférence (par exemple, VLDB 1993). La deuxième bibliothèque (schéma S2) utilise un modèle orienté-objets et s'intéresse uniquement aux articles publiés lors de conférences. Il y a un objet par type de conférence (par exemple, VLDB). Les actes des conférences (VLDB 1993, VLDB 1994, ...) sont représentés par un attribut multivalué (une valeur par année), où chaque valeur contient un ensemble de références aux articles publiés cette année-là dans cette conférence. Un accord entre les deux bibliothèques nous permet d'assurer que les deux bases de données enregistrent les mêmes conférences.

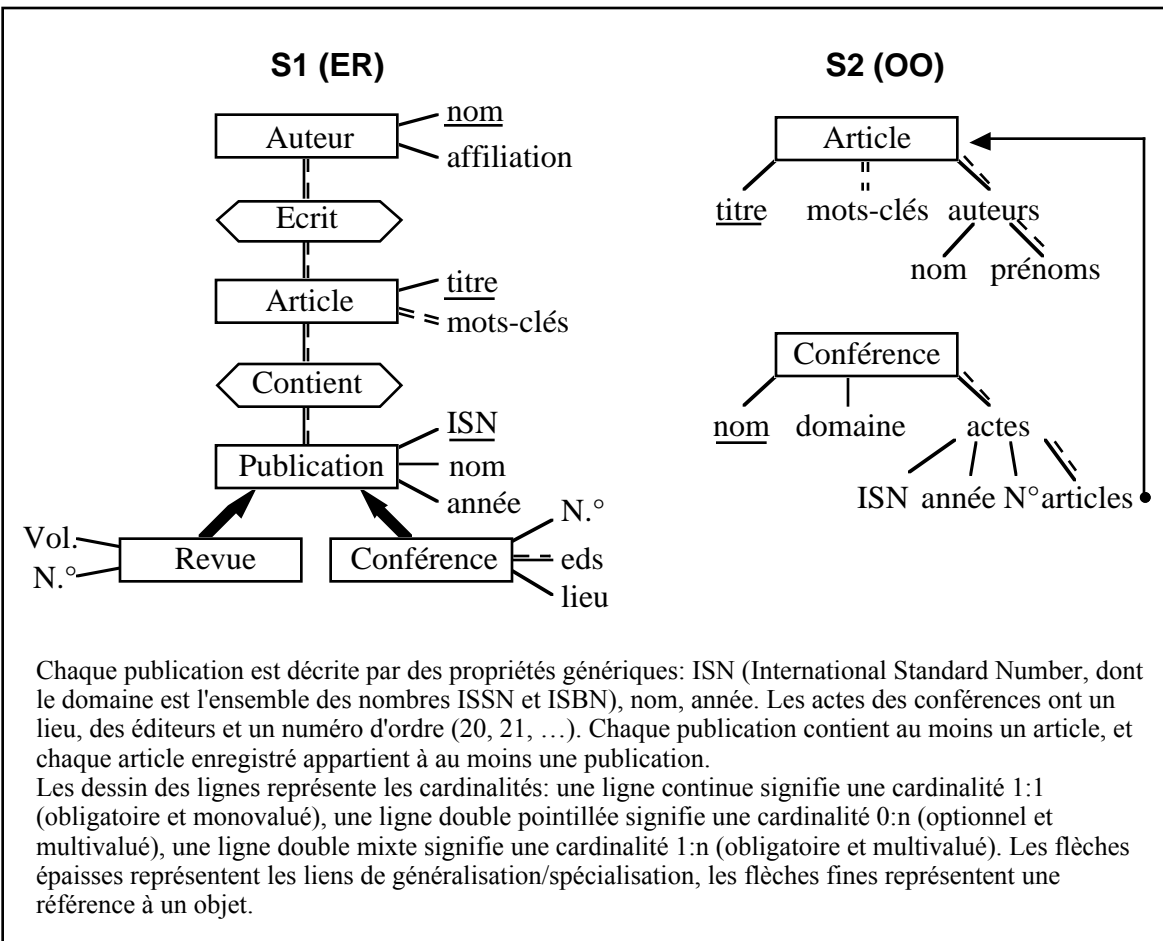


Figure 2. L'exemple des bibliothèques

3. L'étape de pré-intégration

Dans la plupart des cas, les bases de données à intégrer auront été développées indépendamment et seront par conséquent hétérogènes. La différence la plus évidente est lorsque les bases de données existantes ont été installées avec des SGBD utilisant des modèles de données différents (relationnel, CODASYL, orienté- objets, ...). Mais, même en admettant que toutes les bases de données soient converties dans un même modèle, des différences subsisteront dans la richesse de la description obtenue. En effet, certains modèles de données possèdent dans leurs concepts plus de sémantique que d'autres modèles. Par exemple, un schéma entité-association utilise des concepts séparés pour les entités et pour les associations, tandis que le schéma relationnel équivalent va décrire les mêmes données sans faire de distinction explicite entre entités et associations. Pour parvenir à un même niveau de compréhension des deux schémas, on doit être à même d'identifier, dans le schéma relationnel, quelles relations décrivent des entités et quelles relations décrivent des associations. Une autre cause d'hétérogénéité réside dans le non-déterminisme du processus de modélisation. Deux concepteurs qui doivent représenter la même réalité avec le même modèle de données vont inévitablement créer deux schémas différents. Nous discutons ci-dessous des manières d'aborder ces trois problèmes.

3.1. Hétérogénéité des modèles de données

Le problème ici est la transformation des structures de données et des opérations d'un SGBD dans les structures de données et opérations d'un autre SGBD. La grande majorité des chercheurs en intégration supposent que les schémas initiaux sont tous exprimés dans un même modèle de données, appelé modèle commun, sur lequel le logiciel d'intégration est construit. L'étape de traduction devient un pré-requis indispensable à l'intégration et est traitée comme un problème à part. Les traductions requises sont celles entre les modèles de données locaux et le modèle commun. Malheureusement, il existe peu d'outils

capables d'effectuer ces traductions automatiquement (sauf pour passer d'un schéma entité-association à un schéma relationnel). Les développements en cours se focalisent sur les traductions entre modèles orienté-objets et relationnel. Seuls quelques chercheurs ont aussi étudié le problème complémentaire de la traduction des opérations. Ceci est pourtant indispensable pour obtenir un système totalement polyglotte, c'est à dire un système dans lequel chaque participant parle son propre langage.

L'un des points non résolus reste le choix du modèle de données commun. Typiquement, deux choix ont leurs partisans. La majorité préfère actuellement l'approche orienté-objets car elle possède tout les concepts sémantiques des autres modèles et que des méthodes peuvent être utilisées pour implanter des règles de traduction spécifiques. Le problème est de trouver lequel des nombreux modèles orienté-objets existants est le meilleur dans ce rôle. Un second problème est que plus un modèle est riche en concepts, plus le processus d'intégration sera complexe puisqu'il devra résoudre les nombreuses divergences dues aux différents choix de modélisation faits par les divers concepteurs. Pour rendre l'intégration plus simple, l'alternative est de choisir un modèle de données avec un minimum de sémantique, où la représentation des données est assurée par des faits élémentaires pour lesquels il n'y a pas d'alternative de modélisation. Les modèles binaires sont alors les meilleurs candidats comme modèle commun.

Plusieurs chercheurs étudient la spécification d'outils génériques, capables de réaliser n'importe quelle traduction. Les outils traditionnels mettent en oeuvre des algorithmes spécifiques traduisant des schémas exprimés dans un modèle M_i en des schémas exprimés dans le modèle commun MC (et vice versa). Ajouter un nouveau modèle de données M_j à la fédération signifie ajouter deux nouveaux algorithmes: M_j à MC et MC à M_j . Les approches génériques utilisent un méta-modèle (i.e. un modèle adapté à la description des modèles de données) comme pivot pour les traductions [URB 91]. Le méta-modèle connaît tout les concepts de modélisation et sait comment transformer des structures de données entre elles (par exemple, comment combiner des tuples plats pour obtenir des tuples imbriqués). Ajouter un nouveau modèle M_j signifie dans ce cas décrire les concepts de M_j avec les concepts du méta-modèle. Par exemple, une relation du modèle relationnel sera décrite comme un tuple plat d'attributs à valeurs atomiques. Traduire un schéma d'un modèle M_i dans un modèle M_j se ramène à un processus de restructuration de données à l'intérieur de la base de données du méta-modèle.

3.2. Hétérogénéité des puissances d'expression

Les traductions soulèvent des problèmes difficiles et il est peu probable qu'elles puissent être totalement automatisées. En effet, les modèles de données ne peuvent exprimer toute la sémantique du monde réel. L'incomplétude de leurs spécifications conduit à des ambiguïtés sur l'interprétation d'un schéma. Des interactions avec les administrateurs de données sont nécessaires pour résoudre ces ambiguïtés, et acquérir ainsi des informations supplémentaires sur la sémantique d'un schéma: ce processus est appelé enrichissement sémantique.

Des problèmes importants surviennent lorsqu'il s'agit de comprendre un schéma pauvre en sémantique. Le cas le plus difficile se présente lorsque les données sont dans des fichiers, mais les bases de données relationnelles posent également un sérieux défi pour leur ré-interprétation avec des concepts plus sophistiqués (par exemple, avec les concepts orienté-objets). Des techniques de rétro-ingénierie sont actuellement à l'étude pour permettre de distinguer parmi les relations d'un schéma celles qui représentent des objets et leurs attributs (et les regrouper), celles qui représentent des liens d'association, et celles qui représentent des liens de généralisation/spécialisation. Ce processus d'analyse se base sur toutes les informations que peut donner le schéma: clés primaires, clés candidates, clés externes et dépendances. Mais les anciens systèmes relationnels mémorisent uniquement la description des relations et de leurs attributs, sans aucune autre indication. De plus, les relations dans les bases de données existantes ne sont pas nécessairement normalisées. Les techniques de rétro-ingénierie doivent deviner l'information sur les clés et sur les dépendances à partir de l'analyse du schéma, mais aussi des index, des requêtes et des occurrences. Les hypothèses formulées doivent être confirmées par l'administrateur. Par exemple, des conditions de jointure dans les requêtes peuvent indiquer, mais non affirmer, l'existence d'une clé externe; l'utilisation de la clause `DISTINCT` dans une requête SQL peut conduire à la conclusion que l'attribut extrait n'est pas une clé primaire [AND 94].

L'enrichissement sémantique est typiquement un processus de décision humain, utilisé pour fournir plus d'information soit sur la manière dont un schéma décrit la réalité, soit sur le schéma lui-même. Un

exemple du premier type d'enrichissement est la définition explicite de la sémantique d'un objet: par exemple, définir "Employé" comme "les personnes qui ont un contrat de travail en cours avec l'entreprise". Un exemple du second type est de spécifier que l'attribut "directeur" dans la relation "Département" est une clé externe vers la relation "Employé". L'enrichissement sémantique n'est pas spécifique au modèle relationnel. On peut aussi avoir besoin d'étendre des schémas orienté-objets avec des informations sur les cardinalités ou sur les dépendances (qui ne sont pas représentées mais sont nécessaires, par exemple pour une traduction correcte des attributs multivalués). De même, la réingénierie d'un schéma orienté-objets peut promouvoir un attribut en un objet; la question qui se pose immédiatement est de savoir si deux valeurs identiques de cet attribut doivent être traduites en un seul objet ou en deux objets. La décision ne peut pas être automatique, car la réponse dépend de la sémantique du monde réel.

3.3. Hétérogénéité des modélisations

Comme nous l'avons écrit précédemment, le processus de modélisation n'est pas déterministe. Ce relativisme sémantique (i.e. les schémas dépendent du point de vue du concepteur) est le prix à payer pour la richesse sémantique du modèle de données. Les différences peuvent néanmoins être réduites, au niveau de l'organisation en imposant des règles de modélisation (y compris des règles terminologiques), et au niveau technique en appliquant des règles de normalisation. Ces dernières sont bien connues dans le contexte relationnel, mais doivent encore être élaborées pour les bases de données orienté-objets. On peut définir deux familles de règles de normalisation. La première vise à imposer des règles de conception, par exemple:

- un type avec un attribut optionnel doit être remplacé par une structure supertype/sous-type où le sous-type contient nécessairement cet attribut;
- un attribut multivalué doit être remplacé par un objet désigné par un attribut référence;
- un type avec un attribut dont le domaine est énuméré (par exemple, sexe) doit être remplacé par une structure supertype/sous-types. Ces règles imposent une normalisation syntaxique, en ignorant les aspects sémantiques.

L'autre famille de règles met le schéma en conformité avec les dépendances sous-jacentes. Ceci ressemble à la normalisation relationnelle, mais en diffère par le but visé. Les formes normales relationnelles ont pour but de réduire la duplication de données afin d'éviter les anomalies de mise à jour. Les formes normales orienté-objets auxquelles nous nous référons visent à enrichir la sémantique d'un schéma. Un exemple d'une règle de ce type est: s'il existe une dépendance entre deux attributs A et B d'un type d'objet, et que A n'est pas une clé, remplacer ces attributs par un attribut complexe (tuple) composé de A et B. Ces règles de normalisation orienté-objets ne sont pas encore mures, et il reste du travail à faire avant d'arriver à un consensus.

4. L'étape d'identification des correspondances

Lorsque les schémas initiaux ont atteint le niveau de conformité souhaité, l'étape suivante consiste à identifier les éléments communs des bases existantes. Les bases de données contiennent des représentations d'objets du monde réel, avec leurs liens et leurs propriétés. L'intégration de bases de données, toutefois, dépasse les représentations, pour considérer en premier lieu ce qui est représenté plutôt que comment il est représenté. Par exemple, nous voulons savoir si Arthur Durand, représenté dans la base A, est aussi représenté dans la base B, même si les deux occurrences possèdent des attributs complètement différents. Par conséquent, nous disons que **deux bases de données ont quelque chose en commun** si les portions de monde réel qu'elles représentent ont des éléments communs (i.e. une intersection non vide), ou ont des éléments en relation entre eux par un lien d'intérêt pour des applications futures. Ce dernier cas est, par exemple, celui d'une entreprise multi-filiales où la bases de données de chaque filiale enregistre les employés de la filiale et où l'on voudrait dans la base de données intégrée voir un type d'objet Employé représentant la population globale des employés de toutes les filiales.

Nous disons que **deux éléments** (occurrence, valeur, tuple, ...) de deux bases de données **sont en correspondance** s'ils décrivent le même élément du monde réel (objet, lien ou propriété). A chaque fois que cela est possible, les correspondances sont définies, non en extension (au niveau des données, par exemple: *l'article <Intégration de bases de données> de S2 correspond à l'article <Intégration de bases de*

données> de S1), mais en intention (au niveau du type, par exemple: *chaque article de S2 correspond à l'article de S1 qui a le même titre*). La définition intentionnelle d'une correspondance est appelée une **assertion de correspondance inter-schémas**. Le processus d'intégration consiste à trouver ces correspondances entre les éléments des bases de données existantes et pour chaque correspondance à offrir aux utilisateurs de la base fédérée un élément global virtuel (non matérialisé) qui inclut toutes les informations utilisables sur cet élément. La base fédérée enregistre uniquement la définition de l'élément global et des règles de traduction entre l'élément global et ses correspondants locaux. Ces règles font partie du résultat du processus d'intégration.

Les spécifications suivantes doivent être fournies pour définir complètement une assertion de correspondance inter-schémas:

- a/ quels sont les éléments en correspondance
- b/ comment leurs extensions sont liées
- c/ comment les instances correspondantes sont identifiées dans les extensions
- d/ comment leurs représentations sont liées.

Ci-dessous, nous examinons chaque aspect en détail.

a/ quels sont les éléments en correspondance

Les éléments en correspondance peuvent être soit des occurrences ou des valeurs (par exemple les auteurs, les articles), soit des chemins reliant des occurrences et/ou des valeurs (par exemple le chemin liant un auteur et les articles qu'il a écrit). Dans le premier cas, les éléments liés sont simplement désignés par leurs noms qualifiés. Par exemple, une assertion peut relier *S1.Auteur* à *S2.Article.auteurs*, puisque les deux éléments représentent des personnes qui ont écrit un article. Dans le second cas, les chemins sont notés en énumérant les éléments qu'ils traversent. Par exemple, une assertion relie *S1.(Auteur-Ecrit-Article)* à *S2.(auteurs-Article)* pour spécifier que ces deux chemins ont la même sémantique. Ce qui signifie que, si dans S1 l'auteur X a écrit l'article Y et si X et Y appartiennent aussi à S2, alors dans S2 la valeur X apparaît dans l'attribut *auteurs* de l'*Article Y*.

Dans notre exemple, seuls les articles de S1 qui sont publiés dans une conférence apparaissent dans S2. Une définition précise de l'assertion nécessite une spécification précise du sous-ensemble d'articles concernés. Cette spécification peut être faite au moyen d'expressions algébriques: dans ce cas particulier l'opérateur de sélection convient. L'assertion est alors décrite comme suit:

σ ["est un article contenu dans une Publication de type Conférence"] *S1.Article* \equiv *S2.Article*

Les expressions utilisées peuvent inclure des opérateurs plus complexes (union, jointure, ...) pour identifier l'ensemble des objets communs à deux schémas différents. Cette situation est appelée un conflit de fragmentation [DUP 94].

b/ comment leurs extensions potentielles sont liées

Pour construire le schéma intégré, la relation entre les extensions des éléments, dans le monde réel, doit être connue. Etant donné que les extensions sont considérées comme étant des ensembles d'objets du monde réel, chaque assertion décrit la relation ensembliste pertinente: équivalence (\equiv), inclusion (\supseteq), intersection (\cap) ou disjonction (\neq) des deux ensembles. Dans l'exemple, on a: *S1.Article* \supseteq *S2.Article* puisque tous les articles représentés dans S2 le sont dans S1 et que la réciproque est fautive. D'un autre côté, on a: *S1.Conférence* \equiv *S2.Conférence.actes* parce que les deux éléments représentent le même ensemble de conférences malgré le fait que leurs représentations sont structurées différemment dans les deux bases de données.

c/ comment les instances en correspondance sont identifiées.

Quand l'utilisateur cherche à accéder à un objet via le schéma intégré, le système peut avoir à accéder à certaines de ses propriétés dans une base de données et à d'autres dans une autre base. Le système doit donc savoir trouver dans une base l'objet (occurrence ou valeur) correspondant à un objet donné (occurrence ou valeur) d'une autre base. Pour ce faire, chaque assertion doit comprendre une clause de spécification de la correspondance entre instances: nous appelons ceci la clause "Avec Identifiants

Correspondants" (AIC). Dans l'exemple, en supposant qu'il n'y ait pas d'homonymie dans les titres des articles, cela donne:

$S1.Article \supseteq S2.Article$ AIC titre

qui spécifie que les articles des deux bases de données ont un identifiant commun, l'attribut *titre*. C'est le cas le plus simple. Dans le cas général une clause AIC requiert pour chaque base de données un prédicat spécifiant les valeurs/occurrences en correspondance.

d/ comment les représentations sont liées

Les représentations des éléments en correspondance possèdent généralement des propriétés communes, au delà de celles utilisées pour leur identification. $S1.Conférence$ et $S2.Conférence.actes$, par exemple, partagent, en plus de l'identifiant *ISN*, les propriétés *année* et *N°*. Pour éviter la duplication des propriétés dans le schéma intégré, des assertions décrivent de telles correspondances en utilisant une clause "Avec Attributs Correspondants" (AAC). Ainsi, l'assertion de correspondance interschéma s'écrit:

$S1.Conférence \equiv S2.Conférence.actes$ AIC *ISN* AAC *année, N°*

Le format général pour décrire une correspondance entre deux propriétés A and B est: $f(A) R g(B)$, où R est une relation ensembliste ($\equiv, \supseteq, \cap, \neq$) et f et g sont deux fonctions utilisées, si nécessaire, pour résoudre un conflit de représentation. La sémantique de la clause AAC est que, si X et Y sont les éléments vérifiant l'assertion de correspondance, les ensembles de valeurs X.A and Y.B (éventuellement convertis par les fonctions f et g, respectivement) sont liés par la relation R.

4.1. Cohérence des correspondances

Étant donné un ensemble d'assertions entre deux bases de données, il convient de tester s'il est cohérent et minimum. Supposons, par exemple, qu'un schéma contienne un lien de généralisation entre A et B (A est-un B) et que l'autre contienne un lien de même type entre C et D (C est-un D). Un exemple d'assertions de correspondance incohérentes est: $A \equiv D, B \equiv C$. Les deux ne peuvent être simultanément vraies. Certaines assertions sont déductibles d'autres. Par exemple, si, pour les mêmes schémas, la correspondance $A \equiv C$ est déclarée, les correspondances $B \supseteq C$ et $D \supseteq A$ peuvent alors être déduites. Aussi, seules les assertions qui expriment des correspondances non dérivables doivent être explicitées.

Un ensemble approprié d'assertions pour l'exemple des bibliothèques est décrit en figure 3. La figure 4 décrit le schéma intégré construit en prenant en compte ces assertions.

Correspondances S1 / S2 :
1 Auteur \supseteq auteurs AIC nom
2 Article \supseteq Article AIC titre AAC mots-clés
3 Conférence \equiv actes AIC ISN AAC année, N°
4 Publication.nom \supseteq Conférence AIC nom
5 Auteur—Écrit—Article \equiv auteurs—Article
6 Conférence—Publication—Contient—Article \equiv actes—articles:Article
7 Publication.nom—Publication—Conférence \equiv Conférence—actes

Figure 3. L'ensemble des assertions de correspondance pour l'exemple bibliothèques

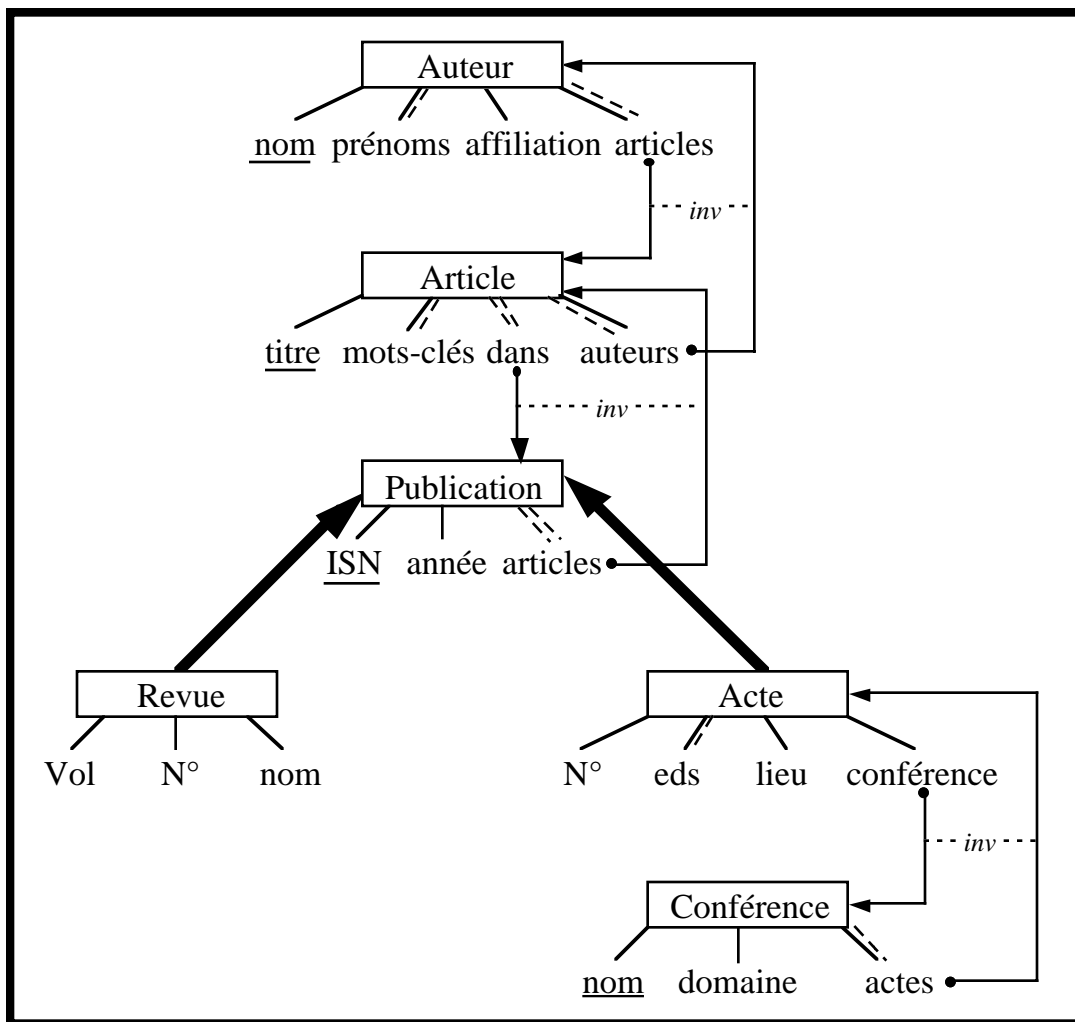


Figure 4. Un schéma intégré orienté-objets pour l'exemple bibliothèques

4.2. Recherche des correspondances

Dans les cas où l'intégration porte sur un nombre important de schémas, comme dans les cas de schémas avec un nombre important de types d'objets, l'identification de toutes les correspondances utiles est loin d'être triviale. Des efforts significatifs ont été faits pour définir des outils pour l'identification automatique des correspondances plausibles [GOT 92]. Ces outils mesurent la similitude entre deux éléments de schémas en recherchant des caractéristiques identiques ou similaires: noms, identifiants, composants, propriétés, attributs (noms, domaines, contraintes), méthodes. Le calcul du taux des similitudes par rapport aux divergences donne une évaluation de la probabilité d'une correspondance. L'étape finale est l'interaction avec l'administrateur des données pour confirmer ou infirmer les correspondances et pour obtenir les informations nécessaires à la définition complète des assertions (en particulier les relations ensemblistes entre les extensions).

La plupart des outils n'analysent que les aspects syntaxiques et structurels, et sont par conséquent confrontés aux problèmes usuels des synonymes et homonymes. Pour limiter ces problèmes certains outils ont recours à des bases terminologiques qui permettent de décrire les termes utilisés dans le domaine de l'application et les liens qui peuvent exister entre eux [FAN 92].

5. Les conflits: taxonomie et solutions

Une fois que les correspondances ont été décrites, l'intégration proprement dite peut commencer. Chaque assertion est analysée pour déterminer quelle est la représentation des éléments en correspondance qui doit être incluse dans le schéma intégré et pour définir quelles sont les règles de traduction entre le schéma intégré et les schémas initiaux. Ces règles seront utilisées par l'évaluateur de requêtes fédérées pour transformer chaque requête définie sur le schéma intégré en un ensemble de

requêtes locales qui sont soumises aux SGBD locaux pour récupérer toutes les informations locales qui permettent de construire l'information souhaitée.

Lorsqu'une assertion décrit les types en correspondance comme étant identiques (i.e. ils ont la même représentation et des populations équivalentes) leur intégration est évidente: le type intégré est identique aux types en entrée et le mapping est l'identité. C'est une simple réplication du type. La plupart du temps, toutefois, les types en correspondance vont présenter des différences, que ce soit dans leurs représentations ou dans leurs populations. Cette situation est appelée un conflit. Plusieurs taxonomies des conflits ont été proposées. Une taxonomie particulièrement détaillée peut être trouvée dans [SET 92]. Pour cet article nous utilisons une version plus simple, décrite dans la table 1.

Table 1 Une taxonomie des conflits entre deux types en correspondance

Classification	les populations du monde réel représentées par les deux types sont différentes
Description	les types ont des ensembles différents de propriétés et/ou leurs propriétés sont représentées de manière différente.
Structure	les concepts utilisés pour décrire les types sont différents
Hétérogénéité	les modèles de données utilisés sont différents
Méta-données	la correspondance relie un type à un méta-type (un ensemble de noms de types)
Données	des instances en correspondance ont des valeurs différentes pour des propriétés en correspondance.

Beaucoup de propositions ont été faites pour la résolution des conflits de classification et de description [KIM 93]. Les conflits de méta-données ont été abordés plus récemment, notamment par [SAL 92]. Les conflits de structure, d'hétérogénéité et de données ont fait l'objet de quelques travaux [SPA 91, SHE 94]. Il n'existe pas de méthode d'intégration complète qui résolve chaque type de conflit, encore moins de méthode qui puisse être automatisée. Ci-après, nous discutons brièvement les différents types de conflits, indiquant les solutions existantes et les problèmes non résolus.

Une autre faiblesse de la recherche actuelle est de ne pas spécifier explicitement le but précis recherché dans cette étape d'intégration proprement dite. Il existe, en effet, plusieurs objectifs possibles, produisant des résultats sensiblement différents. Par exemple, on peut rechercher la simplicité, et donc produire un schéma intégré avec un nombre minimal de types d'objets, de façon à ce que le schéma reste lisible avec un effort raisonnable. On peut rechercher la complétude, au sens où chaque élément des schémas initiaux apparaît dans le schéma intégré. Ceci permet aux administrateurs locaux de retrouver facilement leurs données dans le schéma résultat, et facilite la mise à jour lorsque les schémas des bases locales évoluent. On peut aussi vouloir être exhaustif, en faisant apparaître dans le résultat tous les types d'objets dérivables à partir des types initiaux (ceux qui existent localement et ceux que l'on peut en déduire par leur intersection/union). Ceci alourdit le schéma intégré mais, inversement, prépare l'intégration de nouvelles bases de données qui viendraient s'ajouter à la fédération existante.

5.1. Conflits de classification

On a un conflit de classification lorsque les types en correspondance décrivent des ensembles différents, mais sémantiquement liés, d'éléments du monde réel. Dans l'exemple des bibliothèques, S1 décrit les articles de revues et conférences, alors que S2 ne décrit que les articles de conférences. Un conflit de classification se traduit dans une assertion par l'utilisation d'une relation ensembliste autre que l'équivalence (\supseteq , \cap , \neq).

Une solution standard pour ces conflits est d'inclure dans le schéma intégré la hiérarchie de généralisation/spécialisation appropriée, telle que décrite dans la table 2 [LAR 89, GOT 92]. Cette solution vise l'objectif de complétude: les types initiaux sont copiés dans le schéma intégré et reliés par des liens IS-A. Si nécessaire, un sous type (ou supertype) commun est ajouté pour réaliser la connectivité de la hiérarchie de généralisation/spécialisation. Ceci a pour effet de réduire les traductions entre schéma intégré et schémas initiaux à des simples fonctions d'identité.

Une solution différente, basée sur l'objectif de simplicité, est d'insérer dans le schéma intégré un seul type d'objet (voir table 2b Technique de fusion). Le schéma intégré décrit alors l'union des extensions. Les règles de traduction utilisent l'opérateur de sélection pour dériver les populations initiales de la population globale.

Enfin, l'objectif opposé, à savoir créer un schéma exhaustif (voir table 2b Technique exhaustive), conduit à inclure dans le schéma intégré les types initiaux, plus leur union (un super-type) ainsi que leur intersection et les compléments de l'intersection (trois sous-types).

Table 2: Résolution des conflits de classification		
Conflit	Schéma intégré	
$E1 \supseteq E2$		
$E1 \cap E2$		
$E1 \neq E2$		

2a: la solution standard: préservation des types locaux

Conflit	Schéma intégré	
	Technique de fusion	Technique exhaustive
$E1 \supseteq E2$		
$E1 \cap E2$		
$E1 \neq E2$		

2b: autres solutions

Les stratégies pour les conflits de classification ont parfois été étendues à l'intégration de hiérarchies de généralisation mises en correspondance par plusieurs assertions. Les deux hiérarchies sont fusionnées en déterminant, pour chaque classe d'une hiérarchie, la place que cette classe doit prendre dans l'autre hiérarchie. Le placement est basé sur la sémantique d'inclusion des populations [MAN 88, RED 94]. Il est par contre basé sur la sémantique d'héritage dans les méthodologies d'intégration de vues (où il n'existe pas de populations à intégrer). Le processus de placement obtient de meilleurs résultats si l'on connaît les critères de spécialisation des sous-classes existants dans les schémas initiaux.

5.2. *Conflits structurels*

Un conflit structurel survient lorsque les éléments en correspondance sont décrits par des concepts de niveaux de représentation différents ou soumis à des contraintes différentes: par exemple, une classe d'objets et un attribut, ou un type d'entité et un type d'association. Dans l'exemple des bibliothèques, l'assertion sur les auteurs et celle sur les conférences montrent des conflits structurels. Ce type de conflit a été peu étudié dans la littérature et uniquement pour des modèles de données particuliers [KIM 93, SHO 91].

La solution des conflits structurels [SPA 91] part du principe que le schéma intégré doit pouvoir décrire la population des deux éléments en conflit. L'élément intégré doit donc refléter les possibilités des éléments initiaux: en termes de capacité à décrire l'information (pour laquelle on adopte la borne supérieure minimale) mais aussi en termes des contraintes implicites et explicites attachées aux éléments (en adoptant la borne inférieure maximale). La capacité d'un concept nous dit quelle combinaison de identité/valeur/liens le concept modélise. Par exemple, un attribut dans les modèles orienté-objets et relationnel modélise soit une valeur soit un lien, mais un attribut dans un modèle entité-association modélise uniquement une valeur. Une relation du modèle relationnel modélise une valeur et éventuellement des liens (par les clés externes), mais pas l'identité. Si une assertion entre schémas relationnels met en correspondance une relation (valeur+liens) et un attribut (valeur ou lien), l'élément correspondant dans le schéma intégré sera une relation.

Parmi les contraintes à prendre en compte, on trouve typiquement les contraintes de cardinalité et les dépendances d'existence. Par exemple, l'existence d'un attribut dépend de celle de son propriétaire, mais un objet n'est pas, en général, soumis à des contraintes d'existence. Si, comme c'est le cas pour les auteurs dans notre exemple, une assertion met en correspondance une classe d'objets (S1.Auteur) avec un attribut (S2.Article.auteurs), c'est la classe qui est inscrite dans le schéma intégré (la borne inférieure maximale est dans ce cas l'absence de contraintes).

La table 3 montre le concept à choisir pour le schéma intégré en fonction du type de conflit structurel. Les conflits structurels peuvent être plus complexes que ceux montrés dans cette table. Ils peuvent, par exemple, faire appel à des expressions de jointure [KIM 93, DUP 94]. Une solution générale reste à trouver.

Table 3: Solutions pour les conflits structurels

cas	Schéma1	Schéma2	Schéma intégré
①			
②			
③			
④			

Ces diagrammes sont exprimés sur la base d'un modèle de données générique, où:

- représente un type d'objet (classe, type d'entité, relation, ou type d'article)
- représente une relation entre types d'objets (attribut référence, type de relation, clé externe, ou set CODASYL)
- représente un attribut
- représente une assertion de correspondance de type équivalence

5.3. Conflits Descriptifs

Un conflit descriptif survient dès qu'il y a une différence entre les propriétés des types en correspondance. Ces conflits ont été abondamment étudiés [LAR 89, KIM 93], soit pour proposer une résolution automatique soit pour guider un intégrateur humain. La table 4 résume plusieurs taxonomies existantes.

Table 4: Conflits descriptifs

1. Les types d'objets en correspondance peuvent différer selon leurs:

- noms: homonymes et synonymes
- clés: clé primaire ou clés candidates différentes
- attributs: les ensembles d'attributs sont différents
des attributs en correspondance sont en conflit
(voir point 2)
- méthodes: les ensembles de méthodes sont différents
des méthodes en correspondance diffèrent
(voir point 3)
- contraintes d'intégrité
- droits d'accès

2. Les attributs en correspondance peuvent différer selon leurs:

- noms: homonymes et synonymes
- portées: locale à une base de données ou globale (i.e., conserve sa signification dans la base intégrée)
- structures:
 - simple ou complexe (i.e. composée d'autres attributs)
 - cardinalités: monovalué ou multivalué, facultatif ou obligatoire
 - constructeurs multivalués différents (ensemble, multi-ensemble, liste, tableau)
- types de données: différentes échelles, unités, valeurs par défaut, opérations
- contraintes d'intégrité
- droits d'accès

3. Les méthodes en correspondance peuvent différer selon leurs:

- noms: homonymes et synonymes
- signatures: nombre de paramètres, types de paramètres, résultats

Les solutions traditionnelles sont:

- conflit de nom: renommer en utilisant l'un des noms (en cas de synonyme) ou en utilisant un préfixe (en cas d'homonyme). Le préfixe est généralement le nom de la base de données source.
- conflit de type de données: une fonction de conversion est utilisée pour passer du domaine d'un attribut au domaine de l'autre attribut.
- conflit de clé: une fonction de conversion est utilisée pour associer à chaque valeur d'une clé une valeur de l'autre clé. La fonction doit être une application bijective (1-1) pour être opérationnelle.
- contraintes d'intégrité: utiliser l'approche de la borne inférieure maximale.
- structures d'attributs: aucune solution vraiment disponible. Les méthodologies actuelles traitent les attributs simples monovalués, rarement les attributs multivalués [LAR 89], et ignorent les attributs complexes (i.e. les attributs composés d'autres attributs).

5.4. Conflits d'hétérogénéité des modèle de données

Les méthodologies d'intégration actuelles sont monomodèle. Elles n'intègrent que des schémas qui sont exprimés dans leur propre modèle de données. Comme nous l'avons dit précédemment, les schémas qui ne sont pas exprimés dans ce modèle de données doivent être traduits pendant la phase de pré-intégration, avant d'être pris en considération pour l'intégration.

Une approche différents a été proposée dans [SPA 91]. Les auteurs montrent que les problèmes et les solutions pour chaque type de conflit sont à la base les mêmes quelque soit le modèle de données. Il devrait ainsi être possible d'identifier l'ensemble des règles d'intégration fondamentales qui sont nécessaires et de définir, pour chaque modèle de données spécifique, comment chaque règle s'applique en reformulant la règle selon les particularités du modèle considéré. On peut alors construire un outil capable d'assurer directement l'intégration de schémas hétérogènes et de produire un schéma intégré dans n'importe quel modèle.

Les logiques d'ordre supérieur ont aussi été proposées comme formalismes capables de résoudre tous les types de conflits, y compris d'hétérogénéité [LAK 93]. Un langage de ce type permet à l'utilisateur de définir lui-même le schéma intégré à partir des schémas initiaux hétérogènes sans avoir à les traduire.

5.5. Conflits données/méta-données

Les conflits de ce type surviennent lorsqu'une donnée dans une base est en correspondance avec une méta-donnée (le nom d'un type) dans le schéma d'une autre base. Ces conflits ont été illustrés en utilisant un exemple relationnel de gestion de bourse (figure 5), dans lequel une valeur pour action dans BD1 correspond à un nom d'attribut dans BD2 et à un nom de relation dans BD3.

BD1: Cours (<u>date</u> , <u>action</u> , <u>cotation</u>)				
	951001	IBM	77.72	
	951002	IBM	79.23	
	
	951001	HP	60.02	
	951002	HP	61.45	
	
BD2: Cours (<u>date</u> , <u>IBM</u> , <u>HP</u> , ...)				
	951001	77.72	60.02	...
	951002	79.23	61.45	...

BD3: IBM (<u>date</u> , <u>cotation</u>) HP (<u>date</u> , <u>cotation</u>) 				
	951001	77.72	951001	60.02
	951002	79.23	951002	61.45

Figure 5. Un exemple de conflit données/méta-données

Les langages relationnels traditionnels ne peuvent pas transformer un nom d'attribut ou de relation en valeur, et vice versa. Avec ces langages, on peut facilement passer de BD2 à BD3 et inversement, de même que l'on passe de BD1 à BD2 ou à BD3. Par contre on ne peut pas passer de BD2 ou BD3 à BD1. Ces conflits peuvent être résolus si le langage de correspondance permet la manipulation simultanée des données et des méta-données. L'algèbre relationnelle étendue de [SAL 92] et la logique d'ordre supérieure de [LAK 93] sont des exemples de tels langages.

Ces conflits peuvent aussi être éliminés en utilisant un modèle de données qui ne permet pas de donner des noms significatifs aux éléments d'un schéma [MIL 93]. Plus probablement, les approches orienté-objets fourniront des opérations de transformation de schéma pour effectuer toutes les mises en correspondance nécessaires, en particulier partitionner une classe en sous-classes selon la valeur d'un attribut discriminant (passage de BD1 à BD3), créer une super-classe commune à plusieurs classes avec un nouvel attribut discriminant (passage de BD3 à BD1), etc.

5.6. Conflits de données

Ce dernier type de conflit survient au niveau des instances lorsque des occurrences en correspondance ont des valeurs en conflit pour des attributs en correspondance. Par exemple, un même article est stocké dans les deux bases de données des bibliothèques avec des mots clés différents. Les causes d'un conflit de données peuvent être: erreurs de saisie, sources d'information différentes, versions différentes, mises à jours différées, etc.

Les conflits de données sont normalement détectés lors de l'exécution d'une requête d'accès aux données. Le système peut se contenter de signaler l'erreur à l'utilisateur, mais il peut aussi appliquer une heuristique pour produire une valeur appropriée. Les heuristiques courantes sont: choisir la valeur dans la base qui est la plus "crédible", unifier les valeurs en conflit d'une certaine façon (agrégation des valeurs simples, union des ensembles de valeurs). Une autre possibilité est de fournir aux utilisateurs un langage pour manipuler eux-mêmes toutes les valeurs candidates, l'ensemble des valeurs possibles étant la réponse à la requête chaque fois que le conflit survient [TSE 93].

6. Stratégies d'intégration

Nous avons jusqu'à présent analysé les principes et les techniques de l'intégration de schémas. Dans ce chapitre nous abordons le problème soulevé par la possibilité d'adopter différentes stratégies pour diriger le processus d'intégration. Plusieurs choix sont possibles:

- manuel ou semi-automatique. Puisque seul l'administrateur connaît de façon certaine la sémantique des données, les stratégies manuelles laissent l'administrateur diriger le processus d'intégration. Ces méthodes fournissent un langage de manipulation de schémas que l'administrateur va utiliser pour construire lui-même (si le langage est procédural) ou spécifier (si le langage est déclaratif) le schéma intégré. Les langages procéduraux offrent des primitives de transformation de schéma qui permettent de restructurer les schémas initiaux jusqu'à ce qu'ils puissent être fusionnés en un seul schéma. Le système génère automatiquement les règles de traduction entre les schémas initiaux et le schéma intégré [MOT 87]. Les langages déclaratifs sont plus faciles à utiliser, mais l'établissement des règles de traduction par le système est plus délicat. Les stratégies manuelles supposent que l'administrateur sache comment structurer le schéma intégré qui doit être créé. Inversement, les stratégies semi-automatiques ont pour but de construire un outil qui effectue automatiquement l'intégration lorsque les assertions de correspondance ont été définies. L'outil détermine aussi automatiquement les règles de traduction [SPA 91]. L'administrateur est seulement impliqué dans l'identification des assertions de correspondance, et éventuellement dans le choix de la stratégie d'intégration (voir table 2).
- en un coup ou par incréments. Dans les stratégies en un coup, l'intégration est vue comme un processus atomique qui produit un résultat final. Elles ne se préoccupent pas des possibilités d'utilisation des résultats intermédiaires. Cette vision est mise en défaut par la complexité des applications réelles, pour lesquelles l'intégration peut être un processus très long et pénible, alors qu'il est exclu de suspendre l'exploitation des bases en attendant la fin de l'intégration. Les stratégies par incréments permettent d'effectuer graduellement l'intégration tout en continuant à utiliser les systèmes existants. Elles permettent d'intégrer deux occurrences en correspondance ou deux types en correspondance chaque fois qu'une telle correspondance est détectée et semble intéressante. En d'autres mots, l'intégration peut être effectuée petit à petit au fil des jours [SCH 92].
- ex nihilo ou à partir d'un thesaurus. La plupart des méthodes d'intégration supposent qu'un nouveau schéma intégré est construit directement à partir des schémas initiaux. Si une nouvelle base doit être intégrée après que le schéma intégré ait été construit, le processus est reconduit avec comme schémas initiaux le nouveau schéma et le schéma intégré existant. Une alternative consiste à donner un rôle particulier au schéma intégré existant, celui de thesaurus de toute la connaissance de l'entreprise. Si des nouvelles bases sont à prendre en compte, elles sont utilisées uniquement pour enrichir le thesaurus avec les connaissances supplémentaires qu'elles peuvent contenir. En d'autres mots, elles sont absorbées par le thesaurus. En cas de conflit, c'est le thesaurus qui prime sur les nouvelles bases [COL 91].

7. Conclusion

L'intégration de bases de données est certainement une tâche complexe. C'est pourtant une tâche que les entreprises peuvent difficilement éviter si elle veulent mettre en route de nouvelles applications ou réorganiser le système d'information existant pour une meilleure productivité. Nous avons montré dans cet article que les problèmes à résoudre sont bien identifiables et que des (voire plusieurs) solutions existent pour nombre d'entre-eux. Nous nous sommes attachés à décrire les concepts et techniques fondamentaux, en insistant sur les alternatives et les critères de choix. Des informations plus détaillées peuvent être facilement trouvées dans la littérature.

Bien que la recherche ait été active dans ce domaine depuis vingt ans, avec une croissance significative des efforts ces dernières années, plusieurs problèmes importants doivent encore être résolus. A titre d'exemple nous citerons: l'intégration des objets complexes (tels qu'on les trouve dans les bases orienté-objets), les correspondances complexes entre plusieurs types (et non plus uniquement entre deux types), la prise en compte des contraintes d'intégrité et des méthodes, l'intégration directe de bases hétérogènes. Des études théoriques sont nécessaires pour valider les règles d'intégration et leurs propriétés (commutativité, associativité, ...). Il est donc important que l'effort pour résoudre les problèmes d'intégration soit poursuivi et que les méthodes proposées puissent être évaluées lors d'expérimentations avec des applications réelles.

Références

[AND 94] Andersson M. Extracting an Entity Relationship Schema from a Relational Database Through Reverse Engineering. In *Entity-Relationship Approach - ER'94*, P. Loucopoulos Ed., LNCS 881, Springer-Verlag, 1994, pp. 403-419

[CER 87] Ceri S., Pelagatti G. *Distributed databases: principles & systems*. McGraw-Hill, 1987

[COL 91] Collet C., Huhns M.N., Shen W.-M. Resource Integration Using a Large Knowledge Base in Carnot. *Computer* 24, 12 (Dec. 1991), pp. 55-62

[DUP 94] Dupont Y. Resolving Fragmentation Conflicts in Schema Integration. In *Entity-Relationship Approach - ER'94*, P. Loucopoulos Ed., LNCS 881, Springer-Verlag, 1994, pp. 513-532

[FAN 92] Fankhauser P., Neuhold E.J. Knowledge based integration of heterogeneous databases. In *Proceedings of IFIP DS-5 Conference on Semantics of Interoperable Database Systems* (Nov. 16-20. Lorne, Australia), 1992, pp. 150-170

[GOT 92] Gotthard W., Lockemann P.C., Neufeld A. System-Guided View Integration for Object-Oriented Databases, *IEEE Transactions on Knowledge and Data Engineering* 4, 1 (Feb. 1992), pp. 1-22

[KIM 93] Kim W., Choi I., Gala S., Scheevel M. On Resolving Schematic Heterogeneity in Multidatabase Systems, *Distributed and Parallel Databases* 1, 3, (July 1993), pp. 251-279

[LAK 93] Lakshmanan L.V.S., Sadri F., Subramanian I.N. On the Logical Foundation of Schema Integration and Evolution in Heterogeneous Database Systems. In *Deductive and Object-Oriented Databases*, Ceri S., Tanaka K., Tsur S. (Eds.), LNCS 760, Springer-Verlag, 1993, pp. 81-100

[LAR 89] Larson J.A., Navathe S.B., Elmasri R. A Theory of Attribute Equivalence in Databases with Application to Schema Integration, *IEEE Transactions On Software Engineering* 15, 4, (April 1989), pp. 449-463

[LIT 90] Litwin W., Mark L., Roussopoulos N. Interoperability of multiple autonomous databases, *ACM Computer Surveys* 22, 3 (Sept. 1990), pp. 267-293

- [MAN 88] Mannino M.V., Navathe S.B., Effelsberg W. A Rule-Based Approach for Merging Generalization Hierarchies, *Information Systems* 13, 3, 1988
- [MIL 93] Miller R.J., Ioannidis Y.E., Ramakrishnan R. Understanding Schemas. In *Proceedings of RIDE-IMS'93 Interoperability in Multidatabase Systems* (April 19-20, Vienna, Austria), 1993, pp. 170-173
- [MOT 87] Motro A. Superviews: Virtual integration of multiple databases, *IEEE Transactions On Software Engineering* 13, 7, (July 1987), pp. 785-798
- [RED 94] Reddy M.P., Prasad B.E., Reddy P.G., Gupta A. A Methodology for Integration of Heterogeneous Databases, *IEEE Transactions on Knowledge and Data Engineering* 6, 6, (Dec. 1994), pp. 920-933
- [SAL 92] Saltor F., Castellanos M.G., Garcia-Solaco M. Overcoming Schematic Discrepancies in Interoperable Databases, In *Proceedings of IFIP DS-5 Conference Semantics of Interoperable Databases Systems*, (Nov. 16-20. Lorne, Australia), 1992, pp. 184-198
- [SCH 92] Scholl M.H., Schek H.-J., Tresch M. Object Algebra and Views for Multi-Objectbases. In *Proceedings of International Workshop on Distributed Object Management* (August, Edmonton, Canada), 1992
- [SHE 90] Sheth A., Larson J. Federated database systems for managing distributed, heterogeneous, and autonomous databases, *ACM Computer Surveys* 22, 3 (Sept. 1990), pp. 183-236
- [SHE 92] Sheth A., Kashyap V. So Far (Schematically) yet So Near (Semantically), In *Proceedings of IFIP DS-5 Conference Semantics of Interoperable Databases Systems*, (Nov. 16-20. Lorne, Australia), 1992, pp. 272-301
- [SHE 94] Sheuermann P., Chong E.I. Role-Based Query Processing in Multidatabase Systems. In *Advances in Database Technology - EDBT'94*, Jarke M., Bubenko J. Jeffery K. (Eds.), LNCS 779, Springer-Verlag, 1994, pp. 95-108
- [SHO 91] Shoval P., Zohn S. Binary-relationship integration methodology. *Data & Knowledge Engineering* 6, (1991), pp. 225-250
- [SPA 91] Spaccapietra S., Parent C., Dupont Y. Model Independent Assertions for Integration of Heterogeneous Schemas. *VLDB Journal* 1, 1 (July 1992), pp. 81-126
- [TSE 93] Tseng F.S.C., Chen A.L.P., Yang W.-P. Answering Heterogeneous Database Queries with Degrees of Uncertainty. *Distributed and Parallel Databases* 1, (1993), pp. 281-302
- [URB 91] Urban S.D. A Semantic Framework for Heterogeneous Database Environments In *Proceedings of RIDE-IMS'91 Interoperability in Multidatabase Systems* (April 7-9, Kyoto, Japan), 1991, pp. 156-163