Une approche pour l'échange entre bases de données hétérogènes basée sur des méta-modèles génériques exprimés en langage EXPRESS

Mourad El-Hadj Mimoune, Guy PIERRA**, Yamine AIT-AMEUR***

*Email: ** {pierra, mimoune}@ensma.fr*
Address: LISI-ENSMA
BP 40109 – Site du Futuroscope
86961 FUTUROSCOPE Cedex France
Tel: +33 5 49 49 80 60
Fax: +33 5 49 49 80 64

*Email : * yamine@supaero.fr*
Address: ENSAE-SUPAERO
10 Av E. Belin. BP 4032
31055 Toulouse Cedex 4. France
Tel : +33 5 62 17 80 57
Fax : + 33 5 62 17 83 45

Résumé:

Dans ce travail nous présenterons une approche pour l'échange de données entre bases de données hétérogènes utilisable dans les domaines où les entités et les attributs principaux font l'objet de consensus et peuvent être associés à des identifiants universels.

Notre approche est basée sur l'utilisation de méta-schémas génériques, formalisées dans le langage EXPRESS, et permettant l'échange de n'importe quelle instance de n'importe quel schéma de base de données. Les entités et attributs échangés référencent autant que de besoin les identifiants universels agréés. Elles peuvent également spécialiser les entités et les attributs agréés. La conversion de fichiers d'échange en instructions de définition ou de manipulation de données sur le système receveur peut également être réalisée de façon générique, c'est à dire indépendamment des modèles particuliers échangés. On montre alors l'intérêt du langage EXPRESS pour réaliser directement de tels programmes.

Mots clés : échange de données, intégration de bases de données hétérogènes, modèle de données pivot, fédération des bases de données, PLIB.

1. Introduction

Toute conception de base de données passe par une phase de modélisation conceptuelle, ou sémantique, dans laquelle les méthodes d'analyse et de conception orientée objet tels que UML[13], OMT[6], ou Meurise-objet sont de plus en plus utilisées. Ces méthodes sont toutes basées sur les notions d'entité et d'association auxquelles sont ajoutés des mécanismes d'identification, de généralisation/spécialisation et de polymorphisme. Les modèles, ou schémas conceptuels, obtenus par cette modélisation sont ensuite destinés à être implémentés sur divers SGDB classiques. Coexistent actuellement des systèmes purement relationnels (SGDBR), où la notion d'objet n'est pas présente au sens des langages à objet, des systèmes supportant quelques notions du paradigme objet (tels que POSTGRES ou les SGBD Relationnel Objet), et, des systèmes où les données sont essentiellement des objets comme dans les systèmes O2 et ObjectStore. Ainsi, même lorsque la même méthode de conception est utilisée pour concevoir une base de données, et pour même des modèles objets voisins, l'implémentation est souvent complètement différente. Cela a pour conséquence de produire des bases de données hétérogènes entre lesquelles l'échange de données est difficile voir impossible.

La difficulté d'échange entre bases de données peut résulter de deux choses: la différence des domaines d'application couverts d'une part, et la différence des modélisations pour un même domaine d'application, d'autre part. C'est à ce second cas que nous sous intéressons, et, plus précisément, aux domaines sur les lesquels un consensus existe sur les grandes catégories d'entités qui doivent être représentées et éventuellement leurs attributs fondamentaux. Cette situation se produit souvent dans le cas d'objets du monde réel (exemple : les produits dans le domaine de composants électroniques pour lesquels la norme IEC 61360-4 définit à la fois une classification de référence et les attributs pertinents) ou bien pour des entités intervenant dans des processus stables (exemples: clients, fournisseurs, commandes, factures, produits dans une relation commerciale pour lesquels des dictionnaires similaires sont, ou pourraient être, élaborés). Nous faisons également l'hypothèse que les acteurs de l'échange souhaitent rendre celui-ci possible, par exemple en faisant référence à des dictionnaires

"standards" lorsque ceux-ci existent, mais qu'ils ne souhaitent pas être forcés ni d'utiliser exactement le même modèle ni de mettre en œuvre le même système de gestion de bases de données.

L'approche que nous proposons pour l'échange de données entre bases de données hétérogènes consiste alors à :

- définir un mécanisme d'identification universel pour les concepts essentiels (entités, attributs ou relations) correspondants au domaine de l'échange,
- utiliser un méta-schéma générique, exprimé en EXPRESS, pour la représentation des instances qui doivent être échangées,
- utiliser, si besoin est, c'est à dire si les concepts utilisés dans la base source ne sont pas tous connus de la base cible, un méta-schéma générique, exprimé en EXPRESS, pour échanger le schéma de la base source,
- utiliser le format EXPRESS de fichier d'instances, ou l'API d'accès aux instances EXPRESS, pour réaliser l'échange effectif des données, et, le cas échéant, de modèles,
- exploiter l'une des technologies de mapping disponible en EXPRESS pour réaliser les conversions nécessaires sur les sites émetteurs et fournisseurs.

L'origine de notre intérêt pour le problème d'échange entre bases de données hétérogènes est le besoin de représenter, sous forme informatique, les bibliothèques de composants industriels, telles qu'elles sont décrites naturellement dans les catalogues papiers. Ces descriptions contiennent non seulement des aspects statiques (les propriétés des composants) mais également les aspects dynamiques (par exemple, les fonctions décrivant leur comportement selon leur environnement d'insertion). Le but de nos travaux dans le domaine est de représenter l'ensemble de ces informations sous forme d'un modèle pivot destiné à être utilisé pour l'échange, quels que soient les schémas particuliers utilisés, et les systèmes mis en œuvre, dans chaque environnement particulier.

Notons que le schéma d'un catalogue (i.e. sa structure de classes) dépend du catalogue, de sorte que l'échange portera en général non seulement sur les instances mais également sur le schéma lui-même. Les méta-schémas génériques développés pour ce problème sont en cours de publication sous forme normes ISO connues sous le nom de PLIB. L'objectif de ce papier est à la fois de montrer comment ces méta-schémas génériques peuvent être exploités pour couvrir l'échange de données dans d'autres domaines, et de représenter l'approche que nous avons développé pour réaliser le passage des méta-schémas génériques à un schéma particulier spécifique d'une base de données.

Cet article est structuré de la façon suivante. La section 2 analyse les différentes sources de diversité que l'on peut rencontrer dans des bases de données portant toutes sur le même domaine, considéré selon le même point de vue. Nous présentons sommairement, dans la section 3, le langage de modélisation de données EXPRESS, utilisé pour définir les modèles d'échange de données, et les techniques et outils utilisables pour réaliser des transformations de modèles dans l'univers EXPRESS. Dans la section 4, nous détaillons la modélisation que nous avons développé pour permettre l'échange de données entre bases de données hétérogènes par l'utilisation des méta-schémas générique en langage EXPRESS. Dans la section 5, nous discutons le processus d'échange lui-même, les étapes qu'il compte et les différentes techniques que l'on peut mettre en œuvre. Enfin, nous détaillons dans la section 6 la mise en œuvre que nous proposons pour restaurer les données échangées sur le système receveur.

2. Les différentes dimensions de la diversité

Dans le champ restreint qui nous préoccupe, à savoir bases de données portant sur le même domaine considéré selon le même point de vue, les différents schémas peuvent néanmoins être très divers. Nous analysons ici les différentes dimensions de cette diversité.

Diversité de nomination : Dans chaque base de données, entités (lorsqu'elles sont représentées), tables et attributs sont identifiées par des noms. Des noms différents peuvent être utilisés pour désigner le même concept ou inversement des concepts différents peuvent être désignés par le même nom.

Diversité conceptuelle : Dans le même domaine, et en supposant que les mêmes noms sont utilisés, la structure du modèle conceptuel peut changer. Un modèle peut ne pas présenter de spécialisation particulière par rapport aux concepts et aux noms existants dans l'autre système, il peut au contraire avoir donné lieu a spécialisation. De plus, s'il y a spécialisation, on peut s'être basé sur différents critères pour faire la spécialisation. Par exemple pour une *voiture* on peut spécialiser selon l'origine (*française*, *étrangère*), le type (*voiture de course*, *tout terrain* ou *touristique*), etc. Concernant le choix d'attributs, chaque catégorie

d'utilisateur, et donc chaque schéma s'intéresse à un sous-ensemble particulier des attributs possibles d'une entité.

Diversité structurelle : Dans le même domaine, et en supposant que les mêmes concepts (entités et attributs) sont utilisés, la diversité structurelle ou de représentation est due au choix du schéma final d'implantation de la base de données. La diversité structurelle peut avoir trois causes: le souci d'éviter la redondance, c'est l'aspect normalisation, l'ordre dans lequel les attributs sont représentés dans les tables, enfin le caractère spécifique de chaque SGBD qui entraîne une diversité d'implémentation.

La normalisation est le processus par lequel les données sont organisées dans une base de données. Ceci comprend la définition des tables et l'établissement de relations entre ces tables selon certaines règles pour éliminer la redondance et les dépendances implicites. Cette normalisation peut créer une diversité structurelle du fait que les tables et les relations peuvent être différentes. Ainsi par exemple on peut représenter l'adresse des organisations dans la même table *organisation* (*nom, no, rue, ville, code_postale, pays, tél, fax*) ou par des tables séparées (*nom, add_id*) (*add_id, no, rue, ville, code_postale, pays, tél, fax*). La prise en compte de la redondance a également été considérée dans le modèle PLIB ou deux schémas ont, en fait, été définis, le schéma implicite, ou en intention, dans lequel toute redondance est évité au prix d'une assez grande complexité d'implémentation (toute propriété calculable est associée à sa méthode de calcul, et non à la valeur qui en résulterait pour chaque instance), et le schéma explicite, ou en extension dans lequel les attributs sont tous valués.

Les mêmes tables étant choisies, il n'y a aucune raison que les attributs y soient rangés dans le même ordre.

La diversité d'implémentation est due à l'existence de différents systèmes de gestion de bases de données ayant chacun leurs spécificités. Il est évident que l'implémentation d'un modèle conceptuel, décrit par une méthode orientée objet, ne sera pas le même dans un SGBDR que dans un SGBDOO. Ainsi, dans l'exemple de la section 6, pour représenter l'attribut *possede* de l'entité *service_de_locatio* on utilisera des relations dans SGDBR et des agrégats dans une base de données OO. Il en est de même pour l'héritage. Ce dernier sera simulé par des relations dans des SGDBR, ainsi que dans Oracle et les autres SGDBRO ne supportant pas ce mécanisme, et par l'héritage des classes et des tables dans les systèmes OO et RO supportant le mécanisme d'héritage (exemple : Postgres).

Nous présentons brièvement dans la section suivante le langage EXPRESS pour expliquer ensuite l'usage que nous proposons d'en faire pour l'échange entre bases de données hétérogènes.

3. Le langage EXPRESS

3.1. Concepts d'EXPRESS

EXPRESS est un langage de spécification des données normalisé (ISO10303). Il a été défini initialement pour définir des modèles de données dans les domaines techniques. Il est à présent largement utilisé pour la modélisation des données dans différents domaines[7], [1]. EXPRESS s'inscrit dans le prolongement des travaux sur les modèles de données tels que le modèle entité/association [23], OMT [6], NIAM [9] etc. Il a été défini pour rendre plus précis, et traitable par machine, de tels modèles et pour représenter beaucoup plus librement les contraintes sur les données et donc les conditions de leur intégrité. EXPRESS a été développé dans le cadre du projet STEP (STandard for Exchange of Products data) [24]. Il n'est pas seulement un langage de modélisation conceptuel utilisable pour les échanges entre humains. Il est également un langage de définition de données (DDL) [21] permettant de spécifier les données devant être générées et valider pour les machines.

EXPRESS permet deux niveaux de représentation de l'information :

- une description en intention : Cette description, appelée schéma, correspond au schéma conceptuel d'une base de données. Elle est définie en terme d'un ensemble d'entités, modélisée selon l'approche objets (héritage, liens, liens inverse, dérivation d'attributs …). Les entités sont associées à un ensemble d'attributs typés. Enfin, des contraintes et des fonctions permettent d'associer une sémantique ensembliste à la description en intention. des dérivations peuvent être exprimées,

- une description en extension : Cette description correspond à la population d'une base de données, pour laquelle un format de représentation spécifique à EXPRESS a été défini. Ce format est appelé fichier physique ou fichier d'instances. Il est constitué d'un ensemble d'instances conforme au modèle décrit en intention. Cette description constitue une interprétation particulière, au sens logique, de la description en intention [20].

Dans un schéma de données EXPRESS, Une entité représente un ensemble d'objets ayant des propriétés communes. Ces propriétés sont modélisées par des attributs et des contraintes. L'ensemble des entités est regroupé dans des schémas qui peuvent se référencer. Les domaines de valeurs peuvent également être modélisés par des types.

```
ENTITY A ;
a1 : REAL ;
a2 : OPTIONAL NUMBER ;
a3 : INTEGER ;
a4 : SET OF montype1 ;
a5 : montype2 ;
DERIVE
a6 : REAL := a1*a3 ;
INVERSE
a6 : B FOR bn ;
UNIQUE
a3 ;
WHERE
IF EXISTS (a2) THEN a1 * a2 > 0 ; END_IF ;
END_ENTITY ;
ENTITY B ;
b1 : montype3 ;
…
bn : A ;
END_ENTITY ;
TYPE montype = INTEGER;
WHERE
 Wr: SELF >0;
END_TYPE;
```

Dans l'exemple ci-dessus, A et B sont des entités. Elles possèdent des attributs *ai*, *bi* typés. Les types de données sont soit des types simples (real, integer), des collections d'un type donné (l'attribut a_4 de A) ou bien des types nommés (type défini par utilisateur montype)ou encore une entité(attribut b_n de B). L'attribut a6 est un attribut dérivé calculé à partir de la multiplication des deux attributs a_1 et a_3. Ceci permet d'exprimer un invariant de données sous forme fonctionnelle. Cette expression peut être remplacée par une fonction plus complexe écrite en utilisant le langage impératif d'EXPRESS, proche de PASCAL qui permet de décrire aussi bien des fonctions de dérivation que des contraintes logiques. Les contraintes logiques constituent l'autre classe d'invariant de données. Elles sont introduites par les clauses WHERE et RULE qui décrivent respectivement des invariants locaux à une entité et globaux à un schéma.

Les relations d'héritage sont exprimées par les mots clés SUPERTYPE et SUBTYPE :

```
Entity E1
SUPERTYPE OF (E11 ANDOR E12)
SUBTYPE of( E) ;
….
END_ENTITY ;
```

Où E_1 est la classe mère de E_{11} et E_{12} et est fille de E. Seule la déclaration de la ou des super-classes est obligatoire. La clause SUBTYPE permet de préciser qu'une instance peut appartenir simultanément à plusieurs sous-classes.

Dans le fichier physique les instances d'entités sont décrites par les valeurs de leurs attributs explicites.

```
#1 = A(Va1, ……, Vai) ;
#2 = B (Vb1,……, #1) ;
```

Les Va_i et Vb_i représentent les valeurs des attributs explicite *ai* de A et *bi* de B. chaque instance est associée à un identificateur qui permet de la référencer (#1 et #2). Les attributs dérivés et inverses ne sont pas représentés dans les fichiers physiques car ils peuvent être directement calculés sur le système receveur.

3.2. Techniques associées au langage EXPRESS

Le fait pour EXPRES d'être associé à une syntaxe et à une sémantique précise a permis de développer tout un ensemble d'outils pour manipuler des modèles et des données EXPRESS. Ainsi, à tout schéma EXPRESS sont associés :

- une interface normalisée d'accès aux données conforme à ce schéma, disponible dans plusieurs syntaxes, c'est le SDAI pour Standard Data Access Interface,
- une structure de fichier texte pour l'échange de données conforme à ce schéma.

Il existe alors des outils permettant automatiquement de :

- de générer un SDAI et les APIs d'accès aux données,
- de créer un fichier physique à partir d'une SDAI,
- de créer un SDAI à partir d'un fichier physique échangé.

D'autre part, EXPRESS étant un langage de spécification de haut niveau, puissant car il permet de manipuler globalement les collections et ensembles, il a été complété par :

- le langage EXPRESS-C, qui permet de manipuler les données conforme à un modèle EXPRESS en utilisant directement les rotations EXPRESS,
- le langage EXPRESS-X qui permet d'exprimer de façon déclarative les correspondances entre deux schémas et de générer automatiquement un programme capable de transformer toutes les données conforme à un schéma aux données conforme à l'autre schéma.

Enfin, la possibilité d'exprimer à l'aide de fonction le calcul des attributs dérivés permet une véritable programmation événementielle au sein du modèle de données [7], [19]. Nous utilisons précisément cette approche pour l'une des conversions nécessaires à l'échange entre base de données hétérogènes.

3.3. L'express-G

L'EXPRESS-G est une représentation graphique de la représentions textuelle du langage EXPRESS. Il permet une représentation synthétique d'un modèle de données EXPRESS. De plus, ce formalisme peut être utilisé dans les phases préliminaires de conceptions de modèles de données. EXPRESS-G permet une représentation des concepts structurels et descriptifs du modèle de données par une annotation graphique, ce qui augmente sa lisibilité et sa compréhensibilité. Par contre les aspects procéduraux (dérivation et contraintes) ne peuvent être représentés. L'exemple suivant illustre une représentation EXPRESS-G d'un modèle de données simple portant sur des entités géométriques.

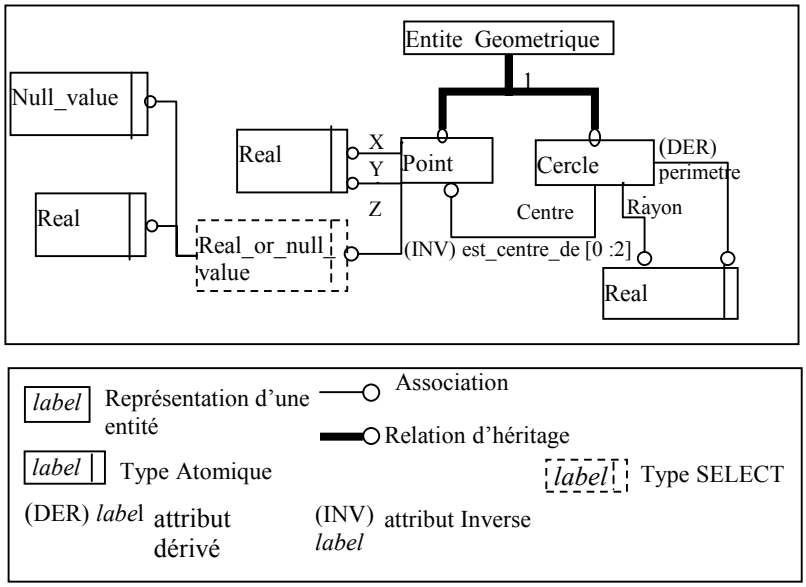

Figure 1 représentation graphique EXPRESS-G des entités géométriques

Dans cet exemple une *entite_geometrique* peut être soit un *cercle* soit un *point*. Un *cercle* possède un *centre*, un *rayon* et un attribut dérivé *perimetre.* L'entité *point* possède les coordonnées *X, Y, Z* dont la coordonnée *Z* peut avoir soit une valeur réelle soit une valeur nulle introduite par le type SELECT *real_or_null_value* qui représente une union de types. Enfin un point peut être le centre d'un maximum de deux cercles par l'attribut inverse *est_centre_de.*

Nous avons décrit ci-dessus les concepts nécessaires pour la compréhension du reste de l'article. Pour plus d'informations sur le langage EXPRESS consulter [2] [5].

4. Notre approche pour l'intégration

L'approche que nous proposons est résumée par trois modèles EXPRESS (simplifiés). Nous présentons d'abord ces modèles, puis nous discutons comment les différents aspects de la diversité y sont pris en compte.

4.1. Les modèles de l'échange

4.1.1. Nomination des concepts : définition d'un identificateur universel

Afin d'éviter les problèmes liés aux différences de noms, un schéma universel de nomination permettant de partager des noms a été défini. Ce schéma permet d'identifier trois catégories de concepts : les sources d'identification ou de modèles (*supplier_BSU*), les entités (*class_BSU*) et les propriétés (*property_BSU*).

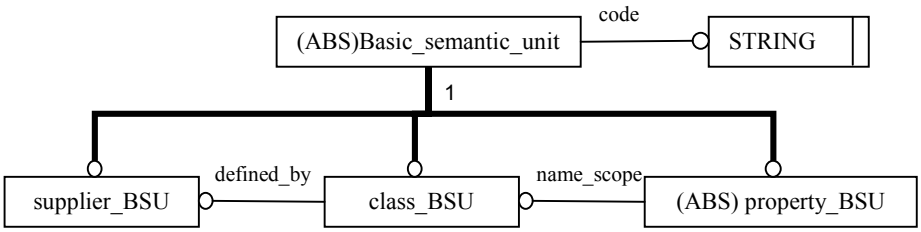

Figure 2 identification universelle des concepts

L'identifiant de chacun des concepts hérite d'un attribut code. L'identification d'une source d'information est un simple code, mais la manière d'affecter ce code, défini dans la norme ISO13584-26, assure son unicité. L'identifiant d'une entité est constitué d'un code et d'une référence à la source d'information qui l'a définie (l'attribut *defined_by*). Il est donc suffisant que chaque source d'information assure l'unicité des codes d'entité qu'elle attribue pour assurer une unicité globale des instances de *class_BSU*.

Enfin, l'identification d'une propriété est constituée d'un code et d'une référence à une *class_BSU*. Il est donc suffisant pour assurer l'unicité de cet identifiant que la source d'information attribue des codes d'attributs uniques pour chacune des classes que cette source d'information définit.

On suppose que des dictionnaires existent pour les entités et les attributs faisant l'objet de consensus et que ces dictionnaires définissent les identifiants de ces entités et attributs. Pour les entités propres à un système, grâce à l'identification de la source d'information, elles sont clairement identifiées en tant que telles et elles feront l'objet d'un traitement adapté. Dans le cas particulier des bibliothèques de composants industriels de tels dictionnaires, existent effectivement (exemple : la norme IEC 1360-4), ou sont en cours de développement pour différents sous-ensembles du domaine.

4.1.2. Echange d'instances : méta-schéma générique

Le modèle utilisé pour échanger des instances est un schéma générique susceptible de représenter n'importe quelle instance de n'importe quelle base de données dont les entités et les attributs sont identifiés par un identificateur universel défini par le modèle précédent. Une instance y est représentée, outre l'identificateur universel de sa classe par des couples (attributs, valeurs)

L'attribut est identifié par son attribut universel, et la valeur est représentée comme une valeur appartenant de tous les types de base. Le schéma ci-dessous simplifie ce modèle en supposant que les type de base sont seulement les types chaîne, entier, et référence à une instance.

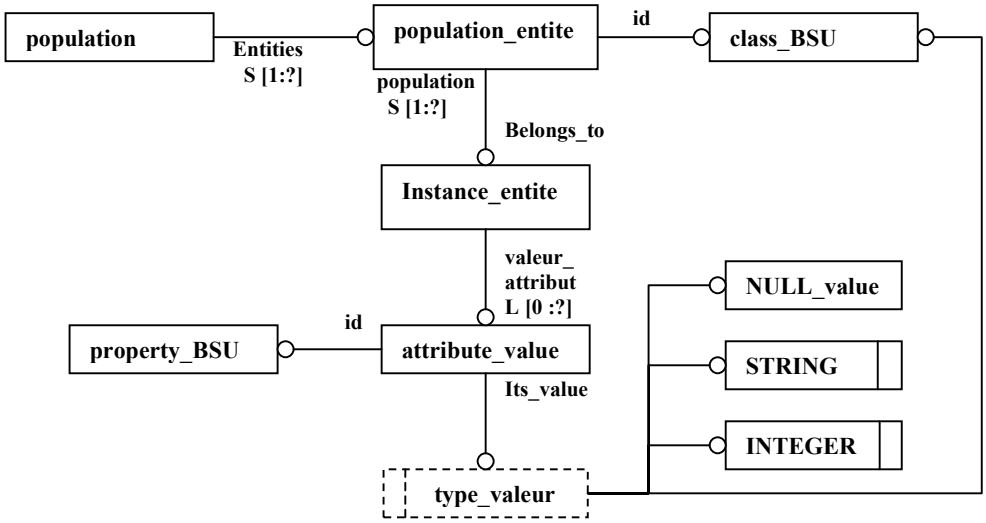

Figure 3 méta-schema générique d'échange d'instances

Ce schéma est illustré par un exemple de mise en œuvre dans la section 6.

4.1.3. Echange de modèles : méta-schéma générique

Si tous les acteurs de l'échange se sont mis d'accord sur un ensemble d'entités et d'attributs susceptibles d'être échangés et sur leurs identificateurs universels, il n'est pas nécessaire d'échanger le modèle du système source pour pouvoir interpréter l'échange.

Au contraire, si le système source contient des attributs additionnels, ou des entités additionnelles, il est nécessaire d'échanger leurs identifications et leurs modèles de définition pour permettre leur stockage ou leur interprétation sur le système receveur. Ainsi si une entité A, connue de tous, a été sous-typé en une entité A1, chaque instance de A1 étant décrite par les attributs $p_1, p_2, ..., p_n$ connues de tous et héritées de A, et les attributs q_1, q_2, q_n définie par le système source, le système receveur pour décider :

- Soit de projeter les instances de A1 sur la définition de A et de les charger dans la population de A,
- Soit de créer une nouvelle entité A1 conforme à la définition échangée et dont la définition sera elle-même stockée dans la base de destination.

Le schéma ci-dessous donne une version un peu simplifiée du méta-schéma générique d'échange de modèles, en supposant, par exemple, que seuls les types entier, chaine et références existent..

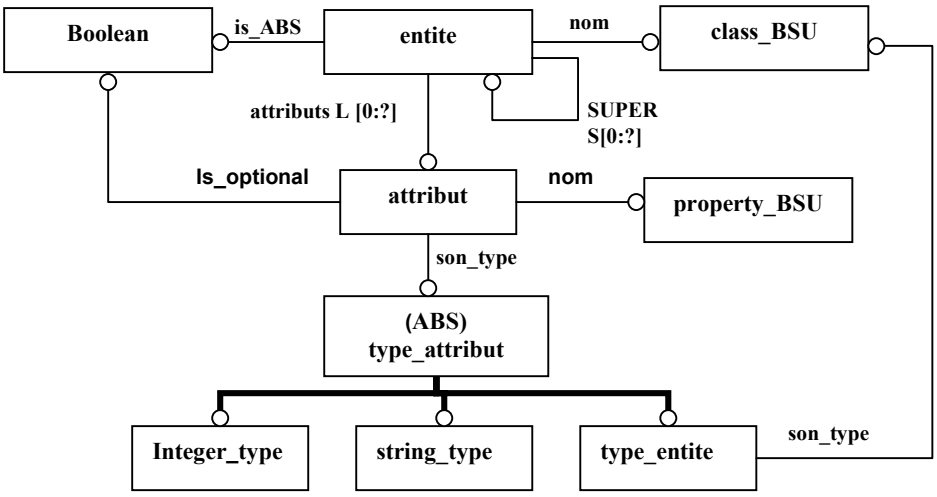

Figure 4 méta-schéma générique d'échange des modèles

Dans ce schéma chaque entité est identifiée par un identifieur absolu (*class_BSU défini dans la figure 2)* et possède un ensemble d'attributs. L'attribut *super* permet la représentation de l'héritage et une entité peut être définie comme abstraite (attribut *is_ABS*). Chaque attribut est identifié par un identifieur absolu(*property_BSU également défini dans la figure 2*) et par un type *son_type*. Un attribut peut être obligatoire ou bien optionnel (l'attribut *is_optinal*).

4.2. Prise en compte des diversités

- *Diversité des noms* : le schéma de normalisation de concepts permet d'atteindre, pour autant évidemment que l'ensemble des acteurs le souhaitent, les objectifs suivants :

1. Le même concept, entité ou attribut, est toujours identifié de la même façon,
2. Un concept nouveau ne peut pas être confondu avec un autre concept, et la source de sa définition est clairement identifiée.

- *Diversité conceptuelle* :

- Toute instance appartenant à une entité connue et associée à des attributs connus peut être interprétée sans ambiguïté sur le système receveur quels que soient les attributs représentés et quel que soit l'ordre dans lequel ils apparaissent.

- Si une entité a été spécialisée d'une entité connue, et sous réserve que son modèle soit échangé, il est possible de représenter les instances soit comme des instances de l'entité connue qui a été spécialisée, soit de représenter effectivement la nouvelle entité, y compris sa définition et le lien de généralisation/spécialisation avec l'entité connue.

- *Diversité structurelle* : concernant enfin la diversité structurelle :

- Chaque instance étant représentée explicitement, complètement, et indépendamment des autres instances, les structures des tables propre à la base de données source n'apparaît plus dans le format d'échange.

- Les attributs valués étant identifiés par un identificateur explicite et non d'après leur position, l'ordre des attributs n'a aucune importance.

- Enfin, si les systèmes sources et cibles n'utilisent pas exactement les même attributs pour les instances d'une certaine entité, la stratégie programmée sur le système receveur pourra, par exemple, consister à négliger les attributs supplémentaires et à représenter par une valeur nulle les attributs manquants.

5. Le Processus d'échange

Le processus d'échange comporte toujours l'échange d'instances. Il comporte éventuellement un échange de modèles.

Le processus d'échange comprend au maximum cinq étapes différentes, présentées sur la figure 5, dont trois sont toujours présentes (1, 3 et 5). Le processus le plus complet suppose qu'existent trois modèles EXPRESS :

- le modèle pivot représenté à la section 4.1,
- un modèle EXPRESS, dit "EXPRESS-source", représentant, le plus à l'identique possible, le schéma interne de la base de données source
- un modèle EXPRESS, dit "EXPRESS-cible", représentant le schéma interne de la base de données cible.

Les données passent alors par cinq formes :

- données de système source,
- instances EXPRESS du modèle EXPRESS-source,
- instances EXPRESS du modèle pivot,
- instances EXPRESS du modèle EXPRESS-cible,
- données de système cible.

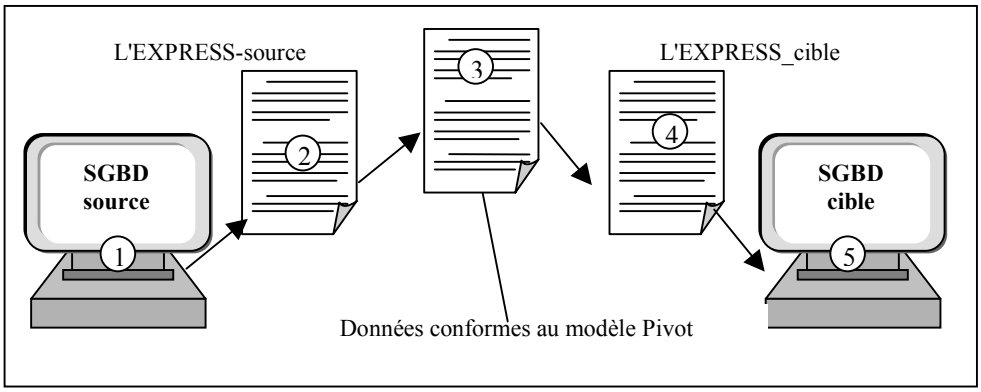

Figure 5 processus d'échange de données entre deux systèmes hétérogènes

Le passage de la base de données source au modèle EXPRESS-source sera implémenté dans un langage de programmation propre à la base source. Le programme nécessaire, générique exploitera le dictionnaire de cette base (également nommé métabase [14]) . Le résultat sera un fichier d'instance du modèle EXPRESS-source. Ce dernier sera ensuite converti grâce à l'une des techniques de conversion disponible dans le contexte du langage EXPRESS. Une fois les instances du modèle pivot disponibles, la génération de la base de données cible pourra être réalisée soit directement, soit en passant par l'intermédiaire d'un fichier d'instances du schéma EXPRESS-cible.

Dans les deux cas, le passage final d'instances d'un modèle EXPRESS à des données dans le système cible nécessitera de générer des instructions de manipulation des données (INSERT …) et éventuellement de définition de données (CREATE ….) nécessaires.

Trois techniques peuvent être utilisées :

- programmation en EXPRESS-C du parcours de la population EXPRESS et génération d'instructions en LMD et LDD,
- programmation dans le langage de programmation de la base de données et accès aux instances EXPRESS à travers l'API correspondant au schéma EXPRESS pertinent (EXPRESS-cible ou modèle pivot),
- génération des instructions de LMD et de LDD par programmation événementielle au sein d'un gestionnaire de données EXPRESS.

C'est cette dernière méthode que nous illustrons dans la section 6.

6. Exemple de mise en œuvre

Dans le cas où les données correspondraient à des entités et à des attributs connus de tous, l'échange entre bases de données hétérogènes se limite à l'échange des instances. Nous donnons dans ce qui suit un exemple simplifié portant sur l'échange d'instances correspondant à un exemple définit dans la figure 6 ci-dessous. Ce schéma décrit un service de location de voitures où chaque service est géré par une organisation et possède un ensemble de voitures à louer. De leur coté, les entités *organisation* et *voiture* sont décrites par un ensemble d'attributs (*nom, puissance,...*).

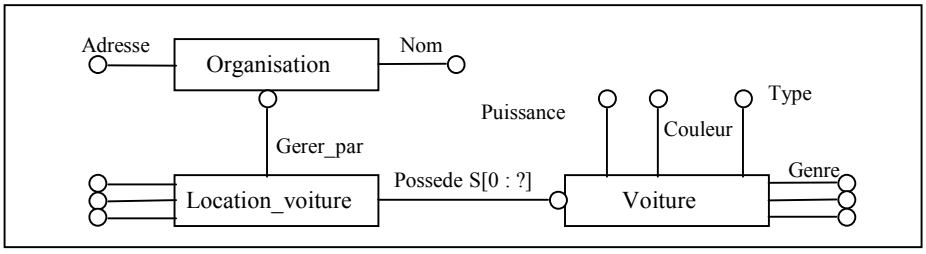

Figure 6 un modèle de données présentant l'univers des services de location de voitures

L'approche que nous proposons consiste à ajouter au modèle de données pivot, défini à la figure 3 et destinée à être utilisé pour l'échange d'instances, des attributs dérivé. Ces attributs dérivés ne devant pas figurer dans les instances d'échange, le même fichier d'échange représente des instances licites aussi bien pour le modèle initial, défini à la figure 3, que pour le modèle modifié. Si le fichier d'échange est alors considéré comme un ensemble d'instance du modèle modifié, il est possible, dans un système EXPRESS qui connaît le modèle, d'interroger pour chaque instance la valeur de l'attribut dérivé. De cette manière l'ensemble du code SQL correspondant à la conversion de l'ensemble des instances peut être généré. La figure ci-dessous présente un exemple de cette approche.

```
ENTITY population_entite;
Id : class_BSU;
Population : set [1:?] of instance_entite;
DERIVE
SQL_table_name : STRING := SELF.Id ;
SQL_schema : STRING := 'CREATE TABLE ' + SELF.SQL_table_name + '(' +
     compute_attributes(SELF.population) + ');' ;
SQL_population : STRING := concatener_SQL_population (SELF.population);
END_ENTITY ;
*
ENTITY instance_entite;
Valeur_attribut : LIST [1:?]of atribute_value;
DERIVE
SQL_population : STRING := 'INSERT INTO ' + SELF.belongs_to.SQL_table_name + 'value '
     + reunir_valeur_attribut (SELF.valeur_attribut) + ');' ;
INVERSE
Belongs_to : population_entite FOR population ; ……
END_ENTIY ;

ENTITY population;
Entities : set[0:?] of population_entite;
END_ENTITY;
```

Le nom de la représentation (i.e. la table) est le même que le nom du concept qu'elle représente (i.e. l'entité *voiture*). L'attribut *SQL_schema* de l'entité *population_entite* permet de générer l'instruction CREATE TABLE qui correspond à l'entité *voiture*. La fonction *compute_attributes* permet le calcul des attributs et leurs types, nécessaires pour la création d'une table SQL, en parcourant l'attribut *valeur_attribut* (un couple (attribut, valeur)) des entités. On suppose ici que chaque instance est définie par le même ensemble d'attributs. L'attribut *SQL_population,* de l'entité *population_entite,* permet de rassembler toutes les commandes d'insertions des différentes instances de la *population* en concaténant le contenu des attributs *SQL_population* de chacune des *instance_entite* référencée par l'attribut *population.*

Le modèle d'échange d'instances conforme au schéma générique ci-dessus est le fichier physique suivant :

```
#1 = population_entite (#2, (#3)) ;
#2 = class_bsu ('voiture') ;
#3 = instance_entité ((#6, #7)) ;
#4 = property_bsu ('genre', #2) ;
#5 = property_bsu ('couleur', #2) ;
#6 = attribute_value (#3, 'Peugeot') ;
#7 = attribute_value (#5, 'rouge') ;
#8 = population_entite (#10, (#9)
#9 = instance_entité ((#13, #14)) ;
#10 = class_bsu ('location_de_voiture) ;
#11 = property_bsu ('gerer_par', #10) ;
#12 = property_bsu ('possede', #10) ;
#13 = attribute_value (#11, #15) ;
…….. ;
#14 = attribute_value (#12, (#2)) ;
#15 = instance_entité (#16, (#19 ,#20)) ;
#16 = class_bsu ('organisation')
#17 = property_bsu ('nom', #16) ;
#18 = property_bsu ('addresse', #16) ;
#19 = attribute_value (#3, 'ADA') ;
#20 = attribute_value (#4, '9 av Northampton 86000') ;
```

Il faut noter que dans un cas réel les noms des entités ('voiture', 'organisation' …) et des attributs ('nom', 'adresse') devraient être remplacés par leurs identifiants universels. Ces identifiants seraient définis par des dictionnaires. Notons que de tels dictionnaires existent déjà pour les composants électroniques (norme IEC 61360-4) et pour les produits et les services d'une manière générale (UNSPSC pour Universal Standard Products and Services Classification). La figure 7 montre une partie du dictionnaire de IEC 61360-4 qui identifie à la fois des entités et des attributs.

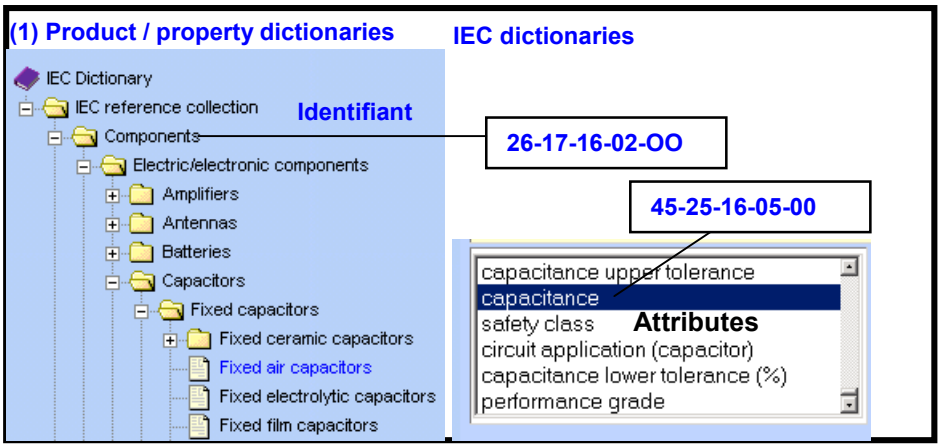

Figure 7 définition des concepts (entités et attributs) dans le dictionnaire IEC

Dans notre exemple on pourrait se référer au dictionnaire UNSPSC pour définir des identifiants pour les entités *voiture* et *location_voiture*. La classification des produits et des services dans UNSPSC est illustrée dans la figure 8 et permet d'associer le code 25-10-15-03 au concept "voiture" et le code 78-11-18-06 au concept "location de voiture".

Segment	Family	Class	Commodity	BTI	Title
25	0	0	0	0	Commercial and Military and Private Vehicles and their Accessories and Components
26	0	0	0	0	Power Generation and Distribution Machinery and Accessories
25	10	0	0	0	Motor vehicles
25	11	0	0	0	Marine transport
25	12	0	0	0	Railway and tramway machinery and equipment
25	10	15	0	0	Passenger motor vehicles
25	10	15	1	0	Minibuses
25	10	15	2	0	Busses
25	10	15	3	0	Automobiles or cars
25	10	15	4	0	Station wagons
25	10	15	5	0	Minivans or vans
25	10	15	6	0	Limousines
25	10	15	7	0	Light trucks or sport utility vehicles
25	10	15	8	0	Sports car
25	10	16	0	0	Product and material transport vehicles
78	11	18	6	0	Vehicle rental or leasing

Figure 8 classifications des "voitures" et "location voiture" dans UNSPSC

En revanche, dans les domaines d'application où de tels dictionnaires n'existent pas, une description similaire des concepts devra être réalisée. De cette manière l'utilisation des schémas génériques pour l'échange des instances et des modèles sera possible.

7. Conclusion

L'échange de données entre bases de données hétérogènes se heurte à la diversité des modèles conceptuels pouvant être définis pour un même domaine et la diversité des systèmes de gestion de bases de données supports : relationnels, relationnels-objets et orientés objets. Dans ce travail nous avons proposé une approche basée sur l'utilisation d'un modèle pivot constitué de deux méta-schémas génériques susceptible être utilisés pour n'importe quel échange. Ces méta-schémas référencent, pour identifier les entités et les attributs échangés, des dictionnaires de données supposés existant et chargés d'associer à chaque concept un identifiant universel. Ces méta-schémas sont exprimés en EXPRESS ce qui permet d'utiliser les différentes techniques de programmation disponibles dans l'univers EXPRESS pour réaliser les conversions nécessaires. Cette approche permet aussi bien un échange de données simples (au niveau instances) qu'un échange de schémas de bases de données (dans le cas où ces schémas devraient également être échangés).

Dans le cas où toutes les entités et tous les attributs sur le système receveur sont connus du système émetteur, un échange d'instances suffit. Dans le cas contraire (système émetteur contiendra des entités

additionnelles et/ou des attributs additionnels), l'approche proposée permet d'échanger les schémas eux-mêmes pour permettre leur interprétation et leur stockage dans le système receveur. Dans tous les cas, qu'il ait ou non échange de schéma, les concepts (entités et attributs) étant identifiés par des identifiants universel, l'ordre des attributs d'une entité n'a aucune importance.

Le passage de la base de données source au modèle pivot peut être réalisé directement, ou bien en passant par l'intermédiaire d'un fichier d'instances du modèle EXPRESS-source qui est une représentation EXPRESS de la métabase de la base de données source. Le passage du modèle pivot à une base de données cible peut être réalisé soit directement, soit en passant par l'intermédiaire d'un fichier d'instances du schéma EXPRESS-cible qui est une représentation EXPRESS de la métabase de la base de données cible. Dans les deux cas, cela nécessitera la génération des instructions de manipulation des données (INSERT …) et éventuellement de définition de données (CREATE ….) nécessaires.

Dans l'univers EXPRESS différentes techniques peuvent être utilisées pour générer les instructions de définition de données et de manipulation de données nécessaires pour restaurer le contenu échangé sur le système cible. Dans cet article nous avons proposé d'utiliser la technique des attributs dérivés d'EXPRESS pour effectuer cette génération. Cette technique permet de réduire la complexité du programme de génération en le découpant en fragments élémentaires chargé chacun de la conversion d'un ou d'un petit nombre de type d'entités.

L'approche proposée ici a été validée dans le domaine de l'échange des bibliothèques de composants industriels. Pour ce domaine d'application nous avons proposé une représentation de l'ensemble des informations échangées, grâce aux méta-schémas génériques, dans la base de données du système relationnel objet POSTGRE. Nous avons également montré comment un fichier d'instances EXPRESS pourrait être automatiquement converti en instructions de manipulation de cette base de données grâce à l'utilisation d'attributs dérivés ajoutés au modèle de données EXPRESS.

8. Références

[1] Ait-Ameur, Y., Pierra, G., Sardet, E., "An object oriented approach to represent behavioural knowledge in heterogeneous information systems ", Proc. of the 6th International Conference on Object Oriented Information Systems, OOIS 2000, London, 18-20 December 2000, D. Patel, I. Choudhuny, S. Patel and S. de Cesare, Eds., Springer, ISBN 1-85233-420-7, pp. 315-332

[2] D. A. Schenk and P. R. Wilson, "Information Modeling: The EXPRESS Way". Oxford University Press, 1994.

[3] M. El-Hadj Mimoune, Y. Ait-Ameur, G. Pierra et J.C. Potier, "Integration of component description in product data management systems", Proc. of ISPE International Conference on Advance in Concurrent Engineering, CE 2000, Technomic Publ. Co., Lancaster, USA, pp. 370-380.

[4] K. R. Dittrich and A. Geppert, "Object-Oriented DBMS and Beyond", proceeding of the Conference on Current Trends in Theory and Practice of Informatics, pp. 275-294, 1997.

[5] ISO 10303-11, "Industrial Automation Systems and Integration -- Product Data Representation and Exchange -- Part 11: Description methods: The EXPRESS language reference manual", 1994.

[6] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, and W. Lorrensen. Object-Oriented Modelling and Design. Prentice Hall, International Edition, 1991.

[7] A. Plantec, " Utilisation de la norme STEP pour la spécification et la mise en œuvre de générateurs de code", thèse, Université de Rennes 1, 1999.

[8] G. Pierra, "Modelling classes of prexisting components in a CIM perspective: The ISO 13584/ENV 400014 approach", revue internationale de CFAO et d'Infographie, vol 9, pp. 435-454, 1994.

[9] H. Habrias, "Le modèle relationnel binaire. La méthode NIAM", Eyrolles, 1988.

[10] G. Pierra, H. U. Wiedmer, eds, Industrial Automation Systems and Integration, Parts Library, Methodology for Structuring Parts Families, ISO 13584-42, ISO, Geneve, 1998 (132 p.)

[11] G. Pierra, Y. Ait-Ameur and E. Sardet, Eds, Industrial Automation Systems and Integration, Parts Library, Logical Model of Supplier Library, ISO DIS 13584-24, ISO, Geneva, 2001 (594 p.)

[12] G. Pierra, "Représentation et Echange de données techniques", Mécanique et Industrie, Mec Ind, 1, pp. 397-414, (2000).

[13] J.Rambaugh, I. Jacobson, and G. Booch, "The Unified Modeling Language Reference Manual" Addison-Wesley, 1999.

[14] G. Gardarin, " Bases de données", Editions Eyrolles, Paris, 2001.

[15] T. Schreuber, B. Wielinga, J. Breuker, KADS: "A Principled Approach to Knowledge-based System Development", Academic Press, London, Forthcoming, 1992.

[16] ISO 10303-21, "Industrial automation systems and integration - Product data representation and exchange - Part 21: Implementation methods: Clear text encoding of the exchange structure (Physical file)", 1994.

[17] M. Bouzeghoub, G. Gardarin, P. Valduriez, "Les objets", Editions Eyrolles 1997.

[18] E. Sardet, G. Pierra, H. Murayama, Y. Oodake, Y. Ait-Ameur, "Simplified Representation of Parts Library : Model, Practice and Implementation", proceeding of PDT Days, Brussels, QMS edition, 2001.

[19] M. El-Hadj Mimoune, Y. Ait-Ameur, G. Pierra, " Modélisation du contenu des catalogues de composants industriels : de la représentation implicite à la représentation explicite", ISPS'2001, Alger, pp. 15-26, 2001.

[20] Y. Ait-Ameur, "Développements Controlés de Programmes par Modélisation et Vérifications de Propriétés", Habilitation à diriger les recherches, Université de Poitiers, 2000.

[21] A. Herbst, "Long-Term Database Support for EXPRESS Data". In 7th Int'l. Working Conf. on Scientific and Statistical Database Management, pp. 207-216, Charlottesville, VA, September 1994. IEEE Computer Society Press.

[22] S Lampierre, F Coufin, J.-P. Frachet, " Méta-modélisation pour l'intégration par les données des activités d'ingénierie des systèmes industriels : application à la réalisation d'un atelier logiciel d'aide à la conception des systèmes d'information.

[23] Peter Pin-Shan Chen, "The Entity-Relationship Model. Toward Unified View of Data" ACM Transaction on Database Systemq, 1(1) :9-36, 1976.

[24] M. Bouazza, "la Norme STEP", Hermes, 1995.

[25] F. Feru, C. Viel, "Echanger avec le protocole d'application 203 de STEP : Echange et partage de données CAO et GDT", Association GOSET, 1998.