Un système d'intégration des métadonnées dédiées au multimédia

Samir Amir

Un système d'intégration des métadonnées dédiées au multimédia

THÈSE

présentée et soutenue publiquement le 06 Décembre 2011

pour l'obtention du

Doctorat de l'Université des Sciences et Technologies de Lille

(spécialité informatique)

par

Samir Amir

Composition du jury

Président :	Sophie Tison, (Professeur)	Université de Lille I
Rapporteurs :	Hervé Martin, (Professeur)	Université de Grenoble
	Florence Sèdes, (Professeur)	Université Toulouse III
Examinateurs :	Mohamed Quafafou, (Professeur)	Université de la Méditerranée
Co-encadrant	Ioan Marius Bilasco, (Maitre de conférence)	Université de Lille I
Directeur de thèse :	Chabane Djeraba, (Professeur)	Université de Lille I

UNIVERSITÉ DES SCIENCES ET TECHNOLOGIES DE LILLE
Laboratoire d'Informatique Fondamentale de Lille — UMR 8022
U.F.R. d'I.E.E.A. – Bât. M3 – 59655 VILLENEUVE D'ASCQ CEDEX
Tél. : +33 (0)3 28 77 85 41 – Télécopie : +33 (0)3 28 77 85 37 – email : direction@lifl.fr

À mes parents

À mes frères et sœurs

À ma femme

Remerciements

Tout d'abord, je remercie vivement le professeur Chabane Djeraba de m'avoir accueilli au sein de son équipe pour réaliser ma thèse. Je le remercie pour m'avoir guidé, conseillé et soutenu tout au long de la thèse.

Je remercie mon co-encadrant Ioan Marius Bilasco pour ses nombreux conseils et son soutien. Grâce à lui j'ai beaucoup appris sur la recherche. Je lui en suis très reconnaissant.

Mes remerciements vont également à Thierry Urruty pour sa collaboration sur de nombreux travaux. Je le remercie pour son aide et son soutien pendant ces trois années.

Je tiens à remercier les professeurs Hervé Martin et Florence Sédes pour avoir accepté d'être les rapporteurs de mon mémoire de thèse, et le professeur Mohamed Quafafou pour avoir accepté d'être examinateur à ma soutenance.

Je remercie vivement le professeur Sophie Tison, directeur du Laboratoire d'Informatique Fondamentale de Lille, d'être la Présidente de mon jury de thèse et pour m'avoir accueilli au sein de son laboratoire pendant mes trois années de thèse. Je remercie aussi l'ensemble des membres du LIFL, doctorants, chercheurs et personnels administratifs.

Je tiens à saluer l'ensemble de l'équipe Fox-Miire dans laquelle j'ai réalisé ma thèse. Plus particulièrement, je remercie Emilie Mesureur pour ses corrections, Taner Danisman pour l'aide apportée durant ma contribution sur le projet MIDAS, ainsi que les autres collègues Yassine, Ismail , Nacim, Rémi, Tarek, Jean et Céline pour les discussions diverses et variées.

Résumé

L'objectif principal de mon travail de thèse est de développer un système d'intégration des métadonnées dédiées au multimédia. Nous proposons, dans un premier temps, un métamodèle pour l'agrégation des métadonnées hétérogènes. Ce métamodèle regroupe plusieurs types de métadonnées nécessaires pour la création, la livraison et la consommation des contenus multimédia via des réseaux et des terminaux hétérogènes. Le métamodèle proposé est extensible et permet également l'intégration d'autres standards de métadonnées existants.

La deuxième partie de ma thèse porte sur l'intégration automatique de métadonnées hétérogènes. Nous proposons un outil de matching des schémas baptisé MuMIe (Multi-level Metadata Integration) qui tient compte de l'hétérogénéité au niveau des schémas et des langages de description. La technique proposée transforme les schémas provenant de différents langages (XML, RDF, etc) en graphes. Ensuite, des mesures de similarité sont effectuées sur ces graphes permettant de trouver les correspondances entre les nœuds via l'utilisation de plusieurs informations sémantiques et structurelles.

Mots clés : métadonnées, interopérabilité, intégration des métadonnées, matching des schémas, langages de description.

Abstract

The main objective of my thesis is to develop a multimedia metadata integration system. First, we propose a metamodel for the aggregation of heterogeneous metadata. This metamodel contains several types of metadata necessary for the creation, delivery and consumption of multimedia content through heterogeneous networks and terminal capabilities. The proposed metamodel is extensible and also allows the integration of other existing metadata standards.

The second part of my thesis focuses on the automatic integration of heterogeneous metadata. This is done by developing a new tool for schema matching named MuMIe (Multi-level Metadata Integration). The proposed approach takes into account the heterogeneity on two levels (schemas and description languages). It converts heterogeneous schemas into graphs and computes similarity measures on these graphs to find the correspondences between nodes. This is done via the use of several structural and semantic information.

Keywords : metadata, interoperability, metadata integration, schema matching, description language.

Table des matières

Table des figures

Liste des tableaux

Chapitre 1

Introduction

1.1 Contexte et problématique

Les contenus multimédia sous forme d'images, vidéos, audios et documents jouent un rôle de plus en plus important dans la vie quotidienne. En parallèle, la manière dont les contenus sont créés, diffusés et consommés a grandement évolué, en obligeant les créateurs et les diffuseurs de contenus de faire face à une variabilité importante en termes d'infrastructure (réseaux Internet, 3G, etc), de besoins informationnels ponctuels des utilisateurs, de leurs intérêts et de la manière dont ils interagissent avec les contenus (simple consultation, annotations, partage, etc). La consommation est contextualisée, car on ne consomme plus pareil le contenu au bureau, à la maison ou en voyage. L'utilisation optimale de contenus par différents acteurs de la communauté multimédia (ex : consommateurs, développeurs, services, applications, etc.), par rapport au contexte, nécessite des informations sur les contenus eux-mêmes, ainsi que sur le contexte de consommation. Ces informations descriptives sont communément appelées *métadonnées*. Comme illustré dans la Figure 1.1, trois grandes catégories de métadonnées se dessinent généralement dans le paysage du Web :

– les métadonnées décrivant les contenus,
– les métadonnées décrivant le contexte matériel,
– les métadonnées décrivant l'utilisateur.

Dés la création du contenu, on génère des métadonnées descriptives sur ce contenu (ex : date de création, sujet, taille, etc) et parfois des informations de haut niveau (ex : nom d'une personne apparaissant dans une vidéo). Ces métadonnées offrent les bases pour le développement des systèmes de navigation et de recherche sémantique dans les contenus multimédia telle que la recherche d'un concept dans une vidéo ou la classification des contenus en fonction de leur sémantique. Avec l'avènement du Web 2.0, parmi les métadonnées décrivant les contenus, nous devons intégrer également les métadonnées communautaires résultant des in-

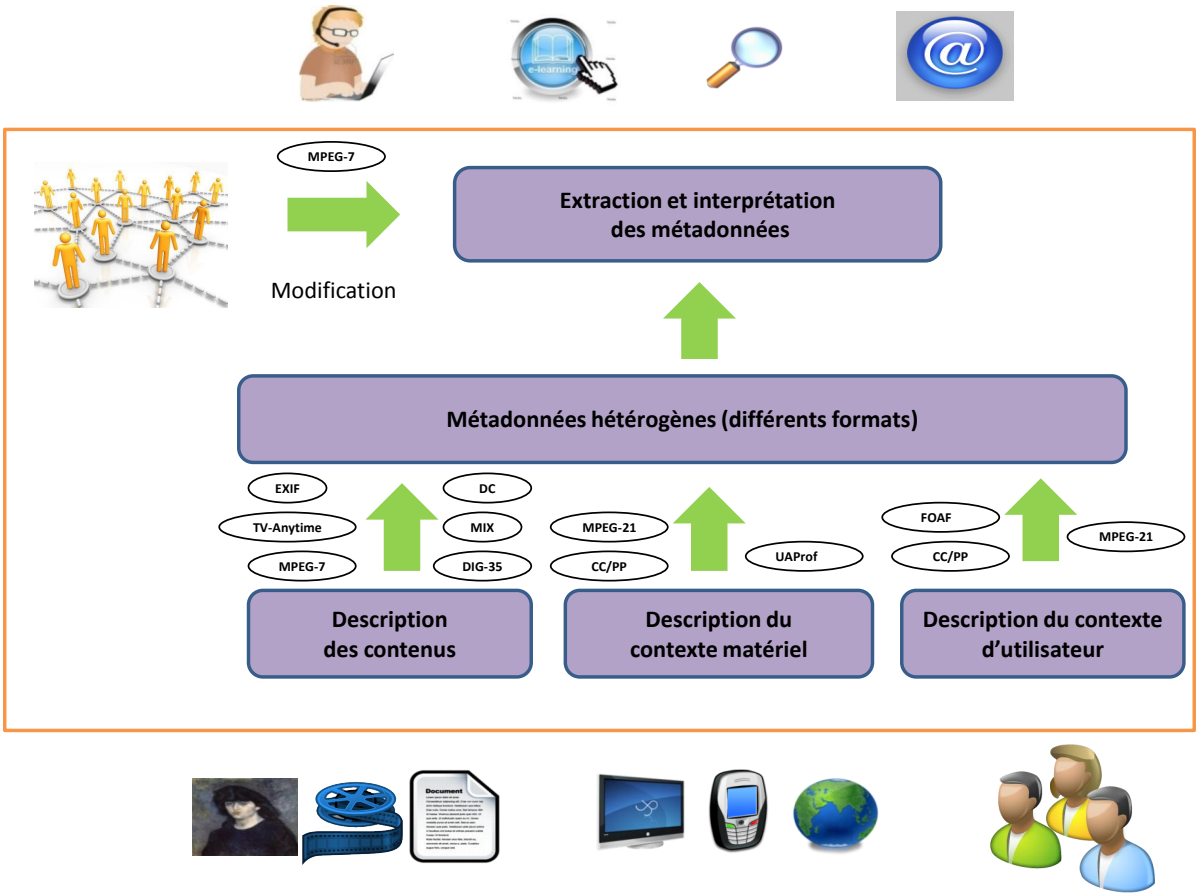

FIGURE 1.1: Les différentes sources des métadonnées hétérogènes.

teractions entre l'utilisateur et le contenu qu'il consomme. Les utilisateurs de ces contenus multimédia ont parfois le droit de commenter, voter et annoter ces contenus enrichissant ainsi les descriptions des contenus ciblés.

L'hétérogénéité des plate-formes et des terminaux des utilisateurs ainsi que leurs préférences posent des importants défis en ce qui concerne la diffusion adaptée de contenus. Souvent à cause des contraintes matérielles ou des préférences et intérêts utilisateurs qui varient dans le

temps, le contenu initial doit être adapté afin de convenir au contexte effectif de déploiement. Ainsi, il est nécessaire de disposer des descriptions précises sur les éléments du contexte auquel le contenu doit être adapté (préférences utilisateurs, caractéristiques des terminaux et de réseaux de transmission, etc).

Généralement les métadonnées utilisées suivent les standards et les recommandations des divers organismes afin de pouvoir être interprétées par le plus grand nombre de développeurs et d'applications. Par exemple, les standards MPEG-7, DIG-35, EXIF ou Dublin Core permettent de décrire les caractéristiques de contenus, alors que MPEG-21 DIA, CC/PP ou UA-Prof servent à décrire le contexte ou encore FOAF pour décrire les liens entre utilisateurs au sein d'une communauté. Cependant, étant donné que ces formats ont été conçus pour couvrir des besoins informationnels spécifiques (par exemple MPEG-7 pour le contenu, CC/PP pour la caractérisation des plate-formes de diffusion), l'utilisation de plusieurs formats en même temps est incontournable. Par conséquent, l'utilisation des métadonnées devient assez complexe et génère un problème d'hétérogénéité également au niveau des langages de représentations. Les différents utilisateurs et applications utilisant ces formats de métadonnées souhaiteraient à moindre coût être capables d'interpréter toutes les informations contenues par ces métadonnées afin de pouvoir les exploiter de manière optimale. Cependant, cet objectif est loin d'être réalisable compte tenu de l'hétérogénéité sémantique, syntaxique et structurelle des métadonnées conçues par des communautés indépendantes.

Le problème général de notre thèse est l'hétérogénéité des métadonnées. Afin de résoudre le problème de l'hétérogénéité, plusieurs solutions d'intégration ont été proposées. Ces solutions se regroupent en deux catégories principales :

- la modélisation conceptuelle, faite par des experts humains dont le but est de trouver une représentation générique pour la fusion des métadonnées.
- l'intégration automatique basée sur le matching (mise en correspondance) des schémas.

Dans ce cadre, nous proposons deux contributions qui appartiennent aux deux catégories citées précédemment. Dans un premier temps, un modèle générique et extensible nommé CAM4Home est proposé afin d'offrir une vue unifiée sur l'ensemble de métadonnées décrivant le cycle complet de vie d'un contenu multimédia.

A titre d'exemple, la Figure 1.2 identifie les différents types d'informations nécessaires à la réalisation d'une application sociale (ayant une dimension communautaire) concernant la présentation des joueurs de football (images, vidéos et statistiques fournis par des Web Services) multicanaux.

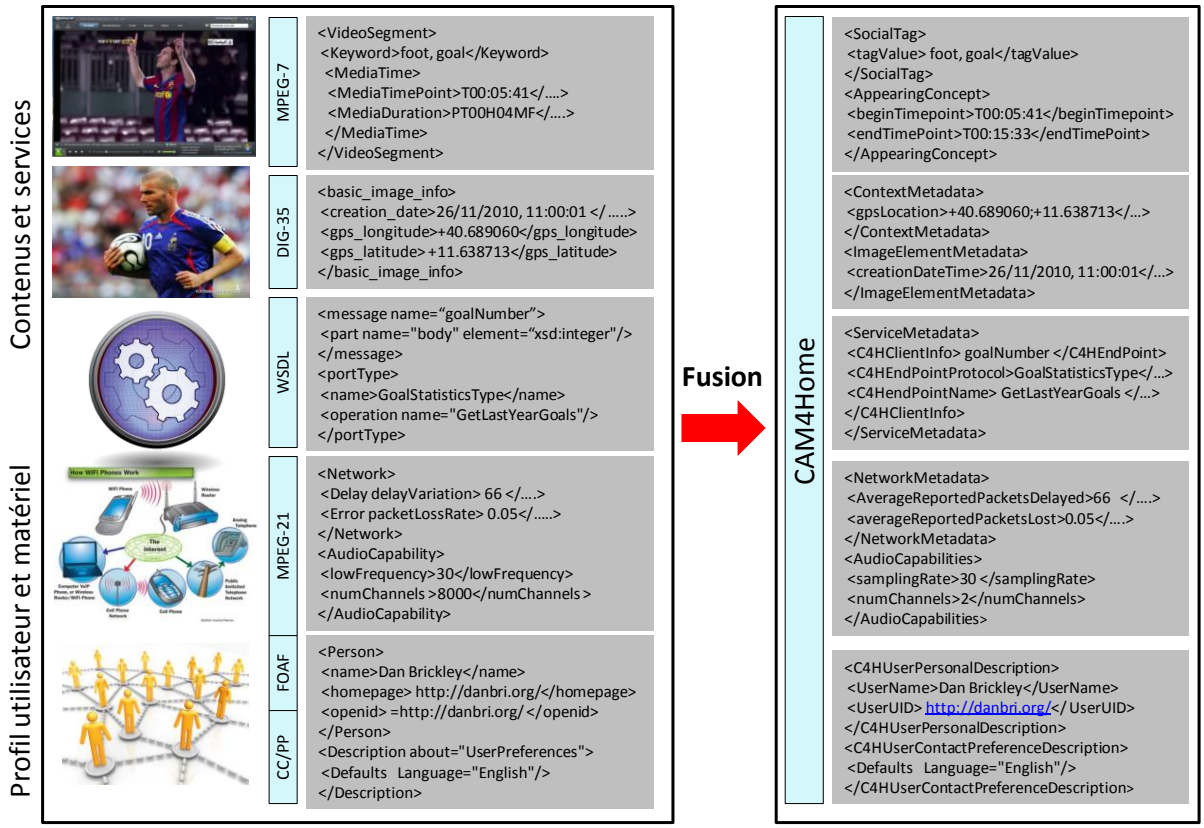

FIGURE 1.2: La fusion des métadonnées au sein de CAM4Home.

Dans cet exemple, nous nous intéressons à :

- des vidéos décrites en MPEG-7 qui définissent des extraits de la vidéo (*VideoSegment*) annotés avec des mots clés (*Keywords*).
- des images décrites en DIG-35 dont la description indique la date (*creation_date*) et le lieu (*gps_longitue*, *gps_latitude*) de la prise de vue.
- des services décrits en WSDL qui permettent d'accéder à des statistiques sur les joueurs.
- des descriptions MPEG-21 DIA décrivant les caractéristiques réseaux (*Network/Delay*) ainsi que les caractéristiques du terminal d'accès (*AudioCapability*).
- des descriptions CC/PP introduisant des informations sur les préférences utilisateur.
- ainsi que des descriptions FOAF sur l'identité des utilisateurs et leurs relations avec d'autres membres d'une communauté.

Le métamodèle CAM4Home offre une vue commune sur l'ensemble des éléments précédemment cités : la description des contenus, des services, des profils utilisateur, communautaires et matériel. Ainsi, les développeurs d'applications multimédia n'ont plus à acquérir des compétences sur une large palette des standards apportant chacun leurs propres spécificités, philosophie, vocabulaires et structuration. Ils peuvent se focaliser sur une vision commune de l'ensemble des besoins classiquement rencontrés lorsque l'on vise des applications multicanaux.

Cette première partie de notre contribution émerge du projet européen CAM4Home du programme ITEA2-EUREKA réalisé entre 2008 et 2010 en regroupant 19 partenaires académiques et industriels travaillant dans la création et la distribution de contenus et services multimédia (voir l'Annexe A pour plus de détails). Nos contributions principales dans le projet CAM4Home concernent la modélisation et la spécification du métamodèle CAM4Home. Nous avons également contribué aux développements autour d'une plate-forme à base de Web services conçue autour du métamodèle. Cette plate-forme implémente cette vision unifiée et

extensible sur les métadonnées nécessaires à la réalisation des applications multimédia sociales et multi-support.

FIGURE 1.3: Illustration de l'hétérogénéité des descriptions

Le métamodèle ne résout que partiellement le problème d'hétérogénéité. Il offre une vue unifié sur les métadonnées, mais le problème de la disponibilité des métadonnées subsiste. La création des métadonnées est un processus coûteux en temps et ressources. Ainsi, il est essentiel de pouvoir réutiliser les métadonnées existantes, même si elles sont représentées dans d'autres formats. Pour cela, nous proposons une deuxième contribution qui vise à faciliter l'intégration des métadonnées issues des standards différents.

```
<Creator>
 <Name>
  <GivenName>Dupont</GivenName>
  <FamilyName>Rob</FamilyName>
 </Name>
</Creator>
```

```
<image_creator>
 <type>
  <given> Dupont </given>
  <family> Rob </family>
 </type >
</image_creator>
```

FIGURE 1.4: Représentation MPEG-7 et DIG-35 pour l'information *Creator*

Ce besoin est illustré par la Figure 1.4 où une même information relative au créateur d'une image (*Rob Dupont*) peut être exprimée de manière différente en MPEG-7 (à gauche) et en DIG-35. D'autres représentations de cette même information peuvent exister dans d'autres formats comme montré dans la Figure 1.3 où l'on retrouve également une description en Dublin Core et en CAM4Home. Dans ce contexte, afin d'interroger les quatre formats décrits précédemment, il est nécessaire d'avoir des connaissances précises sur les quatre descriptions citées précédemment pour pouvoir formuler les requêtes correspondantes, ainsi que sur les langages d'interrogation adéquats. Par exemple, pour accéder au nom et prénom du créateur de la prise de vue :

– en MPEG-7 nous devrons écrire une requête XQuery de la forme :

```
let $s := doc ("MPEG-7.xml")/Creator/Name
return concat ($s/GivenName/text(),' ',$s/FamilyName/text())
```

FIGURE 1.5: Requête de type XQuery en format MPEG-7

– en CAM4Home nous devons écrire une requête SPARQL de la forme :

```
Select ?nom, ?prenom
Where  ?upd rdf:type C4HUserPersonalDescription,
       ?udp name ?nom
       ?udp familyName ?prenom
```

FIGURE 1.6: Requête de type SPARQL en format CAM4Home

Afin de faciliter l'utilisation des métadonnées existantes, nous proposons une méthode d'intégration de métadonnées hétérogènes qui permet de trouver les mappings entre des métadonnées issues des standards hétérogènes de manière automatique. Cette deuxième proposition, conçue au sein de notre équipe en prolongement du projet CAM4Home ouvre la voie vers un accès uniforme aux métadonnées (comme illustré dans la Figure 1.7). Avec le calcul des correspondances entre standards, il est envisageable en partant de l'un d'entre eux de découvrir les éléments des autres standards (pas forcement maîtrisés par le développeur). Grâce à ce

système de médiation, il suffit à l'utilisateur de connaître un seul format de métadonnées. Dans ce travail de thèse, nous ne concentrons exclusivement sur le calcul des ces correspondances. La réécriture automatique de requêtes est un sujet à part entière qui vient se positionner en aval de travail que nous présentons dans cette thèse.

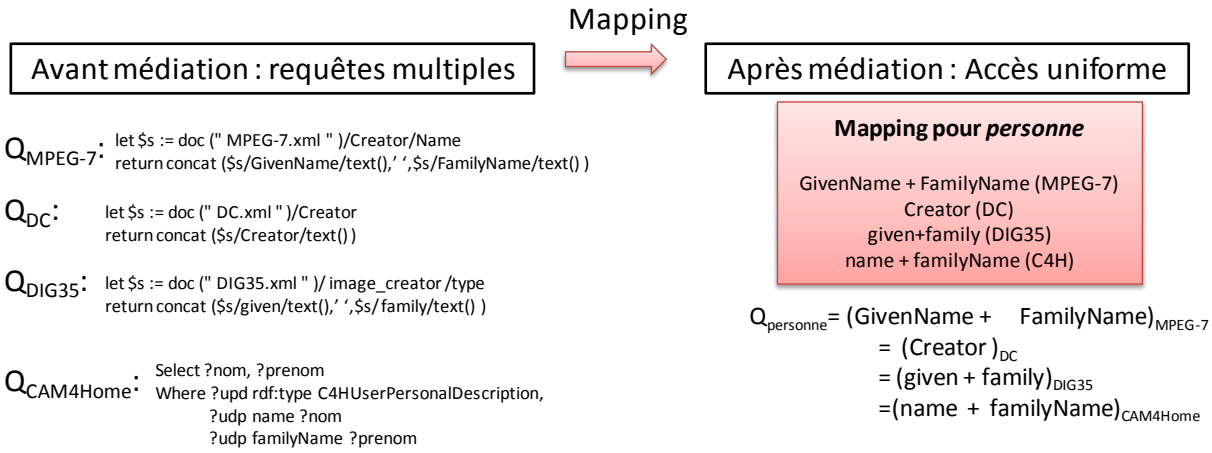

FIGURE 1.7: Utilisation du système de mapping pour fournir un accès uniforme

Dans la section suivante nous synthétisons les objectifs de notre travail, par rapport à chacune des contributions présentées ci-dessus.

1.2 Objectifs de la thèse

Notre premier objectif s'articule autour d'un métamodèle qui doit prendre en considération les points suivants :

- assurer une description sémantique riche des contenus multimédia.
- fournir les métadonnées nécessaires pour pouvoir livrer ces contenus sur des plateformes hétérogènes.

– fusionner les métadonnées hétérogènes.
– fournir une structure simple à utiliser qui assurent également l'extensibilité du métamodèle.
– intégrer des métadonnées spécifiques depuis d'autres standards.

Notre deuxième objectif est le développement d'une méthode automatique pour l'intégration des métadonnées. Nous souhaitons détecter les mappings entre les métadonnées requises par une application ou un utilisateur donné et celles encodées en différents formats (système de médiation des métadonnées). Ce système doit tenir compte de plusieurs contraintes :

– l'hétérogénéité structurelle où les schémas des métadonnées peuvent avoir des niveaux hiérarchiques et des types de données différents.
– l'hétérogénéité des langages de description, dûe à l'utilisation des langages différents pour l'encodage des métadonnées qui disposent des descriptions sémantiques et structurelles différentes.
– les mapping complexes (n : m), où un élément au niveau du schéma médiateur peut correspondre à plusieurs éléments au niveau des autres formats (et vice versa).
– minimiser l'intervention de l'utilisateur en fournissant un taux de précision élevé.

1.3 Contributions et originalité

Pour atteindre notre premier objectif nous avons proposé un métamodèle qui offre une description riche en termes d'information sémantique et contextuelle. Ce métamodèle, nommé CAM4Home [1] (Collaborative Aggregated Multimedia for Digital Home) regroupe plusieurs types de métadonnées nécessaires pour la création, la livraison, la modification, l'interprétation et la consommation des contenus multimédia (Figure 1.8).

1. http ://www.CAM4Home-itea.org/

FIGURE 1.8: Schéma général du métamodèle CAM4Home.

La structure du métamodèle CAM4Home est composée de plusieurs niveaux d'abstraction dont le but est de permettre d'intégrer plusieurs concepts de métadonnées. En plus, de la description offerte par le métamodèle CAM4Home, ce dernier permet également d'intégrer d'autres formats de métadonnées déjà existants (ex : MPEG-7, TV-Anytime, EXIF, etc).

Notre deuxième contribution qui est indépendante mais en cohérence avec le travail réalisé dans le cadre du projet CAM4Home, concerne une approche d'intégration dont le but est de trouver les correspondances sémantiques entre les métadonnées hétérogènes et les besoins des utilisateurs. Tout d'abord, cette approche commence par la transformation des métadonnées vers le même espace de présentation. Cela permet de résoudre le problème de l'hétérogénéité

au niveau des langages de description des métadonnées. Ensuite, les informations sémantiques, structurelles et syntaxiques disponibles au niveau des schémas sont utilisées pour trouver les mappings entre les métadonnées hétérogènes. Enfin, les mappings détectés sont retournés à l'utilisateur pour la validation. Le schéma général de cette approche est illustré par la Figure 1.9.

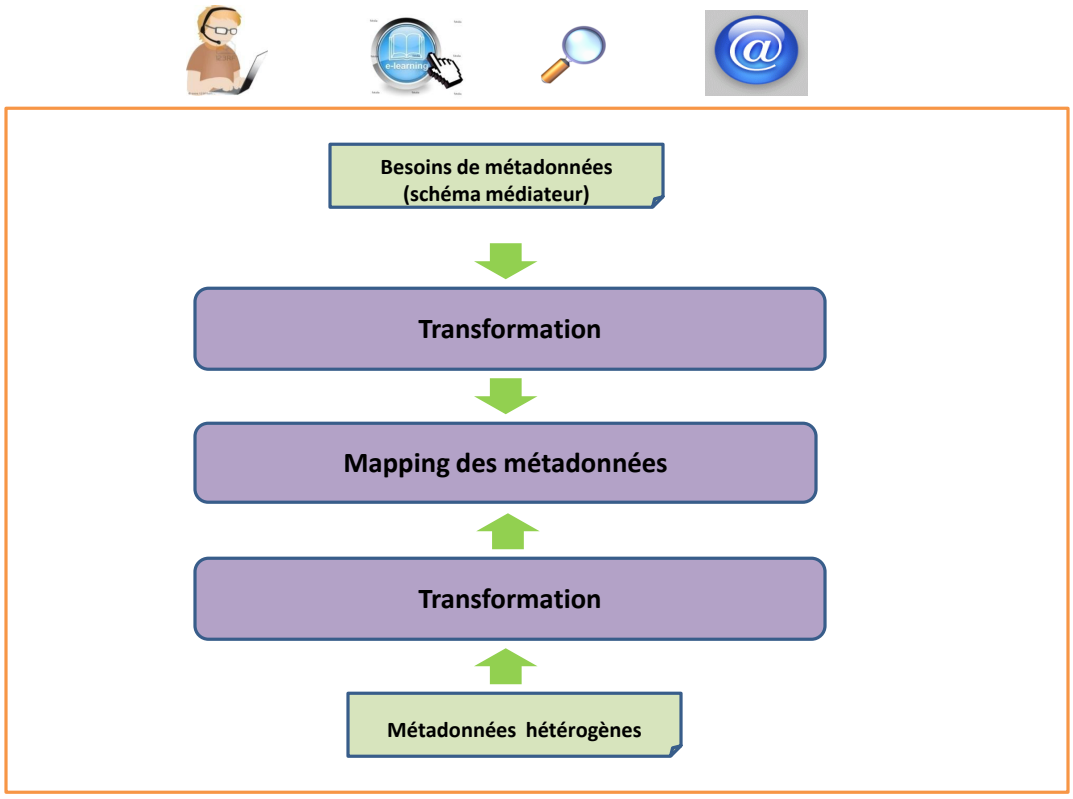

FIGURE 1.9: Schéma général de l'approche d'intégration automatique.

La contribution de notre travail se résume aux points suivants :

– *métamodèle unifié pour la globalité de cycle de vie des contenus multimédia :* Ce métamodèle définit un ensemble de descripteurs permettant la fusion des métadonnées hé-

térogènes. La structure de ce métamodèle offre son extensibilité grâce à la définition de plusieurs concepts génériques.

- *intégration multi-langages des métadonnées :* Nous proposons une méthode d'intégration des métadonnées baptisée *MuMIe* (Multi-level Metadata Integration), basée sur la matching des schémas. Cette méthode prend en entrée plusieurs types de schéma, spécifiés en utilisant des langages différents. Grace à des règles pré-définies, nous transformons les schémas en graphes étiquetés. Nous appliquons des mesures de similarité sur ces graphes pour trouver les correspondances entre les métadonnées.
- *détection des mappings complexes :* grâce à l'utilisation des propriétés hiérarchiques et sémantiques disponibles dans les schémas, plusieurs règles ont été définies afin de détecter les mappings complexes.
- *utilisation de plusieurs types d'information sémantique et structurelle :* les informations structurelles (relations entre les éléments des schémas de métadonnées) et sémantiques (noms des éléments et commentaires) des métadonnées ont été exploitées afin d'augmenter la précision du matching.
- *expérimentation :* nous nous sommes également attachés à l'étude expérimentale du comportement et des performances de MuMIe sur différents types de schéma de métadonnées et avec différentes mesures de qualité. En effet, nous avons réalisé nos tests sur des schémas hétérogènes sur les deux niveaux. Avec chacun des jeux de tests, nous avons étudié les résultats obtenus et validé notre proposition.

1.4 Plan de la thèse

Après l'introduction de ce mémoire de thèse qui précise le contexte et le schéma général de l'approche proposée, nous détaillons dans le Chapitre 2, les caractéristiques des métadon-

nées multimédia en donnant des exemples illustratifs sur les différents types de métadonnées considérées. Nous montrons également quelques exemples de technologies utilisées pour l'encodage de ces métadonnées.

Dans le Chapitre 3, nous discutons la notion d'interopérabilité dans les systèmes d'information. Les différents types d'hétérogénéité des métadonnées que nous adressons sont également présentées. Ensuite, nous passons en revue les méthodes de l'état de l'art, utilisées pour réaliser l'interopérabilité des métadonnées. Enfin, nous discutons les limitations de ces méthodes qui sont principalement réalisées manuellement.

Le Chapitre 4 présente un état de l'art sur le matching des schémas. Une étude comparative entre quelques approches de matching que nous considérons représentatives est faite dans ce chapitre. Nous montrons quelques inconvénients de ces méthodes et nous discutons leurs limitations, plus particulièrement dans le cadre de l'intégration des métadonnées multimédia hétérogènes.

Dans le Chapitre 5, nous proposons le métamodèle CAM4Home dédié à l'agrégation des métadonnées. Ce métamodèle offre une solution d'intégration des métadonnées nécessaires pour la diffusion des contenus multimédia sur des plate-formes hétérogènes. Nous discutons la structure et la spécification de ce métamodèle en mettant en exergue ses apports par rapport l'état de l'art.

Dans le Chapitre 6, nous présentons notre solution d'intégration automatique des métadonnées baptisée *MuMIe*. Nous montrons en détails les différentes étapes d'intégration qui sont basées sur une stratégie de matching novatrice où plusieurs types d'information sémantique, syntaxique et structurelle sont exploités.

Le Chapitre 7 présente une évaluation expérimentale de MuMIe. Dans ces expérimentations, nous présentons la vérité terrain que nous avons utilisée pour la validation nos résultats. Nous étudions également l'influence des différents paramètres sur les performances du sys-

tème. Enfin, une étude comparative est faite avec d'autres approches de matching.

Nous concluons ce mémoire de thèse en résumant les contributions scientifiques et les résultats obtenus. Nous présentons aussi des perspectives de recherche ouverte par ces travaux.

Chapitre 2

Introduction aux métadonnées

Sommaire

2.1 Introduction

Depuis quelques années, le multimédia occupe une place de plus en plus importante dans notre vie et requiert une utilisation plus large des métadonnées destinées à améliorer la gestion de ces types de contenus. Les métadonnées véhiculent diverses catégories d'informations relatives aux types de contenus multimédia, aux informations sémantiques associées et aux caractéristiques des terminaux qui utilisent ces contenus.

L'objectif de ce chapitre est d'expliquer très clairement la vision que nous avons des métadonnées, d'en déterminer les principaux éléments constitutifs et leur rôle dans la gestion des contenus multimédia. Nous mettrons ainsi en exergue certaines des définitions les plus communément reconnues et utilisées par la communauté multimédia. Nous illustrerons également dans ce chapitre quelques exemples portant sur les différents contextes d'utilisation des métadonnées et les langages de description fournis.

2.2 Définition des métadonnées

Dans la littérature, plusieurs définitions des métadonnées ont retenu notre attention. Selon [Gre03], les métadonnées sont des données structurées relatives à un objet, supportant des fonctions associées à cet objet. Ces données, représentées par un ensemble de descripteurs, sont structurées à l'aide d'un schéma de métadonnées. Selon [Sim02], les métadonnées sont des informations ajoutées à un objet pour en retirer des informations sur son contenu. Par exemple, dans le cas d'un fichier informatique, le nom, la taille et la date de création représentent des métadonnées de ce fichier. Celles-ci peuvent être classées par rapport aux fonctions qu'elles doivent soutenir (descriptives, structurelles, administratives, etc.) [Org04] ou par rapport à leur niveau d'abstraction sémantique (bas niveau, haut niveau) [WK03]. Les métadonnées de bas niveau ont moins de valeur pour l'utilisateur humain par rapport à celles du haut niveau (date

de création, résumé, etc). Les métadonnées de haut nouveau sont très riches sémantiquement, elles décrivent des entités sémantiques dans un monde narratif (événement, objet, concept, contexte, etc.) ainsi que leurs attributs et relations [BZCS01]. La qualité et l'expressivité de la sémantique des métadonnées est essentielle pour gérer efficacement les entités décrites par les métadonnées.

La définition des métadonnées, à la fois la plus répandue et la plus imprécise, est la suivante : les métadonnées sont des données représentant d'autres données, décrivant des objets numériques (document web, vidéo, image, etc.) ou non (livre, tableau, etc.). Des descriptions détaillées sur les différentes définitions liées à ce concept peuvent être trouvées dans [LAM01] où les auteurs ont effectué un état de l'art des diverses interprétations afférentes au terme de *métadonnées* et ont montré leur influence dans les différents systèmes.

2.3 Les différents types de métadonnées

Dans cette section, nous proposons une catégorisation des différents types de métadonnées faite en fonction des contenus et des contextes décrits par ces métadonnées. Des exemples illustratifs sont donnés pour chaque catégorie afin de montrer les spécificités des divers standards multimédia permettant de représenter la catégorie en question.

2.3.1 Description des contenus

Certains types de métadonnées sont génériques et ne dépendent pas du type de contenu qu'elles décrivent telle que la date de création d'un objet multimédia, le nom du créateur, le droit d'accès, etc. D'autres informations sont plus spécifiques à un type de contenu donné (ex : informations spatiales pour les images). Nous présentons ci-après quelques exemples correspondants à quatre types de contenus multimédia (vidéo, image, audio et document).

2.3.1.1 Document

Afin d'aider les bibliothécaires et les documentalistes dans l'archivage et la récupération des documents, plusieurs formats de métadonnées ont été proposés en fournissant plusieurs types de description pour ces documents. La Figure 2.1 montre un exemple de description de document au format Dublin Core [Hau05]. Les informations décrites portent sur le contenu (*Title*, *Description*, *Subject*) (lignes 5, 9-10), la propriété intellectuelle (*Creator*, *Rights*) (lignes 6 et 12) et la version (*Date*, *Identifier*) (lignes 8 et 11). La Figure 2.2 illustre un autre exemple de description des documents en format IPTC (International Press Telecommunications Council) [wg04]. Les attributs *contentCreated* et *altID* (lines 1-3 et 7-9) portent sur la version du document. La description des contenus ainsi que les droits d'accès sont définis par les attributs *headlines* et *copyrightNotice* respectivement (lines 4-6 et 10-12).

```
01: <OAI-PMH ....>
02:   ...
03:  <metadata>
04:   <oai_dc:dc ...>
05:     <dc:title>introduction à java </dc:title>
06:     <dc:creator>David James</dc:creator>
07:     <dc:coverage>France</dc:coverage>
08:     <dc:date>2006</dc:date>
09:     <dc:description> innitiation à java + exercices résolus </dc:description>
10:     <dc:subject>programmation</dc:subject>
11:     <dc:identifier>http://rangiroa.essi.fr/cours/langage/99-java-intro.pdf</dc:identifier>
12:      <dc:right>accès libre </dc:right>
13:   </oai_dc:dc>
14:  </metadata>
15:    …
16: </OAI-PMH>
```

FIGURE 2.1: Exemple de description d'un document en format Dublin Core.

```
01:  <contenCreated>
02:     2006
03:  </contentCreated>
04:  <headline >
05:     initiation à java + exercices résolus
06:  </headline >
07:  <altID >
08:     http://rangiroa.essi.fr/cours/langage/99-java-intro.pdf
09:  </altID >
10:  <copyrightNotice >
11:     accès libre
12:  </copyrightNotice >
```

FIGURE 2.2: Exemple de description d'un document en format IPTC.

2.3.1.2 Audio

La navigation dans les fichiers audio nécessite également une description sémantique afin de faciliter la recherche d'informations dans ce type de contenu multimédia.

```
01:<core:AudioElementMetadata>
02: <core:hasAppearingConcepts>
03:  <core:AppearingConcept>
04:   <core:name>
05:    Person
06:   </core:name>
07:   <core:description>
08:    Monsieur Dupont
09:   </core:description>
10:   <core:uri >
11:    http://www.facebook.com/dupont
12:   </core:uri>
13:   <core:uriDescription>
14:      Un pointeur vers le profil facebook de Monsieur Dupont
15:   </core:uriDescription>
16:   <core:beginTimePoint >
17:    125
18:   </corebeginTimePoint>
19:   <core:endTimePoint >
20:    21
21:   </core:endTimePoint>
22:  </core:AppearingConcept>
23: </core:hasAppearingConcepts>
24:</core:AudioElementMetadata>
```

FIGURE 2.3: Exemple d'une description audio en format CAM4Home

A titre d'exemple, nous montrons dans la Figure 2.3 une description en format CAM4Home d'un segment audio contenant les informations sur le début et la fin de ce segment (lignes 16-18 et 19-21) ainsi que le nom et les coordonnées de la personne à qui appartient le discours (lignes 4-6, 7-9 et 10-12).

2.3.1.3 Image

La manipulation d'images nécessite l'utilisation de métadonnées permettant la recherche et l'organisation de ce type de contenu multimédia [DSL05]. A ce stade, plusieurs formats de description des métadonnées liées aux images ont été proposés (ex : DIG35 [wg02a] , EXIF [wg02b], etc.). Nous présentons dans la Figure 2.4 une description d'une image en format DIG35 à propos de *Zinedine Zidane* (lignes 20-23), l'image est en format JPEG et à une résolution de 1600 x 1200 (ligne 5 et 9-12). Le format de métadonnées décrit également les coordonnées GPS du lieu où l'image a été prise (lignes 31 et 32), la position (X, Y) où la personne concernée est localisée dans l'image (lignes 26-27) ainsi que le format d'encodage de sa couleur qui est le RGB (lignes 16-18).

2.3.1.4 Vidéo

En raison du nombre important de documents audiovisuels numérisés disponibles sur Internet, le câble et les bouquets de télévision par satellite, l'utilisation des métadonnées est devenue nécessaire pour faciliter la recherche et la gestion de ces types de contenus. Plusieurs standards de description vidéo ont été proposés (ex: MPEG-7 [CPSZ01], TV-Anytime [PS00], etc). Ces standards fournissent des informations sur :

- La création et la production des objets multimédia.
- L'utilisation des contenus multimédia : elles comportent les droits d'accès, des informations financières, des droits de publication, etc.

```
01:<DIG35>
02:  ......
03: <BASIC_IMAGE_INFO>
04:  <FILE_FORMAT>
05:   <FILE_NAME>image.jpg</FILE_NAME>
06:   <FORMAT_TYPE>JFIF</FORMAT_TYPE>
07:   <VERSION>1.02</VERSION>
08:  </FILE_FORMAT>
09: <IMAGE_SIZE>
10:  <WIDTH>1600</WIDTH>
11:  <HEIGHT>1200</HEIGHT>
12: </IMAGE_SIZE>
13:  <COMPRESSION>JPEG</COMPRESSION>
14: </BASIC_IMAGE_INFO>
15: <COLOR_INFO>
16:  <COLORSPACE>
17:   <PROFILE_NAME>sRGB</PROFILE_NAME>
18:  </COLORSPACE>
19: </COLOR_INFO>
20: <PERSON_NAME>
21:  <NAME_COMP TYPE="Given">Zine eddine</NAME_COMP>
22:  <NAME_COMP TYPE="Family"> Zidane </NAME_COMP>
23: </PERSON_NAME>
24: <POSITION>
25:  <RECT>
26:   <X>0.2</X>
27:   <Y>0.2</Y>
28:   <WIDTH>0.1</WIDTH>
29:   <HEIGHT>0.4</HEIGHT>
30:  </RECT>
31:  <GPS_LONGITUDE>+40.689060 <GPS_LONGITUDE/>
32:  <GPS_ALTITUDE> +11.638713<GPS_ALTITUDE/>
33:  .......
34:</DIG35>
```

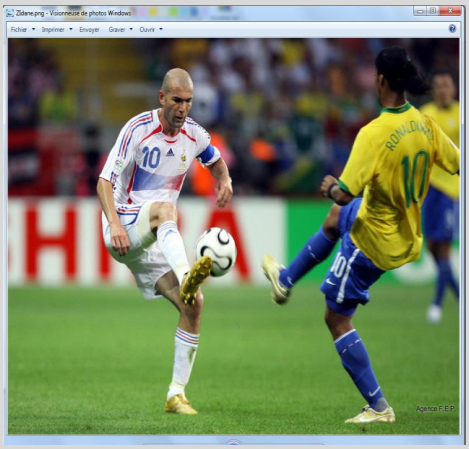

FIGURE 2.4: Exemple d'une description Image en format DIG35.

– La description des caractéristiques de stockage : format, éléments pour identifier le média, etc.
– La description sémantique : ces informations décrivent les segments qui peuvent représenter des composantes spatiales, temporelles ou spatio-temporelles du contenu audio-visuel. Chaque segment peut être décrit par les caractéristiques suivantes : la couleur, la texture, la forme, la motion, d'autres caractéristiques audio, etc, et quelques informations

sémantiques élémentaires.

La Figure 2.5 présente une description, au format MPEG-7, d'une vidéo de *Lionnel Messi* (lignes 7-8). La vidéo contient deux mots-clés (*foot*, *goal*) (lignes 10-11) , les éléments *mediaTimepoint* et *mediaTimeDuration* (lignes 14-17) décrivent le temps de début du segment vidéo en question ainsi que sa durée. Ces informations sont utilisées pour la recherche sémantique des contenus multimédia.

```
01:  <MediaLocator>
02:   <MediaUri>video_Messi.mpg</MediaUri>
03:  </MediaLocator>
04:   ……
05:  <VideoSegment>
06:   <TextAnnotation>
07:  <FreeTextAnnotation>Lionnel Messi
08:  </FreeTextAnnotation>
09:  <KeywordAnnotation>
10:   <Keyword>foot</Keyword>
11:   <Keyword>goal</Keyword>
12:  </KeywordAnnotation>
13:  </TextAnnotation>
14: <MediaTime>
15:  <MediaTimePoint>T00:05:41:001F1000</MediaTimePoint>
16:  <MediaDuration>PT00H04M17S999N1000F</MediaDuration>
17: </MediaTime>
18: </VideoSegment>
```

FIGURE 2.5: Exemple d'une description vidéo en format MPEG-7.

2.3.2 Description des contextes de diffusion

Le contexte de diffusion comprend les caractéristiques physiques des terminaux ainsi que les profils des utilisateurs. Pour faire face aux hétérogénéités des appareils utilisés pour le stockage, la livraison, la transmission, le codage et la consommation des objets multimédia, plusieurs types de métadonnées dédiées à la description des contextes ont été développés. Des services destinés à l'adaptation ont recours à ces métadonnées afin de choisir les contenus convenables au contexte de l'utilisateur.

2.3.2.1 Profil matériel

Le profil matériel comprend principalement les caractéristiques physiques des supports de transmission (réseaux) et des terminaux (mobiles, téléviseurs, etc). Nous montrons dans la Figure 2.6 une description en format MPEG-21 [BdWH+03] des caractéristiques audio et d'affichage qui seront utilisées pour adapter les contenus destinés à un utilisateur donné (résolution d'affichage, caractéristiques audio, etc). Un deuxième exemple dans ce contexte est montré dans la Figure 2.7. Cette Figure est un encodage en format CC/PP (Composite Capability/Preference Profiles) [SH01] des caractéristiques d'affichage (résolution).

```
01:<DIA>
02: <Description xsi:type="UsageEnvironmentType">
03:  <UsageEnvironmentProperty xsi:type="TerminalsType">
04:   <Terminal>
05:    <TerminalCapability xsi:type="DisplaysType">
06:     <Display id="primary_display">
07:      <DisplayCapability xsi:type="DisplayCapabilityType">
08:       <Mode>
09:        <Resolution horizontal="720" vertical="480"/>
10:       </Mode>
11:      </DisplayCapability>
12:     </Display>
13:    </TerminalCapability>
14:   <TerminalCapability xsi:type="AudioOutputsType">
15:    <AudioOutput xsi:type="AudioOutputType">
16:    <AudioOutputCapability xsi:type="AudioOutputCapabilitiesType"
17:    lowFrequency="30" highFrequency="8000" numChannels="2"/>
18:     </AudioOutput>
19:    </TerminalCapability>
20:   </Terminal>
21:  </UsageEnvironmentProperty>
22: </Description>
23:</DIA>
```

FIGURE 2.6: Exemple de description d'un profil matériel avec MPEG-21

2.3.2.2 Profil utilisateur

Ce type de métadonnées peut servir à guider l'adaptation d'un contenu présenté à un utilisateur donné en fonction des logiciels disponibles sur son terminal ou bien de ces intérêts

```
01:<ccpp:component>
02:    <rdf:Description
03:       rdf:about="http://www.example.com/profile#TerminalHardware">
04:      <rdf:type
05:        rdf:resource="http://www.example.com/schema#HardwarePlatform" />
06:      <ex:displayWidth>320</ex:displayWidth>
07:      <ex:displayHeight>200</ex:displayHeight>
08:    </rdf:Description>
09: </ccpp:component>
```

FIGURE 2.7: Exemple de description d'un profil en format CC/PP

et préférences. Un profil comporte un certain nombre de noms, d'attributs et de valeurs associées, lesquels sont utilisés par un serveur afin de déterminer la forme la plus appropriée de la ressource à remettre à un client. Ce profil est structuré, permettant au client de décrire ses capacités par rapport à un profil normalisé accessible au serveur d'origine ou à un autre émetteur de données de ressources à partir d'un ensemble restreint de caractéristiques qui s'ajoutent ou se distinguent du profil normalisé.

La Figure 2.6 montre un exemple de description du profil en format CAM4Home[BAB+10]. Dans cet exemple, plusieurs informations concernent le profil de l'utilisateur. Parmi ces informations on peut citer le logiciel disponible chez l'utilisateur (lignes 5-11), les types de contenus préférés (ligne 14-18). Une autre description en format CC/PP [SH01] est également montrée dans la Figure 2.9. Cette figure décrit les types et les versions des logiciels disponibles chez l'utilisateur.

2.4 Les éléments constitutifs des métadonnées

En analysant les exemples de métadonnées fournis précédemment, nous pouvons identifier les caractéristiques communes suivantes : chaque description est composée d'un ensemble d'éléments (ex : Title, Synopsis, Name) ainsi que de la valeur de ces éléments (ex : Lionel Messi, Une compilation...., Sports). Les éléments sont organisés dans le cadre d'un schéma de

```
01:<C4HDeviceMetadataContainer>
02: </C4HUserProfile>
03:  ...
04:   <C4HSoftwareDescription>
05:    <hasAvailableSoftwares>
06:     <AvailableSoftware>
07:      <software>Mozilla Firefox</software>
08:      <version>3.0.4</version>
09:     <AvailableSoftware>
10:    </hasAvailableSoftwares>
11:   </C4HSoftwareDescription>
12:   ...
13: <C4HUserPreferenceDescription>
14:  <hasPreferredMedia>
15:    <PreferredMedia>
16:     <mediaMimeType>image/gif</mediaMimeType>
17:     <preferenceOrder>2</preferenceOrder>
18:    </PreferredMedia>
19:    ...
20:    </hasPreferredMedia>
21:    ...
22:   </C4HUserPreferenceDescription>
23: </hasUserPreferenceDescription>
24: ...
25:</C4HUserProfile>
```

FIGURE 2.8: Exemple de description d'un profil utilisateur en format CAM4Home

```
01:  <ccpp:component>
02:    <rdf:Description
03:       rdf:about="http://www.example.com/profile#TerminalSoftware">
04:     <rdf:type
05:        rdf:resource="http://www.example.com/schema#SoftwarePlatform" />
06:     <ex:name>EPOC</ex:name>
07:     <ex:version>2.0</ex:version>
08:     <ex:vendor>Symbian</ex:vendor>
09:    </rdf:Description>
10:  </ccpp:component>
```

FIGURE 2.9: Exemple de description d'un profil utilisateur en format CC/PP

métadonnées pouvant être standardisé, comme dans le cas du TV-Anytime ou MPEG-7. Nous pouvons également déduire que les éléments des métadonnées ont été préalablement spécifiés en utilisant un langage déterminé. Le Schéma XML [FY04] pour TV-Anytime et le Schéma RDF [BG04] pour CAM4Home. La Figure 2.10 illustre les trois éléments constructifs des métadonnées et le liens entre eux : l'ensemble des valeurs des attributs (*instances*), la définition

des éléments (*schémas*) et le langage de définition du schéma (*langage de description*). Nous nous concentrons, dans ce qui suit, sur chacun de ces éléments.

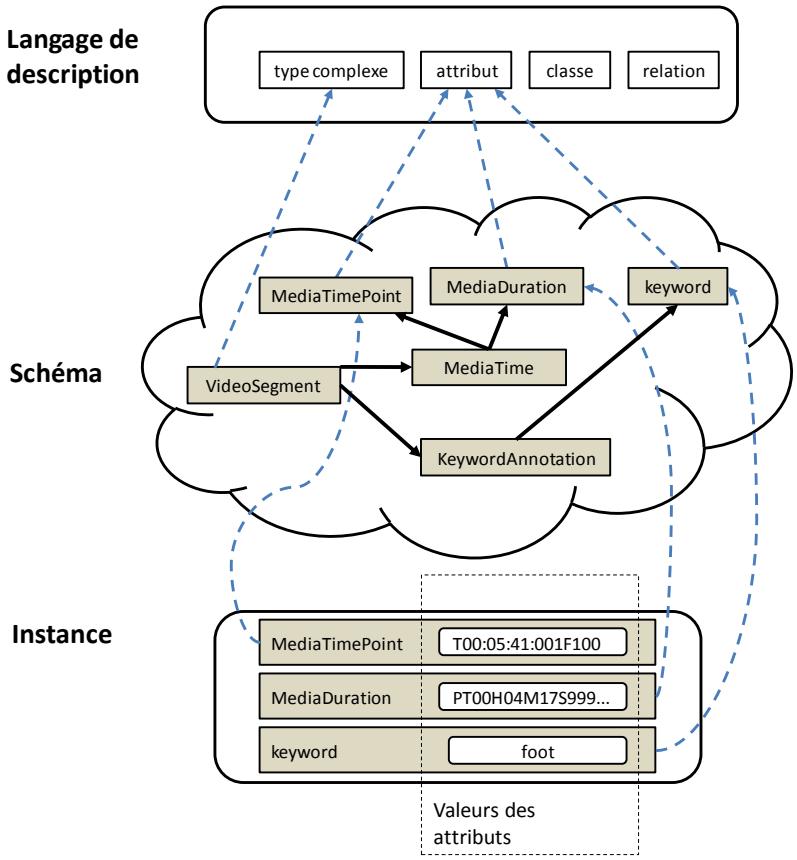

FIGURE 2.10: Les éléments constitutifs des métadonnées

2.4.1 Les langages de description

La définition sémantique du terme "langage" implique que, pour communiquer les uns avec les autres, un accord sur la signification des primitives de langage doit exister. C'est également le cas pour les langages de description des schémas. Généralement, il existe des standards pour

ces langages ou au moins une sorte de consensus [1]. Les langages de description sont classés selon leur pouvoir expressif, sur trois niveaux (Figure 2.11) :

- *Niveau nommage et adressage :* il est représenté par le standard d'adressage des ressources du Web URI (Universal Resource Identifier) et la norme Unicode pour le codage des caractères.
- *Niveau syntaxique :* il est représenté par la définition des espaces de noms qui permettent d'identifier les ressources du Web, le langage XML, XML schéma et le langage de requêtes XML Query.
- *Niveau sémantique :* ce niveau est représenté d'une part par les langages de représentation d'ontologies RDF/RDFS, OWL et SKOS, et d'autre part par les langages de règles, de logique, de preuves et de confiance KL-One, KIF et DAML + OIL [AvH08].

Dans le reste de cette section, nous présentons en détail les langages de description cités précédemment en commençant par le Schéma XML qui est très répondu parmi les standards du web.

2.4.1.1 XML/Schéma XML

L'énorme évolution de la technologie nécessite une description des données d'une manière à faciliter leur échange entre les différents systèmes d'information. XML (Extensible Markup Language) [FY04] est proposé comme un moyen d'accomplir cette tâche, en utilisant un langage de marquage basé sur le texte pour décrire la structure des données ordonnées. Les fichiers XML sont liés à des définitions des types de document (DTDs) qui décrivent la structure des documents XML. Mais les DTDs manquent d'expressivité pour décrire des structures de données de haut niveau. Le Schéma XML (XSD) [FY04] est un ensemble plus riche en structures, en types, et en contraintes pour décrire des données. Il propose des mécanismes de haut

1. Le consortium W3C, par exemple, ne publie pas des *standards* mais des *recommandations*.

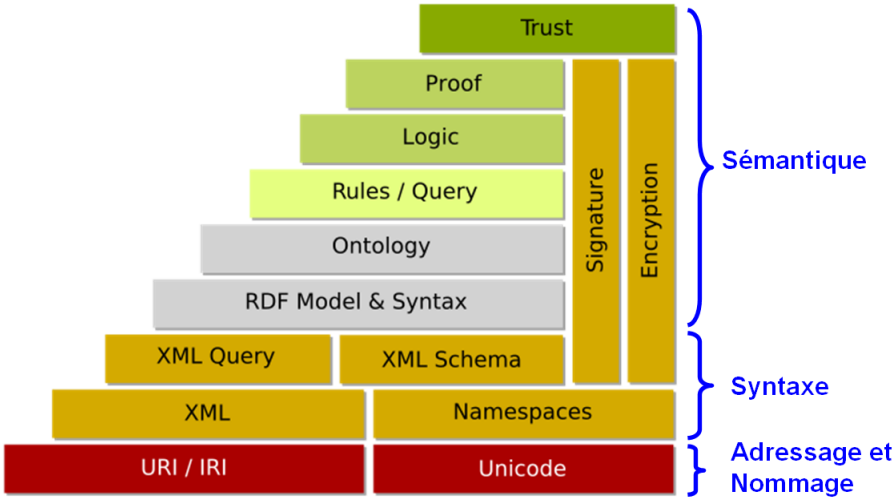

FIGURE 2.11: Les catégories des langages de description.

niveau pour la définition et la structuration de documents XML. La Figure 2.12 montre une spécification XML Schéma d'un type complexe nommé *person*. Ce type est composé d'autres éléments précisant l'identité de la personne (*name*, *address*, *city* et *country*). Cette spécification montre que ces éléments sont de type *string* et doivent apparaître dans l'ordre défini par la balise *xsd:sequence*.

```
01:<xs:element name="Person">
02:  <xs:complexType>
03:    <xs:sequence>
04:      <xs:element name="name" type="xs:string"/>
05:      <xs:element name="address" type="xs:string"/>
06:      <xs:element name="city" type="xs:string"/>
07:      <xs:element name="country" type="xs:string"/>
08:    </xs:sequence>
09:  </xs:complexType>
10:</xs:element>
```

FIGURE 2.12: Exemple d'une description en format Schéma XML.

2.4.1.2 RDF et RDF Schéma

RDF [BG04] (Resource Description Framework) est un formalisme graphique pour représenter des métadonnées. Il est basé sur la notion de triplet (sujet, prédicat, objet). Le sujet et l'objet sont des ressources liées par le prédicat. RDF peut utiliser la syntaxe XML, mais il ne donne aucune signification spécifique pour le vocabulaire comme sous classe de, ou le type. Les primitives de modélisation offertes par RDF sont très basiques. RDF Schéma (RDFS) [BG04] est un langage qui étend RDF avec un vocabulaire de termes et les relations entre ces termes, par exemple : *Class*, *Property*, *type*, *subClassOf*, *subPropertyOf*, *range* et *domain*. RDFS est reconnu comme un langage d'ontologie qui définit :

– Des classes et des propriétés.

– Les sous-classes, les super-classes, les sous-propriétés, et les super-propriétés.

La Figure 2.13 présente une description de type Schéma RDF. Elle est composée de trois classes ; une classe mère (*person*) et deux sous classes (*woman* et *man*). Une propriété de type *mariedWith* a été définie pour relier les classes *woman* et *man*.

2.4.1.3 OWL

En réponse aux limitations de RDF, le groupe de travail sur le Web Sémantique du W3C à proposé OWL [2] qui peut être utilisé pour représenter explicitement le sens des termes des vocabulaires et des relations entre ces termes. OWL vise également à rendre les ressources sur le Web aisément accessibles aux processus automatisés, d'une part en les structurant d'une façon compréhensible et standardisée, et d'autre part en leur ajoutant des méta-informations. Pour cela, OWL a des moyens plus puissants pour exprimer la signification et la sémantique que XML, RDF. De plus, OWL tient compte de l'aspect diffus des sources de connaissances et permet à l'information d'être recueillie à partir de sources distribuées, notamment en permet-

2. http ://www.w3.org/TR/owl-features/

```
01:<rdf:RDF>
02: <rdf:Description rdf:ID="person">
03:  <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
04: </rdf:Description>
05: <rdf:Description rdf:ID="man">
06:  <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
07:  <rdfs:subClassOf rdf:resource="#person"/>
08: </rdf:Description>
09: <rdf:Description rdf:ID="woman">
10:  <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
11:  <rdfs:subClassOf rdf:resource="#person"/>
12: </rdf:Description>
13: <rdf:Property rdf:about="#mariedWith" >
14:  <rdfs:domain rdf:resource="#man"/>
15:  <rdfs:range rdf:resource="#woman"/>
16: </rdf:Property>
17:</rdf:RDF>
```

FIGURE 2.13: Exemple d'une description en format Schéma RDF.

tant la mise en relation des ontologies et l'importation des informations provenant explicitement d'autres ontologies. La Figure 2.14 montre un exemple d'une spécification OWL. Cette description définie une propriété d'équivalence entre les classes *Image* et *Picture* ainsi qu'une propriété de disjonction entre la classe *man* et les deux classes *woman* et *animal*.

```
01: <owl:Class rdf:ID="Image">
02:   <owl:equivalentClass rdf:resource="&Picture;Image"/>
03: </owl:Class>
04: <owl:Class rdf:ID="man">
05:   <owl:disjointWith rdf:resource="#woman"/>
06:   <owl:disjointWith rdf:resource="#animal"/>
07: </owl:Class>
```

FIGURE 2.14: Exemple d'une description OWL.

2.4.1.4 Autres langages

Plusieurs autres langages existent dans la littérature, parmi eux on peut citer le DAML + OIL qui est un langage construit sur des normes antérieures du W3C telles que RDF et RDF Schéma, et étend ces langages avec des primitives de modélisation plus riches. DAML+OIL a été conçu à partir du langage d'ontologie DAML-ONT (DARPA Agent Modelling Language-Ontology, Octobre 2000) en vue de combiner plusieurs composants du langage OIL [HFB+00]. OIL [3] (Ontology Inference Language) est une représentation basée sur le Web, et une couche d'inférence pour des ontologies. Il combine les primitives de modélisation des langages à base de cadres (frames) avec la sémantique formelle et le raisonnement basé sur la logique de description.

Le KL-ONE [BBM+89] est un exemple de langage basé sur la logique de description [BCM+03]. Il s'agit d'une formalisation de représentation de la connaissance à base de cadres (frame-based). Ce système maintient la définition des concepts par un simple nommage, et l'indication de la correspondance des concepts dans une hiérarchie de généralisation/spécialisation. De nouveaux termes peuvent être définis par des opérations de conjonction des concepts. Par exemple l'opérateur « and » peut être utilisé pour préciser qu'un nouveau concept est une spécialisation commune de plusieurs autres concepts. De nouveaux rôles peuvent être introduits pour représenter les relations qui peuvent exister entre des individus dans le domaine modélisé. Les définitions des concepts peuvent inclure des restrictions sur les valeurs possibles, sur les nombres de valeurs, ou sur le type de valeur quun rôle peut avoir pour un concept.

Le SKOS est un modèle de données permettant la représentation standard et la publication de vocabulaires structurés tels les thésaurus et les taxonomies. SKOS est construit sur la base du langage RDF. Il s'agit d'une recommandation du W3C depuis août 2009. Il peut être utilisé seul ou combiné avec des langages tel que OWL. SKOS définit une classe Concept de type *owl*

3. http ://www.ontoknowledge.org/oil/

:Class à laquelle peuvent se rattacher de nombreuses propriétés.

KIF (Knowledge Interchange Format)[Mar02] est un langage basé sur les prédicats du premier ordre avec des extensions pour représenter des définitions et des connaissances. Il possède plusieurs atouts, entre autres :

- sa lisibilité, il est facilement compréhensible par un humain (même si ce n'est pas son but premier) ;
- sa facilité à être utilisé. Il est compréhensible par une machine ce qui permet un raisonnement logique ;
- il permet d'introduire de nouvelles représentations de connaissance, sans changer de langage.

2.4.2 Les schémas

Un autre élément constitutif des métadonnées est leur définition. Cet élément est appelé schéma de métadonnées. Il s'agit tout simplement d'un ensemble d'éléments porteurs d'une *définition sémantique* précise, éventuellement *reliés par une structure* [RB01]. La sémantique d'un schéma est définie par la signification de ses éléments. Un schéma définit généralement le nom des éléments avec leur sémantique, et éventuellement les contraintes imposées à ces éléments (ex : cardinalité, valeurs autorisées). En matière de codage, un schéma peut définir des règles syntaxiques. Si aucune règle n'est définie par le schéma, ce dernier est appelé *syntaxe indépendante* [HK10].

À ce stade, il est important de mentionner que la notion de schéma de métadonnées, provenant du domaine des bases de données, est souvent nommée différemment dans d'autres contextes ou communautés. Ainsi, dans le domaine de la gestion de connaissances, on utilise le mot *vocabulaire des métadonnées* [BS01] pour ce que nous appelons schéma de métadonnées. Ces communautés ont tendance à considérer les métadonnées d'un point de vue linguis-

tique plutôt que technique. Le terme *ontologie* [KP92] est une autre notion très courante dont l'origine se trouve dans le domaine de l'intelligence artificielle. Elle se définit comme la *spécification d'une conceptualisation* [Gru93]. À la base, une ontologie se définit de la même manière qu'un schéma, à savoir un ensemble d'éléments reliés par une structure. Les auteurs dans [NK04] ont par ailleurs identifié plusieurs caractéristiques qui distinguent les ontologies des schémas de métadonnées :

- En premier lieu, les ontologies sont des systèmes logiques définissant un ensemble d'axiomes permettant un raisonnement automatisé sur un ensemble de faits donnés.
- Deuxièmement, les ontologies ont généralement des modèles de données riches en termes d'informations sémantiques utilisées dans la conception. Cela implique l'utilisation de plusieurs primitives de représentation capables de décrire des telles informations (ex : propriétés inverses, classes disjointes, etc).
- Troisièmement, le développement des ontologies est une approche beaucoup plus décentralisée et collaborative que le développement du schéma, ce qui génère un potentiel de réutilisation d'autres définitions d'ontologies existantes. En conséquence de ces différentes perceptions, les termes utilisés peuvent se trouver sous des noms différents pour définir les métadonnées (ex : les langages de définition des vocabulaires et ontologies).

En dépit de ces perceptions différentes, nous croyons que le terme *schéma de métadonnées* est approprié dans le cadre de cette thèse puisque nos travaux portent sur l'interopérabilité des métadonnées, principalement d'un point de vue technique. Toutefois, dans ce travail, le terme *schéma de métadonnées* ne concerne pas uniquement les schémas traditionnels de données, mais il s'agit aussi d'un dénominateur commun pour tous les termes mentionnés précédemment.

2.4.3 Les instances

Les instances matérialisent les métadonnées (classes, attributs, propriétés, etc) par un ensemble de valeurs. La création d'une instance d'un schéma est soumise à des règles imposées par les schémas de métadonnées auxquels elles sont reliées. Si une telle relation existe, on peut dire que les métadonnées correspondent à un schéma.

Des outils permettant de vérifier la validité d'une instance par rapport à son schéma sont employés afin de préserver la cohérence des métadonnées. A titre d'exemple, on peut citer le Pellet [SPG+07] qui transforme les instances en une présentation basée sur la logique de description [BCM+03]. Des règles de déductions sont ensuite utilisées en fonction du schéma afin de vérifier la validité d'une instance donnée par rapport à ce schéma.

2.5 Conclusion

Ce chapitre a présenté le concept de métadonnées en commençant par discuter quelques définitions qui nous ont parues pertinentes. Nous avons exposé ensuite quelques particularités des métadonnées. Ainsi, la Section 2.4 met en évidence les éléments constitutifs des métadonnées. A travers cette section, nous avons vu qu'il existe différentes techniques de modélisation, de représentation et d'encodage des métadonnées sur trois niveaux. La diversité des techniques mentionnées dans ce chapitre est la cause de l'hétérogénéité des métadonnées que nous détaillons dans le chapitre suivant. Nous soulevons les différents types d'hétérogénéité des métadonnées. Nous montrons également toutes des solutions de l'état de l'art que nous jugeons pertinents pour palier à ces multiples problèmes d'hétérogénéité.

Chapitre 3

L'interopérabilité des métadonnées

Sommaire

3.1 Introduction

L'omniprésence des ressources numériques et des terminaux à caractéristiques variées a généré un intérêt croissant pour l'utilisation des métadonnées, allant de la description de la sémantique présente dans les contenus (ex : concept dans une image, place où se trouve le concept, etc.) jusqu'à la description des caractéristiques relatives aux entités physiques transmettant ou consommant les contenus multimédia (ex : réseau, téléphone mobile, télévision, etc.).

Si on anticipe la croissance des métadonnées dans les années à venir, on peut prévoir qu'il sera de plus en plus difficile d'accéder de manière uniforme aux objets multimédia en raison du nombre de communautés indépendantes qui combinent les termes descriptifs des métadonnées à partir de plusieurs vocabulaires, et utilisent différentes structures et langages de description comme nous l'avons illustré dans le chapitre précédent. Les objets multimédia sur le Web (ex : une vidéo ou une image) et les contextes d'utilisation (ex : préférences des utilisateurs) peuvent être décrits par plusieurs types de métadonnées comme illustré dans la Figure 3.1. L'interprétation de ces métadonnées par un utilisateur nécessite une connaissance à priori de tous les formats possibles, ce qui n'est pas facilement réalisable. L'interopérabilité des métadonnées est donc un enjeu crucial.

La littérature montre que le terme "interopérabilité des métadonnées" a un sens très large, et implique un certain nombre de problèmes à résoudre [HK10]. Au premier niveau, les différents agents d'un système d'information doivent être capables d'accéder et d'échanger les métadonnées. Au deuxième niveau, chaque agent doit pouvoir traiter les informations portées par les métadonnées reçues d'un autre agent. Au troisième niveau, on doit s'assurer que les machines et les individus interprètent correctement la sémantique des métadonnées.

Nous débutons ce chapitre par préciser la définition de l'interopérabilité, puis nous passons

en revue les méthodes utilisées pour réaliser l'interopérabilité des métadonnées.

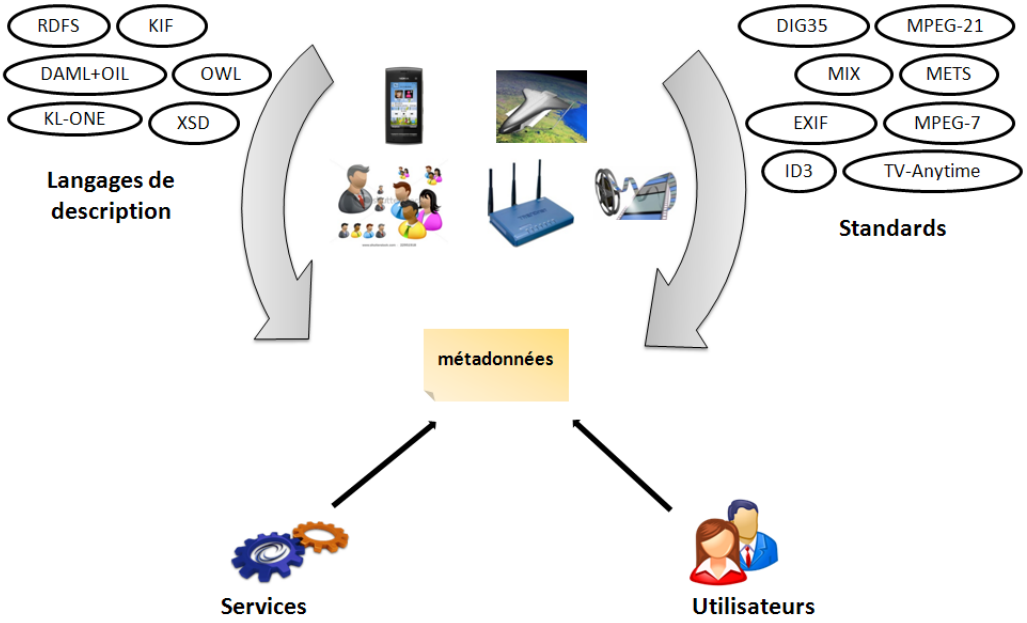

FIGURE 3.1: Schéma illustrant les différentes sources d'hétérogénéité.

3.2 Définition de l'interopérabilité

Le terme interopérabilité désigne l'accès uniforme et transparent à des données hétérogènes à l'aide de mécanismes assurant un fonctionnement coordonné et coopéré entre les différents systèmes d'information [EPV10].

Dans la littérature, nous pouvons également trouver des définitions similaires : Oxford [1] présente l'interopérabilité comme l'habileté à utiliser des ressources dans la conjonction. Webster [2] la définit par l'habilité d'un système à utiliser les parties d'un autre système. D'autres

1. http ://www.oxfordreference.com
2. http :http ://www.websters-online-dictionary.org

définitions existent dans les divers domaine. Par example, dans le domaine des bibliothèques numériques, les auteurs dans [TCD+02] considèrent l'interopérabilité comme le potentiel à franchir les frontières entre les différents contextes des systèmes d'information. D'autres auteurs du même domaine [Org04], définissent l'interopérabilité comme la faculté d'échanger des métadonnées entre deux ou plusieurs systèmes sans ou avec une perte minimale d'informations et sans aucun effort particulier pour les deux systèmes.

Plusieurs classifications de l'interopérabilité existent dans l'état de l'art. Dans le domaine de l'entreprise, les auteurs dans [ISO02] définissent trois types d'interopérabilité : l'interopérabilité des données, l'interopérabilité des applications et l'interopérabilité des processus. La notion d'interopérabilité peut en outre être subdivisée en trois niveaux principaux selon [TCD+02] : le niveau du transport et de l'échange d'information (ex : les protocoles), le niveau de représentation des métadonnées (ex : la syntaxe), et le niveau des schémas de métadonnées. Selon [EIF04], l'interopérabilité peut être divisée en trois niveaux : le niveau organisationel, le niveau technique et le niveau sémantique. L'auteur dans [Mil00] présente aux côtés de l'interopérabilité *technique* et *sémantique* plusieurs catégories d'interopérabilité : interopérabilité politique, humaine, intercommunalité, juridique et internationale.

L'auteur dans [Euz01b] donne une définition issue du domaine de bases de données, dans laquelle cinq niveaux d'interopérabilité ont été définis :

- *encodage* : la capacité de créer une représentation en caractère.
- *lexical* : la capacité créer une représentation en mots (ou symboles).
- *syntaxique* : la capacité de faire une représentation sous forme des phrases structurées.
- *sémantique* : la capacité de créer une signification propositionnelle d'une représentation.
- *sémiotique* : la capacité de créer une signification pragmatique d'une représentation donnée.

Dans le contexte de notre travail, et en tenant compte des caractéristiques techniques des

métadonnées décrites dans le chapitre précédent, nous adoptons dans ce qui suit la définition proposée dans [HK10] qui considère l'interopérabilité comme une propriété qualitative des métadonnées qui permet aux systèmes et aux applications de fonctionner avec - ou d'utiliser - ces métadonnées dans les limites du système. Cette définition est plus générale et comprend tous les niveaux possibles.

Dans la prochaine section, nous présentons une analyse détaillée des différentes sources de l'hétérogénéité des métadonnées.

3.3 Les différents types d'hétérogénéité

Les types d'hétérogénéité qui peuvent affecter les métadonnées ont été largement étudiés dans la littérature. Les auteurs dans [NO95] ont fourni une classification des différents conflits qui sont à l'origine de l'hétérogénéité dans les bases données. D'autres domaines tels que l'intelligence artificielle ont également montré leur intérêt au problème de l'hétérogénéité [Wac03] [VJCS97]. Dans la Figure 3.2, nous établissons une classification des différents types d'hétérogénéité mentionnés dans la littérature. Ceux-ci peuvent être divisés en trois classes : *l'hétérogénéité syntaxique*, *l'hétérogénéité structurelle* et *l'hétérogénéité sémantique*.

a) L'hétérogénéité syntaxique : Ce type d'hétérogénéité est directement liée aux langages de description et a un effet direct sur la dont on accède aux informations. Par exemple, avoir des noms différents pour les entités des langages de description (ex : SPARQL pour RDF et XQuery pour XML)

b) L'hétérogénéité structurelle : L'hétérogénéité structurelle se produit en raison d'incompatibilités organisationnelle entre les schémas. La combinaison ou l'arrangement des éléments de ces schémas forme une certaine structure pour représenter un domaine d'intérêt particulier. Les conflits liés à l'hétérogénéité structurelle se produisent lorsque l'organisation d'une

même information est établie de diverses manières et détails d'un schéma à un autre.

c) L'hétérogénéité sémantique :

L'hétérogénéité sémantique des métadonnées survient généralement lorsque le sens de la spécification varie en fonction des contextes et des interprétations. L'un des principales causes de l'hétérogénéité sémantique est l'utilisation de noms et de formats différents pour la description d'éléments identiques.

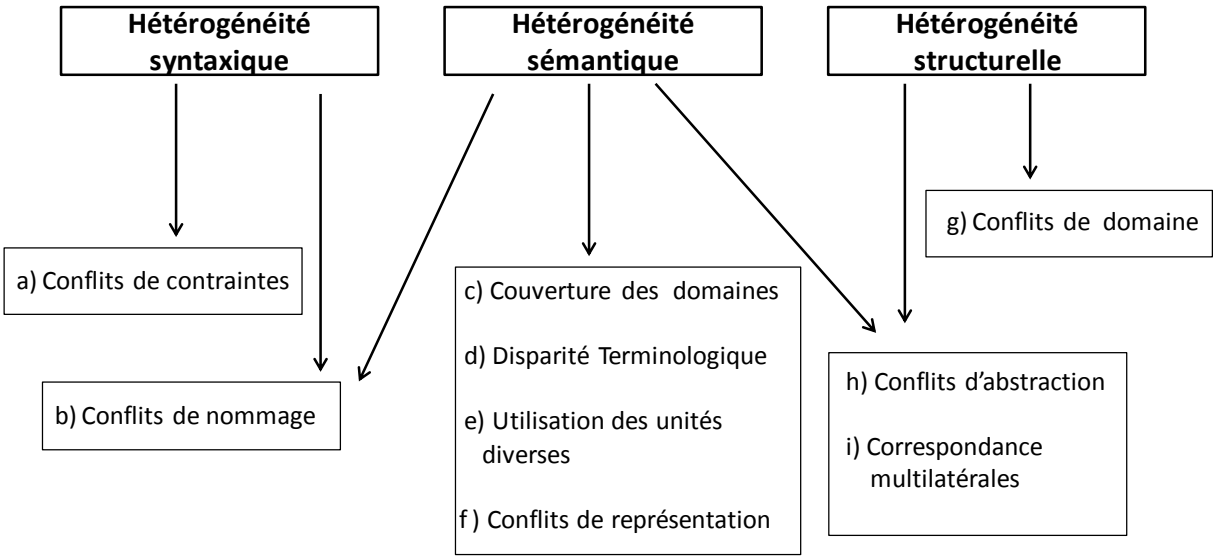

FIGURE 3.2: Classification des différents types d'hétérogénéité.

3.4 Les sources de l'hétérogénéité

Les conflits qui sont à l'origine de l'hétérogénéité des métadonnées ont été bien identifiés dans la littérature. Comme illustré dans le Figure 3.2, chaque type de conflit peut appartenir à une ou plusieurs catégories d'hétérogénéité en même temps. Dans cette section, nous discutons ces différentes sources d'hétérogénéité.

- *a) Les conflits de contraintes :* ce type d'hétérogénéité se produit parce que des modèles distincts offrent différentes possibilités d'exprimer les contraintes pour la définition des éléments. Avec le schéma XML, par exemple, on peut définir l'ordre des éléments avec la propriété séquence (*xsd:sequence*). Cette dernière n'est pas supportée par le OWL et le RDFS.
- *b) Les conflits de nommage :* ce type de conflit apparaît lorsque deux entités, représentant la même information, sont nommées différemment. Cela peut se produire au niveau des langages de description et des schémas de métadonnées. Le langage UML, par exemple, définit le concept *class*, alors que le Schéma XML utilise *complexType* pour capturer le même genre d'information. De même, au niveau des schémas, des modèles de métadonnées distincts peuvent attribuer des noms différents aux éléments porteurs des mêmes informations. Par exemple, quelques communautés [wg08] utilisent le terme *type* pour décrire le type de contenu multimédia. D'autres [LBSM08] utilisent le terme *format* ou *genre* pour décrire la même information.
- *c) La couverture des domaines :* lorsqu'il n'existe aucune correspondance entre les éléments d'un schéma de métadonnées, on parle alors de conflit dû aux couvertures des domaines. Cela se produit quand des informations reflétées par un modèle sont exclues par un autre modèle, et ce même si les deux modèles ont été conçus pour le même domaine sémantique. A cause de sa complexité, ce type de conflit n'est pas abordé dans cette thèse.
- *d) Les disparités terminologiques :* ce type d'hétérogénéité survient lorsque les communautés développant les métadonnées utilisent des noms homonymes dans leurs spécifications. Autrement dit, des noms identiques mais dont le sens dépend du contexte d'utilisation.
- *e) Les conflits dus à l'utilisation d'unités diverses :* ce type d'hétérogénéité se produit

en raison de l'utilisation d'unités différentes pour mesurer la valeur des éléments. Par exemple, un modèle de métadonnées peut utiliser le *pixel* comme unité pour mesurer la taille d'une image tandis qu'un autre modèle va utiliser le *centimètre*.

- *f) Les conflits de représentation :* les conflits de représentation sont le résultat de l'utilisation de formats différents pour coder la même information. Par exemple, deux valeurs de date peuvent être représentées différemment par chaque système (ex : date = 23/03/2011 ou date = 23.03.2011).
- *g) Les conflits de domaine :* le pouvoir expressif des langages de description varie d'un langage à l'autre. Pour cela, certains langages ont la capacité de définir des propriétés sémantiques qui ne sont pas supportées par d'autres langages. On peut par exemple, avec le langage OWL, spécifier que deux classes d'un modèle soient disjointes. Cependant, ce type d'information n'est pas supporté pas le Schéma XML.
- *h) Les conflits d'abstraction :* les conflits d'abstraction appartiennent à la catégorie des conflits de représentation. Ils apparaissent lorsque la même information possède des structures hiérarchiques différentes d'un modèle à l'autre. Un exemple de ce type de conflit au niveau des langages de description est la capacité de définir des attributs différemment dans plusieurs langages. Tandis que le langage OWL définit deux concepts distincts (*owl:class* et *owl:objectProperty*) pour spécifier un objet (classe) et une relation, le Schéma XML incorpore ces deux concepts sous une seule primitive (*xsd:attribute* ou *xsd:element*). Ce type de conflit intervient également au niveau des schémas dans lesquels on peut trouver des modèles de description de métadonnées qui regroupent le créateur d'une ressource numérique, par exemple, en une seule entité qui est *creator*, comme dans le cas du standard Dublin Core. Cependant, d'autres modèles comme TV-Anytime [PS00] et MPEG-7 [CPSZ01] distinguent les entités *Persons* et *Organization* qui sont considérées comme deux entités étendant le concept *Agent*.

– *i) Les correspondances multilatérales :* un autre type de conflit dû à la représentation des modèles, est directement lié aux conflits d'abstraction. Il s'agit des conflits de correspondances multilatérales. Chaque élément dans un modèle de métadonnées peut correspondre à plusieurs éléments dans un autre modèle, et vice versa. Un exemple de ce type de conflit est l'entité *creator* du schéma Dublin Core, qui correspond à l'ensemble des trois entités *IMAGE_CREATOR*, *OriginalWorkAuthor* et *ImageCreator* du standard DIG35 [wg02a].

Plusieurs techniques d'interopérabilité ont été proposées afin de faire face au problème de l'hétérogénéité. Dans la section suivante, nous analysons certaines de ces techniques.

3.5 Les différentes techniques d'interopérabilité

Les experts travaillant dans le domaine de l'interopérabilité des métadonnées ont développé des solutions permettant de surmonter les différents types d'hétérogénéité décrits précédemment. L'objectif de cette section est de mettre en évidence un cadre de classement des techniques existantes en fonction de leurs caractéristiques communes. L'interopérabilité des métadonnées peut être obtenue en éliminant ou en réduisant les hétérogénéités structurelles et sémantiques au niveau des langages de description, des schémas et des instances.

Dans les sections suivantes, nous nous concentrerons sur chacune de ces techniques et discuterons leurs caractéristiques. Nous commençons par l'introduction des approches d'homogénéisation qui visent à convertir les langages de description vers un format pivot. Ensuite, nous introduisons les approches ontologiques tout en donnant des exemples illustratifs de chaque approche. Enfin, nous définissons la notion du mapping entre les schémas qui est la base des systèmes de médiation ainsi que les bases de données fédérées introduits dans le reste de cette section.

3.5.1 L'homogénéisation des langages de description

Le but des méthodes d'homogénéisation est la conversion de langages hétérogènes vers le même langage a été proposée. Parmi ces méthodes, on peut citer les travaux présentés dans [GC05] et [YSL07], où les auteurs proposent des règles de conversion des Schémas XML en OWL. Ces règles permettent de capturer une partie de la sémantique implicite des Schémas XML. Dans [Fon97], les auteurs proposent une méthodologie permettant de traduire les schémas et les convertir en base de données relationnelles.

La spécification du métamodèle pour la définition des ontologies [LBSM08] offre un ensemble de métamodèles et de correspondances permettant de traduire les métamodèles UML et les langages d'ontologie comme les Schémas RDF et OWL. Dans [MVIM07], les auteurs présentent une approche permettant de générer automatiquement des Schémas RDF à partir de spécifications de Schémas XML. Les auteurs dans [GDDD04] présentent une approche permettant la création d'ontologies OWL depuis des modèles UML. La spécification XMI (XML Metadata Interchange) [Wei09] décrit la production de documents XML à partir des modèles UML et produit le Schéma XML pour la validation de ces documents. Une stratégie de mapping entre les objets relationnels et les bases de données relationnelles a été donnée dans [Amb03]. Les auteurs dans [LMC03] présentent une méthodologie pour générer automatiquement une représentation basée sur le Schéma XML depuis des schéma relationnels.

En raison d'incohérences structurelles et sémantiques substantielles parmi les langages de définition des schémas, les travaux précédemment mentionnés n'ont pas été particulièrement probants pour l'intégration des métadonnées hétérogènes car la traduction intégrale d'un langage à un autre entraîne la perte d'informations sémantiques et structurelles précieuses.

3.5.2 Approches ontologiques

La communication, le partage et l'échange d'information entre les différents systèmes d'information sont les caractéristiques communes que doit tenir en compte le développeur des métadonnées. Dans ce contexte, les ontologies constituent une solution pour enrichir les connaissances disponibles sur les métadonnées par rapport à certain domaine.

Dans cette section, nous présentons les méthodes d'interopérabilité basées sur des approches ontologiques allant de la création d'un schéma de métadonnées jusqu'à la création des métamodèles.

3.5.2.1 Schémas de métadonnées

Le but du développement d'un schéma de métadonnées est de fournir une vue standardisée sur un domaine donné. Dans le Tableau 3.1, nous présentons une sélection de standards de métadonnées utilisés dans divers domaines. Pour chaque norme, nous indiquons son domaine d'application, les types d'informations que couvre le langage de description utilisé pour définir les métadonnées, ainsi que l'organisme qui a effectué la standardisation.

TABLE 3.1: Selection de standards de métadonnées

Standard	**Langage de description**	**Domaine d'application**	**Usage**	**Organisme de standardisation**
Dublin Core	RDFS, XSD	indépendant du domaine	description d'une variété de ressources	ISO/NISO
Learning Object Metadata (LOM)	RDFS, XSD	eLearning	description des ressources numériques et non numériquess	IEEE
DIG35	XSD	eLearning	description des image utilisés dans le système eLearning	DIG35
CC/PP	RDFS	multimedia	description du contexte d'utilisation du multimedia	W3C
Geographic Information metadata	XSD, GML	systèmes d'informations géographiques	description de la documentation géographique	ISO
MPEG-7	XSD, DDL	contenus multimédia	description de la sémantique	MPEG

La plupart des systèmes standardisés de métadonnées sont conçus pour un domaine spécifique et dans un but précis. Le standard Dublin Core [Hau05] est un exemple de schéma

suffisamment générique pour décrire une variété de ressources indépendamment de domaines.

En tant que représentant du domaine e-Learning, le standard LOM (*Learning Objects Metadata*) [WG-02] normalise l'agrégation et le séquençage des objets d'apprentissage.

Le standard DIG35 est un schéma basé sur le Schéma XML, son but est d'offrir une description riche pour les images numériques (ex : contexte et contenu de l'image). Enfin, à partir du domaine des systèmes d'informations géographiques, notre sélection contient le standard (ISO 19115 :2003) conçu pour la documentation des ressources numériques dédiées à l'information géographique.

Afin d'offrir une extensibilité aux schémas de métadonnées, un autre type de schémas appelé *schéma hybride* est apparu. En plus des descriptions offertes par ces schémas, ils fournissent des outils permettant de définir de nouveaux concepts de métadonnées. Par exemple, le standard MPEG-7 possède un langage nommé DDL (*MPEG-7 Description Definition Language*), qui est une extension du Schéma XML. Ce langage représente une base solide aux utilisateurs désirant créer leurs propres schémas de métadonnées, compatibles avec le standard MPEG-7.

L'existante de plusieurs communautés de standardisation de schémas peut parfois être une autre source d'hétérogénéité. Chaque schéma est conçu pour un domaine donné qui ne peux pas couvrir tous les besoins informationnels des clients. La construction des métamodèles génériques est une solution alternative.

3.5.2.2 Métamodèles

Un accord sur un métamodèle donné permet implicitement l'interopérabilité en créant des correspondances entre les schéma via un métamodèle commun. Dans le reste de cette section, nous allons présenter les différents types de métamodèles existants dans la littérature.

a) Les métamodèles abstraits : La spécification d'un métamodèle abstrait de métadonnées est une façon de réaliser l'interopérabilité au niveau des langages de description. Ce métamodèle sert de référence technique pour la mise en œuvre des métadonnées dans les systèmes d'information. Si un accord sur un tel métamodèle existe, toutes les métadonnées sont alors exprimées en termes d'éléments de ce métamodèle. Les métadonnées seront interopérables, au moins sur le plan structurel, car elles sont techniquement représentées de la même manière. Le métamodèle DCMI [PNN+05] est un exemple de métamodèle abstrait. Il définit un modèle d'information qui permet une meilleure représentation du standard Dublin Core.

b) Les métamodèles conceptuels globaux : L'introduction d'un métamodèle conceptuel global est une façon de réaliser l'interopérabilité au niveau des schémas. Ce métamodèle présente un ensemble d'éléments génériques qui formalisent les notions dans un certain domaine et définit les concepts qui apparaissent dans un contexte d'intégration donné. Ensuite, chaque élément de métadonnées à intégrer est aligné à un concept plus générique qui lui correspond au niveau du modèle conceptuel.

Le CIDOC CRM [Hau05] est un exemple de ce type de métamodèle destiné au patrimoine culturel. Il définit 81 entités et 132 propriétés, la plupart d'entre elles sur un niveau très abstrait (ex : *physical thing*, *section definition*, etc). Un autre exemple de métamodèle conceptuel global est le métamodèle FRBR [SGotFRfBR97] pour les notices bibliographiques. Il a été défini par la Fédération Internationale des Associations de Bibliothécaires et d'Institutions. Avec ses quatre entités clés (*work, expression, manifestation, item*), il représente une vue générale de l'univers bibliographique, indépendamment de toute norme de catalogue [Til04]. Le SUMO [3] (Suggested Upper Merged Ontology) est également un métamodèle global conceptuel qui favorise l'interopérabilité des données, la recherche et traitement de l'information linguistique [NP01]. Il définit les concepts de haut niveau tels que *object*, *continuousObject*,

3. http ://ontology.teknowledge.com/

process, ou *quantity*. Le DOLCE (Descriptive Ontology for Linguistic and Cognitive Engineering) [CON04] est un autre exemple de métamodèle conceptuel global.

c) Profils d'application : Un profil d'application représente une autre stratégie pour réaliser l'interopérabilité des métadonnées [HP00]. Il s'agit d'un schéma composé d'éléments de métadonnées provenant d'un ou plusieurs systèmes standardisés, optimisés pour un domaine d'application donné, et dont le but est la réutilisation des métamodèles existants.

Les profils d'application sont créés par les développeurs d'applications qui définissent la façon d'appliquer les métadonnées d'un standard donné, dans un contexte donné. Dans un profil d'application, on ne peut pas créer de nouveaux éléments de métadonnées qui n'existent pas ailleurs. Si nécessaire, un nouveau schéma de métadonnées contenant ces éléments doit être créé. L'affinement des éléments appartenant à un autre standard est autorisé. Les développeurs, par exemple, peuvent configurer la plage de valeurs autorisées et réduire ou préciser la définition sémantique des éléments de métadonnées. Les profils d'application sont créés dans le but d'être réutilisés et sont adaptés à des communautés spécifiques d'utilisateurs.

Les auteurs dans [BLMS09] ont proposé un profil d'application dédié à l'intégration de plusieurs standards de métadonnées. L'architecture de ce métamodèle est utilisé dans un système d'information multimédia nommé LINDO [4] dont le but est la gestion de plusieurs moteurs d'indexation multimédia.

d) Les cadres de métadonnées : Un cadre de métadonnées peut être considéré comme un squelette sur lequel divers objets sont intégrés pour une solution donnée [CZ06]. Il s'agit d'une autre façon de réaliser l'interopérabilité au niveau des schémas. Il prévoit généralement un modèle de données composé d'un ensemble de termes abstraits, et une description de la syntaxe et de la sémantique de chaque élément du modèle. Là encore, l'idée est d'intégrer les métadonnées existantes par l'alignement de leurs éléments de modèle à un ensemble d'éléments définis

4. http ://www.lindo-itea.eu/

par le cadre de métadonnées. Les standards MPEG-21 [BdWH+03], METS (Metadata Encoding and Transmission Standard) [LOC07] et le système OAIS (Open Archival Information System) [CCS02] sont des exemples de cadres de métadonnées.

3.5.3 Le mapping

Le mapping des schémas représente un autre moyen pour résoudre le problème de l'interopérabilité permettant de faire face aux hétérogénéités entre les schémas de métadonnées. L'objectif du mapping entre deux schémas hétérogènes S_1 et S_2 est de trouver des correspondances sémantiques entre ces deux schémas [PL98]. Une connaissance à priori de S_1 implique une capacité d'interprétation de schéma S_2.

On peut citer comme exemple de mapping de schémas le travail effectué dans [LBSM08] où les auteurs ont proposé un système baptisé *Ontology for Media Resources*. Ce système aborde le problème d'interopérabilité entre les différents formats de métadonnées par la création d'un ensemble de propriétés considérées comme le minimum d'informations nécessaires à l'utilisateur. Un mapping manuel est ensuite effectué entre cet ensemble de propriétés et les propriétés équivalentes, encodées en différents formats. L'ensemble des propriétés définies par le groupe est considéré comme un schéma assurant un accès uniforme pour les différents standards de métadonnées.

Le mapping des schémas est principalement utilisé dans les approches de médiation et les bases de données fédérées qui seront discutées dans les sections suivantes :

3.5.3.1 Approche de médiation

L'approche de médiation consiste à construire une infrastructure intermédiaire commune facilitant l'accès aux bases de données hétérogènes [Wie92]. Un médiateur comprend un schéma global dont le rôle est de fournir une vue unifiée sur les données hétérogènes afin

d'uniformiser l'accès à ces données. L'approche de médiation présente l'avantage de pouvoir construire un système d'interrogation des sources de données sans toucher aux données qui restent toujours stockées dans leurs sources d'origine d'une façon distribuée.

Afin de pourvoir interroger les sources hétérogènes via un un schéma médiateur, les requêtes envoyées par les clients doivent être réécrites en fonction des alignements réalisée entre le schéma médiateur et les sources hétérogènes.

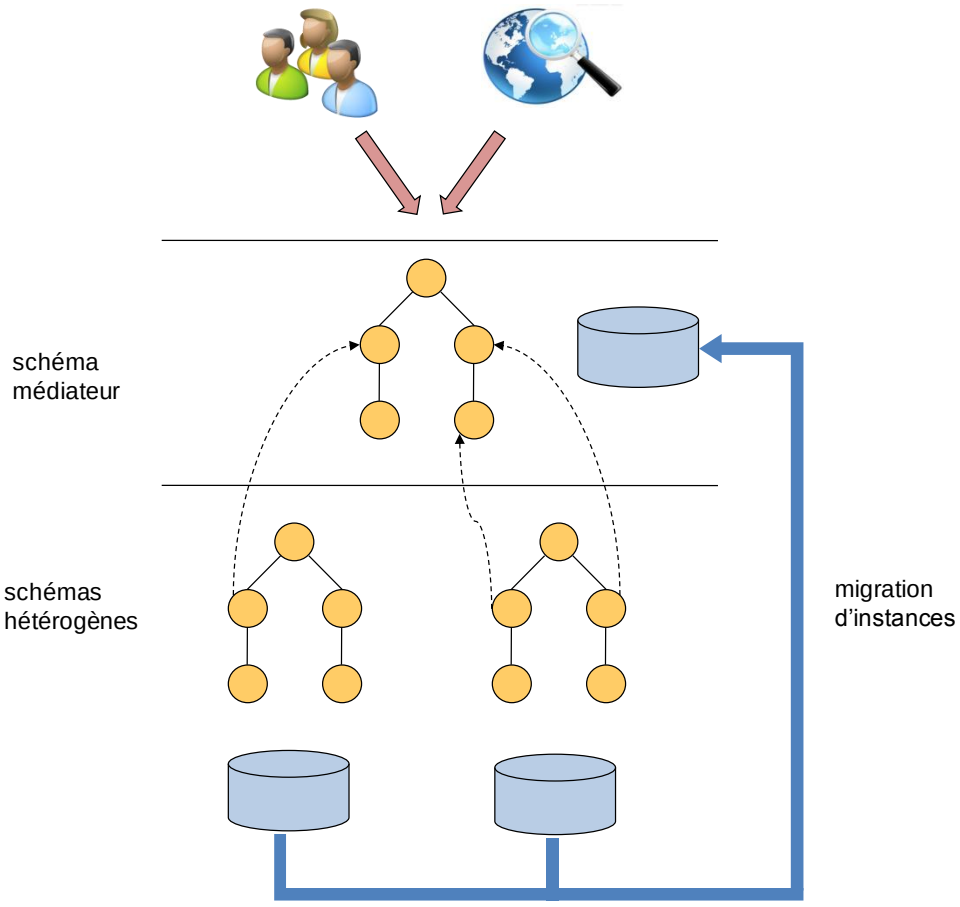

FIGURE 3.3: Architecture d'un système de médiation.

Cependant, dans un système de médiation des métadonnées (Figure 3.3), les instances ne sont pas disponibles au niveau du schéma médiateur. Ce schéma joue le rôle d'une interface commune pour interroger les métadonnées hétérogènes. Un mapping est effectué entre les métadonnées hétérogènes et le schéma médiateur non instancié. Les instances correspondants à ces métadonnées sont migrées vers le schéma médiateur afin qu'elles puissent être interrogeables par les utilisateurs. Dans ce contexte, nous allons nous concentrer dans la suite de ce manuscrit sur l'interopérabilité au niveau des langages description et des schémas. La migration d'instance ne sera pas abordée dans ce manuscrit.

3.5.3.2 Bases de données fédérées

Une base de données fédérée est une base de données repartie, hétérogène, constituée de sources de données de natures variées : fichiers HTML, XML, etc.

L'objectif de fédération des bases de données est de fournir aux utilisateurs une vue intégrée de différentes données hétérogènes. L'élaboration d'une base de données fédérée nécessite une architecture qui assure la communication entre les différentes bases de données [GSDN99]. Cette architecture s'articule en trois niveaux :

- **un niveau présentation** formé de composants qui permettent de formuler des requêtes dans le langage de la base de données fédérée ;
- **un niveau médiation** formé de médiateurs qui se chargent de collecter les demandes des utilisateurs déjà enregistrées par les composants de présentation et de les traduire dans le langage de chaque source de données
- **un niveau adaptation** formé de composants qui permettent la communication entre une source de données et les médiateurs (Figure 3.4).

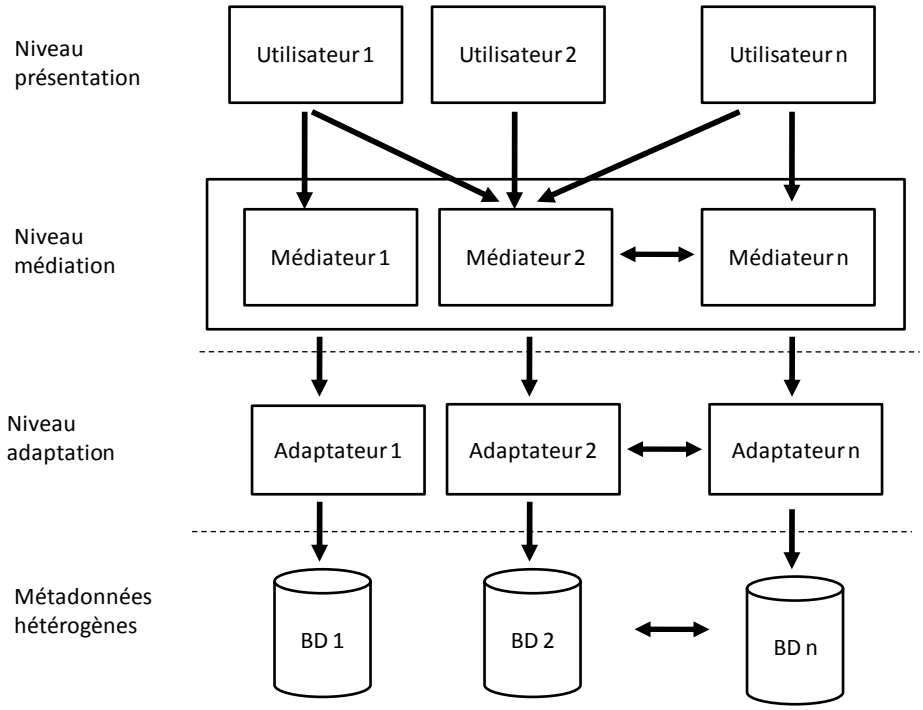

FIGURE 3.4: Architecture d'une base de données fédérée.

3.6 Conclusion

Dans ce chapitre, nous avons présenté une taxonomie complète des différentes approches dont le but est de réaliser l'interopérabilité des métadonnées. Nous avons vu que certaines techniques sont appliquées sur un ou plusieurs niveaux constitutifs des métadonnées. Nous avons commencé par la présentation de quelques approches d'homogénéisation de langage de description. Nous avons montré qu'il existe une hétérogénéité structurelle et sémantique au niveau des langages de description ce qui provoque forcément une perte d'information lors du passage d'un langage à un autre. Dans la Section 3.5.2, nous avons discuté les approches ontologiques en commençant par la standardisation des schémas de métadonnées et puis la méta-modélisation. Nous avons montré que l'introduction d'un schéma ou d'un métamodèle

commun ne résout pas complètement le problème d'hétérogénéité car leurs spécifications sont réalisées pour couvrir des informations spécifiques. En plus, la complexité de l'intégration de nouvelles informations dans les schémas ou les métamodèles augmente avec la diversité et le nombre d'informations. Dans ce contexte, l'extensibilité n'est pas toujours garantie. Les approches à base de mappings présentées dans la Section 3.5.3 est une autre solution pour réaliser l'interopérabilité. Il s'agit du calcul d'un ensemble d'alignements effectués entre les schémas sans pour autant créer de nouveaux schémas ou modifier les schémas initiaux. Le mapping est la base des systèmes de médiation et base de données fédérées discutés dans ce chapitre et qui possèdent des architectures qui leur permettent d'utiliser les mappings convenablement. Dans cette section nous nous sommes concentrés sur les avantages apportés par les techniques d'intégration à base de mapping en s'affranchissant des difficultés liées au processus de calcul de mappings. L'inconvénient des approches à base de mapping réside dans le fait que les mappings sont généralement définis manuellement par des experts humains ce qui est coûteux en termes de temps et de ressources. Dans l'état de l'art certaines solutions apportent des éléments de réponses par rapport au calcul de mise en correspondance de schémas. Dans la section suivant nous présentons un recueil de ces méthodes en mettant en exergue les caractéristiques et les lacunes de chacune en rapport avec le calcul de mappings manière (sémi) automatique.

Chapitre 4

La mise en correspondance des schémas

Sommaire

4.1 Introduction

Nous avons présenté, dans le chapitre précédent, quelques travaux dont le but est de réaliser l'interopérabilité des métadonnées. Ces travaux d'intégration ne se sont pas avérés très performants car ils ont été effectués manuellement, ce qui entraîne une perte de temps assez important. Par ailleurs, avec le nombre de communautés de métadonnées existantes et les environnements dynamiques comme le multimédia, le processus d'intégration doit être mise à jour à chaque apparition d'un nouveau format, ce qui rend la tâche très coûteuse. A ce stade, les techniques mise en correspondance des schémas (matching) sont apparues pour trouver les mappings de manière automatique [ES07] [RB01].

Dans ce chapitre, nous discutons quelques travaux de l'état de l'art sur le matching. Nous commençons par donner un aperçu des techniques de base, puis nous décrivons plusieurs méthodes qui combinent ces techniques.

4.2 La mise en correspondance des schémas

La mise en correspondance (*ou le matching*) de schémas décrit dans la Figure 4.1 consiste en un processus de manipulation des schémas, qui prend en entrée deux schémas hétérogènes et retourne un ensemble de correspondances, nommées *alignements* ou *mappings*. En raison de la complexité du matching des schémas, l'intervention d'experts humains est souvent nécessaire. A ce stade, plusieurs travaux, dont le but et d'automatiser le processus de matching et de réduire l'intervention humaine, ont été effectués [RB01] [KS05a] [SE05].

Le processus de matching des schémas est généralement composé de trois étapes [KS05b] : la première étape consiste à extraire des schémas, les informations qui seront utilisées dans le processus de matching. Ce processus peut, par exemple, n'exploiter qu'un sous-ensemble de ces informations (les noms des entités, les relations, etc). Les primitives des extraites schémas

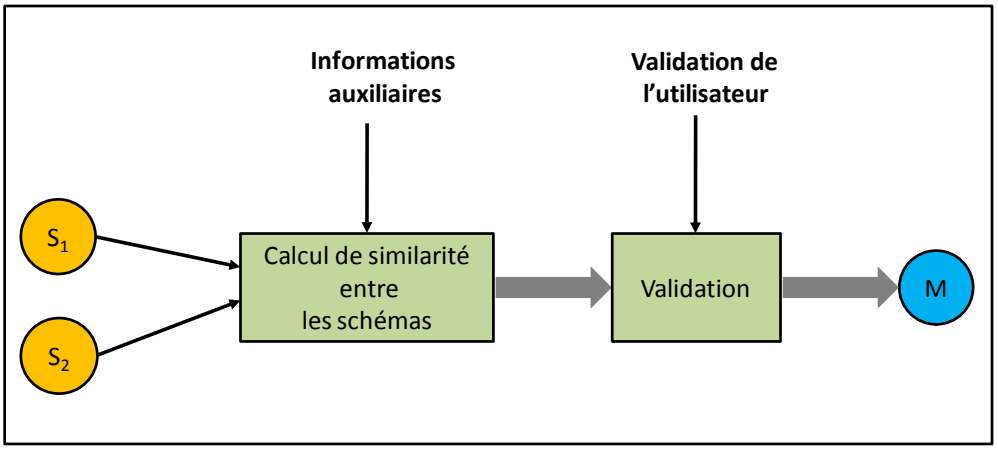

FIGURE 4.1: Le process de matching des schémas

dépendent des stratégies de matching qui seront ensuite mises en œuvre. Elles peuvent correspondre aux identifiants comprenant les URIs ou les labels (noms des entités), aux propriétés sémantiques et structurelles ou aux relations entre les éléments.

La deuxième étape consiste à calculer la similarité entre les éléments appartenant aux deux schémas à aligner. Cette similarité peut être calculée de différentes façons. Il existe de nombreuses fonctions de similarité qui, souvent, reposent sur le calcul d'une distance sémantique δ. La distance sémantique entre deux éléments (x, y) est une valeur normalisée comprise entre 0 et 1. La mesure de similarité notée σ est inversement proportionnelle à δ.

$$\sigma(x,y) = 1 - \delta(x,y) \tag{4.1}$$

Enfin, la dernière étape du processus de matching est l'étape d'interprétation et de validation des résultats. Elle consiste à déduire des mappings à partir de mesures de similarité préalablement calculées. La plupart de ces approches comparent les mesures obtenues à des seuils. Ensuite, les éléments ayant de fortes valeurs de similarité sont retournés à l'utilisateur

pour validation.

De nombreuses méthodes d'intégration des métadonnées mettent en œuvre ce processus générique de matching en exploitant des techniques variées. Plusieurs classifications de techniques de matching ont été proposées dans la littérature, notamment celles dans [RB01], [KS05a], et [SE05]. Nous proposons, dans un premier temps, d'introduire les techniques de base utilisées par la plupart de méthodologies de matching, puis nous présentons la manière dont ces techniques peuvent être combinées pour définir une stratégie de matching en nous appuyant sur certains travaux existant dans l'état de l'art [ES07].

4.2.1 Les techniques de base

Dans cette section, nous introduisons quelques techniques utilisées dans la plupart des méthodologies de matching. Ces techniques appartiennent à trois catégories : *techniques linguistiques*, *structurelles* et *techniques basées sur les instances*. Dans ce chapitre nous allons nous baser sur les deux premières catégories car les instances ne sont pas exploitées dans un système de médiation de métadonnées (Section 3.5.3.1).

4.2.1.1 Les techniques linguistiques

Le but de ces techniques est de trouver des correspondances entre les descriptions textuelles des entités (ex : noms, commentaires, etc). La plus part des méthodes utilisent ces mesures linguistiques ; on peut citer les approches TranScm [MWJ99] et SKAT [MZ98].

Les techniques de matching linguistique sont basées sur une comparaison syntaxique ou sémantique. La mesure de *similarité syntaxique* entre des noms d'éléments peut être calculée en fonction du nombre de caractères communs. La distance de Levenshtein, par exemple [Lev66], mesure la similarité entre deux chaînes de caractères. Elle est égale au nombre minimal de caractères qu'il faut supprimer, insérer ou remplacer pour passer d'une chaîne à l'autre.

D'autres approches comme celle décrite dans [LDKG04], utilisent les méthodes de normalisation avant le calcul de la similarité entre noms comme par exemple, la lemmatization, la division en unités lexicales (tokenization).

L'utilisation des commentaires dans le processus de matching est basée sur des techniques linguistiques. Un exemple de ces techniques est la méthode de TF-IDF [vR79] issue du domaine de traitement de l'information et qui est utilisée pour calculer l'importance des mot dans un texte.

Dans la *similarité sémantique*, des ressources externes comme le thésaurus WordNet [Fel98] et le PMI (Point Mutual Information) [Tur01a] sont souvent utilisées. Elles servent à trouver les relations sémantiques entre les différents concepts. Ces relations sont de type synonymes, hyperonymes, etc. Cela permet de résoudre les conflits de nommage des éléments.

Les techniques linguistiques fournissent des scores de similarité entre les éléments en prenant en compte uniquement le sens de l'élément et non pas celui de son voisinage. Par exemple, si on calcule la similarité entre les deux chemins (*chapitre, auteur, nom*) et (*entreprise, chef, nom*), la similarité entre les deux éléments *nom* est égale à 1, sachant que le sens est totalement différent ; le premier représente le nom de l'auteur d'un chapitre et le deuxième représente le nom d'un chef d'entreprise. Dans ce contexte, l'information structurelle (relation entre nœuds) peut être utilisée pour filtrer les faux candidats dans un processus de matching.

4.2.1.2 Les techniques structurelles

Les techniques de matching basées sur la structure considèrent que la similarité entre deux entités (n_1, n_2), appartenant à deux schémas différents (S_1, S_2), dépend des ensembles d'entités connectées à (n_1, n_2). Certaines méthodes [MGMR02] utilisent la notion de voisinage entre les éléments en considérant que deux éléments distincts appartenant à deux schémas différents sont similaires si leur éléments adjacents sont similaires. D'autres [GSY04] utilisent les re-

lations hiérarchiques, en considérant que deux éléments sont identiques si leurs parents sont identiques. D'autres approches dont le but est d'introduire une flexibilité dans le calcul de la similarité structurelle on été également introduites. Ces approches permettent de faire un matching entre les chemins de nœuds représentant la structure même, si leurs longueurs ne sont pas identiques ou s'ils contiennent des nœuds qui ne sont pas communs. Par exemple, les auteurs dans [DMM+03] considèrent que la ressemblance entre les chemins dépend des quatres critères suivants :

- le chemin p_j contient le maximum de nœuds de p_i dans le bon ordre ;
- l'occurrence des nœuds de p_i est plus proche du début de p_j qu'à celle de sa fin ;
- l'occurrence des nœuds de p_i et p_j sont proches l'un de l'autre. Cela veut dire que la ressemblance entre deux chemins p_i et p_j est inversement proportionnelle au nombre de nœuds qui ne sont pas communs entre les deux chemins ;
- si plusieurs chemins ont une forte ressemblance avec p_i, le chemin idéal est le plus court.

Afin répondre aux quatre critères mentionnés ci dessus, les auteurs ont défini les quatre paramètres suivants :

- **La plus longue sous-séquence commune :** la plus longue sous-séquence commune (lcs) entre deux chemins p_i et p_j est proportionnelle au nombre de nœuds communs dans le même ordre entre p_i et p_j . Pour calculer cette valeur, nous utilisons l'algorithme basé sur la programmation dynamique [Hir75]. On note $lcs_n(p_i, p_j)$, la plus longue sous-séquence commune entre p_i et p_j normalisée par la longueur de p_i.

$$lcs_n(p_i, p_j) = |lcs(p_i, p_j)|/|p_i| \tag{4.2}$$

  Si on considère les deux chemins $p_i = (video, hasDescription, metadata)$ et p_j = *(metadata, video, hasDescription)* ; la plus longue sous-séquence commune entre les deux chemins est $(video, hasDescription)$. Donc $|lcs(p_i, p_j)| = 2$ et $lcs_n(p_i, p_j) = 2/3 = 0.66$.

– **Le positionnement moyen :** afin de répondre au deuxième critère de matching, les auteurs calculent tout d'abord le positionnement moyen entre p_i et p_j. Cette valeur est calculée en fonction de $lcs(p_i, p_j)$ et considère que le matching idéal entre deux chemins (p_i, p_j) est celui qui commence au premier nœud de p_i sans discontinuité. Si on considère que $p_i = (video, metadata)$ et p_j = *(video, hasDescription, metadata)* sont les deux chemins en question, la position moyenne optimale notée aop est (1+2+3)/3 =2. A ce stade, la valeur de positionnement moyen notée ap est ((2+3+4)/3)=3. Enfin, nous calculons le coefficient pos indiquant comment le positionnement réel des nœud est situé par rapport à la solution optimale aop par le biais de la formule suivante :

$$pos(p_i, p_j) = 1 - ((ap - aop)/(|p_j| - 2 * aop + 1)) \tag{4.3}$$

– **LCS avec un nombre minimum de trous :** pour répondre au troisième critère, le score $gap(p_i, p_j)$ est utilisé pour s'assurer que les occurrences de p_i et p_j sont proches les unes des autres . Elle est donnée par :

$$gap(p_i, p_j) = gaps/(gaps + lcs(p_i, p_j)) \tag{4.4}$$

si $p_i = (hasDescription, metadata, information)$ et $p_j = (video, hasDescription, metadata, information)$, on compte une valeur de gaps = 1. Pour s'assurer que la valeur du score gap est inférieure à 1, la valeur de $gaps$ est normalisée.

– **La différence de longueur :** $ld(p_i, p_j)$ donne des valeurs importantes aux chemins (p_i, p_i) dont les longueurs sont éloignés. Il s'agit de la différence de longueur entre le chemin p_i et $lcs(p_i, p_j)$, normalisée par la longueur de p_i :

$$ld(p_i, p_j) = (|p_j| - lcs(p_i, p_j))/|p_j| \tag{4.5}$$

Pour obtenir la similarité entre les chemins (p_i, p_j), les quatre scores sont combinés afin de répondre à l'ensemble des critères :

$$ps(p_i, p_j) = \delta lcn_n(p_i, p_j) + \varphi pos(p_i, p_j) - \theta gap(p_i, p_j) - \lambda ld(p_i, p_j) \tag{4.6}$$

Dans la section suivante, nous montrons comment les méthodes basiques sont utilisées pour définir une stratégie de matching.

4.2.2 Les stratégies de matching

Les techniques décrites précédemment sont des éléments de base pour le matching. Ces techniques peuvent être appliquées de façon séquentielle pour, parallèlement former une stratégie de matching. Dans cette section, nous décrivons quelques une des stratégies les plus connues dans le domaine du matching des schémas. Par la suite, nous analysons de manière générale les stratégies introduites et nous discutons leur avantages et inconvénients.

4.2.2.1 Cupid

Cupid s'appuie sur un algorithme hybride qui combine plusieurs méthodes basiques de matching [MBR01]. Il est destiné à être générique à travers les modèles de données, et a été appliqué sur des modèles aux formats XML et SQL. Cupid est basé sur la comparaison de schémas, sans l'utilisation d'instances. Le processus de matching est composé de trois phases :

– a) la première est le matching linguistique où les noms des éléments sont décomposés en unités lexicales, puis un dictionnaire externe est utilisé pour résoudre les conflits de nommage (ex : abréviation, synonymes, etc). Ensuite, pour réduire le temps de calcul,

Cupid catégorise les éléments en tenant compte de leurs types de données et valeurs linguistiques (la comparaison est faite uniquement entre les éléments des deux schémas, appartenant aux catégories similaires). Cupid calcule la similarité entre les chaînes de caractères des noms d'éléments. Le résultat de cette phase est une table de coefficients de similarité linguistique entre les éléments des deux schémas. La valeur de cette similarité varie entre 0 et 1.

- b) la deuxième phase transforme les schémas d'origine en arbres sur lesquels des mesures structurelles sont appliquées. Cupid considère que la plus grande partie du contenu des informations est représentée par les feuilles de ces arbres, et que ces feuilles sont soumises à un plus petit nombre de variations entre les schémas que les structures internes. Ainsi, la similarité des intra-noeuds s'appuie sur la similarité de l'ensemble de leurs feuilles.
- c) la troisième phase utilise les scores de similarité obtenus dans les deux premières étapes afin de sélectionner les mappings corrects.

4.2.2.2 Similarity Flooding

Les auteurs dans [MGMR02] proposent un algorithme de matching basé sur la structure, nommé *Similarity Flooding*. Cet algorithme a été appliqué sur des schémas couvrant plusieurs langages de description, y compris SQL, RDF et XML. Il est basé sur l'hypothèse suivante : "deux éléments appartenant à deux modèles différents sont similaires lorsque leurs éléments adjacents sont similaires". Le processus de matching commence par la transformation des deux schémas en graphes directs. Ensuite, il mesure la similarité linguistique entre les noms d'éléments (calcul de la distance entre les chaînes de caractères) par l'application d'un calcul itératif. Finalement, plusieurs types de filtrage structurel sont appliqués pour éliminer les mappings indésirables.

4.2.2.3 Coma++

Coma++ est une évolution de Coma qui améliore ses algorithmes et son interface graphique [DR02]. Il offre une bibliothèque extensible d'algorithmes de matching, un framework pour combiner les résultats obtenus depuis plusieurs méthodes basiques de matching, et une plate-forme d'évaluation de ces méthodes. L'utilisateur possède une interface permettant de changer les paramètres et les coefficients de matching. Coma++ a recours à plusieurs informations (instance, structure, noms, commentaires, etc).

4.2.2.4 S-match

S-Match [GSY04] est un algorithme de matching qui peut trouver plusieurs types de relations sémantiques entre les schémas : relation d'équivalence, plus général et moins général. S-Match réduit le problème de matching à un problème de satisfiabilité, chaque taxonomie dans les schémas étant présentée comme une formule de description logique. Ensuite, WordNet [Fel98] est utilisé pour trouver les relations sémantiques entre les concepts appartenant aux deux schémas à aligner. Chaque relation reliant deux éléments est transformée en relation logique entre les deux descriptions correspondant à ces éléments. Finalement, afin de considérer ces relations comme des mappings, des outils de satisfiabilité sont utilisés pour vérifier la cohérence de ces mappings.

4.2.2.5 GLUE

Le système GLUE [SS04] est un système de matching des schémas qui utilise plusieurs techniques d'apprentissage automatique. Il trouve le plus grand nombre de concepts semblables entre les deux schémas et calcule la probabilité de la distribution commune des concepts en utilisant une approche d'apprentissage multi-stratégies pour la mesure de la similarité.

GLUE comporte trois stratégies d'apprentissage :

- L'exploitation des fréquences de mots dans le contenu des instances par l'utilisation du théorème de Naïve Bayes ;
- La comparaison des noms des instances ;
- La combinaison des prévisions des deux stratégies précédentes pour donner du poids a chaque stratégie suivant sa fiabilité à être vraie ou fausse.

4.2.2.6 LSD

LSD (Learning Source Description) [DDH01] est un système de matching basé sur des méthodes d'apprentissage automatique. LSD trouve des correspondances entre une nouvelle source de données et un schéma global défini précédemment. LSD est basé sur la combinaison de plusieurs matchers obtenus durant l'étape de prétraitement. A partir d'un mapping initial (généralement fournit par l'utilisateur), l'étape de prétraitement analyse les instances de la source de données afin de découvrir les caractéristiques et les règles de matching. Ces règles sont ensuite utilisées pour trouver les mappings entre les sources de données et le schéma global. LSD calcule la similarité entre les noms des éléments en utilisant la distance d'édition, cette distance étant également utilisée pour calculer la similarité entre les instances. La classification naïve bayésienne, à son tour, découvre la similitude entre les éléments du schéma en tenant compte de leur décomposition en unités lexicales. Enfin, les mappings validés par l'utilisateur sont pris en considération pour améliorer la précision du matching.

4.2.2.7 RiMOM

RiMOM (Risk Minimisation based Ontology Mapping) [LZLT06] est une stratégie de mapping qui se base sur la conception bayésienne de la théorie de la décision. RiMOM considère le matching comme un problème décisionnel. Le processus d'intégration se fait en deux

étapes, une recherche de similarités entre les concepts, suivie d'une recherche entre les propriétés. RiMOM utilise plusieurs outils d'estimation dont les résultats sont combinés par une méthode d'interpolation linéaire. Ensuite les meilleurs résultats sont propagés sur la structure de l'ontologie. Dans le cas des propriétés, des heuristiques prennent en compte les résultats obtenus au niveau des concepts pour éviter les correspondances qui entreraient en contradiction les unes avec les autres. Le système implémente trois méthodes de propagation, la propagation concept à concept, la propagation propriété à propriété et la propagation concept à propriété. Ces méthodes sont utilisées alternativement selon des heuristiques. Le processus est itératif, avec une validation des résultats à chaque itération.

4.2.2.8 ASMOV

ASMOV (Automated Semantic Matching of Ontologies with Validation) [JMK07] est un algorithme de matching utilisant des informations lexicales (noms d'éléments et dictionnaire externe), des informations structurelles (type de données, propriété entre éléments), des informations sur la structure (adjacence) et la similarité des instances. Le résultat final est une somme pondérée de tous le scores mentionnés précédemment. Les entités ayant un score de similarité maximal sont considérées comme des candidats au mapping. ASMOV utilise un processus de validation sémantique pour ces candidats via la validation de la cohérence de ces mappings par rapport aux deux schémas.

4.2.3 Caractéristiques des méthodes de matching

Le Tableau 4.1 décrit les caractéristiques de quelques méthodes de matching existantes dans l'état de l'art. Ces caractéristiques peuvent être organisées en trois catégories :

– *Les types de schémas supportés par les systèmes de matching :* ce sont les langages de description utilisés pour la définition de ces schémas. Nous avons sélectionné majo-

ritairement des méthodologies de matching destinées aux hiérarchies textuelles (H.T), aux langages sémantiques (RDFS/OWL), aux schémas relationnels (Rel.), aux langages orientés objet (O.O.) et aux taxonomies (XSD). Le Tableau 4.1 montre qu'il y a des méthodes qui sont spécifiques à plusieurs langages en même temps (ex : Similarity Flooding et RiMOM). Cependant, d'autres systèmes sont destinés à un seul langage et ne peuvent pas être génériques (ex : Glue, SEMINT, etc).

– *Cardinalité du mapping :* Afin de résoudre le problème de correspondances multilatérales entre les schémas, la détection des mappings complexes est incontournable. Cependant, tous les algorithmes ne prennent pas ces contraintes en considération. Nous pouvons constater dans le Tableau 4.1 que la plupart des méthodes trouvent des correspondances complexes (n : m), mais ces correspondances sont globales (communes entre plusieurs nœuds).
– *Types d'informations utilisées dans le processus de matching :* le Tableau 4.1 montre, pour chaque méthode, les types d'informations exploitées dans le processus de matching. On peut remarquer qu'il y a des informations qui sont plus ou moins utilisées par ces méthodes. Cela n'est pas dû uniquement à la stratégie de matching mais également à la disponibilité de ces attributs. Cupid et SF, par exemple, sont deux méthodes de matching destinées aux schémas qui ne prennent pas les instances en considération car ces dernières ne sont pas toujours disponibles. Le type de données est un autre exemple d'information qui n'est pas disponible au niveau de quelques langages sémantiques (ex : RDFS). Cependant, les noms d'éléments sont un exemple d'information disponible dans tous les schémas, ce qui la rend largement exploitable dans la plupart des processus de matching.

Étant donné un des objectifs de cette thèse et de développer une solution d'intégration automatique pour la médiation des métadonnées. Deux conditions doivent être vérifiées :

TABLE 4.1: Comparaison des méthodes de matching

Méthode	**Types de schémas supportés**	**Cardinalité du mapping**	**Type d'informations utilisées**
Similarity Flooding	Rel, XSD, O.O, RDF	1 :1 local m :n local	noms d'éléments, structure (l'adjacence entre les nœuds)
Cupid	XSD, O.O	1 :1 local n :1 global	noms d'éléments, structure (feuilles, type de données)
S-Match	H.T, OWL, RDFS	1 :1 local n :1 global	noms d'éléments, ressources externes (WordNet), relations hiérarchiques
GLUE	H.T	1 :1 local n :1 global	nom d'éléments, instances
RiMOM	H.T, RFDS, OWL	1 :1 local n :1 global	Ressources externes, (WordNet), noms d'éléments, cardinalité
LSD	XSD	1 :1 local n :1 global	noms d'éléments, instances
Coma	XSD, Rel	1 :1 local m :n global	types de données, structure noms d'éléments, (feuilles, fils immédiats)
SEMINT	Rel	m :n local m :n global	noms d'éléments, types de données, instances

a) La méthode de matching doit être basée uniquement sur les schémas (pas d'utilisation d'instances) car dans un système de médiation de métadonnées, le schéma médiateur n'est pas instancié (voir Section 3.6).

b) La réalisation d'un système d'intégration multi-niveaux nécessite une méthode de matching qui doit supporter plusieurs types de schémas (schémas définis en utilisant des langages de description différents).

Si on prend en considération les deux conditions précitées, on peut constater que peu de méthodes de matching satisfont ces deux conditions. Parmi ces méthodes, on trouve notam-

ment : Similarity Flooding [MGMR02], Cupid [DR02], et S-Match [GSY04]. Ces approches n'utilisent pas la majorité des informations structurelles et sémantiques (ex : propriétés d'équivalence, caractéristiques de généralisation, etc.). De plus, le gros inconvénient de ces méthodes est leur façon d'utiliser les informations structurelles. Par exemple, les auteurs de [MBR01] considèrent que la plus grande partie du contenu des informations est représentée par les feuilles du schéma, et que ces feuilles sont soumises à un plus petit nombre de variations entre les schémas que les structures internes. Ainsi, la similarité des intra-nœuds s'appuie sur la similarité de l'ensemble de leurs feuilles. Cela ne s'applique pas toujours puisque l'on peut trouver des concepts équivalents apparaissant dans des structures totalement différentes, et des concepts totalement indépendants appartenant à des structures isomorphiques.

La méthode développée dans [MGMR02], basée sur l'idée de la propagation de similarité, possède un inconvénient majeur. Le concept de base de l'algorithme est que l'adjacence contribue à la propagation de la similarité. Ainsi, l'algorithme réagira de manière inattendue lorsque les informations d'adjacence ne seront pas préservées.

L'approche proposée dans [GSY04] a uniquement recours à la relation "parent-enfant" pour calculer les contextes de similarité structurelle. Le travail dans [GSY04] est limité uniquement aux structures arborescentes des schémas et ignore les relation inter-nœuds (propriétés).

4.3 Conclusion

Nous avons présenté dans ce chapitre les différents aspects des méthodes de matching. Nous avons tout d'abord défini la notion de matching en introduisant les différentes informations basiques qui peuvent utilisées dans ce processus. Nous avons également introduit quelques stratégies de matching en expliquant leur manière de combiner les informations ba-

siques. Ensuite, une étude comparative a été faite entre ces stratégies. Dans cette étude, nous avons présenté les limites des méthodes existantes et notamment dans l'utilisation des informations structurelles et sémantiques. L'intégration multi-niveaux des métadonnées nécessite une prise en considération de plusieurs langages de description en même temps, ce qui n'est pas disponible dans la plupart des méthodes de matching existantes.

Aperçu de la proposition

Nous avons présenté dans les chapitres précédents un état de l'art sur l'hétérogénéité des métadonnées et des différentes méthodes d'intégration pour réaliser leur interopérabilité. Ces méthodes appartiennent à deux catégories :

- *Méthodes d'intégration manuelle* réalisées par des experts humains (standardisation, métamodélisation, transformation, etc).
- *Méthodes d'intégration automatique* basées sur le matching des schémas. Leur but est de minimiser l'intervention humaine.

Afin de faire face aux limitations des méthodes d'intégration citées dans les chapitres précédents, nous présentons dans le reste de ce manuscrit nos deux contributions qui s'inscrivent respectivement dans les deux catégories citées précédemment. La première solution concerne un métamodèle dédié à l'agrégation des métadonnées multimédia hétérogènes. Ce métamodèle présenté dans le Chapitre 5 regroupe plusieurs types de métadonnées nécessaires pour la création, la livraison, la modification, l'interprétation et la consommation des contenus multimédia sur des plate-formes hétérogènes. La structure de ce métamodèle lui offre une extensibilité en permettant d'intégrer facilement d'autres informations plus spécifique.

La deuxième contribution appartient à la classe des méthodes d'intégration automatique de métadonnées.Une nouvelle stratégie de matching des schémas dont le but est de réaliser l'interopérabilité sur les deux niveaux (schémas et langages de description) est proposée dans le Chapitre 6. Cette méthode remédie aux lacunes des méthodes de matching discutées dans la Section 4.3.

Dans la suite de ce manuscrit de thèse, nous présentons en détail nos deux contributions et nous montrons leurs réalisations techniques et validations expérimentales.

Chapitre 5

CAM4Home : un métamodèle pour la fusion des métadonnées hétérogènes

Sommaire

5.1 Introduction

Tout au long des chapitres précédents, nous avons présenté les différents types des métadonnées nécessaires pour gérer le cycle de vie d'un objet multimédia de sa création jusqu'à sa consommation par les clients (utilisateurs ou logiciels), ainsi qu'que les problèmes liés à l'hétérogénéité au niveau des schémas et des langages de description.

Nous avons également montré qu'il existe plusieurs standards dont le but est de fournir une représentation largement acceptée des métadonnées. La plupart de ces standards ont été conçus pour définir un type particulier de métadonnées. Certains couvrent la sémantiques des contenus, d'autres offrent une description contextuelle pour fournir les informations nécessaires destinées à l'adaptation des contenus. A ce stade, l'utilisation commune des ces standards est incontournable pour assurer une couverture complète des différents types des métadonnées nécessaires à la manipulation des objets multimédia. Cependant, l'un des inconvénients majeurs d'une telle solution est d'obliger les clients à avoir une connaissance à priori sur tous les standards utilisés. De plus, l'utilisation des standards de très large échelle comme le MPEG-21 nécessite des nombreux pré-requis et une capacité d'interprétation de ce dernier même s'il est partiellement utilisé.

La nécessité d'avoir une description unifiée contenant les besoins informationnels requis pour le cycle de vie d'un objet multimédia a été soulevée par l'ensemble des partenaires du projet européen CAM4Home (annexe A). Un ensemble de métadonnées, dont le rôle est d'assurer la description sémantique et contextuelle requise durant la création, la modification, la distribution, la livraison, l'interprétation et la consommation des contenus multimédia à été défini. Ces métadonnées illustrées par la Figure 5.1 sont classifiées en métadonnées basiques, métadonnées supplémentaires et métadonnées externes. Les *métadonnées basiques* sont destinées à la description sémantique des objets multimédia. Cette description est principalement

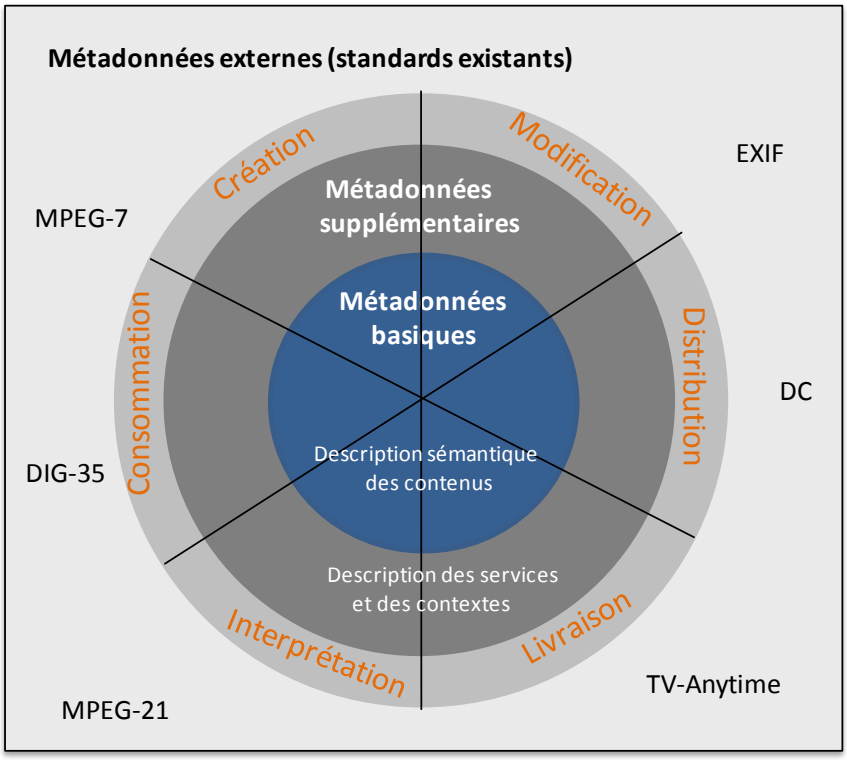

FIGURE 5.1: Les métadonnées nécessaires pour le cycle de vie des contenus.

utilisée pour la recherche et la navigation sémantique et concerne les information décrivant les contenus. La description des profils des utilisateurs ainsi que les caractéristiques matérielles et logicielles de leurs plateformes sont regroupées dans la catégorie des *métadonnées supplémentaires*. Enfin, afin de donner à ces utilisateurs une capacité d'intégrer d'autres formats existants, nous avons défini la troisième catégorie qui est les *métadonnées externes*.

L'intégration de ces trois types de métadonnées est assurée par le métamodèle CAM4Home dont le but est de proposer une description générique et extensible des informations entourant le cycle de vie des contenus multimédia. Dans ce chapitre, nous présentons ce métamodèle en détaillant ses parties constitutives. Nous commençons par introduire la notion d'agrégation des objets multimédia. Ensuite nous présentons le découpage en deux niveaux abstraits et

concrets de chaqu' une des parties (basique, supplémentaire et externe). Ce découpage garantit l'extensibilité et la généricité du métamodèle. Enfin, nous montrons la réalisation technique de ce métamodèle qui a été faite en utilisant le Schéma RDF comme langage de description tout en justifiant nos choix.

5.2 L'agrégation des contenus multimédia

L'objectif du métamodèle CAM4Home est la fusion des métadonnées hétérogènes nécessaires dans toutes les étapes du cycle de vie des contenus multimédia. Le métamodèle CAM4Home est basé sur un concept novateur nommé CAM (Collaborative Aggregated Multimedia). Chaque contenu multimédia (image, vidéo, document, audio, application ou service) est décrit par un ensemble de métadonnées. L'agrégation d'un contenu multimédia avec ses métadonnées spécifiques est appelée *CAMObject*. Plusieurs *CAMObjects* sont regroupés dans des paquets d'informations nommées *CAMBundles*. Un *CAMBundle* contient également un ensemble de métadonnées descriptives pour l'ensemble des *CAMObjects* qu'il contient (ex : relation entre deux *CAMObjects*).

La Figure 5.2 illustre l'agrégation des *CAMObjects* dans un *CAMBundle*. En haut de l'image, deux échantillons de *CAMObjects* se réfèrent respectivement à un service de jeux en ligne et une vidéo publiée par un utilisateur. Le *CAMBundle* regroupe les deux *CAMObjects* en leur ajoutant des métadonnées spécifiques, en permettant également aux utilisateurs de la communauté à la fois de commenter et d'ajouter des tags à l'ensemble de la collection. Cette dernière caractéristique apport la dimension collaborative au métamodèle. La structure du métamodèle CAM4Home a l'avantage de permettre une agrégation flexible des services hétérogènes et des éléments multimédia en une seule et unique description tout en gardant la possibilité donnée aux utilisateurs de faire des annotations (ex : commentaires, tags, etc).

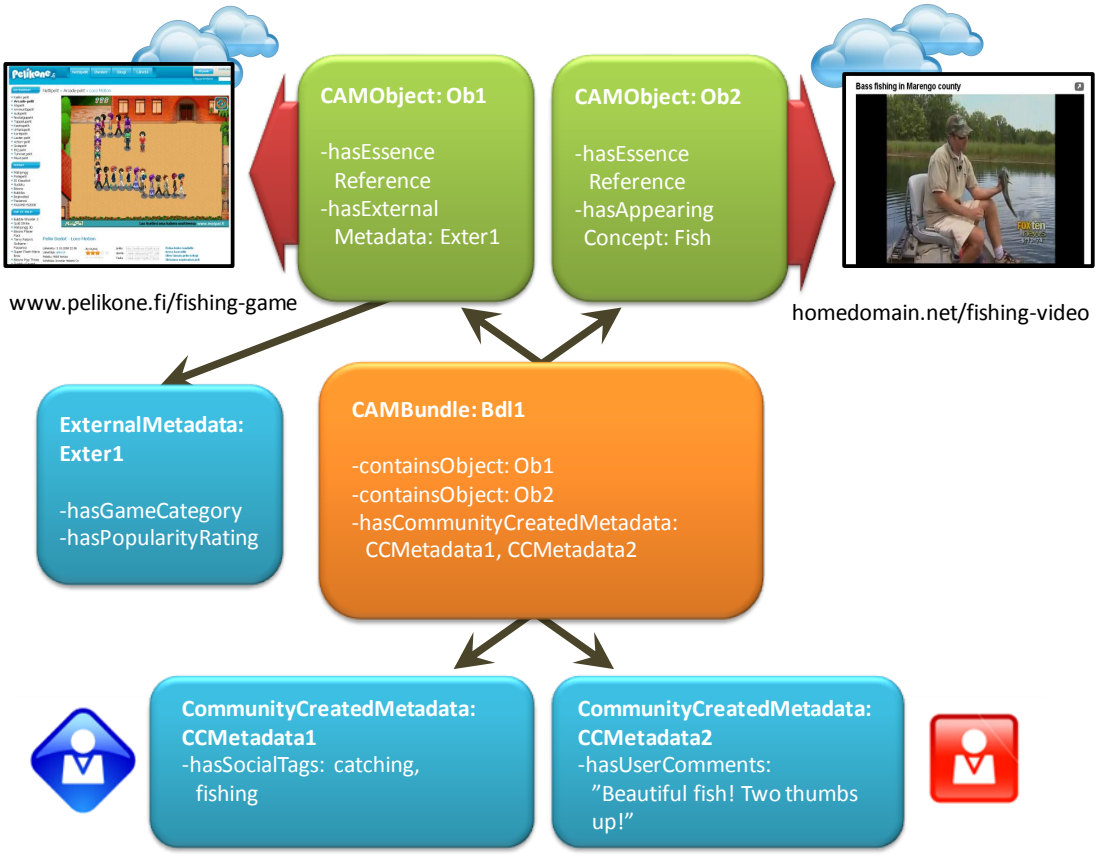

FIGURE 5.2: Exemple de CAMBundle et CAMObject.

En analysant le cycle de vie des *CAMObjects* et des *CAMBundles*, nous pouvons définir la nature des informations nécessaires pour garantir la cohérence du cycle de vie de celles-ci, de leur création à leur consommation. Par exemple, lors de la phase d'interprétation, les métadonnées décrivant les propriétés physiques du contenu sont comparées à d'autres propriétés décrivant le contexte d'utilisation (ex : caractéristiques des appareils, des réseaux, profil des utilisateurs), puis une décision est prise concernant la consommation du contenu dans son form at actuel. Si l'appareil et/ou le réseau ne sont pas capables de lire directement le contenu en l'état, la plateforme ou les services d'adaptation du *CAMBundle* sont sollicités (pour un

transcodage, un transrating etc.) afin de permettre une consommation correcte du contenu. Outre une plateforme de services d'adaptation, le bundle lui-même peut contenir des services d'adaptation spécifiques conçus pour présenter les différents objets multimédia qui composent le bundle.

Lors de la phase d'adaptation, les services ont besoin d'informations sur le contenu et ses propriétés physiques, ainsi que sur les caractéristiques des appareils, des réseaux et du profil de l'utilisateur, afin d'appliquer une transformation du contenu en adéquation avec le contexte de consommation. Ce dernier porte sur les intérêts et les préférences de l'utilisateur et de la communauté, les propriétés logicielles et matérielles, les propriétés du réseau et les services d'adaptation disponibles.

Après cet aperçu des principaux concepts du métamodèle CAM4Home, nous présentons en détail la construction de ce dernier.

5.3 Le métamodèle CAM4Home

Le métamodèle CAM4Home est composé d'un ensemble de structures et de règles nécessaires à l'utilisation des métadonnées permettant de décrire les contenus multimédia ainsi que leur sémantique. Il fournit les concepts de base et les informations nécessaires relatives aux métadonnées dédiées à la distribution collaborative des contenus multimédia. La Figure 5.3 donne un aperçu du métamodèle CAM4Home, composé de deux parties qui assurent l'extensibilité : *le métamodèle abstrait* et *le métamodèle concret*. Chaque métamodèle est composé de trois parties : *le métamodèle abstrait/concret de base*, *le métamodèle abstrait/concret supplémentaire* et *le métamodèle abstrait/concret externe*. Nous rappelons que nous avons adopté ce découpage afin de garantir l'extensibilité et la généricité du métamodèle. Nous présentons dans la suite les différentes parties constitutives du métamodèle CAM4Home.

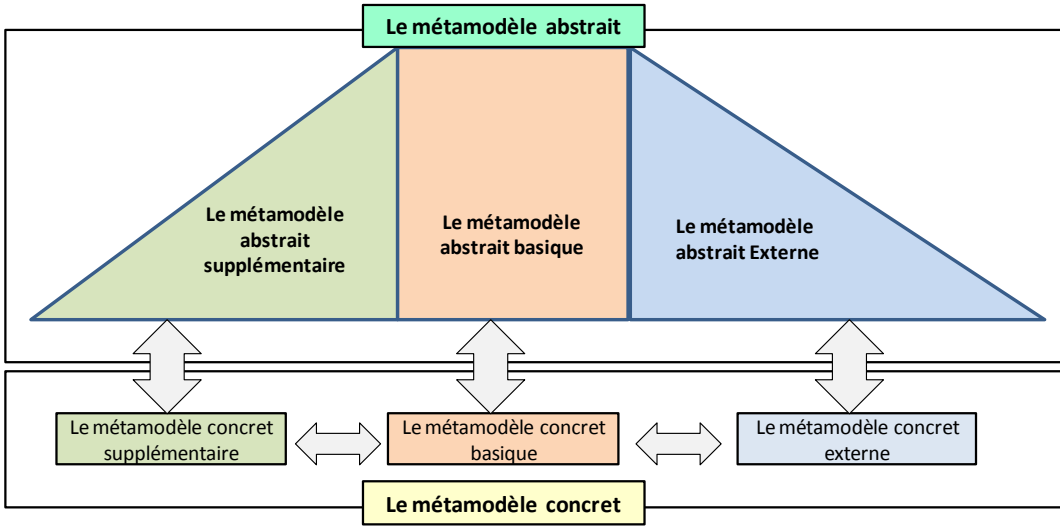

FIGURE 5.3: Le métamodèle CAM4Home.

5.3.1 Le métamodèle abstrait

Le métamodèle CAM4Home est basé sur une description haut niveau assurée par le métamodèle abstrait CAM4Home. Ce dernier agit comme un élément de liaison entre les différentes catégories du métamodèle CAM4Home permettant son extension avec de nouvelles structures en fournissant plusieurs types d'associations. Le métamodèle abstrait définit une catégorisation générique des entités concrètes du métamodèle CAM4Home ainsi que l'association entre ces entités dans le niveau abstrait. Il assure les fonctions suivantes : la description générique, la réutilisation et le partage de connaissances.

Le métamodèle abstrait offre une base commune pour la distribution, la livraison intelligente et la consommation de contenus multimédia qui peut être étendu et spécialisé pour répondre à des sous-domaines plus spécifiques sans perdre la compatibilité du système. Lorsque les concepts communs et leurs relations dans un domaine donné sont spécifiés, le métamodèle de ce domaine peut être efficacement réutilisé pour définir d'autre modèles de métadonnées

pour les autres sous-domaines. Puisque les concepts du domaine sont bien définis, le métamodèle abstrait permet alors une interprétation des métamodèles de sous-domaine étendu. Les restrictions explicites au niveau du métamodèle sont propagées aux entités descendantes. Ainsi, l'extension du métamodèle abstrait garantit l'intégrité du modèle en empêchant ces instanciations interdites.

Le métamodèle abstrait CAM4Home est composé de trois parties : *le métamodèle abstrait basique* qui fournit des informations génériques destinées à la description des contenus, *le métamodèle abstrait supplémentaire* qui introduit une description au plus haut niveau pour les services et les informations contextuelles (materiel, profil d'utilisateur, etc). La troisième partie constitutive du métamodèle abstrait est *le métamodèle abstrait externe*. Ce dernier catégorise les formats de métadonnées existants qui peuvent être associés à la description des contenus et des contextes de diffusion.

5.3.1.1 Le métamodèle abstrait basique

Le métamodèle abstrait basique montré dans la Figure 5.4 regroupe les métadonnées dédiées à la description des contenus (sémantique et structure) et les conteneurs de ces métadonnées (*ContentMetadataContainer*). Un conteneur de métadonnées peut avoir des métadonnées structurées (*ContentMetadata*) ou des métadonnées simples (*Literal*).

La classe *CoreMetadata* est la classe mère de toutes les métadonnées reliées aux contenus. Le conteneur de métadonnées *ContentMetadatContainer* regroupe les métadonnées complexes *ContentMetadata* via l'utilisation de la propriété *hasStructuredMetadata*. Un type de métadonnées complexe (structuré) est une classe contenant un ensemble de propriétés décrivant les contenus et elle peut être spécialisée. Par exemple, *AppearingConcept* est un type complexe de métadonnées contenant des informations sur un concept donné dans une image ou vidéo (ex : le lieu où le concept apparaît, sa position dans l'image, etc). Un conteneur de métadonnées

peut également contenir des métadonnées simples via la relation *simpleMetadata* (ex : title, description, etc).

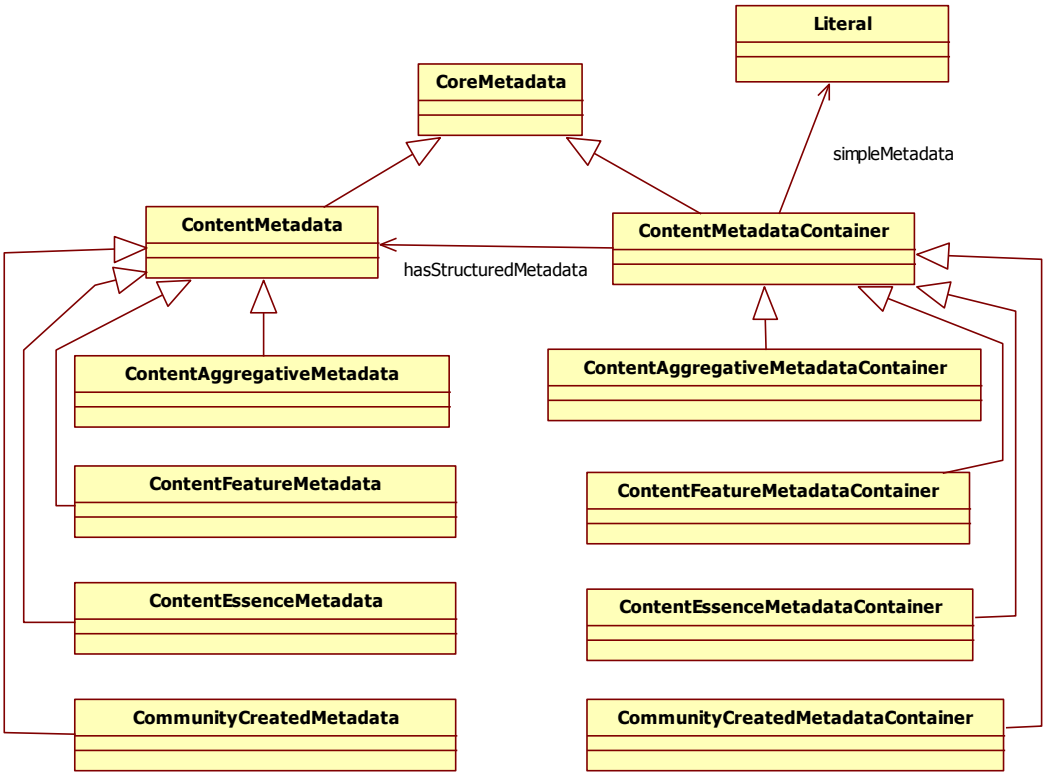

FIGURE 5.4: Le métamodèle abstrait basique.

Les classes *ContentMetadata* et *ContentMetadatContainer* peuvent être spécialisées en fonction du type de métadonnées adressé. Les deux classes *ContentAggregativeMetadata* et *ContentAggregativeMetadataContainer* sont un exemple de cette spécialisation. Elles décrivent l'agrégation des contenus multimédia. Les classes concrètes étendues de ces deux classes sont *Relationship* et *CAMBundleMetadata* respectivement.

La classe *ContentEssenceMetadata* et son conteneur sont utilisés pour décrire le contenu physique et sa localisation (ex l'URL du contenu). *CommunityCreatedMetadata* contient les

informations décrivant les métadonnées créées par la communauté consommant les contenus multimédia (ex : commentaires, tags, ect). La classe *ContentFeatureMetadata* et son conteneur sont utilisés pour la description des caractéristiques des objets multimédia tel que le nom de l'auteur ou une description textuelle du contenu. *ContentFeatureMetadata* est spécialisée pour représenter les concepts qui sont directement reliés aux *CAM Objects*. La classe *ContentFeatureMetadataContainer* est la classe mère de *CAMElementMetadata* et *CAMBundleMetadata*, qui regroupent respectivement les informations décrivant les *CAM Objects* et les *CAM Bundle*. Nous montrons dans la Section 5.3.2.1 des exemples de classes concrètes qui sont crées à partir des classes abstraites introduites précédemment.

5.3.1.2 Le métamodèle abstrait supplémentaire

Cette partie du métamodèle abstrait permet de réaliser l'interopérabilité entre les services afin d'assurer une bonne manipulation des contenus. Elle concerne les métadonnées qui ne sont pas spécifiques à la description des contenus (ex : profils des utilisateurs, profils des appareils, caractéristiques des réseaux, etc). Le métamodèle abstrait supplémentaire (partiellement décrit dans le Figure 5.5) fournit toutes les informations nécessaires reliées aux contextes d'utilisation des contenus multimédia. Les différentes entités de ce métamodèle telles que *User*, *Community*, *Device*, *Network* ou *Service* peuvent être reliées aux différents profils de métadonnées afin de rendre compte des diverses associations entre les éléments du contexte.

La Figure 5.6 montre comment les différentes entités de ce métamodèle sont reliées entre elles. Par exemple, la propriété *UserDeviceReference* relie les deux entités *utilisateur* et *appareil*. D'autres relations dans ce contexte ont été également définies (ex : *BelongsToCommunityReference*, *UserNetworkReference*).

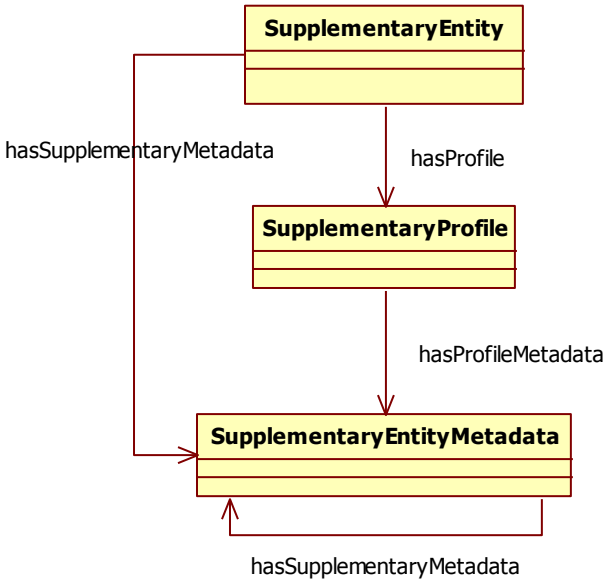

FIGURE 5.5: Une partie du métamodèle abstrait supplémentaire.

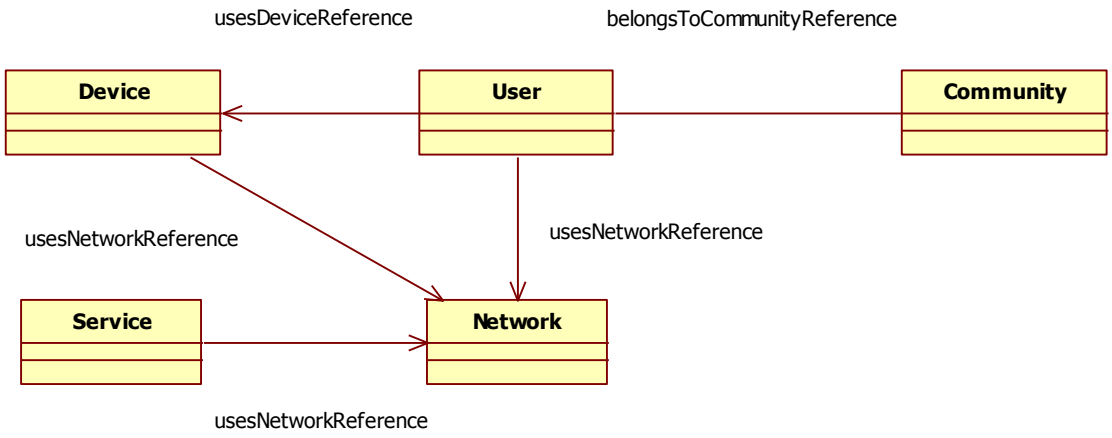

FIGURE 5.6: Les relations entre les entités du métamodèle abstrait supplémentaire.

5.3.1.3 Le métamodèle abstrait externe

Le métamodèle abstrait externe fournit les structures génériques dédiées à l'intégration des formats externes de métadonnées pour le métamodèle CAM4Home. La capacité d'inclure des métadonnées externes au métamodèle CAM4Home est la caractéristique clé qui permet aux utilisateurs de ce métamodèle d'intégrer des standards de métadonnées existants. Deux catégories de métadonnées externes sont définies : les *métadonnées externes basiques* et les *métadonnées externes supplémentaires*. La première catégorie est reliée aux informations décrivant les contenus (*CAMBundle* et *CAMObject*). Quant à la deuxième catégorie, elle est destinée aux métadonnées décrivant les informations contextuelles.

Afin de catégoriser les différents types de métadonnées à intégrer dans le métamodèle CAM4Home, nous avons défini plusieurs sous-classes pour chaque type de métadonnées externes. Par conséquent, nous avons, au niveau abstrait, des classes de métadonnées externes qui sont reliées aux métamodèles basiques et supplémentaires. Ces classes supportent l'intégration des métadonnées externes liées aux contenus ou aux contextes d'utilisation tels que MPEG-7, MPEG-21 et DIG35.

5.3.2 Le métamodèle concret

Le métamodèle concret regroupe l'ensemble des classes instanciables. Il s'agit d'une spécialisation du métamodèle abstrait discuté précédemment et qui permet de répondre aux besoins collectés auprès de divers acteurs du projet CAM4Home. Il est constitué de trois parties complémentaires :

a) Le métamodèle concret basique contenant les informations décrivant la sémantique des *CAMObjects* et *CAMBundle* telles que les relations entre les *CAMObjects*, les informations sur la création, le droit d'auteur, etc.

b) Le métamodèle concret supplémentaire, qui est une spécialisation du métamodèle supplémentaire abstrait. Il regroupe l'ensemble des classes instanciables dédiées à la description des informations contextuelles.

c) Le métamodèle concret externe est utilisé pour lier les formats de métadonnées existants et offrir une scalabilité au métamodèle CAM4Home en lui permettant d'intégrer d'autres standards.

Dans les sections suivantes, nous allons discuter plus en détails ces trois parties constitutives du métamodèle concret CAM4Home.

5.3.2.1 Le métamodèle concret basique

La Figure 5.7 montre une partie du métamodèle concret basique. Il contient les classes et les propriétés décrivant les caractéristiques des contenus (les *CAMBundles* et les *CAMObjects*). Les caractéristiques des *CAMObjects* sont représentées par la classe *CAMElementMetadata*. Quant aux *CAMBundles*, leurs caractéristiques sont représentées par la classe *CAMBundleMetadata*. La relation *containsCAMObjectReference* relie les classes *CAMBundleMetadata* avec *CAMElementMetadata* et la relation *isMetadataOf* qui relie *CAMElementMetadata* avec *CAMElement* sont également montrées dans la Figure 5.7

Chaque classe abstraite décrite dans la Section 5.3.1.1 possède des classes filles spécifiques. A tire d'exemple, la classe *CAMElementMetadata* est une classe concrète étendue de la classe *ContentFeatureMetadataContainer*. Elle est la classe mère de plusieurs sous classes contenant des caractéristiques spécifiques aux contenus multimédia et aux services (*MultimediaElementMetadata*, *ServiceElementMetadata*). Chaque une de ces deux classes a d'autres classes dérivées qui contiennent des informations spécifiques à chaque type de contenu (Image, vidéo, service, etc).

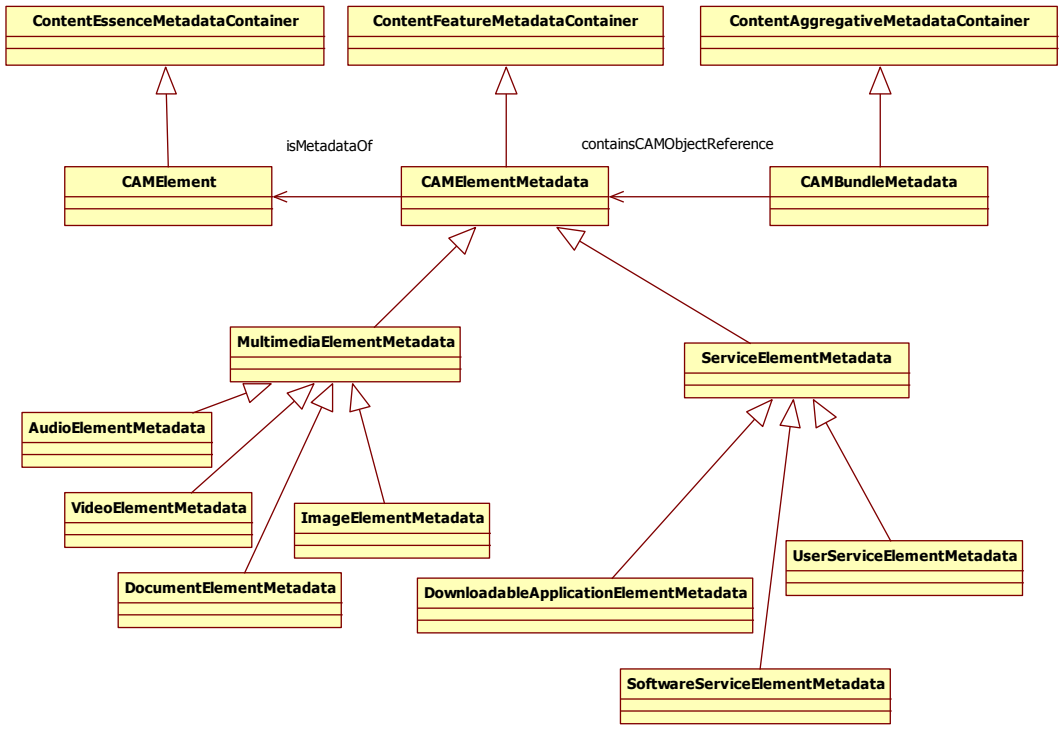

FIGURE 5.7: Relations entres les classes concrètes du métamodèle basique.

5.3.2.2 Le métamodèle concret supplémentaire

Le métamodèle concret supplémentaire est une spécialisation du métamodèle abstrait supplémentaire. La Figure 5.8 montre comment les classes correspondant aux informations décrivant les communautés d'utilisateurs (appartenant au métamodèle concret supplémentaire) sont reliées à leurs classes correspondant au niveau de métamodèle abstrait. La classe abstraite *Community* est spécialisée par la classe *C4HCommunity* qui contient une ou plusieurs classes de type *C4HCommunityProfile* décrivant le profil de la communauté. A travers l'utilisation de la propriété *hasCommunityProfileMetadata*, la classe abstraite *CommunityProfile* est reliée à la classe *CommunityMetadata*. La même méthodologie de modélisation a été appliquée à

toutes les entités du métamodèle abstrait supplémentaire montrées dans la Figure 5.6 (Device, Network, User, Services).

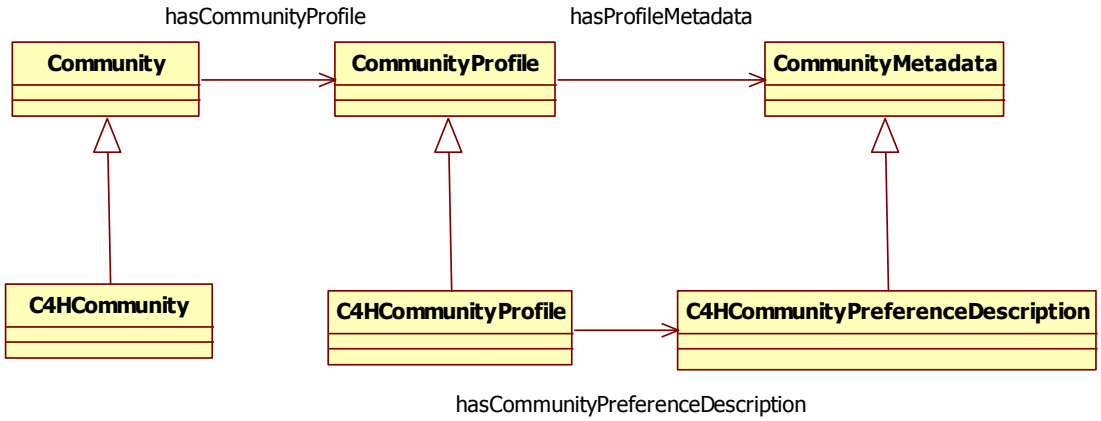

FIGURE 5.8: Les classes concrètes correspondant à la communauté.

5.3.2.3 Le métamodèle concret externe

Le métamodèle concret externe, illustré par la Figure 5.9, est considéré comme un répertoire pour les standards existants qui sont jugés pertinents pour le métamodèle CAM4Home. Ces standards sont catégorisés en fonction des types de données qu'ils décrivent (contenus ou contextes). L'ensemble des classes concrètes externes n'est pas exhaustif, d'autres standards peuvent être intégrés dans le métamodèle CAM4Home grâce à l'utilisation des deux classes génériques, *ExternalCoreMetadata* et *ExternalSupplementaryMetadata* qui sont les classes mères de tous les standards destinés respectivement à la description des contenus et des contextes. La Figure 5.9 montre une partie du métamodèle concret externe. Il représente les classes de métadonnées externes correspondant à la description des contenus (ex : images, video, etc) et des profils d'appareils (MPEG-21, CC-PP, etc).

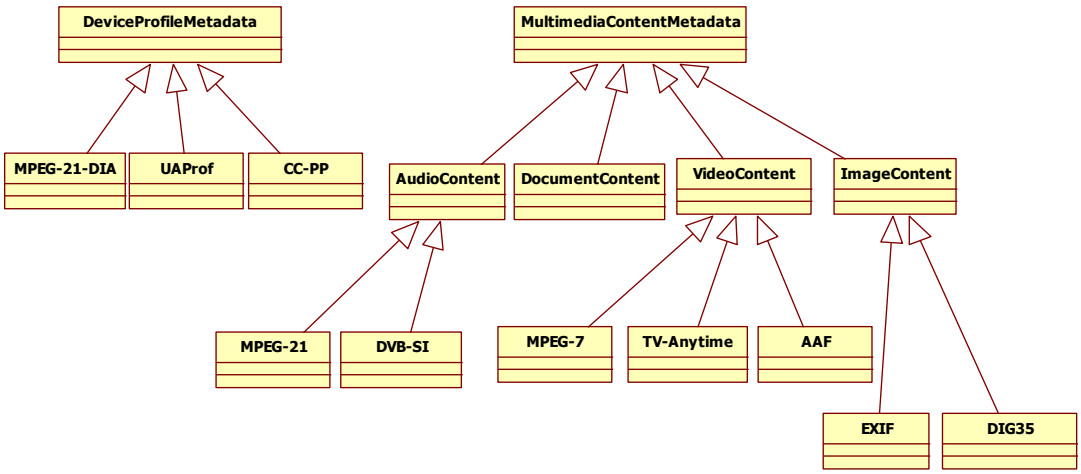

FIGURE 5.9: Une partie du métamodèle concret externe.

5.4 La spécification du métamodèle CAM4Home

Nous avons présenté dans la section précédente le métamodèle CAM4Home qui offre des solutions en terme d'intégration de métadonnées ce qui permet disposer une vue unifiée sur les métadonnées multimédia. Cette section présente la réalisation technique de ce métamodèle. Nous avons choisi le Schéma RDF comme langage de description pour cette réalisation. Les classes représentées dans le métamodèle sont implémentées en utilisant le concept *rdf:class* du Schéma RDF. Quant aux relations, elles sont représentées grace à l'utilisation du *rdf:property*.

Avant de présenter la spécification technique de ce métamodèle, nous discutons tout d'abord les technologies alternatives que nous avons explorées. Nous avons considéré de larges standards de métadonnées couvrant plusieurs types d'informations comme le MPEG-21. Le MPEG-21 couvre des topiques similaires à ceux couverts par CAM4Home pour de la description des contenus et des contextes. Cependant, MPEG-21 ne fournit pas des descriptions à propos des services web ou sur les données créées par la communauté (ex : commentaires des utilisateur, leurs votes, etc). Le couplage entre les standards MPEG-21 et les autres standards comme

WSDL ou OWL-S peut être une solution. L'inconvénient majeur d'une telle solution est que le client doit être capable d'interpréter un nombre important de formats de métadonnées (formats spécifiques à MPEG-21, formats spécifiques à OWL-S, WSDL, etc). Notre objectif est le développement d'une solution d'intégration de métadonnées simple et extensible qui peut être déployable également à des clients légers qui sont typiques dans un environnement réseau à domicile. Nous avons défini un ensemble de descripteurs et de structures de description qui couvrent un large ensemble d'informations nécessaires pour consommer les contenus multimédia dans des environnements hétérogènes. Le métamodèle externe est une partie complémentaire qui permet au métamodèle basique d'inclure d'autres formats de métadonnées.

Nous avons choisi le Schéma RDF comme langage de description pour sa simplicité (par rapport à OWL) et son pouvoir expressif qui couvre toutes les informations décrites par le métamodèle CAM4Home. De plus, le Schéma RDF a plusieurs avantages par rapport au Schéma XML [DHB+00]. Il s'agit d'un langage flexible et extensible, ce qui facilitera l'intégration d'autres informations. Dans la section suivante, nous donnons plus de détails sur la réalisation technique du métamodèle CAM4Home. Des exemples sur l'encodage des classes, des relations et des propriétés sont également fournis.

5.4.1 L'encodage des relations

Afin de préserver la cohérence du métamodèle CAM4Home, plusieurs règles d'intégrité ont été définies. Certaines de ces règles ont été spécifiées en utilisant les relations *composition* et *agrégation* du langage UML [CS00]. Comme le Schéma RDF n'est pas capable de différencier ces deux types de relation, nous avons adopté les conventions de codage suivantes :

– *Les compositions :* sont encodées en utilisant l'encapsulation (déclaration de l'élément dans la classe correspondante) ou via l'utilisation de l'élément *rdf:resource* comme référence. Donc la destruction de l'élément implique la destruction de tout l'objet dans

lequel est contenu cet élément. La Figure 5.10 montre un exemple d'encodage d'une relation de composition entre le concept *VideoElementMetadata* (ligne 1) et *core:AppearingConcept* (ligne 10) à travers la propriété *core:hasAppearingConcepts* (ligne 6).

```
01: <core:VideoElementMetadata rdf:ID="98754512_VEM">
02:  .....
03:  <core:title>
04:    Dummy sport video
05:  </core:title>
06:  <core:hasAppearingConcepts rdf:resource="#joueur1"/>
07:  …
08: </core:VideoElementMetadata>
09:    .....
10: <core:AppearingConcept rdf:ID="joueur1"> ... </...>
```

FIGURE 5.10: Exemple d'encodage d'une relation de type composition.

– *Les agrégations :* sont encodées par l'utilisation du concept *rdf:property*. Cette propriété contient la référence d'une entité du métamodèle, à savoir que, l'entité en question n'est pas encapsulée dans la classe mais à l'extérieur. La Figure 5.11 montre un exemple de ce type d'encodage (agrégation entre *CAMBundleMetadata* et *targetDeviceReference*) où on peut voir que le concept *targetDeviceReference* pointe vers une référence externe qui n'est pas encapsulée dans l'entité *CAMBundleMetadata*, qui est l'identificateur unique de l'entité CAM4Home.

```
01:<core:CAMBundleMetadata rdf:ID="7635241_BDL">
02:<core:targetDeviceReference>1267_NOKIA_DEVICE_UID</...>
03:        ...
04:</core:CAMBundleMetadata>
```

FIGURE 5.11: Exemple d'encodage d'une relation de type agrégation

La définition de ces règles garantit la consistence du métamodèle CAM4Home dans le cas d'ajouts ou de suppressions des entités ou des relations dans le système : tous les membres d'une composition doivent être supprimés dans le cas d'une suppression de la classe mère.

Dans le cas d'une agrégation, la suppression de la classe mère n'affecte pas les autres éléments car ils sont déclarés à l'extérieur de cette classe.

Dans les sections suivantes, nous donnons des exemples d'encodage des classes et des propriétés du métamodèle CAM4Home. Nous introduisons également la spécialisation des concepts abstraits du métamodèle CAM4Home en donnant des exemples de leur instanciation.

5.4.2 L'encodage des classes et des propriétés

La Figure 5.12 montre la définition de la propriété *hasFeatureMetadata* (lignes 7-15) reliant le concept *ContentFeatureMetadata* avec *ContentFeatureMetadataContainer*.

```
01:<rdf:Class
02:  rdf:about="&abstract;ContentFeatureMetadataContainer">
03:  <rdf:subClassOf rdf:resource=
04:    "&abstract;ContentMetadataContainer"/>
05:</rdf:Class>
06:
07:<rdf:Property
08:  rdf:about="&abstract;hasFeatureMetadata">
09:  <rdfs:domain rdf:resource=
10:    "&abstract;ContentFeatureMetadataContainer"/>
11:  <rdfs:range rdf:resource=
12:    "&abstract;ContentFeatureMetadata"/>
13:  <rdfs:subPropertyOf rdf:resource=
14:    "&abstract;hasStructuredMetadata"/>
15:</rdf:Property>
```

FIGURE 5.12: Encodage de la propriété *hasFeatureMetadata*.

Les concepts *rdfs:subClassOf* (ligne 3) et *rdfs:subPropertyOf* (ligne 13) sont utilisés pour relier les classes et les propriétés appartenant au métamodèle basique avec les concepts qui leur correspondent au niveau du métamodèle abstrait.

La Figure 5.13 présente la manière de relier les concepts concrets aux concepts abstraits. Chaque type simple de métadonnées utilisé pour la description des contenus est directement

associé à une sous-propriétés de *simpleFeatureProperty* (lignes 8-10). Les types structurés de métadonnées spécifiques à la description des contenus sont des extensions de la classe *ContentFeatureMetadata* (lignes 12-14). Ces classes sont reliées à leurs conteneurs via l'utilisation de la propriété *hasStructuredMetadata*. Par exemple, dans la Figure 5.13, le concept *hasAppearingConcept* (lignes 15-20) présente une propriété complexe qui relie *AppearingConcept* au *CAMElementMetadata*.

```
01:<rdf:Class rdf:about="&core;CAMElementMetadata">
02:  <rdf:subClassOf rdf:resource=
03:    "&abstract;ContentFeatureMetadataContainer"/>
04:</rdf:Class>

05:<rdf:Property rdf:about="&core;title">
06:  <rdfs:domain rdf:resource="&core;CAMElementMetadata"/>
07:  <rdfs:range  rdf:resource="&xsd;string"/>
08:  <rdfs:subPropertyOf rdf:resource=
09:    "&abstract;simpleFeatureMetadata"/>
10:</rdf:Property>

11:<rdf:Class rdf:about="&core;AppearingConcept"/>
12:  <rdf:subClassOf rdf:resource=
13:    "&abstract;ContentFeatureMetadata"/>
14:</rdf:Class>

15:<rdf:Property rdf:about="&core;hasAppearingConcept">
16:  <rdfs:domain rdf:resource="&core;CAMElementMetadata"/>
17:  <rdfs:range  rdf:resource="&core;AppearingConcept"/>
18:  <rdfs:subPropertyOf
19:    rdf:resource="&abstract;hasFeatureMetadata"/>
20:</rdf:Property>
```

FIGURE 5.13: Les classes CAMElementMetadata et AppearingConcept, et leurs attributs

Les Figures 5.14 et 5.15 donnent des exemples d'instanciation du métamodèle CAM4Home pour l'encodage des *CAMBundles* et *CAMObjects* illustrés dans la Figure 5.2. Le *CAMBundle* (lignes 02-12) regroupe deux *CAMObjects* via la relation *containsCAMObjectReference* et un ensemble de métadonnées descriptives (lignes 07-11). Le *CAMObject*, qui est de type *VideoElementMetadata* (ligne 2) est également décrit par un ensemble de métadonnées spécifiques.

```
01:<rdf:RDF ...>
02: <core:CAMBundleMetadata rdf:about="&inst;B;Bdl1;1">
03:  <core:containsCAMObjectReference>O;Ob1;1</...>
04:  <core:containsCAMObjectReference>O;Ob2;1</...>
05:  <core:hasSharedSocialTags rdf:nodeID="CCMetadata1"/>
06: </core:CAMBundleMetadata>

07: <core:SharedSocialTags rdf:nodeID="CCMetadata1">
08:  <core:serverURI>http://c4h.org/tags</core:serverURI>
09:  <core:hasSocialTag rdf:resource="#catching"/>
10:  <core:hasSocialTag rdf:resource="#fishing"/>
11: </core:SharedSocialTags>
12:</rdf:RDF>
```

FIGURE 5.14: Spécification d'un CAMBundle.

```
01:<rdf:RDF ...>

02: <core:VideoElementMetadata rdf:about="&inst;O;Ob2;1">
03:  <core:title>A sunny weekend</core:title>
04:  <core:creatorReference>c4h:John</core:creatorReference>
05:  <core:legalNotice>free</core:legalNotice>
06:  <core:hasAppearingConcept rdf:nodeID="AP"/>
07:  <core:isMetadataOf rdf:nodeID="VE1">   ...
08: </core:VideoElementMetadata>

09: <core:AppearingConcept rdf:nodeID="AP">
10:   <core:name>fish</core:name>  ...
11: </core:AppearingConcept>

12: <core:VideoElement rdf:nodeID="AP1">
13:  <core:essenceFileIdentifier>
14:    http://homedomain.net/fishing-video </...>
15: </core:VideoElement>

16:</rdf:RDF>
```

FIGURE 5.15: Spécification d'un CAMObject.

5.5 Plate-forme de services CAM4Home

En parallèle avec la spécification RDF, nous avons développé un ensemble de services Web qui assure la gestion des métadonnées CAM4HOME. Les services Web travaillent par rapport

à une instance du métamodèle qui couvrent l'ensemble des métadonnées introduites dans les couches concrètes basique, supplémentaire et externe comme décrites dans les sections précédentes. Cette instance peut évoluer au fur et à mesure que des nouveaux types de métadonnées (ou descripteurs) sont nécessaires. Cette couche de services permet de séparer l'accès aux métadonnées de leur encodage et de leur organisation physique. Ainsi, la formulation de requêtes n'est pas dépendante d'une organisation spécifique, d'une hiérarchisation de fichiers ou d'une structuration d'une base de données stockant les métadonnées. Ceci ouvre la voie à la création collaborative des métadonnées car l'on s'affranchit de stockage physique des métadonnées au sein des fichiers difficilement partageables et modifiables de manière collaborative. Les métadonnées appartiennent à des nuages caractérisant chaque objet (*CAMObject*), agrégation (*CAMBundle*) ou profil décrivant le contexte. Différents acteurs, selon leurs autorisations, peuvent enrichir l'ensemble de métadonnées caractérisant une entité. Sur la Figure 5.16, nous illustrons les services permettant la gestion des descriptions des *CAMObject* et *CAMBundle*. Les services concernant les contenus permettent :

- d'enregistrer (CAM Object/Bundle Registration),
- de rechercher (CAM Object/Bundle Search),
- d'interroger et d'enrichir les métadonnées liées aux contenus (CAM Metadata Processing),
- d'étendre l'instance du métamodèle en enregistrant des nouveaux descripteurs et hiérarchies de descripteurs à travers d'extensions de schémas (CAM Metamodel Management).

Afin de gérer facilement l'extension du modèle, l'entrepôt contenant les descriptions (CAM Metadata Registry) est découpé en quatre parties : une pour le schéma de base ainsi que les extensions, une pour les métadonnées liées aux *CAMObjects*, une pour les métadonnées liées aux *CAMBundles* et une dernière pour les profils décrivant le contexte.

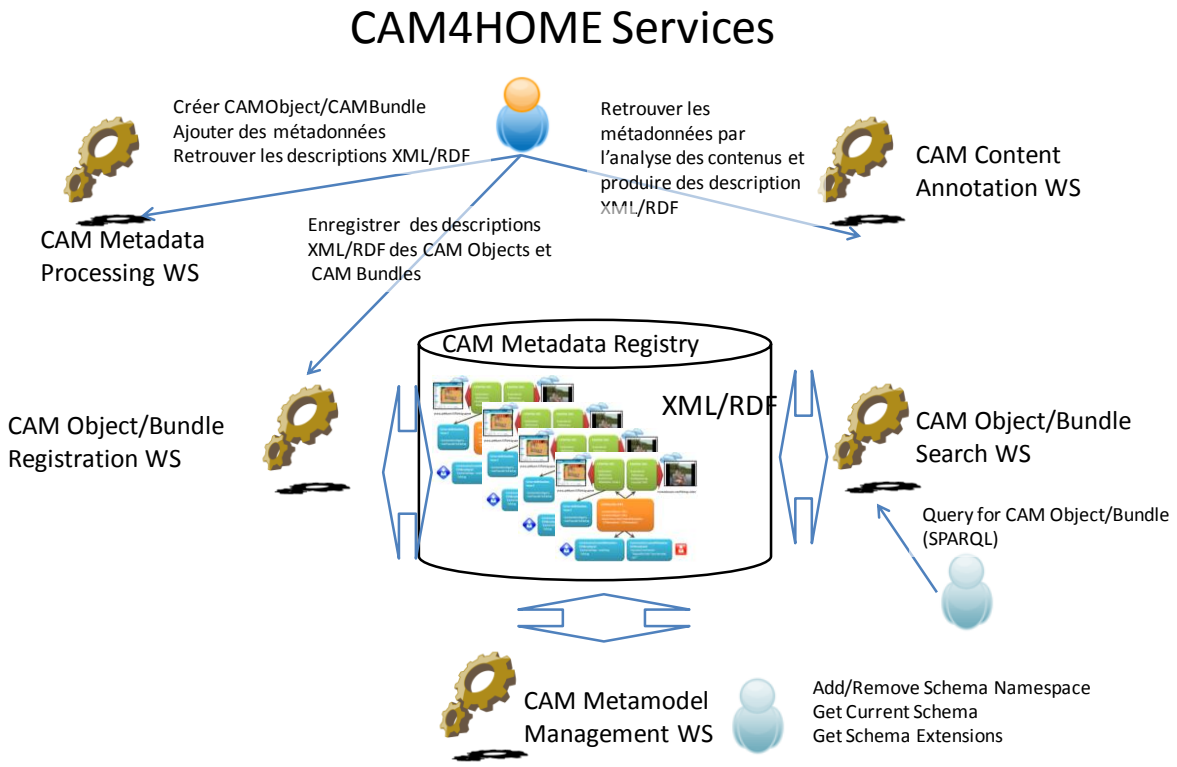

FIGURE 5.16: Les services orientés contenu de la plateforme CAM4HOME.

Le développement de la plate-forme étant le fruit d'un travail collaboratif avec l'Université de Murcie (Espagne), AtosOrigin (Espagne), Université d'Oulu (Finlande) et VTT (Finlande), nous tenons à préciser que nous avons été fortement impliqué dans la mise en place de l'entrepôt, de la spécification du service du gestion du métamodèle (*CAM Metamodel Management* WS), ainsi que dans l'implémentation du services *CAM Metadata Processing* WS (interrogation et ajout des métadonnées issues des auteurs ainsi qu'issues de la communauté des utilisateurs).

Ces services accompagnent l'ensemble du cycle de vie des métadonnées autour d'une entité de la plate-forme CAM4HOME. Dans la Figure 5.17, nous présentons le cycle de vie concernant les descriptions *CAMObject* et *CAMBundle* qui se décline en cinq étapes : création de descriptions (1-4), enrichissement (A-B puis 2-4) interrogation (A-C), déploiement (X), suppression (Z). Les carrés en bas à droite de chaque opération indiquent les services Web concernées : MP pour Interprétation de métadonnées, OR et BR pour Enregistrement de *CAMObject* et *CAMBundles* respectivement, OS et BS pour Recherche de *CAMObject* et *CAMBundles* respectivement.

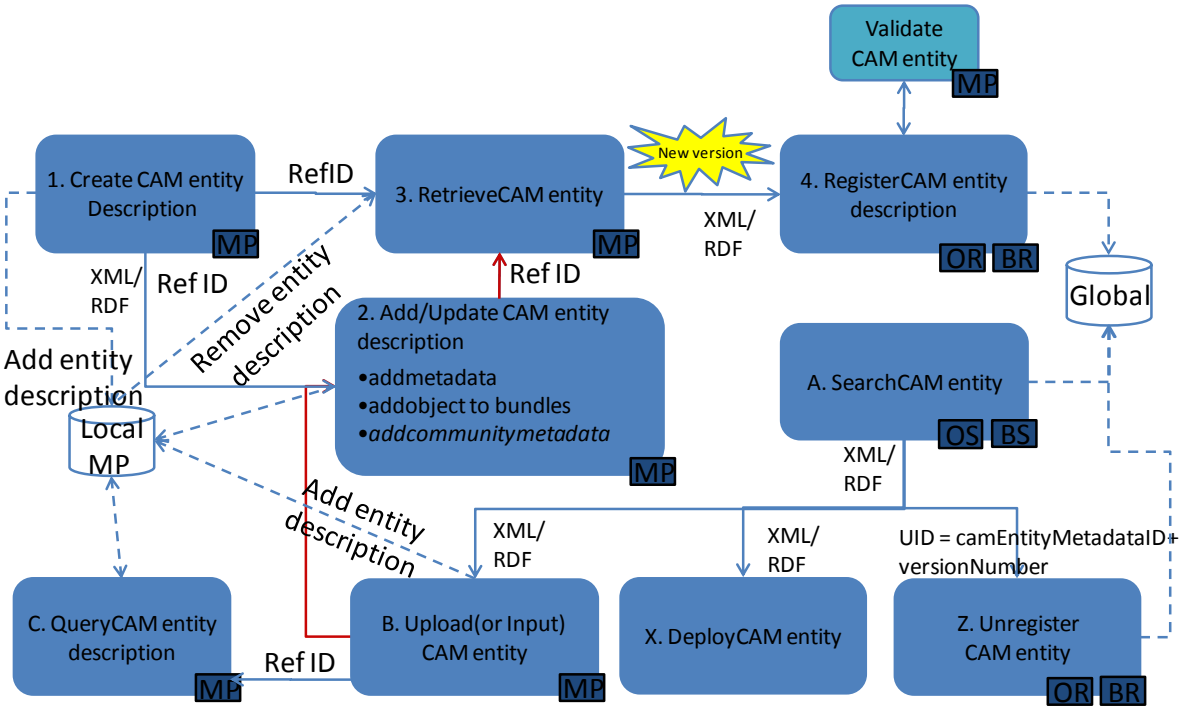

FIGURE 5.17: Cycle de vie des entités contenues de la plateforme.

Dans la suite de cette section nous illustrons :

– la création et l'enrichissement de descriptions *CAMObject* et *CAMBundle*

– l'extension de l'instance du métamodèle

– l'interrogation

5.5.1 Création et enrichissement de descriptions

La Figure 5.18 illustre le processus de création de nouvelles descriptions ou la modification des descriptions existantes. La première étape consiste dans la création (*Create*) ou la récupération (*Upload*) d'une description existante identifiée par un identifiant unique (RefID) au sein des ensembles de descriptions CAM4Home.

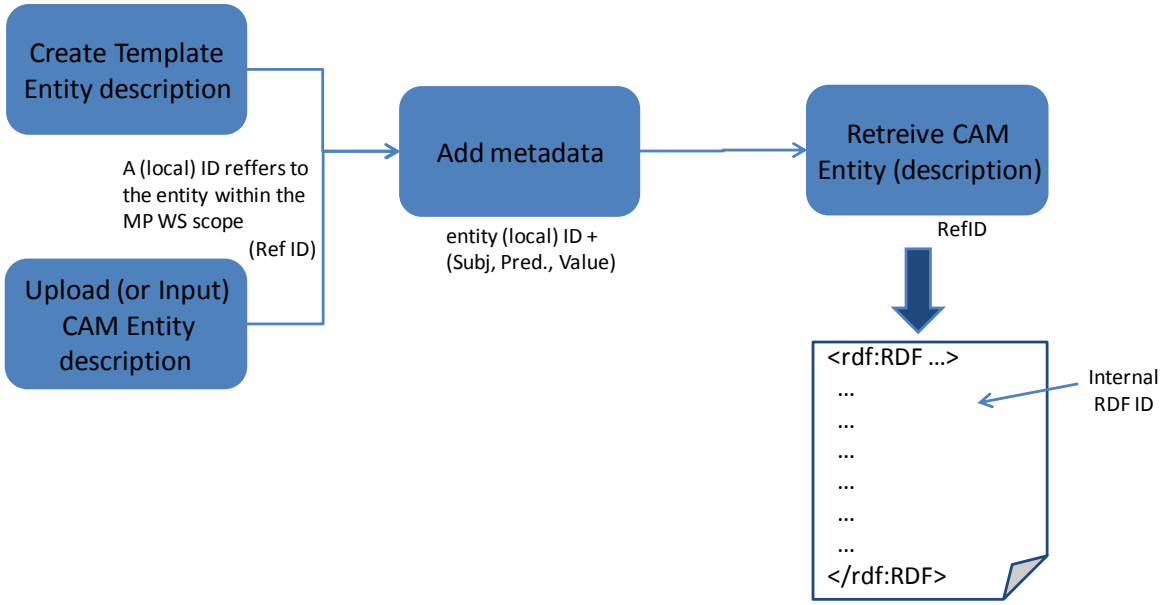

FIGURE 5.18: Création ou mis à jour de la description d'une entité CAM.

Par rapport à cet identifiant nous ajoutons des informations concernant l'entité (*CAMObject* ou *CAMBundle*) ciblée sous la forme de triplets RDF. Il est également possible de récupérer l'ensemble de métadonnées concernant une entité sous forme de flux RDF/XML afin de l'exporter vers un autre entrepôt ou pour l'interroger directement en s'affranchissant des services

d'interrogation proposés.

L'ajout des information à une entité se fait à travers d'une suite de triplets RDF (Sujet, Prédicat, Objet) comme illustré ci-dessus :

– l'ajout d'une propriété simple

```
&inst;#O;zaze-3ez12;0,          <- identifiant (RefID) d'une image (S)
core:thumbnail,                 <- propriété (P)
" http://…/Eiffel_Tower.jpg"    <- valeur (O)
```

– l'ajout d'une propriété complexe

```
(ap1, rdf:type, &core;AppearingConcept)                    <-  créer le conteneur de la propriété
(ap1, core:name, "TourEiffel")                             <- ajouter les divers sous-propriétés au conteneur
(&inst;#O;zaze-3ez12;0, core:hasAppearingConcept, ap1) <- lier le conteneur
```

– l'ajout d'une métadonnée générée (UserComment) par un utilisateur lors de la consultation

```
(c1, rdf:type, &core;UserComment)                  <- créer le conteneur commentaire
(c1, core:comment, "La Tour a 324 m")              <- ajouter la propriété au conteneur
(c1, core:userReference, "c4h::ext::user::sa81")   <- identifier l'auteur
(&inst;#O;zaze-3ez12;0, core:hasSocialTag, ap1)    <- lier le conteneur
```

– l'ajout d'une métadonnée externe (MPEG7_VideoMetadata)

```
(mtd1, rdf:type, &ext;MPEG7_VideoMetadata)                  <- créer le conteneur externe
(mtd1, abs:metadataValue, "<mpeg :7…>…</mpeg7>")            <- ajouter la propriété
(&inst;#O;zaze-3ez12;0, core:hasCoreExternalMetadataTag, mtd1)  <- lier
```

5.5.2 L'extension du métamodèle

Les descriptions CAM4Home peuvent être enrichies à la volée avec des nouveaux descripteurs en fonction des besoins applicatifs. Les concepts de base peuvent être étendus afin d'y introduire des nouveaux descripteurs en utilisant les mécanismes propres au RDF. Ainsi, nous pouvons spécialiser la classe *core:AppearingConcept* par une classe *foot:Joueur* déclarée dans l'espace de nommage " *foot* ". Nous pouvons aussi ajouter des nouvelles propriétés en étendant les propriétés abstraites (*abs:simpleFeatureMetadata*) héritées par *core:AppearingConcept* de *abs:ContentFeatureMetadata*. Dans la Figure 5.19, nous présentons la spécification de la classe *foot:Player* avec ses deux propriétés *foot:nomJoueur* et *foot:nomEquipe* qui s'ajoutent à l'ensemble des propriétés déjà définies (*core:concept*, *core:conceptDesc*, *core:spatialLocationLeft*, *core:beginTimePoint*, etc.) pour *core:AppearingConcept*.

La propriété *nomJoueur* étend la propriété *conceptDesc* et la propriété *nomEquipe* est une nouvelle métadonnée introduite en étendant une propriété abstraite du conteneur *abs:ContentFeatureMetadata* dont hérite *core:AppearingConcept*. Lier les nouvelles propriétés aux propriétés déjà existantes est primordial pour ne pas empiéter sur l'accès aux valeurs de ces propriétés.

Lors de l'interrogation du modèle, nous utilisons les règles d'inférences pour rendre visibles le plus d'information possible en exploitant également les propriétés étendues. Par exemple, lorsque l'on cherche de vidéo où la propriété *conceptDesc* vaut *Zidane*, grâce au moteur d'inférences et au fait que nous avons lié la propriété *nomJoueur* à *conceptDesc*, le moteur retrouvera également la description contenant le conteneur *foot:Joueur*. Ainsi, même les utilisateurs ne connaissant pas l'ensemble des nouvelles propriétés des schémas étendues peuvent en bénéficier.

L'ajout d'information à ces nouvelles propriétés, se fait comme précédemment une fois que la nouvelle extension du schéma est enregistrée auprès du CAM Metamodel Management WS.

```
<!DOCTYPE rdf:RDF [
        <!ENTITY rdf 'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
        <!ENTITY core 'http://www.cam4home-itea.org/model/core#'>
        <!ENTITY XMLSchema 'http://www.w3.org/2001/XMLSchema#'>
        <!ENTITY rdfs 'http://www.w3.org/2000/01/rdf-schema#'>
        <!ENTITY foot 'http://www.lifl.fr/~samir/foot#'>
]>
<rdf:RDF xmlns:rdf="&rdf;" xmlns:rdfs="&rdfs;">
 <rdf:Class rdf:about="&foot;Joueur">
  <rdfs:subClassOf rdf:resource="&core;AppearingConcept"/>
 </rdf:class>

 <rdf:Property rdf:about="&foot;nomJoueur">
  <rdfs:domain rdf:resource="&foot;Joueur"/>
  <rdfs:subPropertyOf rdf:resource="&core;conceptDesc"/>
  <rdfs:range rdf:resource="&XMLSchema;string"/>
 </rdf:Property>

 <rdf:Property rdf:about="&foot;nomEquipe">
  <rdfs:domain rdf:resource="&foot;Joueur"/>
  <rdfs:subPropertyOf rdf:resource="&abs;simpleFeatureContentMetadata"/>
  <rdfs:range rdf:resource="&XMLSchema;string"/>
 </rdf:Property>
</rdf:RDF>
```

FIGURE 5.19: Spécification de la classe *foot:Player*.

Ainsi, nous écrivons un premier triplet RDF pour créer le conteneur, ensuite, nous instancions la propriété *nomJoueur* et finalement, nous lions le conteneur à l'entité concerné à travers la propriété *core:hasAppearingConcept*. La liaison peut se faire car la nouvelle classe *foot:Joueur* étend la classe *AppearingConcept* qui est liée par *hasAppearingConcept* à l'entité *CAMObject* ou *CAMBundle*.

```
(j1, rdf:type, &foot;Joueur)                                    <- créer le conteneur
(j1, foot:nomJoueur, "Zidane")                                  <- ajouter la propriété au conteneur
(&inst;#O;zaze-3ez12;0, core:hasAppearingConcept, j1) <- lier le conteneur
```

5.5.3 Interrogation de descriptions

La première étape du processus d'interrogation consiste dans la recherche de l'identifiant associé aux descriptions concernant le contenu ciblé. C'est les services CAM Object Search WS et CAM Bundle Search WS qui en partant d'une liste de propriétés attributs/valeurs retrouvent les entités satisfaisant les critères imposés. Comme nous l'avons évoqué dans le paragraphe précédent, les recherches se font sur un modèle obtenu par inférence à partir du modèle initial, ainsi l'ensemble de propriété liées à celles de la requête sont assujetti à la recherche. Les propriétés sont remontées vers le haut de la hiérarchie en les généralisant. Une fois les entités retrouvées, nous pouvons accéder aux informations contenues en utilisant l'opération Interpret du CAM Metadata Processing WS. Le service accepte des requêtes en SPARQL permettant d'accéder aux éléments de la description. Ainsi par exemple :

– pour accéder à des métadonnées simples (*core:titre*) nous formulerons de requêtes de la sorte

```
Select ?title
Where &inst;#O; zaze-3ez12;0 core:title ?title
```

– pour accéder à des métadonnées au sein de conteneurs nous matérialisons dans les requêtes les liaisons existantes :

```
Select ? name
Where &inst;#O;zaze-3ez12;0 core:hasAppeaingConcepts ?ap
        ?ap foot:nomJoueur ?name
```

Ainsi, nous avons proposée une solution complète autour du métamodèle, en développant ne plat-forme pour la gestion d'une instance du métamodèle qui peut être étendue sans restriction de fonctionnalités en termes de création, modification et interrogation de métadonnées.

5.6 Conclusion

Nous avons présenté dans ce chapitre un métamodèle qui a pour objectif d'agréger les métadonnées hétérogènes au sein d'un métamodèle unifié afin d'assurer une description riche des contenus multimédia durant leur cycle de vie. Nous avons commencé par introduire la notion d'agrégation des objets multimédia, nous avons ensuite présenté les différentes parties constitutives du métamodèle CAM4Home. La première partie porte sur un métamodèle abstrait décrivant la structure et les propriétés des métadonnées à haut niveau. Le métamodèle abstrait est composé du : métamodèle basique qui assure une description sémantique des contenus, le métamodèle supplémentaire qui contient les informations contextuelles utilisées pour le filtrage et l'adaptation des contenus ainsi que du métamodèle externe qui permet d'enrichir le métamodèle CAM4Home par d'autres métadonnées issues d'autres standards. La deuxième partie constitutive du métamodèle CAM4Home est le métamodèle concret. Il s'agit d'une spécialisation des différentes entités du métamodèle abstrait. Le métamodèle concret est à son tour composé de trois parties concrètes : le métamodèle concret basique, supplémentaire et externe.

Le métamodèle que nous avons proposé est extensible et permet d'intégrer aisément d'autres informations, y compris celles issues des standards existants. Grace à sa flexibilité, nous avons utilisé le Schéma RDF comme langage de description pour la spécification du métamodèle CAM4Home. Enfin nous avons montré quelques exemples illustratifs d'encodage du métamodèle.

Dans le but de simplifier l'utilisation des métadonnées externes via la relation *hasExternalMetadata* qui nécessite une connaissance à priori sur tous les standards utilisés, nous montrons, dans le chapitre suivant, une approche qui permet de trouver les mappings entre les besoins des utilisateurs et les formats de métadonnées externes.

Chapitre 6

MuMIe : un système multi-niveaux pour l'intégration des métadonnées

Sommaire

6.1 Introduction

Nous avons présenté dans le chapitre précédent le métamodèle CAM4Home dédié à l'agrégation des métadonnées hétérogènes. Ce métamodèle offre une description riche en termes de métadonnées et fourni également une possibilité d'intégrer d'autres formats de métadonnées externes par le biais de la propriété *hasExternalMetadata*. Cependant, le client doit avoir une connaissance à priori de tous ces formats externes pour pouvoir les interpréter et les interroger. Dans ce chapitre, nous nous intéressons à l'intégration automatique des métadonnées dans un contexte de médiation. L'utilisateur peut définir ses besoins informationnels en termes de métadonnées. Un système de matching trouve les correspondances sémantiques entre les besoins des utilisateurs et les métadonnées hétérogènes.

Dans l'état de l'art, nous avons présenté les différents types d'hétérogénéité observables au niveau des schémas et des langages de description. Nous avons également montré les limitations des méthodologies de matching existantes. Motivés par les défis principalement présentés dans la Section 4.3, nous abordons dans ce chapitre une nouvelle stratégie de matching que nous baptisons MuMIe (Multi-level Metadata Integration).

Contrairement aux méthodes présentées dans l'état de l'art, l'approche MuMIe illustrée par la Figure 6.1, permet d'obtenir une interopérabilité sur deux niveaux : schémas et langages de description. L'interopérabilité au niveau des langages est faite par la transformation de tous les schémas vers un espace de représentation commun tout en conservant les informations sémantiques et hiérarchiques nécessaires dans le processus d'intégration. Cet espace de représentation joue le rôle d'un squelette pivot composé uniquement des concepts basiques que nous avons définis pour chaque langage. L'apport d'une telle représentation est d'offrir une extensibilité pour notre approche en lui permettant de supporter plusieurs langages de description.

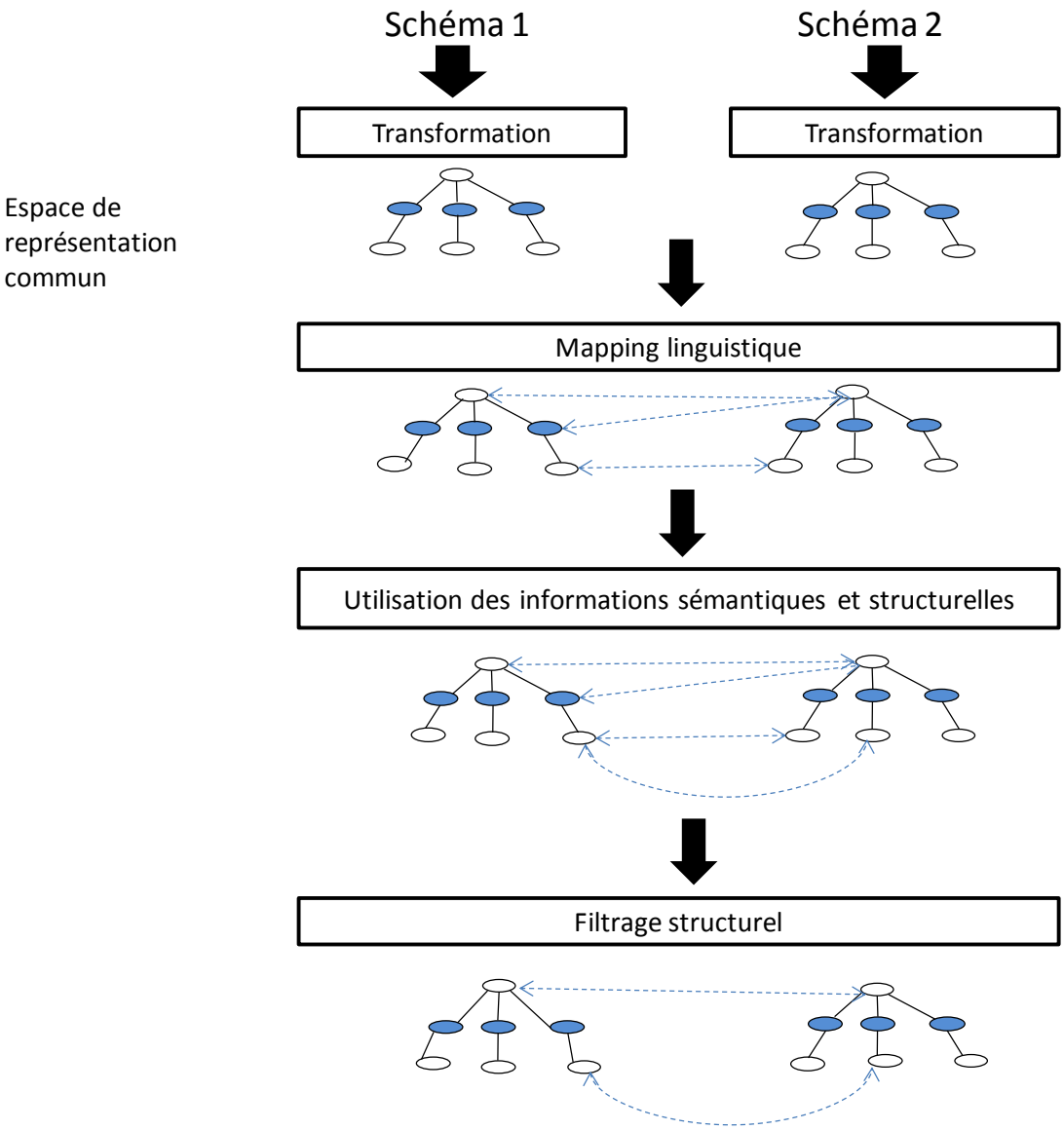

FIGURE 6.1: Schéma général de l'approche MuMIe

Une fois que les schémas issus des langages hétérogènes sont transformés en graphes, nous appliquons des mesures linguistiques pour calculer les similarités entre les nœuds des graphes.

En plus des noms d'entités, nous utilisons également les commentaires correspondant à ces entités comme une autre source d'information linguistique. Un autre apport de notre approche est l'utilisation des informations sémantiques et structurelles conservées durant l'étape de projection, afin de détecter d'autres mappings et spécialement les mappings complexes (n : m).

Etant donné que la similarité entre deux nœuds ne dépend pas uniquement de ces derniers mais également de son voisinage qui définit le contexte, nous utilisons les informations contextuelles afin de calculer la similarité structurelle. Contrairement aux autres méthodes qui utilisent un seul type de voisinage, dans notre méthode nous utilisons plusieurs contextes en même temps (contexte des fils immédiats, des ancêtres et des feuilles). L'ajout de ces informations remédie aux limitations des méthodes utilisant un seul contexte de similarité (SF, Cupid, etc). La similarité entre les contextes est donnée par la distance entre les chemins. Dans notre approche, nous introduisons plusieurs critères de relaxation sur le calcul des distances entre les chemins afin de prendre en compte l'hétérogénéité au niveau des langages de description, ce qui donne une flexibilité dans le processus de matching. Le calcul de la similarité entre les contextes joue le rôle d'un filtre sémantique qui utilise la structure (informations contextuelles) pour éliminer les faux mappings détectés durant le calcul de la similarité linguistique. Enfin, nous réalisons une spécification qui permet de définir la manière de stocker les mappings afin qu'ils puissent être aisément utilisés.

Dans le reste de ce chapitre, nous fournissons en détail les techniques que nous utilisons pour le matching des schémas tout en illustrant nos apports.

6.2 Uniformiser les langages de description

En raison de l'hétérogénéité des langages de définition des schémas, il n'est aucun modèle canonique qui peut supporter l'ensemble des caractéristiques des schémas. Ainsi, nous

proposons dans cette section, une méthode de transformation des langages de description en un espace de représentation commun en capturant uniquement les informations linguistiques, structurelles et sémantiques nécessaires pour le matching.

6.2.1 Construction de graphes

Dans notre approche, illustrée par la Figure 6.2, nous modélisons ces schémas en tant que graphes orientés étiquetés représentant uniquement les concepts basiques des langages de description : les classes, les attributs, les éléments et les propriétés reliant ces entités. Ces concepts sont les informations communes pour tous les langages descriptifs. Autrement dit, ces entités forment un espace de représentation sur lequel toute ontologie est projetable.

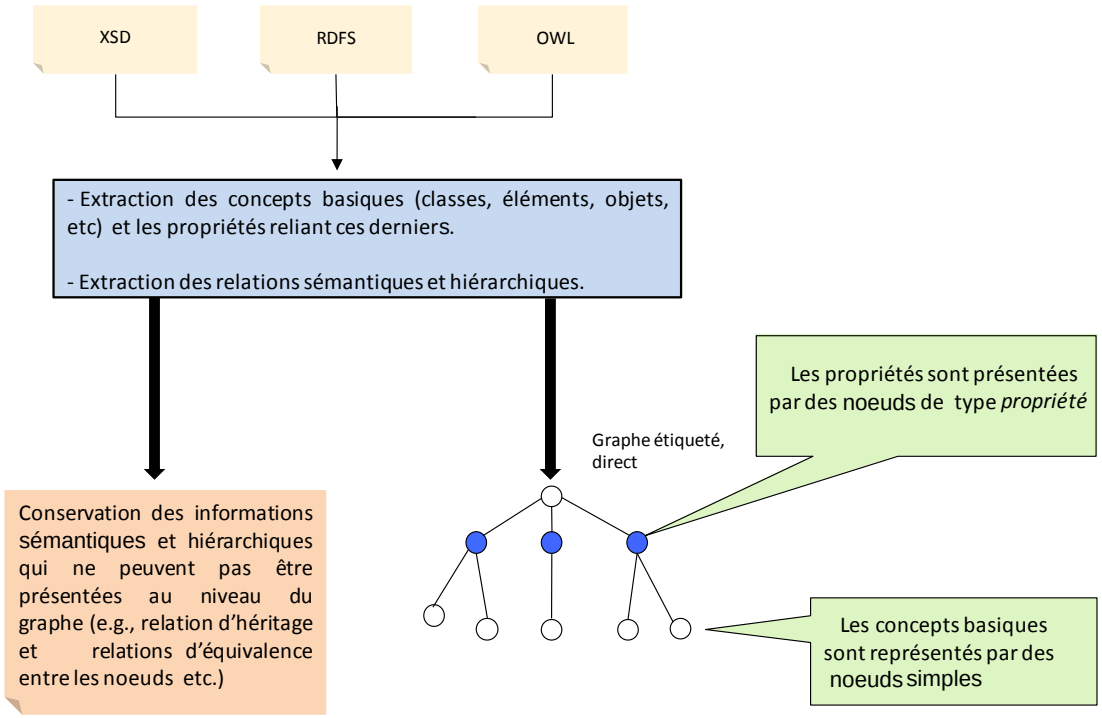

FIGURE 6.2: Illustration de l'étape de construction du graphe.

En se basant sur les travaux de conversion entre les différents langages de description [VDPM+08] [HPSH03] [YSL07], nous avons identifié les mappings entre les concepts basiques et les propriétés sémantiques des trois langages XSD, RDFS et OWL. Ces mappings sont montrés dans le Tableau 6.1.

TABLE 6.1: Les concepts basiques des langages XSD, RDFS et OWL

Type du nœud	**XSD**	**RDFS**	**OWL**
Normal	xsd:element xsd:group xsd:attribute xsd:attributeGroup xsd:complexType	rdfs:class	owl:class
Propriété	le nom du nœud parent + celui du fils	rdf:property	owl:objectProperty

Ce tableau représente un ensemble de mappings que nous avons définies et dans lesquels nous considérons que les éléments *xsd:complexType*, *xsd:attribute*, *xsd:attributeGroup*, *xsd :attributeGroup*, *rdfs:class* et *owl:class* sont équivalents et sont représentés par des nœuds de type *nœud normal* au niveau du graphe. Les propriétés inter-nœuds sont représentées par des nœuds de type *nœud propriété*. Elles correspondent aux entités *rdf:property* et *owl:objectProperty* pour les langages RDFS et OWL. Quant aux propriétés du Schéma XML, nous inspirons de [MVIM07], où les auteurs considèrent que les propriétés inter-nœuds sont implicites et peuvent être capturées en combinant les noms des éléments reliés par ces propriétés.

Afin de clarifier la construction des graphes, nous montrons dans la Figure 6.4 un exemple de graphe construit à partir d'un Schéma RDF dont la spécification est montrée dans la Figure 6.3. Cette spécification contient trois entités : *Image*, *Person* et *URI* qui sont représentées par des *nœuds normaux*. Les propriétés reliant ces concepts : *createdBy* et *hasReference*, sont représentées par des *nœuds propriété*.

```
01:<rdf:RDF>
02:--------------------
03:<rdfs:Class rdf:about="Image">
04:       <rdfs:subClassOf rdf:resource="&abs;Object"/>
05:</rdfs:Class>
06:
07:<rdf:Property rdf:about="hasReference">
08:       <rdfs:domain rdf:resource="Image"/>
09:       <rdfs:range rdf:resource="URI"/>
10:</rdf:Property>
11:
12:<rdf:Property rdf:about="createdBy">
13:       <rdfs:range rdf:resource="Image"/>
14:       <rdfs:domain rdf:resource="Person"/>
15:</rdf:Property>
16:--------------------
17:</rdf:RDF>
```

FIGURE 6.3: Un extrait d'une description RDFS

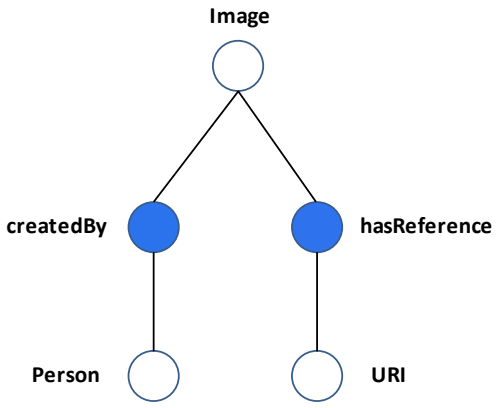

FIGURE 6.4: Graphe construit à partir d'une description RDFS

6.2.2 Informations sémantiques et structurelles

Plusieurs types d'information sémantique et structurelle peuvent être spécifiés avec les différents langages de description. Cependant, dans notre approche, nous ne tenons pas compte

de toutes ces informations car certaines d'entre elles constituent des contraintes non pertinentes pour le processus de matching. Par exemple, si on tient compte de l'information *xsd:sequence* montrée dans la Figure 6.5, qui indique que l'ordre des éléments *firstName* et *lastName* doit être respecté, l'information portée par cette dernière ne sera pas identique à celle présentée dans la Figure 6.6 alors qu'il s'agit sémantiquement de la même entité.

```
01:<xsd:element name="Person">
02:  <xsd:complexType>
03:    <xsd:sequence>
04:      <xsd:element name="firstName" type="xs:string"/>
05:      <xsd:element name="lastName" type="xs:string"/>
06:    </xsd:sequence>
07:  </xsd:complexType>
08:</xsd:element>
```

FIGURE 6.5: Un type complexe *person* avec la balise *xsd:sequence*

```
01:<xsd:element name="Person">
02:  <xsd:complexType>
03:      <xsd:element name="lastName" type="xs:string"/>
04:      <xsd:element name="firstName" type="xs:string"/>
05:  </xsd:complexType>
06:</xsd:element>
```

FIGURE 6.6: Un type complexe *person* sans la balise *xsd:sequence*

Le type de données est un autre exemple d'informations non prises en compte par notre approche car on peut trouver des entités représentant la même information mais avec un type de donnée différent (ex : *xsd:complexType* et *xsd:string*). De plus, le type des données n'est pas pris en compte par tous les langages de description. Par exemple, pour le Schéma RDF, toutes les données simples sont de type *rdfs:literal* sans distinction de type.

Le Tableau 6.2 montre toutes les caractéristiques sémantiques et hiérarchiques utilisées dans le processus de matching. Ces informations ne sont pas représentées par des nœuds dans le graphe mais comme des propriétés de ces derniers. Nous montrons dans la Section 6.3.3 des exemples d'utilisation de ces règles pour détecter les mappings entre les nœuds des graphes.

TABLE 6.2: Les caractéristiques sémantiques et hiérarchiques

XSD	xsd:restriction, xsd:abstraction, xsd:extension, xsd:substitutionGroup
RDFS	rdfs:seeAlso, rdfs:isDefinedBy, rdfs:subClassOf
OWL	owl:sameAs, owl:unionOf, owl:complementOf, owl:intersectionOf, owl:equivalentClass, owl:disjointWith, rdfs:subClassOf, owl:AllDifferent, owl:equivalentProperty, owl:distinctMembers, owl:TransitiveProperty, owl:someValuesFrom owl:differentFrom

6.3 Le processus de matching

Une fois que les schémas sont transformés en graphes, les informations inutiles sont filtrées. Ensuite, nous calculons les similarités entre les nœuds de ces graphes en utilisant plusieurs types d'information linguistique (noms d'éléments, commentaires, etc). Les informations structurelles sont également exploitées pour éliminer les faux candidats issus de la première étape de matching. La Figure 6.7 montre les trois phases de matching : le pré-traitement, le calcul de la similarité linguistique et le calcul de la similarité structurelle.

6.3.1 Pré-traitement

Dans cette étape, nous commençons par l'analyse de toutes les entités linguistiques impliquées dans le processus de matching (les noms des nœuds et les commentaires correspondant). Puis ces entités sont filtrées et normalisées afin de les rendre utilisables lors l'étape de calcul de la similarité linguistique.

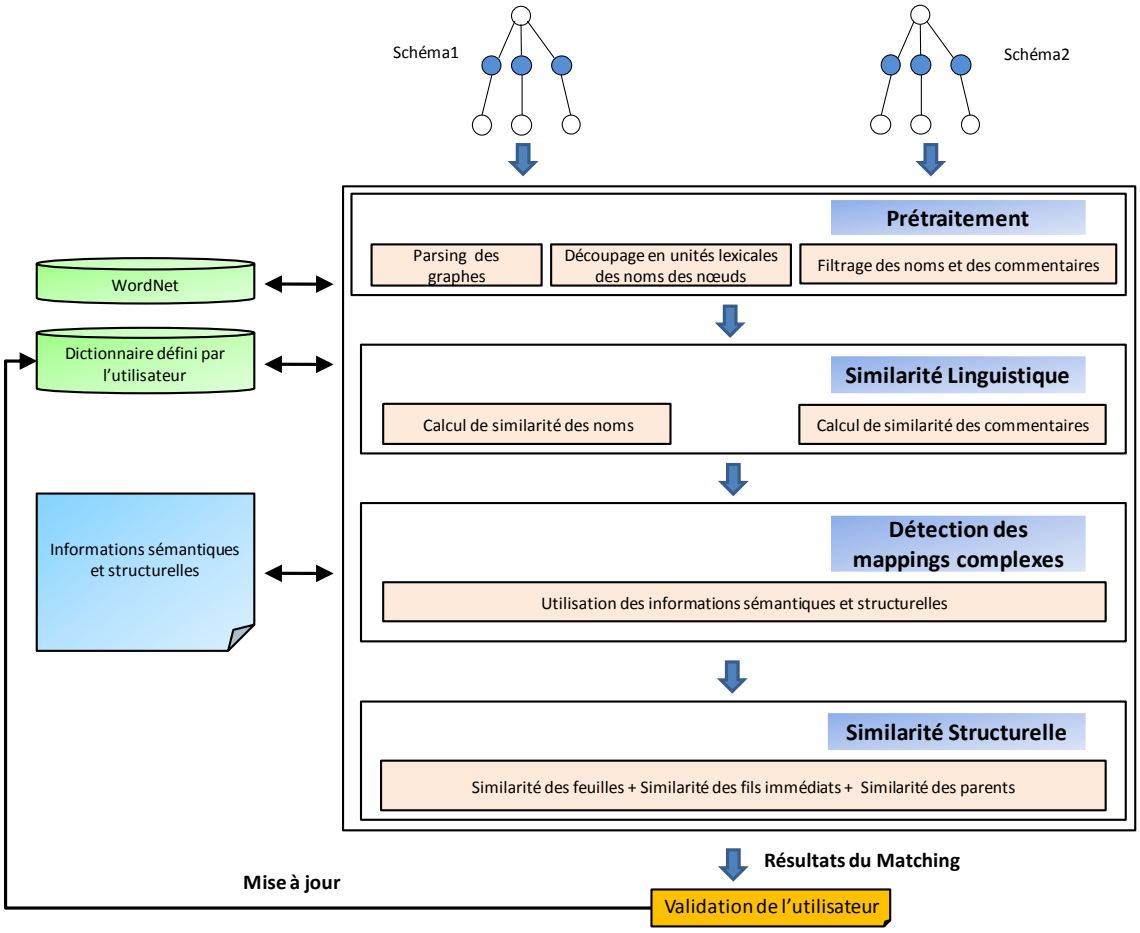

FIGURE 6.7: Les trois phases de matching.

a) Pré-traitement des noms d'entités : le nom d'un nœud est une chaîne de caractères sans espace qui peut être un mot, un terme, ou une expression (une combinaison des mots). Afin de calculer la similarité entre les noms des nœuds, une étape de normalisation est nécessaire. D'abord les noms des nœuds sont découpés en ensembles d'unités lexicales s'ils s'agit de combinaisons de termes reliés par des ponctuations, des chiffres ou des majuscules (Tokenization). Par exemple, *MediaRegionLocator* devient (*Media*, *Region*, *Locator*) ou *Basic_Image_Param* devient (*Basic*, *Image*, *Param*). Une fois que le découpage en unités lexicales est fait, ces unités

sont morphologiquement analysées afin de trouver leurs formes basiques possibles (Lemmatisation). Par exemple, *Locations* est associé à sa forme singulière, *Location*.

b) Pré-traitement des commentaires : les commentaires sont utilisés comme une autre source d'information linguistique via l'utilisation des techniques issues de la recherche d'information. Pour ce faire, les commentaires sont linguistiquement filtrés en éliminant les mots ne portant aucune information utile, comme les articles, les prépositions, les conjonctions, les pronoms et les verbes modaux [vR79].

6.3.2 La similarité linguistique

Dans cette étape, nous calculons la similarité linguistique entre toutes les paires des nœuds dans les deux schémas à aligner. La similarité linguistique est calculée à partir de la similarité des noms et la similarité des commentaires.

a) Similarité des noms : l'objectif de cette étape est de trouver un alignement initial par le calcul de similarité entre les noms des nœuds dans les deux schémas à aligner. Chaque nœud est représenté par un ensemble d'unités linguistiques (*tokens*). Nous commençons d'abord par l'explicitation du sens des unités lexicales par l'utilisation d'une source linguistique. En utilisant WordNet [Fel98], on peut trouver les synonymes d'un terme donné. Cela contribue à résoudre les problèmes d'hétérogénéité survenant lorsque les communautés développant les métadonnées utilisent des termes différents pour décrire la même information. Par exemple, quelques communautés [wg08] utilisent le terme *type* pour décrire le type d'un contenu multimédia. D'autres [LBSM08] utilisent le terme *format* ou *genre* pour décrire la même information. D'autres ressources externes comme DBpedia [ABK$^+$07] ou PMI [Tur01b] peuvent être également utilisées.

Après l'étape d'explicitation, chaque nœud n_i représenté par un ensemble d'unités lexicales M_i aura des ensembles de synonymes *synsets* pour chaque unité lexicale m_i. $M_i^{'}$ est le

résultat final qui regroupe tous les *synsets* résultant de l'explicitation de M_i.

$$M_i^{'} = M_i \bigcup \{m_k | \exists m_j \in M_i \bigcap m_k \in synset(m_j)\}$$

La similarité des noms S_{nom} entre deux nœuds (n_1, n_2) appartenant à deux graphes est calculée en utilisant la distance de Jaro-Winkler (JW) ([BMC+03]) entre chaque unité lexicale $m_i \in M_1^{'}$ et toutes les unité lexicales $m_j \in M_2$ (et vice versa) ([LS08]). Le choix de cette distance s'appuie sur l'étude comparative faite dans [CRF03] où les expérimentations faites par les auteurs prouvent les performances de la distance de Jaro-Winkler dans le processus de matching linguistique. Ensuite pour chaque token, on retient le score (MJW) maximum :

$$MJW(m_i, M_k^{'}) = max_{m_j \in M_k^{'}} JW(m_i, m_j)$$

Finalement, la moyenne des meilleures similarités est calculée :

$$\mathrm{S}_{nom}(n_1, n_2) = \frac{\sum_{m_i \in M_1} MJW(m_i, M_2^{'}) + \sum_{m_j \in M_2} MJW(m_j, M_1^{'})}{|M_1| + |M_2|} \quad (6.1)$$

b) Similarité des commentaires : en raison de la complexité du vocabulaire technique, l'information portée par les noms des nœuds est parfois insuffisante. En conséquence, plusieurs nœuds peuvent avoir une faible valeur de similarité S_{nom} même s'ils décrivent la même information. Afin de faire face à ce manque d'informations, notre méthode utilise les commentaires correspondant aux nœuds (e.g., *rdfs*:*comment*, *xsd :documentation*, etc.) comme une deuxième source sémantique. Nous appliquons la technique TF/IDF [vR79] utilisée dans le domaine de la recherche d'information pour calculer la similarité entre les commentaires. Pour ce faire, tous les commentaires dans les deux schémas à intégrer sont considérés comme des

documents. Chaque nœud sera représenté par un vecteur dont les coordonnées sont les résultats de TF/IDF. Par conséquent, la similarité entre deux nœuds est la distance entre les deux vecteurs correspondant à leurs commentaires. Afin d'illustrer le calcul de ces vecteurs, nous supposons que $v = (w_1, w_2, ..., w_P)$ est le vecteur représentant un nœud donné n. $P = |U|$ est le nombre des mots distincts dans tous les commentaires appartenant aux deux schémas. Le i_{th} élément w_i de v, qui représente le nœud n dans un schéma, est calculé comme suit :

$$w_i = tf_i * idf_i \qquad idf_i = \log_2 \frac{N}{b_i} \tag{6.2}$$

où tf_i est la fréquence du terme. tf_i représente le nombre de fois que le i_{th} mot dans U apparaît dans le commentaire correspondant à n_i. idf_i est la fréquence inverse de document. Elle sert à calculer le logarithme de l'inverse de la proportion de documents du corpus qui contiennent le mot w_i. N est le nombre de commentaires dans U, dans les deux schémas. b_i est le nombre de commentaires qui contiennent le mot w_i au moins une fois.

Comme nous l'avons mentionné précédemment, la similarité entre deux nœuds n_i et n_j est la distance entre les vecteurs correspondant à leurs commentaires v_i et v_j. Cette distance est une similarité cosinus. Elle est calculée comme suit :

$$S_{commentaire}(v_i, v_j) = \frac{\sum_{k=1}^{P} w_{ik} w_{jk}}{\sqrt{\sum_{k=1}^{P} (w_{ik})^2 * \sum_{k=1}^{P} (w_{jk})^2}} \tag{6.3}$$

Le résultat des calculs effectués précédemment est une matrice de similarité linguistique *lSim* calculée à partir de la somme pondérée des similarités des noms et des commentaires :

$$lSim(n_i, n_j) = \mu_1 * S_{nom}(n_i, n_j) + \mu_2 * S_{commentaire}(n_i, n_j) \tag{6.4}$$

où $\mu_1 + \mu_2 = 1$ et $(\mu_1, \mu_2) \geq 0$.

6.3.3 Utilisation des informations hiérarchiques et sémantiques

Les informations sémantiques et structurelles retenues à l'étape de construction des graphes sont utilisées par notre système pour détecter spécialement les mappings complexes (n :m). Pour ce faire, nous introduisons dans notre approche, un ensemble de règles spécifiques à chaque type d'information. Ces règles permettent de déduire d'autres en fonction de ceux déjà détectés dans le calcul de la similarité linguistique. Dans la suite de cette section, nous montrons quelques exemples d'utilisation de ces informations :

- **Relations de généralisation :** la relation de généralisation entre deux types indique que l'un est un sous-ensemble de l'autre (e.g., *xsd:extension*, *rdfs:subClassOf*, *xsd*:*abstraction*, etc.). Cette information nous aide dans la détection des mappings complexes (n :m). Par exemple, si on considère que *ms*:*Agent* est un nœud défini dans un schéma qui sera aligné à MPEG-7, cet attribut aura une valeur importante *lSim* avec le nœud *mpeg7* :*AgentType*. A ce stade, l'union des deux éléments *mpeg7*:*PersonType* et *mpeg7*:*OrganizationType* sera également considérée comme candidate pour le matching avec l'attribut *ms*:*Agent* car *mpeg7*:*OrganizationType* et *mpeg7*:*PersonType* sont des extensions de type *mpeg7* :*AgentType*. D'autres informations structurelles sont également utilisées (e.g., *owl* :*disjointWith*, *owl*:*unionOf*, etc.).
- **Relations sémantiques :** Si on considère que n_i et n_j sont deux nœuds ayant un niveau de similarité important, et n_k est un autre nœud correspondant à une classe ayant une propriété d'équivalence avec celle correspondant à n_i (e.g., *owl*:*equivalentProperty*, *owl* :*equivalentClass*, *xsd*:*substitutionGroup*, *owl*:*sameAs*, etc.), cette information nous permet de déduire que n_k est un autre nœud candidat pour n_j. Notre approche utilise aussi d'autres relations dans la détection des matchings complexes. Par exemple, la propriété *owl*:*differentFrom* permet de faire une distinction entre deux classes C_1 et C_2. Si ces deux classes appartenant à un schéma S_1 correspondent avec une classe C_3 appartenant à un

autre schéma S_2, on considère C_3 correspond la disjonction de ces deux classes.

Les mappings obtenus suite à l'analyse linguistique et sémantique sont souvent nombreux et inadéquats à cause des *faux-amis* et de la largesse consentie, notamment en ce qui concerne l'utilisation des synonymes. Dans notre vision de l'intégration, nous souhaitons élargir les possibilités des mappings au niveau linguistique et éliminer ensuite les faux mappings en appliquant un filtrage qui s'appuie sur des notions structurelles et sémantiques telles que décrites dans la section suivante.

6.3.4 Les informations structurelles

Nous adoptons un calcul de similarité structurelle pour éliminer les faux candidats détectés lors du calcul de la similarité linguistique. Par exemple, les deux attributs de MPEG-7 [BZCS01] *mpeg7:MediaRelTimePoint* et *mpeg7:MediaRelIncrTimePoint* ont une valeur élevée de similarité linguistique avec l'attribut *ms:MediaTimePoint* défini au niveau du schéma médiateur. Alors que leur sens est totalement différent. Afin de faire face à ce problème, nous utilisons les informations structurelles pour filtrer les faux candidats. Par rapport à l'état de l'art nous utilisons trois contextes distincts (ancêtres, fils immédiats, feuilles) ce qui augmente la pertinence du filtrage. Dans cette section, nous commençons par introduire la notion de contexte de nœud, puis nous présentons la méthode que nous utilisons pour calculer la similarité entre les contextes. La similarité structurelle, qui est une combinaison entre les scores relatifs à chaque contexte est ensuite calculée. Cette valeur permet de sélectionner les candidats finaux pour le mapping.

6.3.4.1 Les contextes d'un nœud

Le contexte d'un nœud donné caractérise le voisinage dans lequel se trouve ce nœud. Nous avons montré dans le Section 4.3 les limitations des méthodes existantes qui utilisent la struc-

ture dans le processus de matching. L'une des principales limitations est leur façon d'utiliser l'information structurelle (contextuelle). Afin de remédier à ces limitations, nous considérons dans notre approche la définition du contexte donnée dans [LYHY02] où les auteurs considèrent que le contexte d'un nœud est donné par ces ancêtres, ces fils immédiats et ces feuilles (Figure 6.8) :

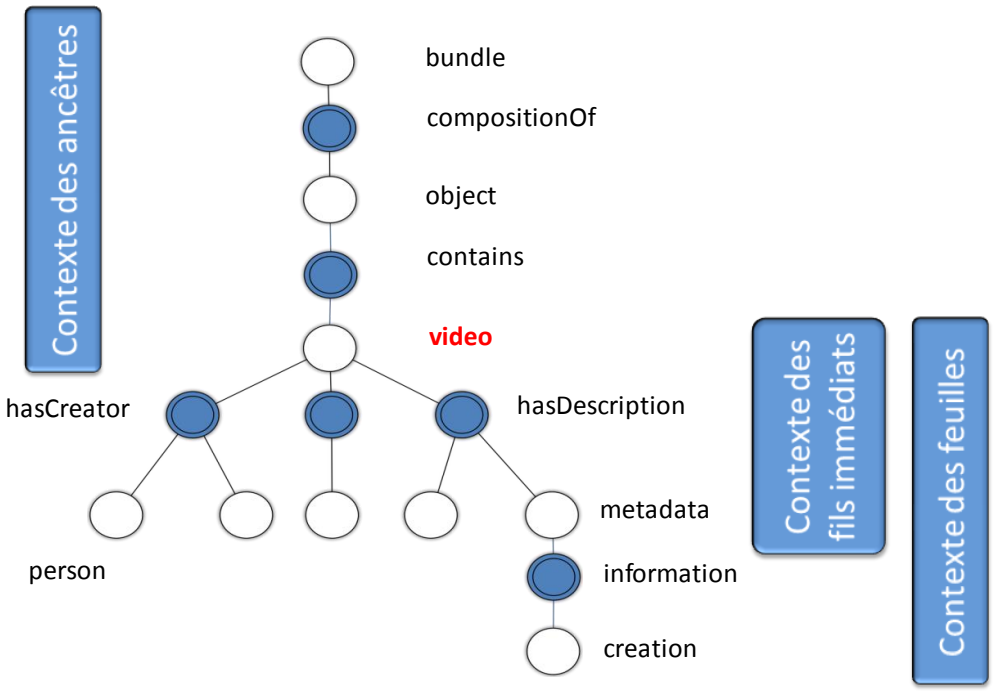

FIGURE 6.8: Les contextes d'un élément.

a) Le contexte des ancêtres : les ancêtres d'un nœud n_i sont définis par le chemin p_i s'étendant de n_i jusqu'à la racine du graphe. Par exemple, le contexte des ancêtres du nœud *video* dans la Figure 6.8 est donné par le chemin (*bundle, compositionOf, object, contains, video*). Cela indique que le nœud *video* décrit la vidéo qui est contenue dans un *object*, et que le *bundle* est une composition d'*objects*.

b) Le contexte des fils immédiats : le contexte des fils immédiats d'un nœud n_i est composé de l'ensemble de nœuds simples qui sont les descendants immédiats de n_i. Les nœuds (*person, hasCreator, video*) et (*metadata, hasDescription, video*) forment le contexte des fils immédiats pour le nœud vidéo.

c) Le contexte des feuilles : les feuilles d'un schéma sont les données atomiques décrites par ce dernier. A savoir, les éléments qui contiennent des valeurs. Le contexte des feuilles d'un nœud n_i est composé par l'ensemble des chemins directs reliant n_i à ces feuilles. Par exemple, les chemins (*creation, information, metadata, hasDescription, video*) et (*person, hasCreator, video*) font parti du contexte de feuilles pour le nœud *video* (Figure 6.8).

6.3.4.2 Relaxation des contraintes linguistiques et structurelles

Dans cette section nous montrons notre contribution dan le calcul de similarité structurelle. Nous proposons une nouvelle adaptation et relaxation des quatre paramètres introduits dans la Section 4.2.1.2 :

- Pour les quatre paramètres, la plus longue sous-séquence commune (lcs) est calculée en fonction de la matrice de similarité linguistique *lSim* calculée dans la section précédente. Cela signifie que deux nœuds (n_i, n_j) sont considérés identiques si la valeur de $lSim(n_i, n_j)$ est supérieure à un seuil donné (ex : 0.8). C'est-à-dire, la comparaison entre les nœuds tient compte de l'information syntaxique et sémantique au lieu de faire une comparaison classique entre les chaînes de caractères.
- Ces quatre paramètres ont été définis pour le schéma XML qui est basé sur une classification taxonomique. Pour cette raison, l'utilisation de ces paramètres dans un graphe étiqueté nécessite une autre relaxation pour le calcul de lcs. Par exemple, si on considère les deux triplets RDF *(Object, ContentOn, URL)* et *(Object, HasReference, URL)*, on

peut remarquer que le nom de la propriété dans le premier triplet (*Content*) correspond à *Object*. Par contre, le nom de la propriété dans le deuxième triplet (*Reference*) correspond à *URL*. A ce stade, nous apportons dans notre approche une nouvelle stratégie de relaxation en considérant les nœuds de type propriété comme des nœuds qui peuvent être permutés avec leur fils ou parent immédiat. Pour ce faire, chaque chemin est divisé en un ensemble de segments, chaque segment étant composé de deux nœuds adjacents et l'un des deux étant de type propriété (Figure 6.9).

Si on considère les deux chemins montrés dans la Figure 6.9 comme deux contextes, le chemin à droite (*Package*, *hasVideo*, *object*, *description*, *metadata*) devient (*Package*, *object*, *hasVideo*, *metadata*, *description*). A ce stade, et la valeur de *lcs* entre les deux chemins sera de 5 au lieu de 0 après les deux types de relaxation introduits précédemment.

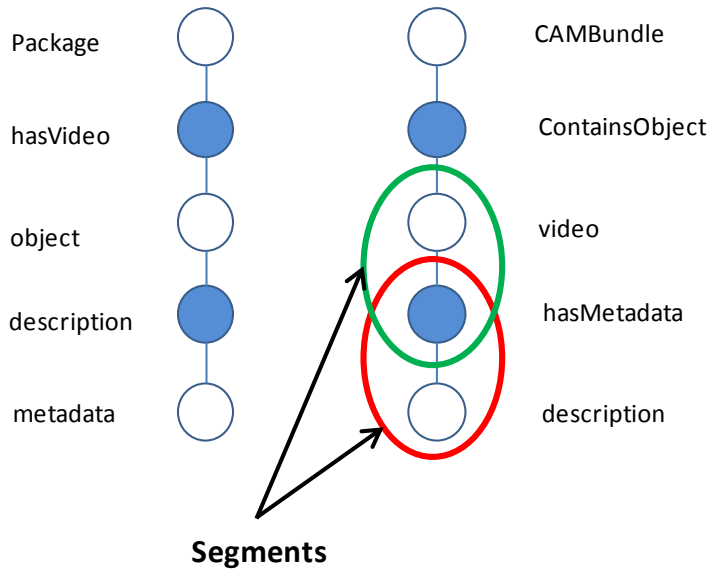

FIGURE 6.9: Exemple de chemins et segments

6.3.4.3 Calcul de la similarité structurelle

La similarité structurelle *nodeSim* est la combinaison des similarités entre les chemins représentant les contexts des nœuds. Dans cette section, nous discutons en détail comment on calcule la similarité entre les *ancetres*, *les fils immédiats* el *les feuilles* appartenant aux deux schémas à aligner.

a) Similarité des ancêtres : le contexte des ancêtres d'un nœud est donné par le chemin reliant la racine au nœud en question. La similarité entre deux contextes des ancêtres est calculée comme la similarité entre les chemins, pondérée par la similarité linguistique des nœuds correspondant (n_i, n_j) :

$$ancSim(n_i, n_j) = ps(n_i, n_j) * lSim(n_i, n_j) \tag{6.5}$$

b) Similarité des fils immédiats : afin d'obtenir la similarité des fils immédiats *immSim* entre deux nœuds (n_i, n_j), on compare les deux sous-ensembles des fils immédiats $S = \{s_1, s_2, \cdots, s_n\}$ et $S^{'} = \{s_1^{'}, s_2^{'}, \cdots, s_m^{'}\}$. Les fils immédiats désignent uniquement les nœuds simples. Les nœuds propriétés (reliant deux nœuds simples) sont inclus dans le calcul de la similarité entre les chemins par l'utilisation de la similarité *SimI* entre chaque paire de nœuds appartenant aux deux ensembles, où :

$$SimI(s_i^{'}, s_j) = ps(s_i^{'}, s_j) * lSim(s_i^{'}, s_j) \tag{6.6}$$

$ps(s_i^{'}, s_j)$ est la similarité entre les deux chemins allant de $(s_i^{'}, s_j)$ à (n_i, n_j). Puis les paires ayant une valeur maximale de similarité sont sélectionnées.

$$MaxSimI(s_i, S^{'}) = max_{s_j^{'} \in S^{'}} SimI(s_i, s_j^{'}) \tag{6.7}$$

$$MaxSimI(s_i^{'}, S) = max_{s_j \in S} SimI(s_i^{'}, s_j) \tag{6.8}$$

Finalement, la moyenne des meilleures similarités est considérée afin de calculer la valeur de similarité correspondant aux fils *immSim* :

$$immSim(n_i, n_j) = \frac{\sum_{i=1}^{|S|} MaxSimI(s_i, S') + \sum_{i=1}^{|S'|} MaxSimI(s_i', S)}{|S_\prime| + |S|} \tag{6.9}$$

c) Similarité des feuilles : si on considère que $l_i \in leaves(n_i)$ est un nœud de type feuille, alors le contexte de l_i est le chemin p_i allant de n_i à l_i. Le contexte des feuilles est donné par :

$$leafSim(l_i, l_j) = ps(p_i, p_j) * lSim(l_i, l_j) \tag{6.10}$$

Afin de mesurer la similarité entre deux feuilles $l_i \in leaves(n_i)$ et $l_j \in leaves(n_j)$, on calcule la similarité des feuilles *leafSim* entre chaque paire des feuilles dans les deux ensembles des feuilles. Puis on sélectionne la paire ayant la valeur maximale de similarité. La moyenne des meilleures similarités est prise.

d) Similarité des nœuds : la similarité des nœuds est obtenue par la combinaison des trois similarités décrites précédemment : similarité des ancêtres, similarité des fils immédiats et similarité des feuilles (Algorithme 1).

$$nodeSim(n_i, n_j) = \alpha * ancSim(n_i, n_j) + \beta * immSim(n_i, n_j) + \gamma * leafSim(n_i, n_j) \tag{6.11}$$

où $\alpha + \beta + \gamma = 1$ et $(\alpha, \beta, \gamma) \geq 0$

Une fois que la similarité structurelle calculée, le système retourne pour chaque nœud normal n_i les K nœuds correspondant aux K-plus grandes valeurs de *nodeSim*. Ces nœuds doivent avoir une valeur *nodeSim* supérieure à un seuil donné. Dans la mesure où la valeur de K est supérieure à 1, c'est l'utilisateur qui sélectionne parmi les propositions. Le dictionnaire externe est mis à jour selon les retours de l'utilisateur.

Algorithm 1 détermination des mappings

S_1 et S_2 les deux schémas à aligner
N_1 et N_2 le nombre de nœuds dans S_1 et S_2
pour tous les paires de nœuds ((n_i, n_j), $n_i \in S_1$, $n_j \in S_2$) **faire**
  si *lSim* $((n_i, n_j) > \tau_1$ **alors**
    calculer les similarité structurelle *nodeSim* entre (n_i, n_j)
    $nodeSim(n_i, n_j) = \alpha * ancSim(n_i, n_j) + \beta * immSim(n_i, n_j) + \gamma * leafSim(n_i, n_j)$

    pour toutes les paires de nœuds (n_i, n_j) **faire**
      si $nodeSim(n_i, n_j)$ est supérieure à un seuil τ_2 **alors**
        (n_i, n_j) sont considérés comme des candidats au mapping
      finsi
    fin pour
  finsi
fin pour

6.4 Représentation des mappings détectés

L'objectif principal de ce chapitre est de proposer une approche pour trouver les mappings de manière automatique entre les métadonnées. En outre, comme nous avons mentionné dans la Section 3.6, les mappings doivent être utilisées par des programmes pour migrer les instances depuis les schémas hétérogènes vers le schéma médiateur. Une représentation standardisée des mappings est essentielle pour pouvoir les manipuler aisément, surtout dans un domaine dynamique comme le multimédia.

Plusieurs travaux existent dans l'état de l'art dont le but est de trouver une représentation scalable du mapping. Les caractéristiques de ces représentations dépendent de celles du mapping, à savoir, les propriétés, les types de relation, etc. Les auteurs dans [LLZL08] par exemple, utilisent OWL comme langage de description pour spécifier le mapping. Ils considèrent que la spécification doit contenir quatre types de mapping : le mapping entre les instances, le mapping entre les classes, le mapping entre les attributs et le mapping entre les relations. L'auteur dans [Euz01a] [Euz04] définit plusieurs niveaux de représentation du mapping :

- L'identifiant unique de la relation entre les entités appartenant aux deux schémas.
- La score du mapping (entre 0 et 1).
- Le type de relation entre les éléments des deux schémas (équivalence, plus générale, moins générale, etc).
- La référence des schémas alignés.
- La cardinalité du mapping entre les entités.
- Les entités concernées par le matching.
- Un ensemble de propriétés optionnelles pour décrire le mapping (ex : commentaires, date de création, etc).

Dans notre approche, nous proposons une représentation simple qui assume que tous les mappings retournés par notre méthode sont de type *equivalentTo*. La distinction entre les éléments des schémas est ignorée car tous les éléments sont transformés en nœuds de graphes. Quant à la cardinalité, elle peut être de type simple (1:1) ou de type complexe (n:m). Afin de faciliter la représentation des mappings complexes, nous les décomposons en mappings de type (1:m) et (n:1) [MMSV02] [BV05]. Les niveaux de représentation que nous prenons en compte dans notre approche sont :

- l'ensemble des entités concernées par le mapping ;
- l'identifiant unique des relations ;
- concernant la cardinalité du mapping, deux catégories sont définies : le mapping simple (1:1) représenté par le concept *SimpleMapping* et le mapping complexe (n:m) et (m:n) représenté par le concept *ComplexMapping*.

Nous utilisons le Schéma XML comme langage de description pour la spécification du mapping. Le Schéma XML couvre tous les besoins mentionnés précédemment.

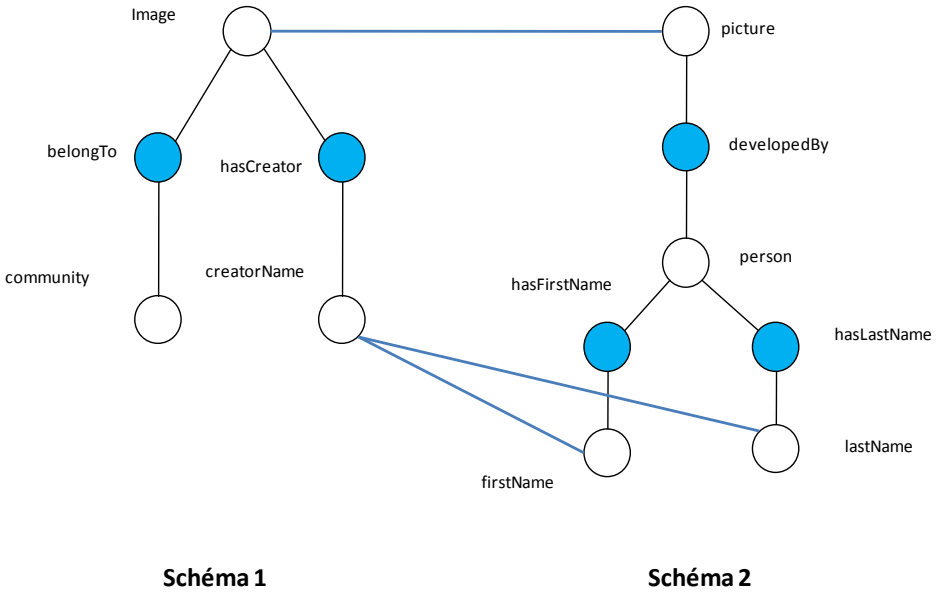

FIGURE 6.10: Exemple de deux schémas alignés

```
01:<mappingResult>
02:  ….
03: <SourceSchema source="Schéma1"/>
04: <TargetSchema source="Schéma2"/>
05:
06:  <simpleMapping ID="#123YFD">
07:  <Sourcenode name="image"/>
08:  <TargetNode name="picture"/>
09:  <simpleMapping>
10:   ….
11:  <complexMapping ID="#123E55">
12:   <Sourcenode name="image/hasCreator/creatorName"/>
13:   <TargetNode name="picture/developedBy/person/hasLastName/lastName"/>
14:   <TargetNode name="picture/developedBy/person/hasFirstName/firstName"/>
15:   </complexMapping>
16:  …..
17:</mappingResult>
```

FIGURE 6.11: Exemple d'une représentation Schéma XML du mapping

La Figure 6.11 montre une représentation en format XML des mappings détectés entre les deux schémas présentés dans la Figure 6.10. Elle représente le mapping simple trouvé entre

les éléments *image* et *picture* ainsi qu'un mapping complexe entre l'entité *creatorName* du schéma S_2 et les deux entités *lastName* et *firstName* du schéma S_2.

6.5 Conclusion

Nous avons présenté dans ce chapitre une méthodologie de matching déployée sur deux niveaux (langage de description et schéma). Cette méthodologie prend en considération les schémas de métadonnées issus de différents langages de description (Schéma XML, Schéma RDF, OWL), les transforme en graphes étiquetés tout en conservant les informations structurelles et sémantiques. Cette représentation garantit l'extensibilité de l'approche proposée en lui permettant de supporter d'autres langages de description. La méthode commence par le calcul de la similarité linguistique entre les nœuds des deux graphes en utilisant les noms et les commentaires correspondant à ces derniers. Dans notre approche, les informations structurelles et sémantiques conservées durant la phase de construction des graphes sont utilisées pour détecter les mappings complexes.

Afin de faire face aux limitations des méthodes utilisant un seul contexte d'information, notre approche utilise plusieurs types d'information contextuelle (contexte des ancêtres, des feuilles et des fils immédiats) pour obtenir un meilleur résultat de matching. Nous avons également introduit une méthodologie d'adaptation et de relaxation dans le calcul de la similarité entre les chemins, ce qui donne à notre méthode plus de flexibilité. Enfin, pour pouvoir aisément manipuler les mappings, nous avons fourni une spécification de ces dernier où le Schéma XML a été utilisé comme langage de représentation du mapping.

Dans le chapitre suivant, nous détaillons les expérimentions effectuées sur plusieurs standards de métadonnées. Les résultats obtenus valident et montrent les performances de l'approche proposée par rapport au méthodes connues dans l'état de l'art.

Chapitre 7

Expérimentation et évaluation

Sommaire

7.1 Introduction

En plus des démonstrations logicielles réalisées dans le cadre du project CAM4Home (annexe B), et qui montré les apports du métamodèle proposé. Nous avons implémenté et testé notre approche de matching proposée sur plusieurs types de métadonnées hétérogènes. L'objectif principal de ce chapitre est de montrer les expérimentations effectuées et les résultats obtenus grace auxquels nous avons validé notre approche.

Ce chapitre est organisé comme suit. Dans la Section 7.2, nous commençons par décrire les caractéristiques des métadonnées utilisées dans nos expérimentations tout en montrant les vérités terrain avec lesquelles nous avons comparé nos résultats. Nous introduisons ensuite les techniques utilisées pour évaluer les performances de notre approche. La Section 7.4 présente les résultats expérimentaux que nous avons obtenus. Une étude comparative est faite avec d'autres méthodes existant dans l'état de l'art pour montrer les apports de notre proposition. Enfin, nous concluons ce chapitre par une discussion des résultats obtenus.

7.2 Caractéristiques des métadonnées utilisées

Nous présentons dans cette section les caractéristiques des métadonnées utilisées dans nos expérimentations. Ces métadonnées sont issues du domaine de multimédia et présentent une hétérogèité sur tous les niveaux. Ensuite, nous discutons la vérité terrain avec laquelle nous avons comparé nos résultats expérimentaux.

7.2.1 Les formats retenus

L'approche MuMIe a été testée sur plusieurs extraits de formats de métadonnées issus du domaine du multimédia. Nous avons utilisé des standards dédiés à la description sémantique

des contenus tel que MPEG-7 et DIG35 ainsi que d'autres formats utilisés pour la description des contextes de diffusion tels que MPEG-21 et CAM4Home. Les formats utilisés montrés dans le Tableau 7.1 présentent une hétérogénéité significative aux deux niveaux. Au niveau des langages de description, on peut remarquer que certains formats tel que MPEG-7 et MPEG-21 sont spécifiés en utilisant le Schéma XML. D'autres formats sont encodés avec OWL et RDFS (ex : DIG35 Ontology et CAM4Home). En plus de l'hétérogénéité linguistique au niveau des schémas, les corpus utilisés présentent une hétérogénéité structurelle significative où l'on peut remarquer que les métadonnées utilisées dans nos expérimentations sont différentes en termes de profondeur et de nombre de nœuds.

TABLE 7.1: Caractéristiques des métadonnées utilisées

Standard de métadonnées	**Profondeur**	**Nombre de nœuds**	**Langage**
MIX [MIX02]	7	41	XSD
DIG35 [wg02a]	9	57	XSD
EXIF [wg02b]	6	63	XSD
MPEG-7 [CPSZ01]	13	115	XSD
IPTC [wg04]	9	65	XSD
MPEG-21 [BdWH$^+$03]	11	143	XSD
PeCMan Ontology [ml08]	7	46	OWL
DIG35 Ontology [ml08]	9	57	OWL
CAM4Home [BAB$^+$10]	17	153	RDFS
TV-Anytime [PS00]	8	71	XSD

7.2.2 La vérité terrain

Afin de valider nos résultats expérimentaux, nous avons considéré plusieurs vérités terrain. En plus de la documentation disponible pour chaque standard, nous nous sommes basés également sur les travaux du groupe *Ontology for Media Resources* décrit dans la Section 3.5.3.

Ce groupe définit un ensemble de propriétés considérées comme le minimum d'informations nécessaires à l'utilisateur pour manipuler les contenus multimédia. Un mapping manuel est ensuite effectué entre cet ensemble de propriétés et les propriétés équivalentes, encodées en différents formats. Dans nos évaluations, nous considérons que les attributs appartenant aux standards différents sont équivalents s'ils correspondent à la même propriété définie par *Ontology for Media Resources*. Huit catégories de métadonnées ont été définies par ce groupe :

- *Identification* : c'est la catégorie qui regroupe l'ensemble des propriétés décrivant l'identité de la ressource (ex : *identifier*, *title*, *language*, etc).
- *Création* : elle contient les informations relatives à la création des objets multimédia (ex : *creator*, *date*, *location*, etc).
- *Description des contenus* : c'est l'ensemble des propriétés dédiées à la description sémantique des contenus multimédia (ex : *keyword*, *description*, etc).
- *Relationnel* : elle décrit les relations entre les objets multimédia.
- *Droits :* elle contient les informations relatives aux droits d'accès.
- *Distribution* : c'est la catégorie des informations relatives à la publication des contenus multimédia (ex : *publisher*).
- *Fragment* : elle regroupe l'ensemble des propriété dont le rôle est la description d'une portion donnée d'un contenu multimédia (ex : segment audio ou vidéo).
- *Propriétés techniques* : contient un ensemble de propriétés techniques (ex : *frameSize*, *samplingRate*, *format*, etc).

Les Tableaux 7.2 et 7.3 montrent une partie du mapping entre les entités *Creator* et *Description* définis par *Ontology for Media Resources* et les entités correspondantes au niveau des standards DIG35, IPTC, MPEG-7 et TV-Anytime. La liste complète du mappings entre le reste des entités et les différents standards peut être trouvée dans [LBSM08].

TABLE 7.2: Le mapping avec l'entité Creator

Le mapping entre l'entité **Creator** et	**Entité**
DIG35	Metadata/Image_Creation/General_Creation_Info/**IMAGE_CREATOR** Metadata/Ipr/Ipr_Names/Ipr_Person/Description/**ImageCreator** Metadata/Ipr/Ipr_Names/Ipr_Person/Description/**OriginalWorkAuthor**
IPTC	newsItem/contentMeta/creator/**name** newsItem/partMeta/creator/**name**
MPEG-7	CreationInformation/Creation/**Creator**
TV-Anytime	BasicDescription/CreditsList/CreditsItem/PersonName/**GivenName** BasicDescription/CreditsList/CreditsItem/PersonName/**FamilyName** BasicDescription/CreditsList/CreditsItem/**OrganisationName**

TABLE 7.3: Le mapping entre l'entité Description

Le mapping entre l'entité **Description** et	**Entité**
DIG35	Metadata/Content_Description/Caption Metadata/ipr/Ipr_Description/Ipr_Caption
IPTC	newsItem/contentMeta/**description** newsItem/contentMeta/headline newsItem/contentMeta/**slugline** newsItem/contentMeta/Dateline newsItem/partMeta/**description** newsItem/partMeta/headline newsItem/partMeta/**slugline** newsItem/partMeta/**Dateline**
MPEG-7	CreationInformation/Creation/**Abstract**
TV-Anytime	BasicDescription/**Synopsis** BasicDescription/**PromotionalInformation** BasicDescription/**Keyword** BasicDescription/ParentalGuidance/ParentalRating/**Name** BasicDescription/AwardList/AwardListItem/**Title**

Dans la prochaine section, nous décrivons les techniques utilisées pour l'évaluation de notre approche.

7.3 Technique d'évaluation

Nous avons utilisé les critères d'évaluation *Précision*, *Rappel* et *F-mesure* introduites dans [MBR01] pour l'évaluation des performances de notre approche. Ces mesures permettent de mesurer le comportement de l'approche dans la détection des mappings par rapports à ceux faits par des experts humains (Figure 7.1).

Précision : la précision est la proportion de mappings corrects parmi l'ensemble de ceux retournés par le système de matching. Cette mesure donne une indication sur la précision du système, elle est inversement proportionnelle au bruit affectant le processus de matching.

$$Précision = \frac{|mappings \text{ détectés} \bigcap mappings \text{ réels}|}{|mappings \text{ détectés}|}$$

Rappel : le rappel est la proportion du mappings corrects générés par le système parmi tous ceux qui sont corrects (en incluant aussi des mappings corrects qui n'ont pas été détectés par le système). Il donne une indication sur l'efficacité d'un algorithme en montrant le pourcentage de mappings ratés.

$$Rappel = \frac{|mappings \text{ détectés} \bigcap mappings \text{ réels}|}{|mappings \text{ réels}|}$$

F-mesure : Afin d'évaluer la qualité de matching, les deux techniques d'évaluation décrites précédemment doivent être prises en considération. A ce stade, plusieurs mesures combinant ces deux techniques on été proposées, en particulier *F-mesure*. F-mesure est une technique d'évaluation utilisée dans le domaine de la recherche d'information [vR79]. Elle combine la

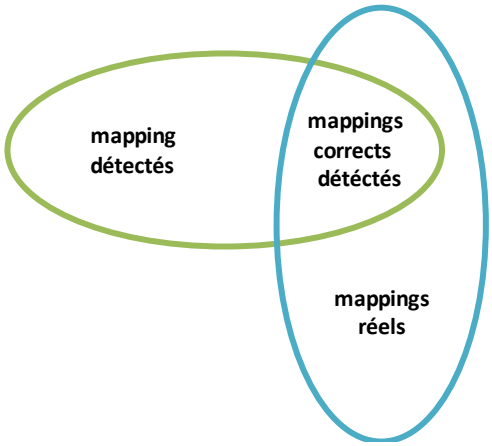

FIGURE 7.1: L'ensemble des mappings

précision et le rappel en leur donnant la même importance. F-mesure permet de comparer les performances des algorithmes par une seule mesure, elle est définie par :

$$F - mesure = \frac{2 * Rappel * Précision}{Rappel + Précision}$$

7.4 Expérimentation

Dans cette section, nous montrons les expérimentations que nous avons effectuées pour valider notre approche. Nous commençons par étudier l'influence des paramètres de similarité linguistique et structurelle. Pour ce faire, nous testons un ensemble de combinaisons de paramètres et nous comparons les résultats des divers combinaisons avec la vérité terrain que nous possédons. Cela nous permet de fixer les valeurs de ces paramètres pour le processus de matching.

Nous détaillons également dans cette section les résultats du matching que nous avons effectué sur un certain nombre de standards de métadonnées. Ces résultats sont discutés et comparés avec d'autres méthodes connues dans l'état de l'art (Cupid et SF).

7.4.1 Paramétrage

Le système de matching proposé est basé sur une combinaison de plusieurs types de similarité. Une similarité linguistique combinant la similarité des noms d'entités et des commentaires, ainsi qu'une similarité structurelle basée sur trois contextes. Afin de connaître l'influence de chaque similarité sur le résultat du matching, nous avons effectué plusieurs expérimentations qui nous ont permis d'attribuer à chaque paramètre le poids correspondant.

7.4.1.1 Paramètres de la similarité linguistique

Afin de choisir le valeur des paramètres μ_1 and μ_2 et connaître l'effet relatif à la combinoson des deux similarités linguistiques S_{nom} et $S_{commentaire}$, nous avons sélectionné deux ensembles d'attributs (T_1, T_2) depuis plusieurs standards de métadonnées. Chaque élément appartenant à T_1 possède un équivalent dans T_2. Nous avons calculé la similarité linguistique ($lSim$) entre les éléments appartenant aux deux ensembles via l'utilisation de plusieurs combinaisons des paramètres μ_1 et μ_2. Ces éléments sont considérés comme linguistiquement identiques si leur similarité ($lSim$) est supérieure à 0.9. La Figure 7.2 montre les résultats de cette expérimentation en terme de précision (pourcentage de mappings corrects linguistiquement), où l'on peut remarquer que les noms d'éléments ont plus d'importance par rapport aux commentaires. Cependant, les commentaires portent toujours une information pertinente qui peut être utile pour la détection de mappings.

7.4.1.2 Paramètres de la similarité structurelle

La définition des paramètres α (le poids des ancêtres), β (le poids des fils) et γ (le poids des feuilles), a été faite par plusieurs expérimentations en utilisant des combinaisons différentes des ces paramètres. Les Figures 7.3 et 7.4 montrent les résultats de matching en termes

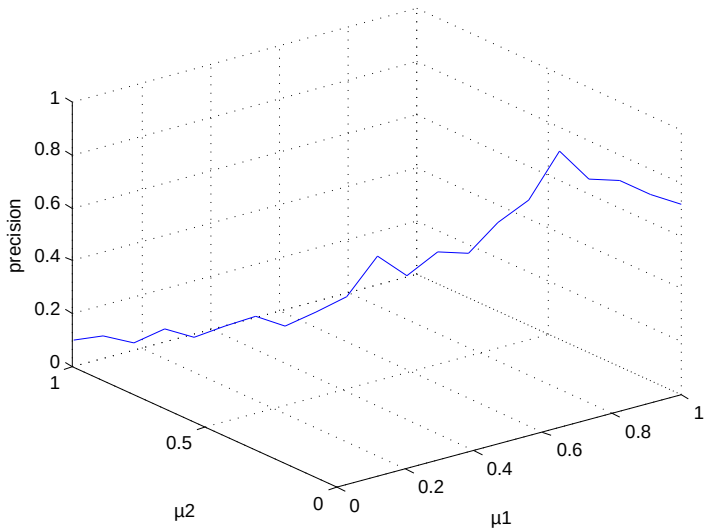

FIGURE 7.2: Influence de μ_1 et μ_2 sur la similarité linguistique

de *F-Mesure* [MBR01] entre les standards (MPEG-7, DIG35) et (DIG35, EXIF) respectivement. Nos expérimentations ont montré qu'une partie importante de l'information structurelle est contenue dans le contexte des nœuds parents où les valeurs implorantes de F-mesure sont localisées pour $\alpha \in [0.45\ 0.6]$; cela explique l'intérêt de quelques stratégies de matching telles que [BSZ03], [MHH$^+$01] et [SM01] qui considèrent que le contexte d'un nœud dépend uniquement de ses parents. En outre, les expérimentations ont montré que les contextes de fils immédiats et de feuilles contiennent de l'information pertinente pour le processus de matching ($\beta \in [0.1\ 0.15]$ et $\gamma \in [0.25\ 0.4]$).

7.4.2 Qualité de matching

L'approche MuMIe a été évaluée en considérant les trois critères d'évaluation mentionnés précédemment (*Précision*, *Rappel* et *F-mesure*). Les paramètres linguistiques et structurels ont

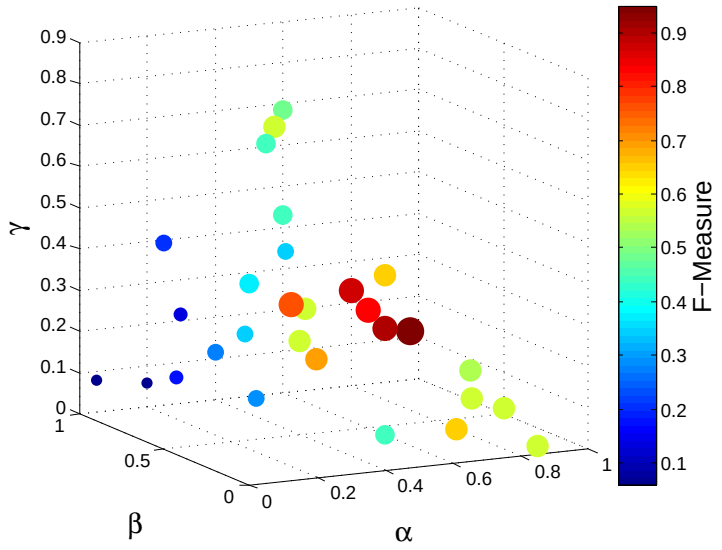

FIGURE 7.3: Influence de α, β et γ sur le matching entre MPEG-7 et DIG35

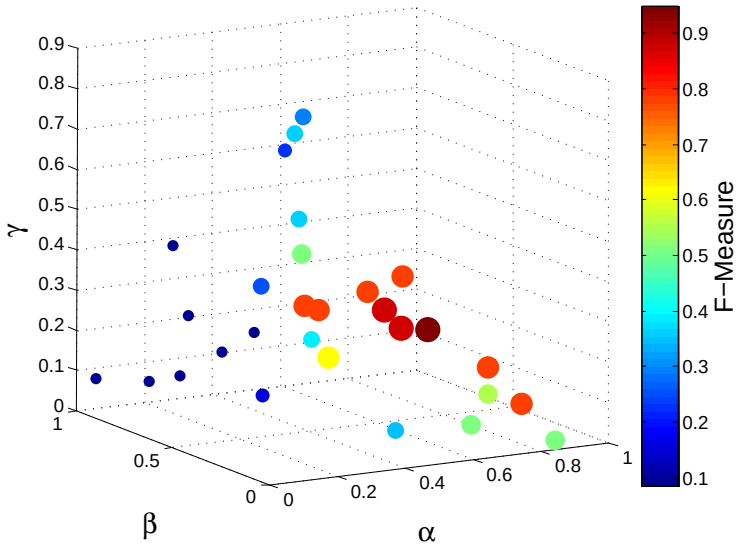

FIGURE 7.4: Influence de α, β et γ sur le matching entre DIG35 et EXIF

été choisis suite aux expériences effectuées dans la Section 7.4.1 ($\mu_1 = 0.85, \mu_2 = 0.15, \alpha = 0.55, \beta = 0.15, \gamma = 0.30$).

Afin de pouvoir comparer nos résultats avec Cupid décrit dans la Section 4.2.2.1, nous avons sélectionné dans un premier temps uniquement des standards dont les schémas ont été définis avec le Schéma XML. Pour ce faire, nous avons calculé le matching entre TV-Anytime et tous les standards dont les schémas ont été définis avec le Schéma XML. Ensuite, afin que nos résultats soient comparables avec SF décrit dans la Section 4.2.2.2, nous avons pris en compte tous les schémas du Tableau 7.1. Nous avons calculé le matching entre le métamodèle CAM4Home et le reste des schémas. Les résultats du matching entre TV-Anytime et les standards définis avec le Schéma XML sont présentés dans le Tableau 7.4. Le Tableau 7.5 montre les résultats du matching entre CAM4Home et le reste des standards.

En analysant les résultats des Tableaux 7.4 et 7.5, on peut remarquer que la valeur moyenne de F-measure est 62 % et 81 % pour les deux tableaux, un résultat qui nous semble intéressant. On peut remarquer également que les résultats du Tableaux 7.5 sont supérieurs à ceux du Tableau 7.4. Cela est dû à la différence entre les schémas médiateurs choisis pour chaque expérimentation. Le Tableau 7.6 montre également un extrait de mappings détectés entre CAM4Home, MPEG-7 et DIG-35.

TABLE 7.4: Résultats du matching avec TV-Anytime

Matching entre TV-Anytime et	**Précision**	**Rappel**	**F-mesure**
MIX	50%	55%	52%
EXIF	62%	65%	63%
MPEG-7	75%	70%	72%
IPTC	43%	47%	45%
DIG35	77%	82%	79%

TABLE 7.5: Résultats du matching entre CAM4Home et le reste des standards

Matching entre CAM4Home et	**Précision**	**Rappel**	**F-mesure**
MIX	92%	89%	90%
DIG35	87%	90%	88%
EXIF	90%	81%	85%
MPEG-7	77%	64%	70%
MPEG-21	72%	60%	65%
PeCMan Ontology	94%	89%	91%
DIG35 Ontology	85%	90%	87%
TV-Anytime	77%	72%	75%

TABLE 7.6: Extrait des résultats de mapping entre CAM4Home, MPEG-7 et DIG35

CAM4Home	**MPEG-7**	**DIG35**
duration	mediaDuration	N/A
creatorReference	creator	image_creator
gpsLocation	location	location
creationDateTime	date	creationTime captureTime
camEntityVersion	version	version
description	abstract	caption
legalNotice	copyrightString	copyright
title	title	ipr_title
entityUID	publicIdentifier entityIdentifier	image_ID

7.4.3 Etude comparative

Nous avons comparé les performances de notre solution avec celles de Cupid et SF. La comparaison a été faite avec ces deux approches car elles sont basées sur le matching des schémas, elles sont multi-niveaux (SQL et Schéma XML pour Cupid, SQL, XML et RDF pour

SF) et elles utilisent une similarité linguistique et structurelle dans leur processus de matching. Les résultats de cette étude comparative en termes de F-Mesure sont montrés respectivement dans les Figures 7.5 et 7.6.

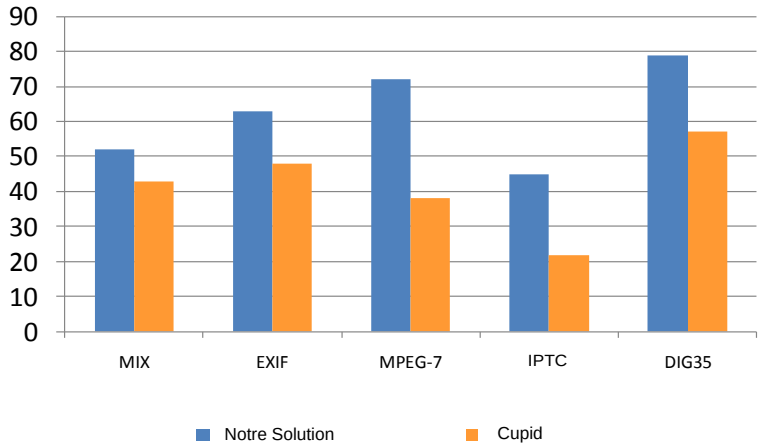

FIGURE 7.5: Etude comparative en terme de F-Mesure avec Cupid (TV-Anytime comme schéma médiateur).

Les résultats expérimentaux ont montré que, sur toutes les standards de métadonnées utilisés, notre solution était meilleure en termes F-Measure. Cela est principalement dû à :

- l'utilisation de l'information linguistique qui est une combinaison des scores obtenus à partir des des commentaires et des noms des nœuds.
- l'exploitation des informations hiérarchiques et sémantiques (Section 6.3.3) permettant de détecter d'autres mappings même si les nœuds correspondants ne sont pas linguistiquement identiques. Ces informations servent également dans la détection des matching complexes.
- l' utilisation de trois contextes pour le calcul de la similarité structurelle, ce qui donne une flexibilité importante avec une meilleure exploitation de l'information structurelle. Alors que Cupid et SF utilisent uniquement un seul contexte de similarité.

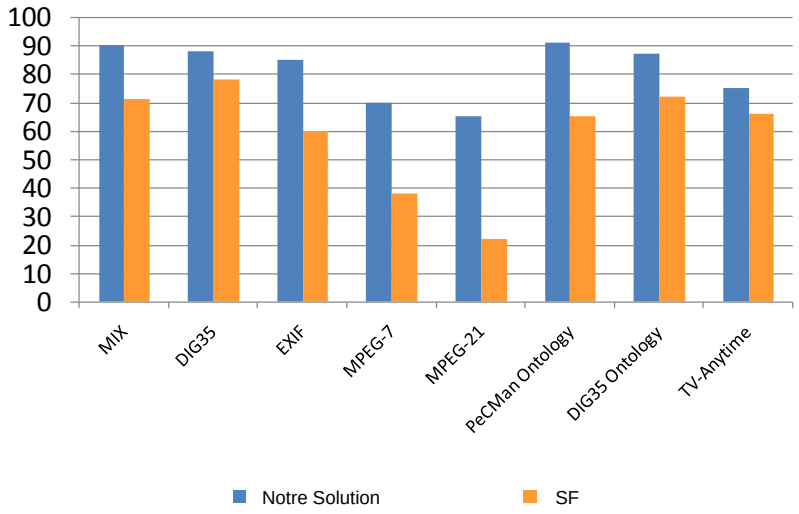

FIGURE 7.6: Etude comparative en terme de F-Mesure avec SF (CAM4Home comme schéma médiateur).

7.5 Conclusion

Nous avons présenté dans ce chapitre les résultats expérimentaux que nous avons réalisés dans le but de valider notre proposition de matching. Nous avons utilisé plusieurs standards de métadonnées hétérogènes sur les deux niveaux. Les résultats obtenus ont montré que la combinaison des informations linguistiques et structurelles augmente de manière significative la détection des mappings corrects entre les métadonnées.

Nous avons remarqué que l'intervention de l'utilisateur est souvent requise si l'on veut atteindre un taux de précision de 100%. Cependant, si l'on tient compte de la définition de l'interopérabilité donné dans [Org04], où les auteurs la définissent comme la capacité d'échanger des métadonnées entre deux ou plusieurs systèmes sans ou avec minimum de perte d'informations et sans aucun effort particulier sur les deux systèmes, on peut dire que notre système peut être considéré comme une solution automatique pour la réalisation.

Chapitre 8

Conclusions et perspectives

Sommaire

8.1 Résultats principaux

En raison de la forte utilisation de métadonnées, il est important de développer un système d'intégration pour assurer leur interopérabilité. L'existence d'un tel système est devenue cruciale afin d'uniformiser l'accès aux métadonnées. Pour cela, nous avons proposé dans ce manuscrit de thèse deux stratégies complémentaires d'intégration des métadonnées.

La première porte sur un métamodèle générique nommé CAM4Home dont le but est d'assurer la diffusion des contenus multimédia sur des plate-formes hétérogènes. Ce métamodèle assure les fonctions suivantes :

- *fournir une vue unifiée des métadonnées multimédia* : nous avons défini tous les types des métadonnées nécessaires pour assurer la diffusion des contenus multimédia sur des plate-formes hétérogènes. Ces métadonnées accompagnent les contenus multimédia durant toutes les étapes de leur cycle de vie, de leur création jusqu'à leur consommation. Trois catégories de métadonnées ont été ainsi définies : les métadonnées basiques dont le rôle est d'assurer une description sémantique des contenus ; les métadonnées contextuelles qui définissent les caractéristiques des contextes de diffusion (logiciels et matériels) et les métadonnées externes issues d'autres standards.
- *garantir l'extensibilité du métamodèle CAM4Home* : grâce à sa structure qui contient deux niveaux d'abstraction et plusieurs niveaux hiérarchiques, CAM4Home peut facilement intégrer d'autres informations répondant à des besoins spécifiques non encore identifiés.
- *l'encapsulation des formats de métadonnées externes* : le métamodèle CAM4Home permet aux utilisateurs d'intégrer dans leur descripteurs d'autres standards de métadonnées déjà existants tel que MPEG-7, DIG-35 et TV-Anytime.
- *flexibilité d'encodage* : contrairement à la plupart des standards de métadonnées qui sont

spécifiés en utilisant le Schéma XML. Nous avons utilisé le Schéma RDF comme langage de description dans notre réalisation technique grâce à sa flexibilité et son pouvoir expressif.

Notre deuxième contribution est la réalisation d'une nouvelle méthode d'intégration automatique des métadonnées. Cette méthode est basée sur une nouvelle stratégie de mise en correspondance de schémas qui prend en considération l'hétérogénéité des métadonnées sur deux niveaux : schémas et langages de description. Pour ce faire, les schémas sont projetés en un seul espace de représentation puis des mesures de similarité linguistiques et structurelles assez poussées sont utilisées pour trouver les correspondances entre les métadonnées hétérogènes. Les principales innovations que nous avons faites dans le développement de cette solution peuvent être résumées comme suit :

- *une méthode de matching multi-niveaux* : notre solution peut supporter des schémas hétérogènes sur les deux niveaux : schéma et langages de description. Pour ce faire, les schémas hétérogènes sont transformés en graphes directs étiquetés. Ces graphes sont des espaces de représentation sur lesquels toutes les ontologies sont projetables. Durant l'opération de transformation, nous capturons uniquement les concepts basiques communs entre les différents langages de description tout en conservant les informations structurelles et sémantiques.
- *l'exploitation des informations linguistiques issues des commentaires* : en plus des noms d'éléments utilisés dans la plupart des stratégies de matching, nous avons utilisé la documentation disponible sous forme de commentaires comme une deuxième source d'informations par le biais de l'utilisation de la technique TF-IDF.
- *l'utilisation des informations sémantiques et hiérarchiques* : grâce à ces informations notre approche offre une meilleure détection des mappings, notamment des mappings complexes. Pour chaque type d'information, une règle de déduction a été définie.

– *l'utilisation de plusieurs contextes de similarité* : contrairement à la plupart des méthodes de matching, notre méthode considère que le contexte d'un nœud dépend de ses ancêtres, de ses fils immédiats et de ses feuilles. Les expérimentations effectuées ont montré les améliorations apportées par ces trois contextes.
– *relaxation des contraintes* : plusieurs stratégies de relaxation ont été introduites dans notre approche. Pour ce faire, nous n'avons pas pris en compte certains types d'information dans le processus de mise en correspondance. Nous avons utilisé une méthode plus flexible dans le calcul de similarité entre les chemins. Plusieurs types d'adaptation et de relaxation ont été introduits sur cette méthode pour qu'elle soit applicable aux langages flexibles tel que RDFS et OWL.

L'approche proposée a été testée sur plusieurs standards issus du monde multimédia tels que MPEG-7, MPEG-21, DIG35, EXIF, CAM4Home, MIX. Ces standards présentent une hétérogénéité significative sur les deux niveaux. Les résultats expérimentaux ont montré les bonnes performances de l'approche proposée. L'étude comparative de notre approche avec d'autres méthodes de matching connues dans l'état de l'art a confirmé la pertinence de la solution proposée, notre solution a obtenu de meilleures performances en terme de *F-mesure*.

8.2 Perspectives

Bien que nous ayons effectué dans cette thèse des recherches approfondies pour proposer une solution d'intégration automatique des métadonnées dédiées au multimédia, d'autres investigations peuvent être menées pour complementer approche. Nous avons analysé certaines des limites de notre travail et nous proposons dans ce qui suit les orientations futures :

– Nous avons présenté une stratégie d'utilisation des informations structurelles pour le filtrage sémantique des mappings. Cette stratégie est basée sur trois contextes de simila-

rité : similarité des ancêtres, similarité des feuilles et similarité des fils immédiats. Ces trois contextes représentent l'information structurelle verticale. Dans nos futurs travaux de recherche, nous prévoyons d'améliorer notre méthode de mise en correspondance par une meilleure exploitation de l'information structurelle par l'utilisation des relations d'adjacence horizontale entre les nœuds.

– Notre étude présentée dans la Section 7.4.1 porte sur l'influence des paramètres α, β et γ sur l'effet du matching. Suite aux résultats de cet apprentissage, nous avons attribué une seule valeur pour chaque paramètre. Cette valeur sera identique pour tous les nœuds du graphe. Cependant, la sélection des paramètres α, β et γ pour chaque paire de nœuds, en fonction de leur position dans le graphe, pourrait sans doute augmenter la qualité du matching.
– La solution d'intégration automatique des métadonnées proposée dans ce manuscrit de thèse est basée sur un système de médiation. Ainsi, le système d'intégration détecte les mappings entre le schéma présentant les besoins d'utilisateur et les métadonnées hétérogènes (Section 3.6). A ce stade, les instances des schémas hétérogènes doivent être migrées vers le schéma médiateur pour qu'elles puissent être interrogées par l'utilisateur. Ainsi a plus long terme nous visons de réaliser cette migration, des règles doivent être définies en fonction des mappings détectés. Ces règles doivent aussi tenir compte de l'hétérogénéité structurelle et sémantique.

Publications Personnelles

Journaux

[AMJMM11] Samir Amir, Ioan Marius Bilasco and Chabane Djeraba. MuMIe : Multi-level Metadata Mapping System. *Journal of Multimedia*, Academy publisher, Vol 6 No 3, pages 225-235, 2011

Chapitres de livres avec comité de programme et actes

[ABSD11] Samir Amir, Ioan Marius Bilasco, Haidar Mohamed Sharif et Chabane Djeraba. Towards a Unified Multimedia Metadata Management Solution. *Intelligent Multimedia Databases and Information Retrieval : Advancing Applications and Technologies*, IGI Global, pages 170-194, 2011.

[ABUMD10] Samir Amir, Ioan Marius Bilasco, Thierry Urruty, Jean Martinet et Chabane Djeraba. Designing intelligent content delivery frameworks using MPEG-21. *The Handbook of MPEG Applications : Standards in Practice*, John Wiley and Sons Ltd : Chichester, pages 455-477, 2010.

Conférences Internationales avec Comité de Lecture

[ABBD11] Samir Amir, Yassine Benabbas Ioan Marius Bilasco et Chabane Djeraba. MuMIe : A New System for Multimedia Metadata Interoperability. *ACM International Conference on Multimedia Retrieval*, 17 - 20 Avril, 2011. Trento - Italy, papier long, pages 1-8 (conférence rang A)

[BYA11] Yassine Benabbas, Samir. Amir, Adel. Lablack et Chabane Djeraba. Human Action RecognItion Using Direction and Magnitude Models of Motion. *International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, 5 - 7 Mars, 2010. Algarve-Portugal.

[ABDED10] Samir Amir, Ioan Marius Bilasco, Taner Danisman, Ismail Elsayad et Chabane Djeraba. Multimedia Metadata Mapping : Towards Helping Developers in Their Integration Task. *The 8th International Conference on Advances in Mobile Computing and Multimedia (MoMM2010)*, 08 - 10 Novembre, 2010. Paris - France, papier long, pages (205-212)

[ABD10] Ismail Elsayad, Jean Martinet, Thierry Urruty, Samir Amir et Chabane Djeraba. Toward A Higher-Level Visual Representation For Content-Based Image Retrieval. *The 8th International Conference on Advances in Mobile Computing and Multimedia (MoMM2010)*, 08 - 10 Novembre, 2010. Paris - France, papier long, pages (213-220)

[ABDUD10] Samir Amir, Ioan Marius Bilasco, Taner Danisman, Thierry Urruty et Chabane Djeraba. Semi-Automatic Multimedia Metadata Integration. *The 17th International Conference on Knowledge Engineering and Knowledge Managemen (EKAW2010)(poster)*, 11 - 15 Octobre, 2010. Lisbonne - Portugal. (conférence rang A)

[AMB10] Samir Amir, Ioan Marius Bilasco, Taner Danisman, Thierry Urruty, Ismail Elsayad et Chabane Djeraba. Schema Matching For Integrating Multimedia Metadata. *IEEE International Conference on Machine and Web Intelligence (ICMWI2010)*, 3 - 5 Octobre, 2010. Alger - Algérie, papier long , pages (234 - 239)

[EMU10] Ismail Elsayad, Jean Martinet, Thierry Urruty, Samir Amir et Chabane Djeraba. Effective Object-Based Image Retrieval Using Higher-Level Visual Representation. *IEEE International Conference on Machine and Web Intelligence (ICMWI2010)*, 3 - 5 Octobre, 2010. Alger - Algérie, papier long (pages 218 - 224)

[BAB10] Ioan Marius Bilasco, Samir Amir, Patrick Blandin, Chabane Djeraba, Juhani Laitakari, Jean Martinet, Eduardo Martínez-Gracía, Daniel Pakkala, Mika Rautiainen, Mika Ylianttila et Jiehan Zhou Semantics for intelligent delivery of multimedia content. *Symposium On Applied Computing (SAC2010)*, 22 26 mars 2010. Sierre, Suisse

[BAD09] Samir Amir, Ioan Marius Bilasco et Chabane Djeraba. A Semantic Approach to Metadata Management in Sensor Systems. *COGnitive systems with Interactive Sensors (COGIS2009)*, 16 18 Novembre 2009. Paris, France

[FAMS06] Erwan Flécher, Samir Amir, Marie Babel et Olivier Déforges. LAR VIDEO : Lossless video coding with semantic scalability. *International Conference on Signals and Electronic Systems (ICSES2006)*, 17 - 20 Septembre, 2006. Varsovie - Pologne.

[BCAT06] Assia Cherfa-Bouzouad, Yazid Cherfa, Samir Amir et Abdelhak Taméra. Evaluation sans référence de la segmentation en contours. *The 3rd International Symposium on Image/Video Communications over fixed and mobile networks (ISIVC2006)*, 13 - 15 Septembre, 2006. Hammamet - Tunisie.

Conférences Nationales avec Comité de Lecture

[ABUD11] Samir Amir, Ioan Marius Bilasco, Thierry Urruty et Chabane Djeraba. Vers une Interopérabilité Multi-Niveaux des Métadonnées, INFORSID'2011, may 24-26, 2011 - Lille, France. *Informatique des Organisations et Systèmes d'Information et de Décision (INFORSID2011)*, 24-26 mai 2011. Lille, France

[ABU10] Samir Amir, Ioan Marius Bilasco, Thierry Urruty et Chabane Djeraba. MuMIe : Une Approche Automatique pour l'Interopérabilité des Métadonnées. *11ème Conférence Internationale Francophone sur l'Extraction et la Gestion des Connaissances (EGC2011)*(papier long), 25 au 28 janvier 2011, Brest, France, pages (347-352)

[ABD09] Samir Amir. Un système d'intégration de métadonnées dédiées au multimédia. *Informatique des Organisations et Systèmes d'Information et de Décision (INFORSID2009)(Forum jeunes chercheurs) pages (491-492)*, 26 29 Mai 2009. Toulouse, France

[AFBD06] Samir Amir, Erwan Flécher, Marie Babel et Olivier Déforges. LAR VIDEO : Codage vidéo sans perte à scalabilité sémantique. *Compression et Représentation des Signaux Audiovisuels (CORESA2006)*, 9 - 10 Novembre, 2006. Caen - France.

Brevets

[JRAD07] Jean-Ronan Vigouroux, Samir Amir et Pascal Bourdon. Method for adjusting the colorimetry of four primaries projectors. *Thomson Corporate Research*, September 2007.

Bibliographie

[ABD09] S. Amir, I. M. Bilasco, and C. Djeraba. A Semantic Approach to Metadata Management in Sensor Systems. In *Cognitive systems with Interactive Sensors (COGIS), Paris (France)*, novembre 2009.

[ABD+10a] S. Amir, I. M. Bilasco, T. Danisman, I. El sayad, and C. Djeraba. Multimedia metadata mapping : Towards helping developers in their integration task. In *ACM MoMM*, pages 213–220, 2010.

[ABD+10b] Samir Amir, Ioan Marius Bilasco, Taner Danisman, Ismail Elsayad, and Chabane Djeraba. Schema matching for integrating multimedia metadata. In *IEEE ICMWI*, pages 234–239, 2010.

[ABK+07] Soren Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, and Zachary Ives. Dbpedia : A nucleus for a web of open data. In *In 6th Intl Semantic Web Conference, Busan, Korea*, pages 11–15. Springer, 2007.

[ABSD10] S. Amir, I. M. Bilasco, MD. H. Sharif, and C. Djeraba. *Towards a Unified Multimedia Metadata Management Solution*. Intelligent Multimedia Databases and Information Retrieval : Advancing Applications and Technologies, IGI Global, 2010.

[ADMR05] David Aumueller, Hong-Hai Do, Sabine Massmann, and Erhard Rahm. Schema and ontology matching with coma++. In *IN SIGMOD CONFERENCE*, pages 906–908. ACM Press, 2005.

[AJP07] Julie Allinson, Pete Johnston, and Andy Powell. A dublin core application profile for scholarly works. *Ariadne*, (50), January 2007.

[Amb03] Scott W. Ambler. *Agile Database Techniques Effective Strategies for the Agile Software Developer*. John Wiley Sons, Heidelberg (DE), 2003.

[Ami09] S. Amir. Un système d'intégration de métadonnées dédiées au multimédia. In *INFORSID*, pages 491–492, 2009.

[ATW+07] Charu C. Aggarwal, Na Ta, Jianyong Wang, Jianhua Feng, and Mohammed Javeed Zaki. Xproj : a framework for projected structural clustering of xml documents. In *KDD*, pages 46–55, 2007.

[AvH08] Grigoris Antoniou and Frank van Harmelen. *A Semantic Web Primer*. MIT Press, Cambridge, MA, 2. edition, 2008.

[AYCS02] Sihem Amer-Yahia, SungRan Cho, and Divesh Srivastava. Tree pattern relaxation. In *EDBT*, pages 496–513, 2002.

[BA05] Hannes Bohring and Sören Auer. Mapping xml to owl ontologies. In *Leipziger Informatik-Tage*, pages 147–156, 2005.

[BAB+10] Ioan Marius Bilasco, Samir Amir, Patrick Blandin, Chabane Djeraba, Juhani Laitakari, Jean Martinet, Eduardo Martínez-Gracía, Daniel Pakkala, Mika Rautiainen, Mika Ylianttila, and Jiehan Zhou. Semantics for intelligent delivery of multimedia content. In *SAC*, pages 1366–1372, 2010.

[BBM+89] Ronald J. Brachman, Alexander Borgida, Deborah L. McGuinness, Peter F. Patel-Schneider, and Lori Alperin Resnick. The classic knowledge representation system or, kl-one : The next generation. In *Preprints of the Workshop on Formal Aspects of Semantic Networks, Two Harbors*, pages 1036–1043. MorganKaufman, 1989.

[BCM+03] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook : Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[BdWH+03] Ian S. Burnett, Rik Van de Walle, Keith Hill, Jan Bormans, and Fernando Pereira. Mpeg-21 : Goals and achievements. *IEEE MultiMedia*, 10.(4) :60–70, 2003.

[BG04] D. Brickley and R.V. Guha. Rdf vocabulary description language 1.0 : Rdf schema, February 2004. `http://www.w3.org/TR/rdf-schema/`.

[BGM08] Elisa Bertino, Giovanna Guerrini, and Marco Mesiti. Measuring the structural similarity among xml documents and dtds. *J. Intell. Inf. Syst.*, 30(1) :55–92, 2008.

[BLMS09] Mihaela Brut, Sébastien Laborie, Ana-Maria Manzat, and Florence Sèdes. Integrating Heterogeneous Metadata into a Distributed Multimedia Information System (regular paper). In *Cognitive systems with Interactive Sensors (COGIS), Paris (France), 16/11/09-18/11/09*, page (electronic medium), http ://www.see.asso.fr, novembre 2009. Société de l'Electricité, de l'Electronique et des Technologies de l'Information et de la Communication (SEE).

[BMC+03] Mikhail Bilenko, Raymond J. Mooney, William W. Cohen, Pradeep D. Ravikumar, and Stephen E. Fienberg. Adaptive name matching in information integration. *IEEE Intelligent Systems*, 18(5) :16–23, 2003.

[BS01] Steven Bird and Gary Simons. The olac metadata set and controlled vocabularies. *CoRR*, cs.CL/0105030, 2001.

[BSZ03] P. Bouquet, L. Serafini, and S. Zanobini. Semantic coordination : A new approach and an application. In *International Semantic Web Conference*, pages 130–145, 2003.

[BV05] Aida Boukottaya and Christine Vanoirbeek. Schema matching for transforming structured documents. In *ACM Symposium on Document Engineering*, pages 101–110, 2005.

[BZCS01] Ana B. Benitez, Di Zhong, Shih-Fu Chang, and John R. Smith. Mpeg-7 mds content description tools and applications. In *CAIP*, pages 41–52, 2001.

[CCS02] CCSDS. Open archival information systems, February 2002. `http://public.ccsds.org/publications/archive/650x0b1.pdf`.

[CON04] WONDERWEB CONSORTIUM. Dolce : A descriptive ontology for linguistic and cognitive engineering, February 2004. `http://www.loa-cnr.it/DOLCE.html`.

[CPSZ01] Shih-Fu Chang, Atul Puri, Thomas Sikora, and HongJiang Zhang. Introduction to the special issue on mpeg-7. *IEEE Trans. Circuits Syst. Video Techn.*, 11(6) :685–687, 2001.

[CRF03] William W. Cohen, Pradeep D. Ravikumar, and Stephen E. Fienberg. A comparison of string distance metrics for name-matching tasks. In *IIWeb*, pages 73–78, 2003.

[CS00] J. Conallen and M. Saez. *Concevoir des applications Web avec UML.* Technologies objet. Solutions UML. Eyrolles, 2000.

[CZ06] Lois Mai Chan and Marcia Lei Zeng. Metadata interoperability and standardization a study of methodology, July 2006. `http://www.dlib.org/dlib/june06/chan/06chan.html`.

[DCWS06] Theodore Dalamagas, Tao Cheng, Klaas-Jan Winkel, and Timos K. Sellis. A methodology for clustering xml documents by structure. *Inf. Syst.*, 31(3) :187–228, 2006.

[DDH01] AnHai Doan, Pedro Domingos, and Alon Y. Halevy. Reconciling schemas of disparate data sources : a machine-learning approach. In *Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, SIGMOD '01, pages 509–520, New York, NY, USA, 2001. ACM.

[DHB+00] Stefan Decker, Frank Van Harmelen, Jeen Broekstra, Michael Erdmann, Dieter Fensel, Ian Horrocks, Michel Klein, and Sergey Melnik. The semantic web - on the respective roles of xml and rdf. *IEEE Internet Computing*, 4 :http ://www.ontoknowl, 2000.

[DHL03] M. Doerr, J. Hunter, and C. Lagoze. Towards a core ontology for information integration. *J. Digit. Inf.*, 4(1), 2003.

[DMM+03] D.Carmel, Y. S. Maarek, M. Mandelbrod, Y. Mass, and A. Soffer. Searching xml documents via xml fragments. In *SIGIR*, pages 151–158, 2003.

[DR02] Hong Hai Do and Erhard Rahm. Coma - a system for flexible combination of schema matching approaches. In *VLDB*, pages 610–621, 2002.

[DSL05] Chabane Djeraba, Nicu Sebe, and Michael S. Lew. Systems and architectures for multimedia information retrieval. *Multimedia Syst.*, 10(6) :457–463, 2005.

[EIF04] EIF. European interoperability framework for pan-european egovernment services., 2004.

[EPV10] Kai Eckert, Magnus Pfeffer, and Johanna Völker. Towards Interoperable Metadata Provenance. In *Proceedings of the ISWC workshop on Semantic Web for Provenance Management (SWPM)*, 2010.

[ES07] Jérôme Euzenat and Pavel Shvaiko. *Ontology matching*. Springer-Verlag, Heidelberg (DE), 2007.

[Euz01a] Jérôme Euzenat. An infrastructure for formally ensuring interoperability in a heterogeneous semantic web. In *SWWS*, pages 345–360, 2001.

[Euz01b] Jérôme Euzenat. Towards a principled approach to semantic interoperability. In *Workshop on Ontologies and Information Sharing, IJCAI01*, page pages, 2001.

[Euz04] Jérôme Euzenat. An api for ontology alignment. In *International Semantic Web Conference*, pages 698–712, 2004.

[EV04] J. Euzenat and P. Valtchev. Similarity-based ontology alignment in OWL-lite. In R. López de Mántaras and L. Saitta, editors, *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI-04)*, pages 333–337. IOS Press, 2004.

[Fel98] Christiane Fellbaum, editor. *WordNet : An Electronic Lexical Database*. MIT Press, Cambridge, MA, 1998.

[Fon97] Joseph Fong. Converting relational to object-oriented databases. *SIGMOD Record*, 26(1) :53–58, 1997.

[FY04] J. Paoli C.M. Sperberg-McQueen E. Maler F. Yergeau, T. Bray. Extensible markup language (xml) 1.0 (third edition) w3crecommendation., 2004. http://www.w3.org/TR/2004/REC-XML-20040204/.

[FZT04] Matthias Ferdinand, Christian Zirpins, and David Trastour. Lifting xml schema to owl. In *ICWE*, pages 354–358, 2004.

[GC05] R. García and O. Celma. Semantic integration and retrieval of multimedia metadata. In *2nd European Workshop on the Integration of Knowledge, Semantic and Digital Media*, pages 69–80, 2005.

[GDDD04] Dragan Gasevic, Dragan Djuric, Vladan Devedzic, and Violeta Damjanovic. Converting uml to owl ontologies. In *WWW (Alternate Track Papers & Posters)*, pages 488–489, 2004.

[Gre03] Jane Greenberg. Metadata and the world wide web. In *Encyclopedia of Library and Information Science*, pages 1876–1888. Marcel Dekker, 2003.

[Gru93] Thomas R. Gruber. A Translation Approach to Portable Ontology Specifications. *Knowledge Acquisition*, 5(2) :199–221, 1993.

[GSDN99] Georges Gardarin, Fei Sha, and Tuyet-Tram Dang-Ngoc. Xml-based components for federating multiple heterogeneous data sources. In *Proceedings of the 18th International Conference on Conceptual Modeling*, ER '99, pages 506–519, London, UK, 1999. Springer-Verlag.

[GSY04] Fausto Giunchiglia, Pavel Shvaiko, and Mikalai Yatskevich. S-match : an algorithm and an implementation of semantic matching. In *ESWS*, pages 61–75, 2004.

[Hau05] M. Hausenblas. Multimedia vocabularies on the semantic web, July 2005. http://www.w3.org/2005/Incubator/mmsem/.

[HFB+00] Ian Horrocks, Dieter Fensel, Jeen Broekstra, Stefan Decker, Michael Erdmann, Carole Goble, Frank van Harlemen, Michel Klein, Steffen Staab, Rudi Studer, Enrico Motta, and I. Horrocks. The ontology inference layer oil, 2000.

[HHP+05] Laura M. Haas, Mauricio A. Hernández, Lucian Popa, Mary Roth, and Howard Ho. Clio grows up : From research prototype to industrial tool. In *In ACM SIGMOD International Conference on Management of Data (SIGMOD*, pages 805–810, 2005.

[Hir75] Daniel S. Hirschberg. A linear space algorithm for computing maximal common subsequences. *Commun. ACM*, 18(6) :341–343, 1975.

[HK10] Bernhard Haslhofer and Wolfgang Klas. A survey of techniques for achieving metadata interoperability. *ACM Comput. Surv.*, 42(2), 2010.

[HL01] Jane Hunter and Carl Lagoze. Combining rdf and xml schemas to enhance interoperability between metadata application profiles. pages 457–466, 2001.

[HP00] Rachel Heery and Manjula Patel. Application profiles : mixing and matching metadata schemas, February 2000. `http://www.ariadne.ac.uk/issue25/app-profiles/`.

[HPSH03] Ian Horrocks, Peter F. Patel-Schneider, and Frank Van Harmelen. From shiq and rdf to owl : The making of a web ontology language. *Journal of Web Semantics*, 1 :2003, 2003.

[Hun03] J. Hunter. Enhancing the semantic interoperability of multimedia through a core ontology. *IEEE Trans. Circuits Syst. Video Techn.*, 13(1) :49–58, 2003.

[Ini07] Advanced Distributed Learning Initiative. Sharable content reference model, July 2007. `http://www.adlnet.gov/scorm/index.aspx`.

[ISO02] ISO15745. International standards for business, government and society., 2002. `http://www.w3.org/TR/2004/REC-XML-20040204/`.

[JMK07] Yves Jean-Mary and Mansur Kabuka. ASMOV : Ontology alignment with semantic validation. In *Proc. of Joint SWDB-ODBIS Workshop on Semantics, Ontologies, Databases, Vienna (Austria)*, September 2007.

[Kle02] Michel C. A. Klein. Interpreting xml documents via an rdf schema ontology. In *DEXA Workshops*, pages 889–894, 2002.

[KP92] Gudrun Klose and Thomas Pirlein. Die modellierung der leu/2-wissensbasis im überblick. In *Ontologie und Axiomatik der Wissensbasis von LILOG*, pages 3–22, 1992.

[KS05a] Yannis Kalfoglou and W. Marco Schorlemmer. Ontology mapping : The state of the art. In *Semantic Interoperability and Integration*, 2005.

[KS05b] Yannis Kalfoglou and W. Marco Schorlemmer. Ontology mapping : The state of the art. In *Semantic Interoperability and Integration*, 2005.

[LAM01] Joanne LAMB. Sharing best methods and know-how for improving generation and use of metadata. 2001.

[LBSM08] W. Lee, T. Bürger, F. Sasaki, and V Malaisé. Use cases and requirements for ontology and api for media object, July 2008. `http://dev.w3.org/2008/video/mediaann/mediaont-1.0/`.

[LC00] Wen-Syan Li and Chris Clifton. Semint : A tool for identifying attribute correspondences in heterogeneous databases using neural networks, 2000.

[LDKG04] Bach Thanh Le, Rose Dieng-Kuntz, and Fabien Gandon. On ontology matching problems - for building a corporate semantic web in a multi-communities organization. In *ICEIS (4)*, pages 236–243, 2004.

[Les86] Michael Lesk. Automatic sense disambiguation using machine readable dictionaries : how to tell a pine cone from an ice cream cone. In *SIGDOC '86 : Proceedings of the 5th annual international conference on Systems documentation*, pages 24–26. ACM Press, 1986.

[Lev66] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady.*, 10(8) :707–710, February 1966.

[LLZL08] Guanyu Li, Lin Li, Jun Zhang, and Zhenghai Luo. Mapping based automatic instance transformation. In *Proceedings of the 2008 Second International Symposium on Intelligent Information Technology Application - Volume 01*, IITA '08, pages 799–803, Washington, DC, USA, 2008. IEEE Computer Society.

[LMC03] Dongwon Lee, Murali Mani, and Wesley W. Chu. Schema conversion methods between xml and relational models. In *Yan / Information and Software Technology 48 (2006) 245252 Knowledge Transformation for the Semantic Web, IOS*, pages 1–17. IOS Press, 2003.

[LOC07] LOC. Mets (metadata encoding and transmission standard), February 2007. `http://www.loc.gov/standards/mets`.

[Low01] Paul Benjamin Lowry. Xml data mediation and collaboration : A proposed comprehensive architecture and query requirements for using xml to mediate heterogeneous data sources and targets. In *34th Annual Hawai'i International Conference On System Sciences (HICSS), Mauii*, pages 3–6, 2001.

[LS08] F. Lin and K. Sandkuhl. A survey of exploiting wordnet in ontology matching. In *IFIP AI*, pages 341–350, 2008.

[LYHY02] Mong-Li Lee, Liang Huai Yang, Wynne Hsu, and Xia Yang. Xclust : clustering xml schemas for effective integration. In *CIKM*, pages 292–299, 2002.

[LZLT06] Yi Li, Qian Zhong, Juanzi Li, and Jie Tang. Result of ontology alignment with rimom at oaei06. In *In Proceedings of the ISWC 2006 Workshop on Ontology Matching*, 2006.

[Mar02] Philippe Martin. Knowledge representation in cglf, cgif, kif, frame-cg and formalized-english. In Uta Priss, Dan Corbett, and Galia Angelova, editors, *Proceedings of the 10th International Conference on Conceptual Structures (ICCS 2002)*, volume 2393 of *Lecture Notes in Computer Science*, pages 77–91. Springer, 2002.

[MBR01] J. Madhavan, P. A. Bernstein, and E. Rahm. Generic schema matching with cupid. In *VLDB*, pages 49–58, 2001.

[MGMR02] Sergey Melnik, Hector Garcia-Molina, and Erhard Rahm. Similarity flooding : A versatile graph matching algorithm and its application to schema matching. In *ICDE*, pages 117–128, 2002.

[MHH+01] Renée J. Miller, Mauricio A. Hernández, Laura M. Haas, Ling-Ling Yan, C. T. Howard Ho, Ronald Fagin, and Lucian Popa. The clio project : Managing heterogeneity. *SIGMOD Record*, 30(1) :78–83, 2001.

[Mil00] Paul Miller. Interoperability. what is it and why should i want it ? June 2000. `http://www.ariadne.ac.uk/issue24/interoperability/intro.html`.

[MIX02] MIX. Mix : Metadata for image in xml schema, February 2002. `http://www.loc.gov/standards/mix/`.

[ml08] multimedia lab. Dig35 : Metadata standard for digital images, February 2008. `http://multimedialab.elis.ugent.be/users/gmartens/Ontologies/DIG35/v0.2/`.

[MMSV02] Alexander Maedche, Boris Motik, Nuno Silva, and Raphael Volz. Mafra - a mapping framework for distributed ontologies. In *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web*, EKAW '02, pages 235–250, London, UK, 2002. Springer-Verlag.

[MvH04] D. L. McGuinness and F v Harmelen. Owl web ontology language overview. feb 2004. `http://www.w3.org/TR/owl-features/`.

[MVIM07] Igor Miletic, Marko Vujasinovic, Nenad Ivezic, and Zoran Marjanovic. Enabling semantic mediation for business applications : Xml-rdf, rdf-xml and xsd-rdfs transformations. In *IESA*, pages 483–494, 2007.

[MWJ99] Prasenjit Mitra, Gio Wiederhold, and Jan Jannink. Semi-automatic integration of knowledge sources. In *In Proc. of the 2nd Int. Conf. On Information FUSION99*, 1999.

[MXWL08] Wenhui Ma, Guangping Xu, Gang Wang, and Jing Liu. Detecting semantic mapping of ontologies with inference of description logic. In *SNPD*, pages 393–398, 2008.

[MZ98] Tova Milo and Sagit Zohar. Using schema matching to simplify heterogeneous data translation. In *VLDB*, pages 122–133, 1998.

[NK04] Natalya Fridman Noy and Michel Klein. Ontology evolution : Not the same as schema evolution. *Knowledge and Information Systems*, 6(4) :428–440, July 2004.

[NO95] Channah F. Naiman and Aris M. Ouksel. A classification of semantic conflicts in heterogeneous database systems. *J. Organ. Comput.*, 5 :167–193, February 1995.

[NP01] Ian Niles and Adam Pease. Towards a standard upper ontology. pages 2–9. ACM Press, 2001.

[NtHT+08] Felix Naumann, Ching tien Ho, Xuqing Tian, Laura Haas, and Nimrod Megiddo. Attribute classification using feature analysis, 2008.

[(OC07a] Online Computer Library Center (OCLC). Dewey decimal classification (ddc), July 2007. `http://www.oclc.org/dewey/`.

[oC07b] Library of Congress. Library of congress subject headings (lcsh), July 2007. `http://www.loc.gov/aba/cataloging/subject/`.

[Org04] National Information Standards Organization. Understanding metadata, July 2004.

[OS99] Aris M. Ouksel and Amit P. Sheth. Semantic interoperability in global information systems : A brief introduction to the research area and the special section. *SIGMOD Record*, 28(1) :5–12, 1999.

[PBP03] Siddharth Patwardhan, Satanjeev Banerjee, and Ted Pedersen. Using measures of semantic relatedness for word sense disambiguation. In *CICLing*, pages 241–257, 2003.

[PL98] Margaret St. Pierres and William P. LaPlan. Issues in crosswalking content metadata standards, July 1998. `http://www.niso.org/press/whitepapers/crsswalk.html`.

[PNN+05] Andy Powell, Mikael Nilsson, Ambjorn Naeve, Pete Johnston, and Thomas Baker. Dcmi abstract model. dublin core metadata initiative (dc), July 2005. `http://dublincore.org/documents/abstract-model/`.

[PS00] Silvia Pfeiffer and Uma Srinivasan. Tv anytime as an application scenario for mpeg-7. In *ACM Multimedia Workshops*, pages 89–92, 2000.

[PVM+02] Lucian Popa, Yannis Velegrakis, Renée J. Miller, Mauricio A. Hernández, and Ronald Fagin. Translating web data. In *VLDB*, pages 598–609, 2002.

[RB01] Erhard Rahm and Philip A. Bernstein. A survey of approaches to automatic schema matching. *VLDB J.*, 10(4) :334–350, 2001.

[SAD10] T. Urruty J. Martinet S. Amir, I. M. Bilasco and C. Djeraba. *Designing Intelligent Content Delivery Frameworks Using MPEG-21*, chapter 20, pages 455–475. John Wiley and Sons, UK, 2010.

[SBMZ03] Luciano Serafini, Paolo Bouquet, Bernardo Magnini, and Stefano Zanobini. An algorithm for matching contextualized schemas via sat. In *In Proceedings of CONTEXT03*, 2003.

[SD08] Dan A. Simovici and Chabane Djeraba. *Mathematical Tools for Data Mining : Set Theory, Partial Orders, Combinatorics*. Springer Publishing Company, Incorporated, 2008.

[SE05] Pavel Shvaiko and Jérôme Euzenat. A survey of schema-based matching approaches. pages 146–171, 2005.

[SE08] P. Shvaiko and J. Euzenat. Ten challenges for ontology matching. In *OTM Conferences (2)*, pages 1164–1182, 2008.

[SGB+08] Marcos Da Silveira, Nicolas Guelfi, Jerry-David Baldacchino, Pierre Plumer, Marc Seil, and Anke Wienecke. A survey of interoperability in e-health systems - the european approach. In Luis Azevedo and Ana Rita Londral, editors, *HEALTHINF (1)*, pages 172–175. INSTICC - Institute for Systems and Technologies of Information, Control and Communication, 2008.

[SGotFRfBR97] International Federation of Library Associations Study Group on the Functional Requirements for Bibliographic Records. Functional requirements for bibliographic records., July 1997. http://www.ifla.org/VII/s13/frbr/frbr.htm.

[SH01] Lalitha Suryanarayana and Johan Hjelm. Cc/pp for content negotiation and contextualization. In *Mobile Data Management*, pages 239–245, 2001.

[sh07] Medical subject headings. U.s. national library of medicine (nlm)., July 2007. http://www.nlm.nih.gov/mesh/.

[Sim02] Cyrille Simard. La normalisation de la formation en ligne : enjeux, tendances et perspectives., 2002.

[SK98] Amit P. Sheth and Wolfgang Klas, editors. *Multimedia Data Management : Using Metadata to Integrate and Apply Digital Media*. McGraw-Hill, 1998.

[SKR01] Hong Su, Harumi A. Kuno, and Elke A. Rundensteiner. Automating the transformation of xml documents. In *WIDM*, pages 68–75, 2001.

[SL90] Amit P. Sheth and James A. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Comput. Surv.*, 22(3) :183–236, 1990.

[SM01] Gerd Stumme and Alexander Maedche. Fca-merge : Bottom-up merging of ontologies. In *IJCAI*, pages 225–234, 2001.

[SPG+07] E Sirin, B Parsia, BC Grau, A Kalyanpur, and Y Katz. Pellet : A practical OWL-DL reasoner. *Journal of Web Semantics*, 5(2) :51–53, June 2007.

[SS04] Steffen Staab and Rudi Studer, editors. *Handbook on Ontologies*. International Handbooks on Information Systems. Springer, 2004.

[TCD+02] Baker. T, Blanchi. C, Brickley. D, DUVAL. E, HEERY. R, JOHNSTON. P, KALINICHENKO L, NEUROTH. H, and SUGIMOTO. S. Principles of metadata registries, white paper, delos network of excellence on digital libraries, July 2002.

[Til04] Barbara Tillett. What is frbr : A conceptual model for the bibliographic universe. feb 2004. `http://www.loc.gov/catdir/cpso/whatfrbr.html`.

[TLL+06a] Jie Tang, Juanzi Li, Bangyong Liang, Xiaotong Huang, Yi Li, and Kehong Wang. Using bayesian decision for ontology mapping. *Web Semant.*, 4 :243–262, December 2006.

[TLL+06b] Jie Tang, Juanzi Li, Bangyong Liang, Xiaotong Huang, Yi Li, and Kehong Wang. Using bayesian decision for ontology mapping. *Journal of Web Semantics*, page 2006, 2006.

[Tol06] Andreas Tolk. What comes after the semantic web - pads implications for the dynamic web. In *PADS*, page 55, 2006.

[TPC04] C. Tsinaraki, P. Polydoros, and S. Christodoulakis. Integration of owl ontologies in mpeg-7 and tv-anytime compliant semantic indexing. In *CAiSE*, pages 398–413, 2004.

[Tur01a] Peter D. Turney. Mining the web for synonyms : Pmi-ir versus lsa on toefl, 2001.

[Tur01b] Peter D. Turney. Mining the web for synonyms : Pmi-ir versus lsa on toefl. In *Proceedings of the 12th European Conference on Machine Learning*, EMCL '01, pages 491–502, London, UK, 2001. Springer-Verlag.

[VDPM+08] D Van Deursen, C Poppe, G Martens, E Mannens, and Rik Van De Walle. Xml to rdf conversion : A generic approach. *2008 International Conference on Automated Solutions for Cross Media Content and MultiChannel Distribution*, pages 138–144, 2008.

[VJCS97] Pepjijn R. S. Visser, Dean M. Jones, Bench T. J. M. Capon, and M. J. R. Shave. An analysis of ontological mismatches : Heterogeneity versus interoperability. In *AAAI 1997 Spring Symposium on Ontological Engineering*, Stanford, USA, 1997.

[vR79] C. J. van Rijsbergen. *Information Retrieval*. Butterworths, London, 2 edition, 1979.

[Wac03] Holger Wache. Semantische mediation für heterogene informationsquellen. *KI*, 17(4) :56–, 2003.

[Wei09] Michael Weiss. Xml metadata interchange. In Ling Liu and M. Tamer Özsu, editors, *Encyclopedia of Database Systems*, page 3597. Springer US, 2009.

[WG-02] IEE WG-12. Ieee standard for learning object metadata, July 2002. `http://ltsc.ieee.org/wg12`.

[wg02a] DIG35 working group. Dig35 : Metadata standard for digital images, February 2002. `http://xml.coverpages.org/dig35.html`.

[wg02b] EXIF working group. Exchangeable image file format, February 2002. `http://www.exif.org/`.

[wg04] IPTC working group. International tress telecommunication council, February 2004. `http://www.iptc.org/IPTC4XMP/`.

[wg08] XMP working group. Extensible metadata platform specification. XMPSpecificationPart2.pdf, February 2008.

[Wie92] Gio Wiederhold. Mediators in the architecture of future information systems. *IEEE COMPUTER*, 25(3) :38–49, 1992.

[WK03] Utz Westermann and Wolfgang Klas. An analysis of xml database solutions for the management of mpeg-7 media descriptions. *ACM Comput. Surv.*, 35(4) :331–373, 2003.

[YSL07] Kai Yang, Robert Steele, and Amanda Lo. An ontology for xml schema to ontology mapping representation. In *iiWAS*, pages 101–111, 2007.

Annexe A

Description du projet CAM4Home

Le but du projet européen CAM4Home (Collaborative Aggregated Multimedia for Digital Home) est la création d'une plateforme intelligente pour la création et la livraison de contenus multimédia via des réseaux et des terminaux hétérogènes. Cette plateforme, illustrée dans la Figure A.1 est composée de deux couches complémentaires :

- *la couche des métadonnées* contenant les informations qui décrivent la sémantique des contenus multimédia et leurs contextes d'utilisation. Les informations sémantiques sont utilisées pour la recherche et la navigation dans les contenus. Quant aux informations contextuelles, elles sont utilisées pour l'adaptation des contenus multimédia selon le profil de l'utilisateur et les caractéristiques matérielles des appareils utilisés pour la livraison et la consommation des contenus multimédia. Notre rôle dans cette tâche était la conception de métamodèle décrivant les informations contextuelles (utilisateur, communauté, appareils, services et réseaux), la spécification des deux couches du métamodèle (abstrait et concret) et d'assurer l'extensibilité du métamodèle.
- *la couche des services* contenant l'ensemble des services responsables de la manipulation des contenus tels que les services d'adaptation des contenus multimédia qui utilisent les métadonnées contextuelles ou les services de recherche sémantique. Notre rôle était de développer certains services de métadonnées (service d'adaptation et de recherche).

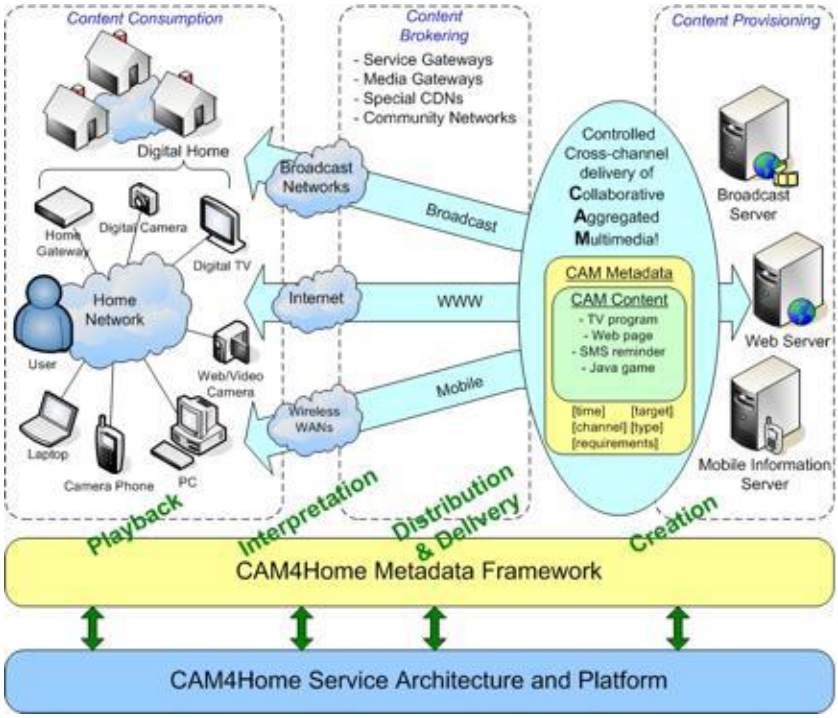

FIGURE A.1: La plateforme du projet CAM4Home.

CAM4Home est basé sur un concept novateur nommé CAM (Collaborative Aggregated Multimedia). Le CAM fait référence à l'agrégation et la composition de contenus multimédia individuels appelés *CAMObjets* dans un paquet d'informations nommé *CAMBundle*. Le projet développe un métamodèle commun pour les contenus CAM qui peut être appliqués à des fins à la fois personnelles et commerciales, et qui permet une interopérabilité entre les standards de métadonnées.

Le projet CAM4Home regroupe 19 partenaires académiques et industriels (Figure A.2) dont 7 Français (CNRS, Thomson, Ouestaudiovisuel, IWEDIA, GET-INT, NDS, France Telecom), 7 Finlandais (VTT, Nokia, Sesca Innovations, Sofia Digital, VLP, Université de Oulu, Swelcom), 3 Espagnols (Atos Origin, Université de Murcie), 1 Slovène (Kapion) et 1 Luxembourgeois (Centre Henri Tudor). L'objectif principal du CNRS dans le projet CAM4Home était la modélisation et la spécification du métamodèle CAM4Home. Le CNRS avait également plusieurs autres tâches dans le développement des services.

Le projet CAM4Home a reçu la médaille d'argent du prix de la réalisation ITEA2 2010.

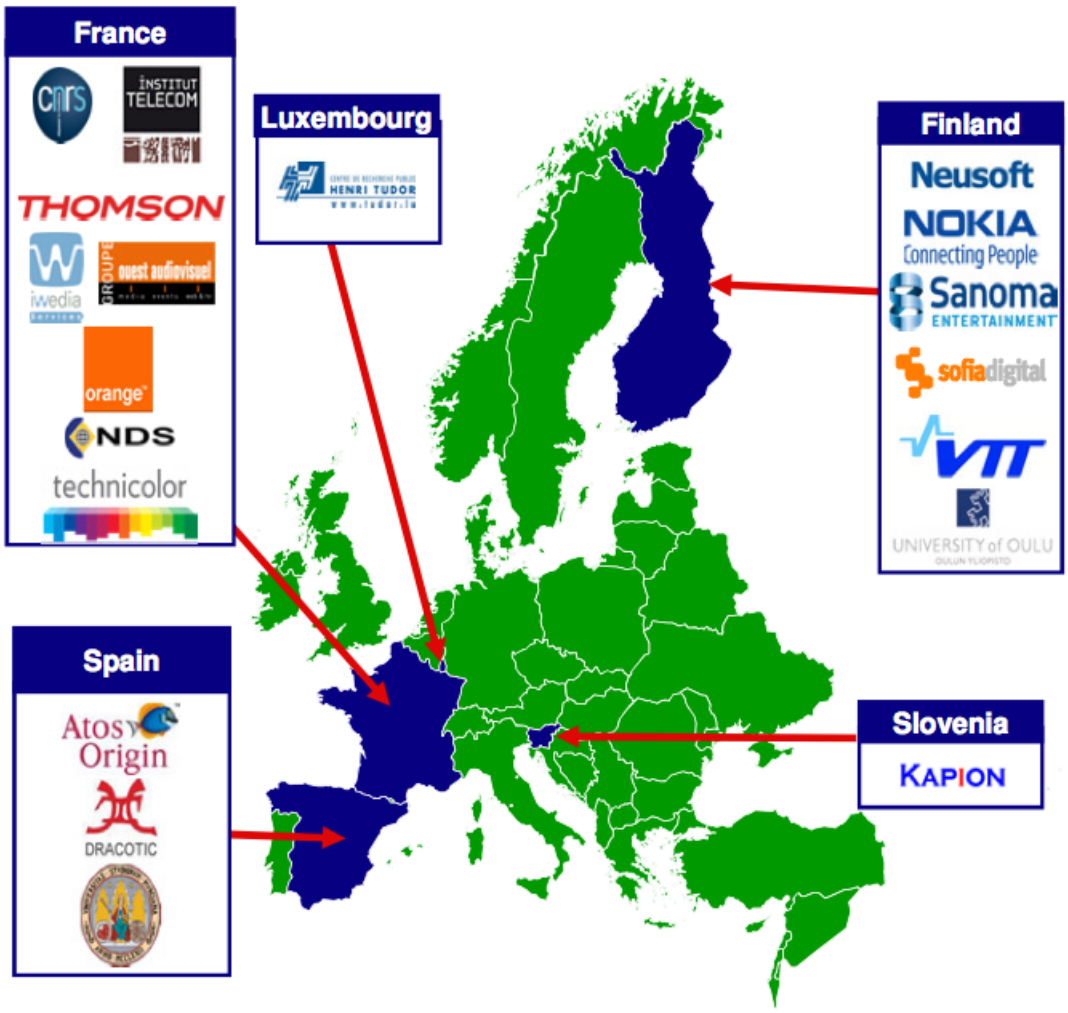

FIGURE A.2: Partenaires du projet CAM4Home.

Annexe B

Démonstrateur CAM4Home

Le métamodèle CAM4Home a été validé par plusieurs démonstrations logicielles faites par l'ensemble de partenaires du projet CAM4Home. La Figure B.1 montre l'architecture globale de la plateforme des services développés pour la réalisation de ce démonstrateur. Chaque démonstrateur s'articule autour de la plateforme afin d'accéder :

- aux informations sur les contenus (Metadata Services),
- aux notifications relatives aux changements intervenus dans la plateforme (enregistrement des nouveaux contenus) ainsi que relatifs au contexte de diffusion (Core Platform Services),
- aux services d'adaptation (Content and Network Services).

Le but des démonstrateurs est de montrer l'intérêt du métamodèle CAM4Home qui permet la diffusion des contenus multimédia sur des plate-formes hétérogènes (mobile, TV, PC, etc). Cela est fait via l'utilisation des services d'adaptation des métadonnées. La plate-forme CAM4Home utilise également le métamodèle pour plusieurs autres scénarios (streaming, jeux en ligne, P2P, etc). Pour chacun de ces scénarios, un ou plusieurs services spécifiques on été développés. A titre d'exemple, on peut citer le service de recommandation, service de streaming, service de recherche de contenus etc.

Afin de montrer un exemple réel d'utilisation du métamodèle CAM4Home, nous décrivons dans la suite de cette section un des scénarios réalisés dans le cadre du projet qui concerne l'agrégation

collaborative de contenus multimédia.

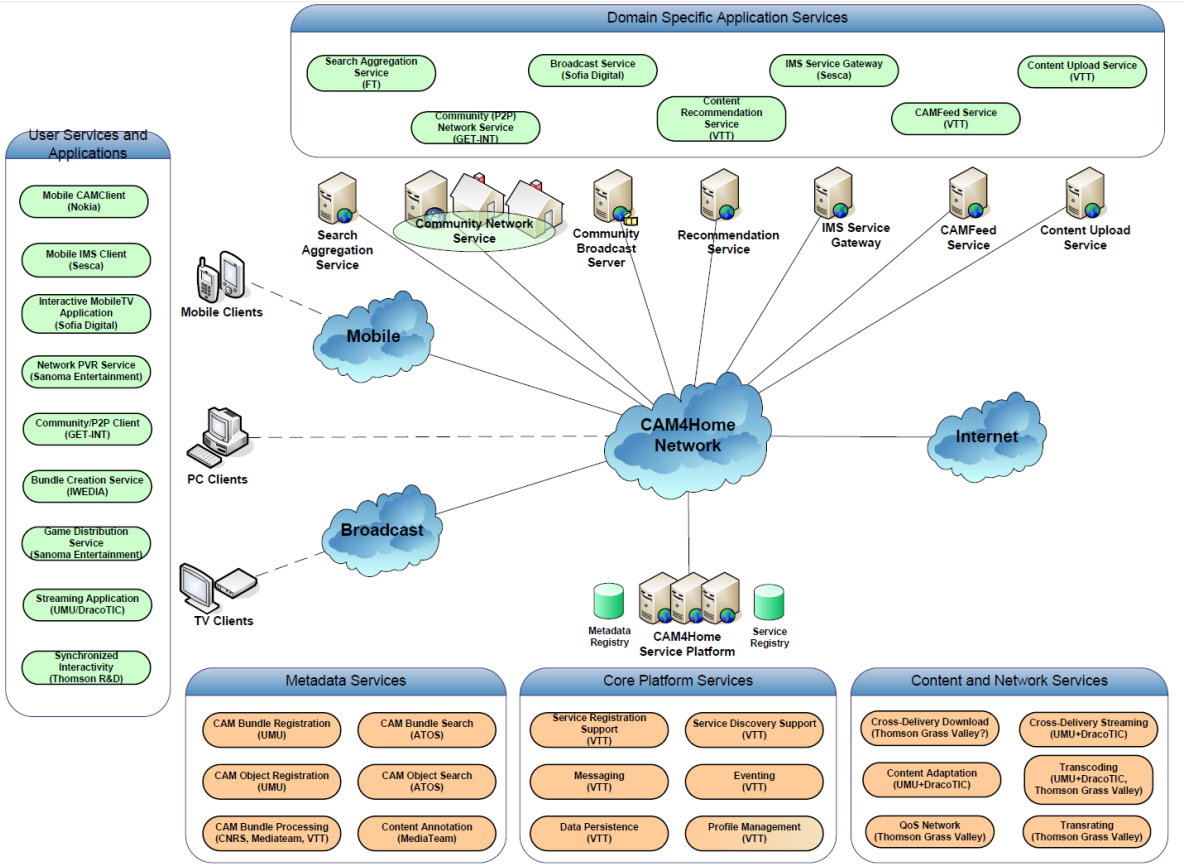

FIGURE B.1: Architecture globale du démonstrateur CAM4Home

B.1 Agrégation collaborative des contenus multimédia

Dans le scénario d'agrégation collaborative de contenus multimédia (illustré par la Figure B.2), les utilisateurs agissent comme des créateurs de nouveaux assemblages multimédia en enrichissant et en agrégeant les contenus existants. Ces assemblages donnent lieu au sein de la plate-forme à la création de nouvelles unités de déploiement indépendantes.

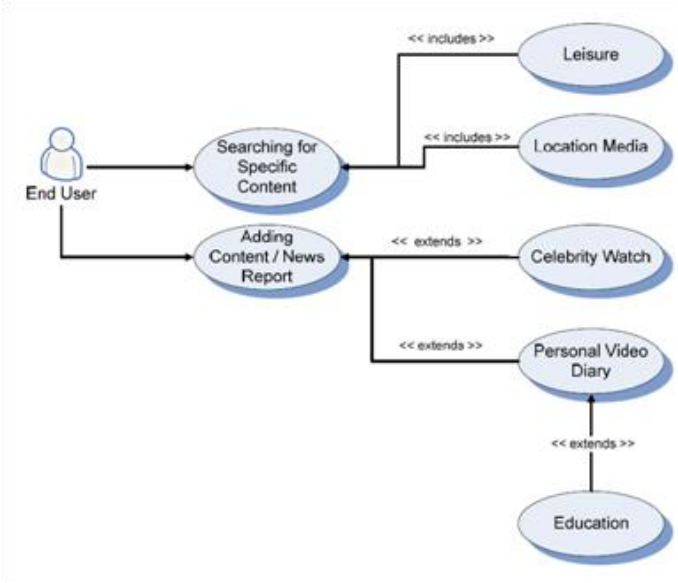

FIGURE B.2: Diagramme cas d'utilisation

Par exemple, un utilisateur U_1 charge des contenus de type, image, vidéo, audio, à propos du football. Chaque contenu est référencé dans la plate-forme sous forme d'un *CAMObject* (représenté par la classe *CAMElementMetadata* au centre de la Figure B.3) et sa description contient différents types de métadonnées :

- Des informations contextuelles sur la création du contenu (location, date, heure) regroupées dans le conteneur *ContextMetadata*,
- Des libellées (*SocialTag : Messi, but*) ou commentaires (*UserComment* : " *Quel joli but* ! ") générés par les utilisateurs lors du visionnage du contenu,
- Des informations de haut niveau (*AppearingConcept : Lionel Messi, Arbitre*, etc).

En utilisant ces métadonnées, le contenu pourra être cherché ou recommandé pour d'autres utilisateurs.

Afin de rendre accessibles les contenus chargés par U_1 à l'ensemble de la communauté, un *CAMBundle* (représenté par la classe *CAMBundleMetadata* au centre de la Figure B.4) assemblant ces conte-

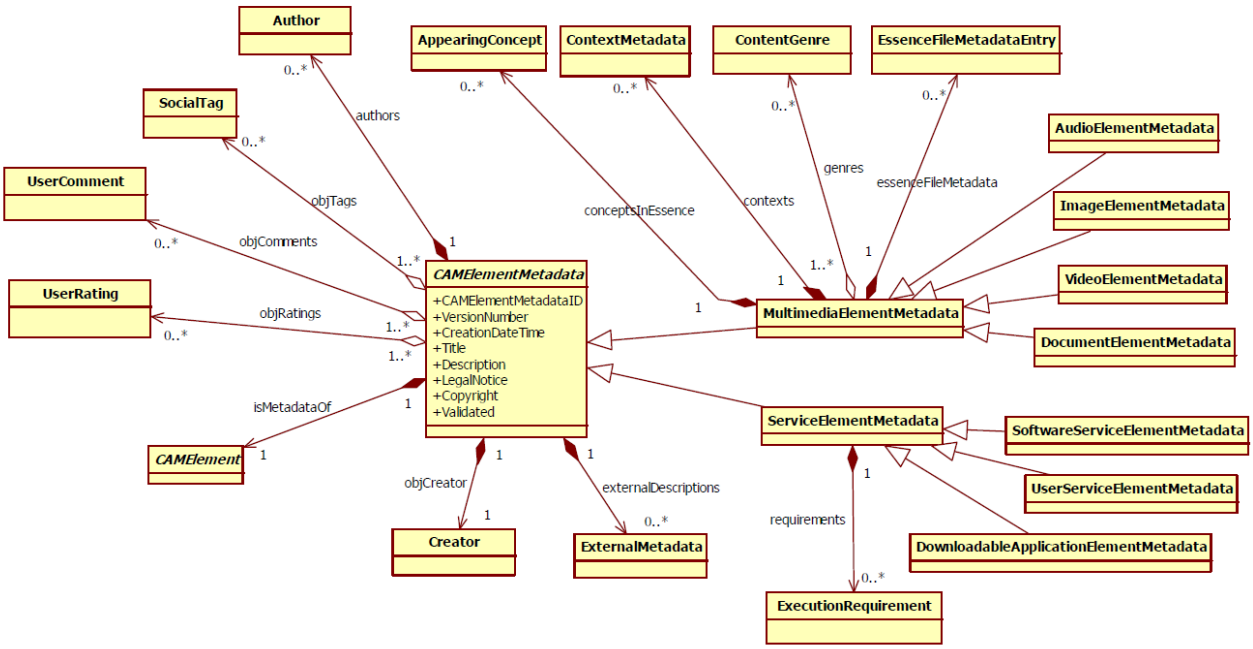

FIGURE B.3: CAMObject et les conteneurs associés

nus est crée. Des métadonnées relatives à ce *CAMBundle* décrivant les relations (Relationship) entre les différents *CAMObject* sont ajoutées afin de faciliter sa recherche. Un profil spécifique à U_1 est crée automatiquement par le système pour pouvoir lui suggérer des contenus qui sont susceptibles de l'intéresser.

Grace à l'utilisation des métadonnées générées par U_1, un autre utilisateur U_2 intéressé par le sport reçoit le *CAMBundle* grâce aux services de recommandation. U_2 décide de visualiser les contenus de U_1. A ce stade, des services spécifiques interviennent pour l'adaptation du *CAMBundle* selon les caractéristiques du terminal de l'utilisateur U_2. Les caractéristiques du terminal U_2 sont accessibles via les profils matériels décrits par des métadonnées supplémentaires.

L'utilisateur U_2 reconnaît l'image de *Lionel Messi* et annote la vidéo créée par U_1 via le descripteur *ApperingConcept(concept=Lionel Messi)*. Des commentaires et des mots clés peuvent être également insérés par U_2 pour personnaliser la description du *CAMBundle* initiale. U_2 ajoute ensuite une deuxième

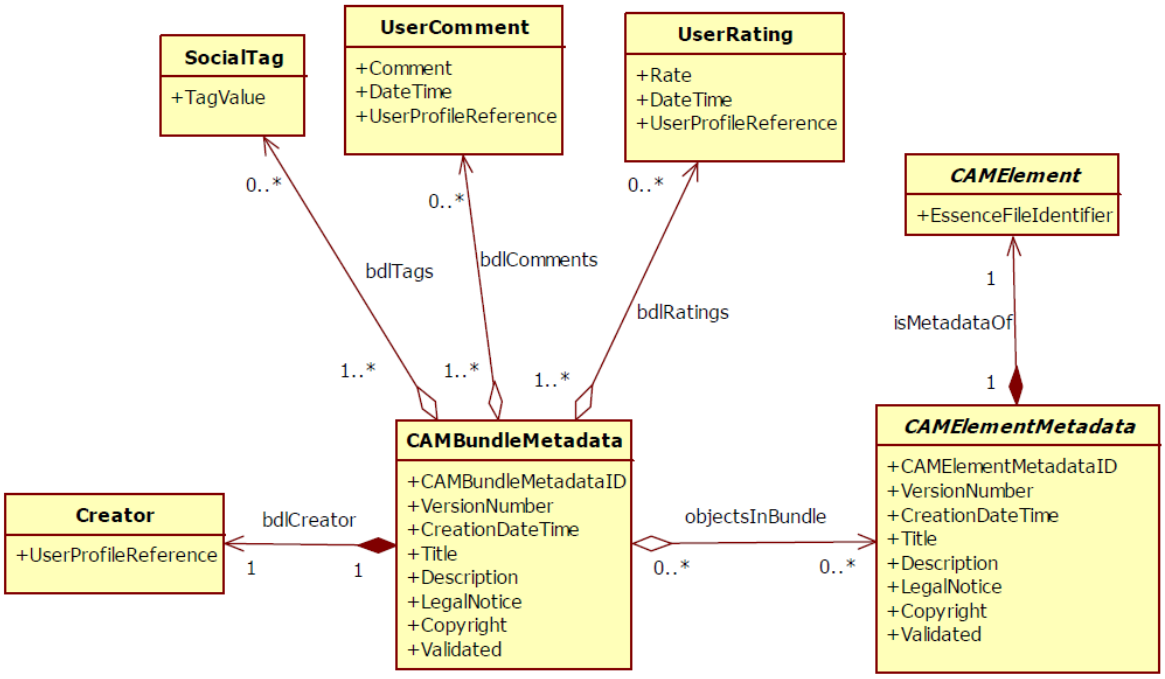

FIGURE B.4: CAMBundle et les conteneurs associés

vidéo au précédent *CAMBundle* en produisant une nouvelle version qu'il publiera pour la rendre accessible par le reste de la communauté.