PREMIERE PARTIE : DOCUMENT A STRUCTURES MULTIPLES : PRESENTATION ET ETAT DE L'ART

DEUXIEME PARTIE : NOTRE PROPOSITION : MODELISATION, INTEGRATION ET EXPLOITATION DE DOCUMENTS A STRUCTURES MULTIPLES

Introduction Générale

I. Contexte et problématique

Le développement considérable qu'ont connu les technologies de numérisation ces dernières décennies a conduit à une croissance quasi exponentielle du nombre de ressources numérisées. Si l'on considère que le concept de document numérique est le vecteur qui représente ces ressources, s'intéresser aux évolutions des ressources numérisées conduit tout naturellement à s'intéresser aux évolutions du concept même de document numérique. Ces évolutions touchent autant au fond (intégration de contenus multimédias notamment) qu'à la forme (conditions de présentation, de stockage et de manipulation) de ces documents. La prise en compte de ces évolutions, pour une gestion efficace des ressources documentaires, constitue la base de nos travaux et des problématiques auxquelles nous nous sommes intéressés. Ces problématiques concernent principalement les aspects structuration des documents.

Un document n'est plus vu comme un tout, ni comme un bloc monolithique, mais comme un ensemble d'entités (fragments ou granules de documents) reliées. L'évolution du document numérique a permis d'intégrer des entités de natures différentes, complexifiant ainsi la notion même de document. Chaque entité peut être un document simple ou complexe, et ainsi chacune des entités peut être à son tour découpée, composée et/ou décrite par d'autres entités. Ces entités sont reliées par des relations permettant de donner une forme au document et à ses entités. Plusieurs types de relations peuvent apparaître, de sorte à ce que plusieurs formes d'un même document émergent.

Chacune des matérialisations du même document est traduite par une structuration particulière (Cf. Chapitre I). La littérature identifie des structures documentaires particulières : structures logique, physique, sémantique…(Poullet et al. 1997). La nature des documents à traiter a induit de nouvelles structurations plus complexes, comme par exemple les structures spatiales, temporelles, qui ne sont pas de simples relations hiérarchiques. Cela ne sera pas sans incidence sur la complexité de leurs représentations informatiques. Ces structures peuvent être définies au niveau global du document comme elles peuvent être associées à l'une des entités qui le compose. ***Ainsi, un document est multistructuré, soit parce que plusieurs structures le décrivent, soit parce que les documents qui le composent ont eux-mêmes structurés ou multistructurés.***

Si l'on s'en réfère aux différentes solutions évoquées dans la littérature (Cf. Chapitre II) pour représenter cette multistructuralité, deux grandes familles peuvent être dégagées : les langages ou les modèles. De façon très générale, le constat qui peut être fait est que les solutions apportées sont intéressantes, mais partielles. Certaines se focalisent sur un type de document particulier, par exemple le document image (Mechkour 1995) ou texte (Navarro et Baeza-Yates 1997), alors que d'autres ne permettent de représenter que certaines structures ou encore sont contraintes par des structures de base (Chatti et al. 2007) ou tout simplement sont dépendantes d'une technologie (Sperberg-McQueen et Burnard 2007).

Le principal défi que nous nous proposons de relever dans cette thèse est la proposition d'une ***modélisation indépendante de tout type de structure documentaire, de tout type de document et de toute technologie*** (Cf. Chapitre III).

La prise en compte de ces différentes structurations d'un même document est primordiale pour une gestion optimale et partagée des fonds documentaires. La représentation de l'ensemble de ces structures ouvre aujourd'hui de nouvelles perspectives en termes d'accès et d'exploitation des informations documentaires (Cf. Chapitre IV et V).

II. Propositions et contributions

Nous cherchons donc à spécifier de nouvelles méthodes de modélisation, d'intégration et d'exploitation des documents en tirant partie de leur caractère multistructurel.

La modélisation que nous proposons, est basée sur la notion de fragmentation. Nous définissons la fragmentation comme étant la possibilité de décrire séparément les différentes entités qui forment un document ainsi que leurs relations, de façon à pouvoir traduire plusieurs types de liens entre ces entités. La diversité typologique des relations est à l'origine de la diversité typologique des structures. Les structures multiples d'un même document peuvent être établies pour traduire différents usages ou l'adaptation à différents contextes du même document. Des structures de même type peuvent être donc définies pour traduire des visions différentes de l'organisation du document. Nous parlons alors de vues de document pour couvrir aussi bien la diversité de vision que la diversité typologique des structures. Ces vues, telles que nous les avons conçues, peuvent être définies non seulement au niveau global du document, mais aussi au niveau de chacune des entités qui le compose.

Faciliter l'accès aux différentes entités est l'un des objectifs de notre modélisation. L'utilisation de la structure permet d'atteindre cet objectif. Cependant, la recherche d'un fragment particulier dans une grande masse de documents nécessite le parcours de toutes les structures de l'ensemble des documents. Réduire le nombre de structures parcourues par leur catégorisation est une bonne solution pour optimiser l'accès à l'information documentaire. Toutefois, rattacher toutes les informations documentaires à une même structure jugée la plus représentative conduit à la perte de certaines spécificités de la structure d'origine. C'est dans cette optique que nous avons modélisé un niveau spécifique pour garder les caractéristiques propres à chaque document et un niveau générique pour les caractéristiques communes à un sous ensemble de documents.

La catégorisation des documents en se basant sur leur structure nécessite le calcul d'une distance entre le représentant de la classe et l'individu. Le représentant d'une classe est matérialisé au niveau générique du modèle et l'individu représente une organisation particulière du document décrite au niveau spécifique du modèle. A partir d'un certain seuil de similarité déterminé par expérimentation, un individu est rattaché à la classe la plus proche. Le représentant de cette classe peut être éventuellement adapté pour couvrir l'organisation spécifique du nouvel individu. Cette adaptation doit être contrôlée afin de ne

pas avoir des classes trop « larges », qui finiraient par perdre en représentativité. Afin de garantir une dispersion minimale au sein de chaque classe, nous avons proposé une démarcher permettant la conservation de la représentativité.

Une fois les deux niveaux spécifique et générique du modèle instanciés, la restitution des fragments de documents peut reposer aussi bien sur une même vue que sur l'ensemble des vues des documents modélisés. Les spécificités du modèle proposé élargissent les possibilités d'exploitation des documents par le biais de deux processus complémentaires : l'interrogation et l'analyse multidimensionnelle. L'originalité de ces processus réside dans l'utilisation des éléments appartenant à une ou plusieurs vues comme paramètres d'interrogation ou d'analyse et la gestion de chevauchement entre les paramètres issus de différentes vues et définis sur un même contenu.

III. Organisation du mémoire

Ce mémoire se compose de cinq chapitres regroupés en deux parties.

La première partie est consacrée à l'état de l'art. Elle décrit le contexte de nos travaux, ainsi qu'un panorama des travaux réalisés dans le domaine de la modélisation et de l'intégration de documents multistructurés. Cette partie se compose de deux chapitres.

Le premier chapitre traite de l'évolution du concept de document du papier vers le numérique, du texte vers le multimédia, et aborde ensuite la description des documents au travers de leur structuration. Nous présentons par conséquent un panorama des typologies et des méthodes de représentation des structures de documents ainsi que les standards documentaires sur lesquels repose la description de l'organisation et du contenu des documents. Nous introduisons à ce niveau les concepts liés à multistructuralité. La dernière section est consacrée à la présentation des principales problématiques posées par la prise en compte des structures multiples de documents.

Le deuxième chapitre est consacré à un panorama des principaux travaux liés à la modélisation, la représentation et l'exploitation des documents multistructurés. Nous avons organisé ces travaux en deux familles. La première famille concerne les travaux proposant des solutions basées sur des langages. La seconde regroupe les approches basées sur des modèles.

La deuxième partie aborde notre proposition visant à la modélisation, l'intégration et l'exploitation des documents multistructurés. Cette partie se compose de trois chapitres.

Le troisième chapitre propose un modèle générique de documents multistructurés. Ce modèle appelé MVDM (**M**ulti **V**iew **D**ocument **M**odel) comprend deux niveaux de description : spécifique et générique. Le niveau spécifique décrit les documents au travers de leurs différentes structures traduites par des vues spécifiques. Le niveau générique représente les caractéristiques des classes de documents ayant des organisations spécifiques similaires.

Le quatrième chapitre est consacré aux différentes techniques proposées pour l'intégration et l'exploitation de documents selon le modèle MVDM. La première partie de

ce chapitre traite en particulier de la problématique de classification de vues. La deuxième partie de ce chapitre détaille les techniques proposées pour l'exploitation des documents intégrés dans la base en tirant profit de leur caractère multistructurel. Il présente les techniques d'interrogation des documents, ainsi que d'analyse multidimensionnelle basée sur les éléments de l'ensemble des structures d'un même document.

Le dernier chapitre présente la validation de nos propositions au travers de l'outil MDOCREP (**M**ultistructured **DOC**ument **REP**ository) que nous avons développé. Nous décrivons notamment les résultats des expérimentations réalisées dans le cadre de la classification des documents et de l'analyse multidimensionnelle des informations documentaires en exploitant leur caractère multistructurel.

PREMIERE PARTIE :
DOCUMENT A STRUCTURES MULTIPLES : PRESENTATION ET ETAT DE L'ART

Chapitre I – Document et structure : concepts de base

Résumé du chapitre. *Ce chapitre présente le contexte de ce mémoire de thèse. Il expose du concept de « document » et son évolution. Il introduit également le concept de structure au travers d'un tour d'horizon des différentes typologies, des différentes formes de représentation et des standards de description de ces structures. Ces notions nous amènent à présenter le concept de document à structures multiples appelé encore document multistructuré.*

Sommaire du Chapitre I.

I. Introduction

Avec la numérisation, le concept de document a connu des mutations profondes qui touchent aussi bien la forme que le fond. Ces évolutions laissent percevoir le document comme un ensemble d'entités. Relier ces entités revient à leurs offrir une organisation. La diversité de relations qui peuvent être établies entre deux entités permet de traduire plusieurs organisations pour un même document. Si une organisation est matérialisée par une structure, la coexistence de plusieurs organisations conduit à la coexistence de plusieurs structures définissant ainsi un nouveau concept : la multistructuralité des documents.

Si l'on se focalise sur l'évolution du document, cette évolution se répercute directement sur la structure. L'intégration des médias agit dans un premier temps sur la forme des structures qui passe des arborescences à des graphes et des forêts et dans un deuxième temps sur la typologie de ces structures. L'objectif de ce chapitre est d'introduire le concept de structuration. Eliciter la notion de structure, ses différentes typologies, ses différentes formes de représentation et les standards qui la normalisent nous amène à introduire le concept de la multistructuralité et les problématiques qui en découlent.

Nous commençons ce chapitre par définir le concept de document. Ceci nous amène à suivre son évolution dans le temps. Afin de décrire ces documents ainsi que les différentes relations qui traduisent leur composition, plusieurs types de structures (logique, physique, sémantique, etc.) peuvent être utilisés. Ensuite, nous faisons un tour d'horizon de ces types de structures et à leurs représentations (liste, arbre, graphe, etc.). Nous concluons ce chapitre par un exposé des principales problématiques relatives à la gestion de structures multiples de documents.

II. Document, annotation et métadonnée

II.1. Document

II.1.1. Définitions

Une définition générique du concept de *document* est donnée par le Larousse : « *tout renseignement écrit ou objet servant de preuve ou d'information* ». L'ISO (**I**nternational **O**rganization for **S**tandardization) (ISO-5127 2001) définit le document comme « *une information enregistrée qui peut être traitée comme une unité dans un processus de documentation* ».

Cette définition met en avant deux points : l'aspect informationnel et l'aspect matériel du document. L'aspect informationnel concerne le contenu et donc les données véhiculées au travers d'un document. Quant à l'aspect matériel, il se rapporte au support de l'information, donc à son stockage. Bachimont (Bachimont 1998) rejoint cette définition. Il définit le document comme « *un objet matériel exprimant un contenu* ».

Dans le cadre d'une réflexion collective d'un réseau thématique pluridisciplinaire du département STIC du CNRS, (Pédauque 2003) distingue trois aspects du document : le document comme forme, comme signe, et enfin comme medium. Le premier aspect renvoie au fait que le document est une forme physique perceptible. Dans ce cas, le document est considéré comme étant un objet matériel ou immatériel. Le deuxième aspect renvoie aux problèmes de manipulation, lecture et interprétation du contenu. Le document est perçu comme une entité porteuse de sens et doté d'une intentionnalité. Enfin, le dernier aspect renvoie à la dimension usage du document. Le document est thématisé comme objet social, objet de négociation et de transaction culturelle et économique.

Un des aspects importants est aussi l'échange. Cet aspect est abordé dans la définition de (Blasselle 1998), qui mentionne qu'un document permet d'assurer la « *transmission de connaissance, de savoir et d'information* ». Cette capacité de transmission est assurée par la présentation de l'information sous une forme persistante et portative du contenu, qui est classiquement le support papier.

II.1.2. Evolution du concept de document

Le numérique a permis la dématérialisation de documents : le conteneur de documents est passé du papier aux supports magnétique et électronique. En même temps, l'information véhiculée, classiquement textuelle, a pu s'enrichir d'éléments issus d'autres média (des images fixes ou animées, des graphiques, du son, de la vidéo, etc.). Ceci a conféré au document de nouvelles capacités aussi bien au niveau du contenu informationnel qu'au niveau des traitements possibles. En outre, un document électronique ne se contente plus de remplir les fonctionnalités d'un simple document papier, il permet d'offrir d'autres possibilités de gestion telles que : une restitution selon différents formats, une communication avec d'autres systèmes informatiques (échanges, partage, etc.), des recherches d'informations, etc. (Dupoirier 1995).

II.1.2.1. Document numérique

La numérisation des documents a amené à de nouvelles définitions du terme document comme celle donnée par l'ISO (ISO-5127 2001) : « *Un document est l'ensemble d'un support d'information et des données enregistrées sur celui-ci sous forme en général permanente et lisible par l'homme et la machine* ». Contrairement au document traditionnel, l'accès à l'information sur les supports numériques n'est jamais direct. « *Il passe au minimum par le décodage d'une représentation sous forme binaire de l'information pour en proposer une présentation sous une forme sémiotique lisible* » (Bachimont et Crozat 2004). Le document numérique peut être considéré ainsi comme une reconstruction dynamique d'un document matériel.

La numérisation peut être effectuée de deux manières différentes : à partir d'un support papier par scannérisation ou directement pensé et numérisé à partir d'un logiciel spécifique.

Quelle que soit la manière dont est obtenu le document numérique, sa migration des formes matérielles (le support papier) aux formes immatérielles (électroniques ou

numériques) présente plusieurs avantages : gains de temps et d'espace, amélioration de la réactivité, réduction des coûts de stockage et d'échange, etc. Rapidement sont venus s'ajouter aux documents issus de la numérisation, des documents directement créés sous forme numérique (éditions informatiques ou bureautiques, messages électroniques, etc.). Le document est donc devenu de plus en plus riche et complexe en intégrant des contenus divers (texte, graphique, image, son, vidéo, etc.). Ces contenus sont regroupés dans ce que l'on appelle désormais des documents multimédias.

II.1.2.2. Document multimédia

Le terme *multimédia* se rapporte à une communication à travers plusieurs types de média utilisés conjointement. Dans la littérature, un média est considéré comme le support sur lequel est physiquement stocké, contenu, ou transféré, une musique, un film, des photos ou plus généralement des données. Par ailleurs, en informatique, un média est défini comme « *un type de données abstrait c'est-à-dire une description d'une structure de données ainsi qu'un ensemble d'opérations définies sur ces données telles que les opérations pour la saisie et la présentation de données* » (Apers et al. 1997).

En nous référant à la définition du concept de *document*, nous pouvons ainsi définir un document multimédia comme étant l'agencement interactif dans le temps et dans l'espace d'éléments distincts de natures différentes (issus de deux ou plusieurs types de média) afin d'enrichir le contenu de l'information diffusée. Ainsi, multimédia est défini par les chercheurs en informatique comme l'étude d'éléments complexes, tels que les images, les vidéos et les sons. Nous présentons dans cette section les médias les plus utilisés.

- Le texte

Un texte est une succession organisée de chaînes de caractères. Ces chaînes de caractères forment des mots, des phrases et des paragraphes selon un langage. Selon (Marcoux 1994), un texte désigne « *un ensemble d'informations qui représente, dans les activités normales de manipulation ou d'acquisition des [connaissances] à l'échelle humaine, une unité que l'on peut raisonnablement considérer comme indivisible et complète* ».

Plusieurs standards tels que RTF (**R**ich **T**ext **F**ormat), PDF (**P**ortable **D**ocument **F**ormat), Text, LaTeX (**La**mport **TEX**) ont été proposés afin de représenter et traiter le média texte. Ces standards sont essentiels, car ils permettent de réaliser sur le document électronique des manipulations fastidieuses à effectuer sur un support papier, d'assurer l'intégrité, la réutilisabilité et la portabilité lors d'échanges de documents, et donc de favoriser le partage de documents.

- L'audio

L'audio représente le support matériel qui permet de véhiculer un message sous la forme d'une information sonore. Cette information sonore est le résultat des phases d'échantillonnage et de quantification d'un signal continu produit par une source sonore. Ainsi, l'étude de ce média revient à sa caractérisation par un ensemble de paramètres tels

que la fréquence d'échantillonnage, la taille de l'échantillon, le pas de quantification, le nombre de canaux utilisés, etc. Par ailleurs, le son requiert de gros volumes. Afin de faciliter le stockage et l'échange de ses volumes, on opte généralement pour la compression. Parmi les formats audio compressés les plus connus, nous citons le format WMA (**W**indows **M**edia **A**udio) de Microsoft, le format AAC (**A**dvanced **A**udio **C**oding) et le format MP3 (**MP**EG-1 audio layer **3**).

Ce média peut jouer différents rôles dans le document : source d'information supplémentaire ou complémentaire à un document textuel, commentaire d'un document image, agrément, canal de sortie supplémentaire lorsque les interfaces visuelles classiques sont saturées, aide aux handicapés visuels, alarme, etc.

- L'image

Selon le Larousse une image est « *une représentation d'un être ou d'une chose par les arts graphiques ou plastiques, la photographie, le film, etc.* ». Par image numérique, on entend toute image qui se présente sous la forme d'une matrice de pixels ou sous une forme vectorielle. Ainsi, elle est acquise, créée, traitée et stockée sous forme binaire.

De nos jours, une image peut être acquise par des dispositifs comme les scanners, les appareils photo, etc. Elle peut être créée directement par des programmes informatiques, via la souris, les tablettes graphiques ou par la modélisation 3D. Le traitement d'image est assuré par des logiciels spécifiques (tel que Adobe illustrator, Photoshop elements, etc.) qui permettent de modifier une image en taille, en couleur, d'ajouter ou supprimer tel ou tel élément, d'appliquer des filtres variés, etc. Une image peut être présentée, codée et stockée selon plusieurs formats (Roxin et Mercier 2004). Parmi les plus répandus, nous pouvons citer les formats BMP (BitMaP), JPEG (**J**oint **P**hotographic **E**xperts **G**roup), TIFF (**T**agged **I**mage **F**ile **F**ormat), PNG (**P**ortable **N**etwork **G**raphics), SPIFF (**S**till **P**icture **I**nterfrange **F**ile **F**ormat), etc.

- La vidéo

Les documents audiovisuels connus sous le nom de *vidéos* sont formés d'une séquence d'images appelées *frames* et d'une bande sonore synchronisée. Cependant, une séquence vidéo n'est pas une simple succession d'images et de sons, mais un arrangement spatial et temporel de modules lui donnant un sens et une structure (Salazar et Valero 1995). Les frames résultent soit des phases d'échantillonnage et de quantification d'un signal vidéo analogique, soit de programmes informatiques générant des images de synthèse. La vidéo se caractérise plutôt par les paramètres de fréquence d'échantillonnage, de taille d'échantillon, de pas de quantification, de taux d'images, etc.

Une vidéo peut être présentée sous la forme de plusieurs conteneurs. Un format conteneur (« *Wrapper* » ou « *Containers* » en anglais) est un format de fichier spécifique aux documents audiovisuels. Il peut contenir divers types de données qui, vu leur taille, doivent être compressées à l'aide de codecs normalisés. Un codec désigne tout procédé capable de compresser ou de décompresser un signal analogique ou numérique. Ces

procédés peuvent être matériels ou logiciels. Les conteneurs les plus répandus sont MPEG (**M**oving **P**ictures **E**xperts **G**roup) et AVI (**A**udio **V**ideo **I**nterleaved) de Microsoft.

II.2. Annotation et métadonnée

Accéder au sens, au contenu de ces documents multimédias est une tâche complexe. Si l'on prend l'exemple d'un document audio, si l'on souhaite savoir quel est le locuteur, il faut mettre en place des algorithmes de reconnaissance difficiles à maitriser. Si on va plus loin et que l'on souhaite connaître le thème, on doit utiliser des systèmes ontologiques ad-hoc. Ainsi, l'annotation offre une véritable valeur ajoutée aux documents numériques explicitant, décrivant sa forme ou son contenu.

Selon W3C, une annotation est « *un commentaire, une note, une explication ou toute autre remarque externe qui peut être attachée à un document web ou à une partie de celui-ci* ». (Bringay et al. 2004) définissent l'annotation comme « *une note particulière attachée à une cible* ». Cette cible peut-être une collection de documents, un document, un segment de document (paragraphe, phrase, mot, image ou partie d'une image, etc.). D'un point de vue général, une annotation est une description du contenu documentaire.

La pratique de l'annotation peut être significative pour une personne dans un environnement particulier. Cependant, cette pratique est d'autant plus importante quand elle est réalisée dans un cadre collaboratif dans une communauté. Cette thématique à été abordée dans notre équipe par le biais de la modélisation et l'exploitation des annotations dites collectives (Cabanac et al. 2009). Ces travaux ont servi également pour une validation sociale des annotations collectives (Cabanac et al. 2010).

Pour caractériser une annotation, (Marshall 1998) considère trois dimensions. Une première dimension s'intéresse au rôle de l'annotation dans la communication avec les autres. Cette dimension définit le public auquel cette annotation est destinée. Une autre dimension reflète le cycle de vie d'une annotation. Dans ce cas, l'auteur indique dans quelle mesure son annotation peut se transformer en une « écriture » et s'intégrer dans le texte original du document. Ceci détermine la durée de vie d'une annotation. La troisième dimension concerne la forme que peut prendre l'annotation. En effet, une annotation peut être *informelle*, également qualifiée de « *cognitivement sémantique* » par (Zacklad et al. 2003), ou *formelle* et dans ce cas elle est structurée puisqu'elle est basée sur une formalisation rigoureuse. (Prié et Garlatti 2004) propose de distinguer l'annotation et la métadonnée. Ils estiment qu'une métadonnée est attachée à une ressource identifiée en tant que telle, a une pertinence *a priori* et saisie suivant un schéma alors qu'une annotation est plutôt identifiée au sein de cette ressource et écrite au cours d'un processus d'annotation.

Dans nos travaux de thèse, nous nous intéressons à ces annotations dites formelles. Ces annotations sont généralement traduites par des métadonnées, car elles fournissent des informations additionnelles sur les données. En effet, une métadonnée (du Grec, « *méta* » : ce qui dépasse, englobe) est une donnée à propos d'une autre donnée. En sciences de l'information, les métadonnées sont « *des ensembles de données structurées décrivant des ressources physiques ou numériques, ou, sur un plan plus fonctionnel, de l'information*

structurée qui décrit, explique, localise la ressource et en facilite la recherche, l'usage et la gestion » (NISO 2004). (Bechhofer et al. 2002) considèrent que l'annotation n'est qu'un moyen particulier d'associer une métadonnée à une ressource sur le web, insistant ainsi sur la fonction et pas sur la notion.

III. Structuration de document

L'intégration des médias et des annotions, notamment les métadonnées, a rendu le document numérique plus complexe avec un contenu difficilement accessible et manipulable. La structuration de ces documents est une bonne solution pour organiser, agencer et par conséquent faciliter l'interrogation du contenu documentaire.

Le mot structure vient du latin « *struere* » qui signifie construire et agencer. Selon le dictionnaire le Larousse, une structure est la « *manière dont les parties d'un ensemble concret ou abstrait sont arrangées entre elles* ». Une structure documentaire reflète donc l'idée d'une entité organisée en éléments. Cette organisation peut être partielle (document semi-structuré) ou complète (document structuré).

Dans ce qui suit, nous décrivons ces différents niveaux de structuration du document. Ensuite, nous détaillons les différentes représentations et les typologies de structures possibles. Enfin, nous présentons les standards documentaires les plus cités dans la littérature.

III.1. Du document non structuré au document structuré

Dans la littérature, trois classes de documents peuvent être distinguées. Selon le niveau de structuration de leur contenu, on peut recenser : (1) les documents non structurés, (2) les documents semi-structurés et (3) les documents structurés.

(1) Les documents *non structurés*, appelés encore documents « *plats* », sont des documents qui n'intègrent aucune marque explicite d'élément de structure. Ainsi, dans ces documents, on ne retrouve pas la disposition et l'emplacement des informations. Le document est présenté comme une suite de caractères (plein texte). Selon (Bringay et al. 2004), un document plat est un document pour lequel ni le lecteur ni le système n'est capable de décrire ou détecter une structuration de son contenu. (Tannier 2006) considère que tout texte ne comportant pas plus que des marquages de ponctuations (virgules, points de toutes sortes, etc.) et/ou de présentation (passages en lignes, espacements divers, énumérations, etc.) est un document plat.

(2) Les documents *semi-structurés* sont des documents caractérisés par leur structure implicitement déclarée, irrégulière, non rigide, inconnue *a priori*. Cette structure peut être éventuellement incluse dans le document, par des attributs implicites, et des éléments ne répondant pas à un typage strict. Selon (Debarbieux 2005), cette structure doit être définie en l'inférant *a posteriori*. Les documents dits semi-structurés ne fournissent pas d'indications concernant la disposition des informations décrites. Dans ce type de document, l'ordre des informations n'a généralement pas d'importance. On ne parle alors plus de texte, mais de données. Ces données n'ont habituellement aucune signification

intrinsèque, c'est-à-dire qu'il est impossible de les considérer sans examiner la structure dans laquelle elles sont inscrites (Tannier 2006). (Fuhr et Großjohann 2001) évoquent deux approches pour appréhender un document semi-structuré selon le besoin et la manière d'aborder la recherche d'information. La première approche, orientée document, considère le document comme un texte dont la principale finalité est la lecture. Dans ce cas, les balises servent à fournir des informations relatives à la structure (paragraphe, section, etc.) et/ou la forme (caractères italiques, gras, etc.). La deuxième approche, orientée donnée, considère le document comme une source de données. Ainsi, le document est utilisé afin de représenter et échanger ces données.

(3) Les documents *structurés* sont des documents qui possèdent une structure explicitement déclarée et connue *a priori*. Celle-ci permet d'identifier les différents éléments ainsi que leur rôle. Dans ce type de documents, la structure fournit des informations relatives à l'emplacement et à l'organisation des éléments.

En conclusion, un document structuré ou semi-structuré intègre des informations additionnelles, telles que des balises qui renseignent sur les différents éléments de structure qu'il peut contenir et éventuellement des annotations à caractère sémantique, etc. Un document est vu comme non structuré *a priori,* mais peut devenir structuré après plusieurs analyses appelées « *élicitations* » à condition qu'il contienne tous les éléments nécessaires et que ceux-ci puissent être extraits.

III.2. Structures documentaires

Une structure documentaire permet de décomposer le contenu en unités élémentaires appelées *éléments*. L'agencement entre ces éléments peut être assuré par plusieurs types de relations (hiérarchiques, temporelles, spatiales, etc.). La diversité des types de relations est à l'origine de la diversité de nature de structures et des représentations associées.

III.2.1. Les différentes représentations de structures

Les diversités typologiques des relations impliquent des représentations structurelles différentes. Les relations non hiérarchiques (temporelles, spatiales) ne peuvent pas être représentées de la même façon que les relations hiérarchiques. Ainsi, nous distinguons quatre formes de représentations de structures à savoir : les listes, les arbres, les forêts, les graphes.

- Liste

La représentation sous forme de liste est la description structurelle la plus simple. Une représentation sous forme de liste ordonnée est considérée comme une succession d'éléments, chacun ayant un rang. Cet ordre séquentiel est bien adapté pour traduire par exemple, des documents plats ou des relations temporelles simples.

- Arbre

En plus de la relation d'ordre, ce type de représentation permet de prendre en compte les relations hiérarchiques. Il traduit les inclusions entre les éléments qui forment le

document. Le parcours préfixé de l'arbre permet de retrouver l'ordre des éléments de la liste. Une telle représentation offre plus de visibilité sur l'organisation générale d'un document.

- Forêt

Les représentations sous forme de liste et d'arbre imposent une relation d'ordre entre les différents éléments d'un même document. Cependant, il existe des éléments juxtaposés, indépendants, dont la position n'est ni déterminée par rapport à un ordre ni par rapport à une hiérarchisation particulière. C'est le cas par exemple d'une présentation vidéo qui accompagne le texte ou des photos qui peuvent apparaître à la fin de chaque page (ou bien des liens vers ces photos) sans être incluses forcément dans des paragraphes bien précis. La prise en compte de tels éléments transforme la structure d'une simple arborescence en une forêt.

- Graphe

La définition de plus d'une relation entre deux éléments, les relations de renvois et les références croisées sont des cas de figure qui peuvent exister à l'intérieur d'un document. Cependant, les types de représentations que nous avons définis précédemment ne peuvent pas traduire ces cas. De telles structures seront représentées alors sous forme de graphe. Par exemple, dans les paragraphes, nous pouvons trouver des renvois vers des articles, des auteurs, des légendes des photos, etc.

- Bilan

Les différentes représentations évoquées dans cette sous section sont en fait complémentaires, la représentation sous forme d'arbre vient pallier les limites de la représentation sous forme de liste, la représentation sous forme de forêt vient compléter la représentation sous forme d'arbre. La représentation sous forme de forêt prend en compte d'autres éléments qui ne pouvaient pas être décrits dans la structure arborescente. La représentation sous forme de graphe permet de traduire des relations logiques plus complexes comme pour les renvois ou les références.

Nous ne pouvons pas favoriser une représentation particulière, le choix dépend du type de documents utilisés. Pour un document simple, la représentation adéquate sera la représentation sous forme de liste. Pour un document plus complexe qui possède des renvois vers d'autres documents, la représentation sous forme de forêt est la plus adéquate. Ceci dit, la représentation arborescente semble la mieux placée entre richesse de description et simplicité d'utilisation.

III.2.2. Typologie des structures

La structuration d'un document consiste à identifier chacun des éléments qui le constituent. Une structure peut prendre plusieurs formes. Elle peut être considérée comme étant un ensemble d'éléments organisés hiérarchiquement (selon une organisation logique) et/ou un enchaînement temporel d'éléments (organisation temporelle) et/ou un agencement d'un ensemble d'objets (organisation physique), etc. Toutefois, certains types

d'organisations sont applicables à des médias spécifiques. Par exemple, une organisation temporelle est spécifique aux documents audiovisuels.

Nous présentons dans cette section, les structures les plus citées dans la littérature. Afin d'illustrer les différences entre ces structures ainsi que leurs spécificités, nous appuierons nos exemples sur un même document de base : « TéléJournal » (cf. Figure I.1). Ce document présente une séquence vidéo d'un journal télévisé et une description des thèmes abordés. Les exemples des différents types de structure seront représentés sous forme d'arborescence ou de graphe selon la nécessité.

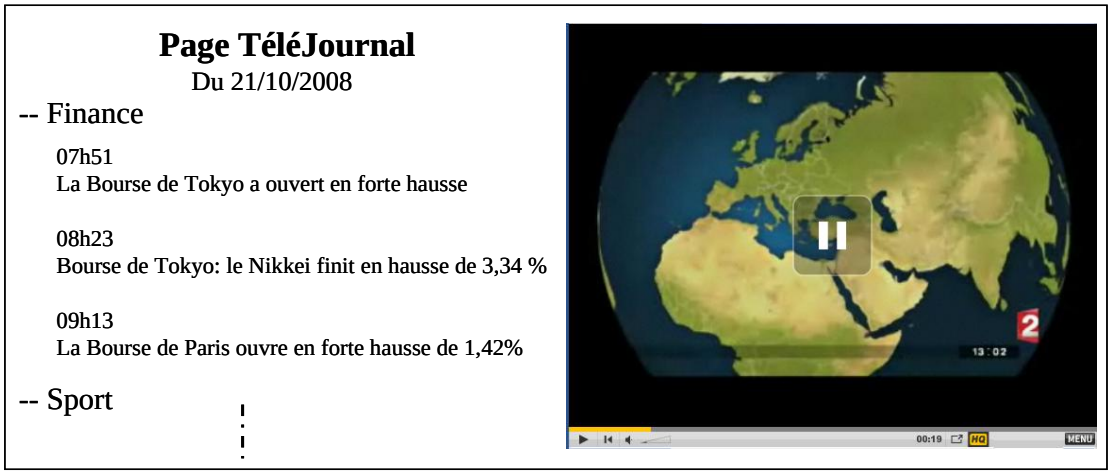


Figure I.1. Extrait d'un document « TéléJournal ».

III.2.2.1. Structure logique

La structure logique reflète « *l'organisation explicite d'abstractions logiques représentant des parties de document* » (Fourel 1998). Ces abstractions logiques correspondent généralement à des entités d'informations qui existent indépendamment les unes des autres du fait qu'elles contiennent suffisamment d'informations pour être compréhensibles. La structure logique permet un découpage de l'information d'un point de vue hiérarchique et logique selon un principe de décomposition plus ou moins fin. Ce mécanisme impose d'identifier de façon non ambiguë les granules d'information composant le document. L'organisation de cette structure est explicite et elle est définie *a priori*.

Selon (Roisin 1999), la structure logique des documents s'appuie sur trois entités :

- les éléments de base non décomposables qui constituent le contenu ;
- les éléments composites obtenus par composition d'éléments de base ou d'autres éléments composites ;
- les attributs qui peuvent être associés aux éléments pour leur adjoindre des informations supplémentaires.

La Figure I.2 présente la structure logique du document de base (Cf. Figure I.1). Ce document est constitué d'un « Titre », d'une « Date », d'un ou plusieurs « Thèmes » et

d'une séquence « Vidéo ». Chaque élément « Thème » est composé d'un élément « Titre » et d'un ou plusieurs éléments « Heure » et « Info ».

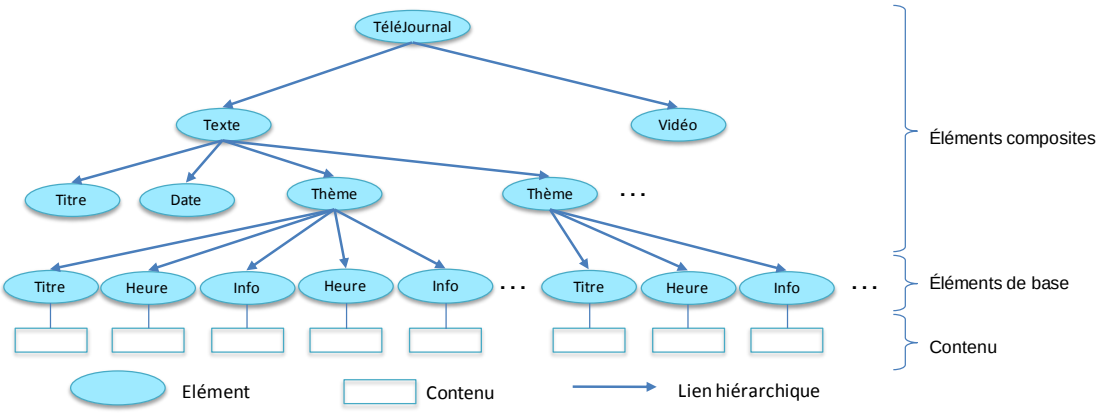


Figure I.2. Structure logique du document « TéléJournal ».

III.2.2.2. Structure physique

Le concept de structure physique est lié à la restitution du document sur un support physique (papier, écran, etc.). Cette structure permet un découpage de l'information suivant sa présentation. Ainsi, une structure physique décrit la mise en page d'un document et définit les différentes zones de ce document ainsi que leurs caractéristiques. Cette structure est traduite par un ensemble de règles de présentation tel qu'une succession de lignes, de paragraphes, de colonnes, de pages, de caractéristiques typographiques, etc. Ces règles sont spécifiées par pavés (ou blocs) d'information. Schématiquement, un bloc est représenté par une zone rectangulaire de taille et de coordonnées précises, destinée à organiser l'information. A un niveau d'abstraction plus fin, un bloc peut lui même être découpé en sous-blocs, chaque bloc élémentaire contenant un granule logique d'information homogène.

Tout comme la structure logique, la structure physique est présentée sous forme d'une arborescence de blocs. Sur un support papier, le découpage s'effectuera par exemple page par page, colonne par colonne, paragraphe par paragraphe, etc. (Cf. Figure I.3). Afin d'expliciter l'exemple, nous avons opté pour une représentation sous forme d'emboitement de bloc.

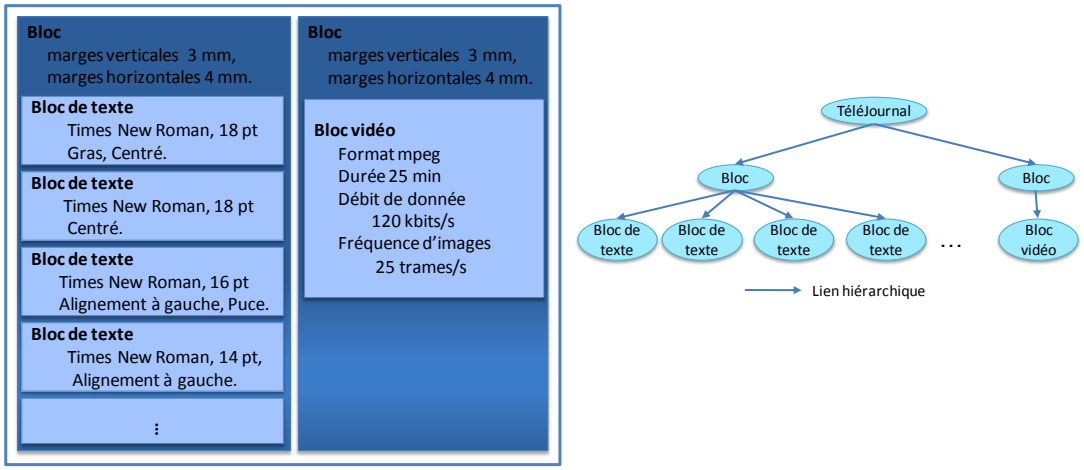


Figure I.3. Structure physique du document « TéléJournal ».

III.2.2.3. Structure sémantique

La structure sémantique reflète l'organisation de l'information contenue dans le document. Elle est définie au travers de la composition sémantique, représentant le sens d'un ou plusieurs éléments de la structure logique (Poullet 1997). Cette composition est traduite généralement par des métadonnées décrivant les éléments de la structure logique.

Figure I.4 présente un extrait de la structure sémantique du document « TéléJournal » en se focalisant sur la description du passage vidéo intégré. Ce passage est composé d'un ensemble de séquences. Chaque séquence est décrite par une bande audio et un ensemble de frames. Dans cet exemple, la sémantique est fournie au travers l'intégration des métadonnées « Locuteur » pour la bande audio et « Région », « Couleur » et « Mot Clé Img » pour chaque frame. Cette description est susceptible d'être affinée selon le besoin et l'usage.

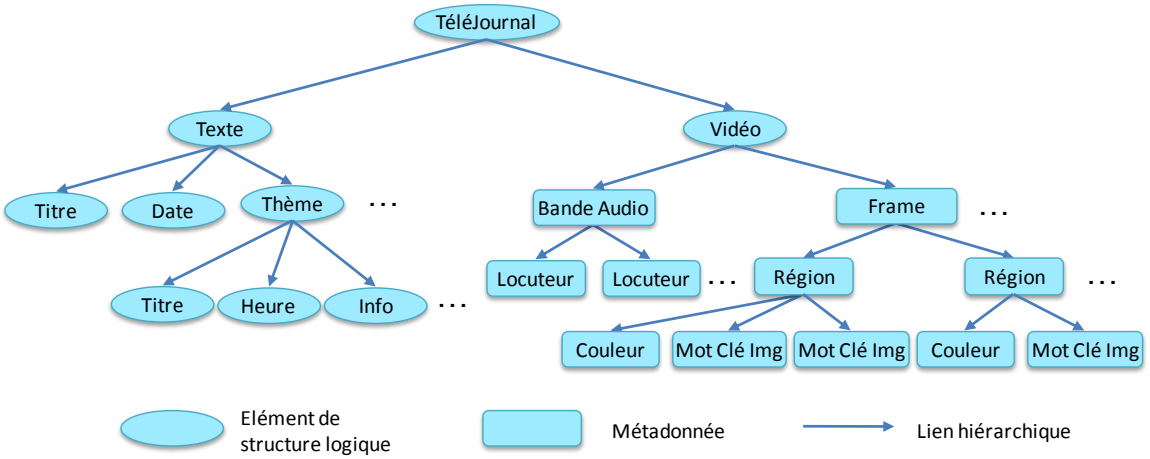


Figure I.4. Structure sémantique du document « TéléJournal ».

III.2.2.4. Structure hypermédia

La structure hypermédia (Auffret et al. 1999) correspond à l'organisation inter et intra média dans un ou plusieurs documents. Cette organisation est traduite par un ensemble de liens qui permettent la navigation inter ou intra média en spécifiant de façon classique une ancre de départ, une ancre d'arrivée et le type du lien. Différents types de liens sont possibles : renvoi, référence, annotation, synchronisation, etc.

Dans ce contexte, la structure la plus utilisée est la structure hypertexte (Julien 1988) (Aguiar et Beigbeder 2004). Cette structure, comme son nom l'indique, est relative au média texte. Elle représente les différents liens dans un document textuel. Les pages web sont des exemples typiques des documents incluant une structure hypertexte.

Dans la Figure I.5, nous nous limitons à la présentation de l'organisation intra document. Nous décrivons ainsi les relations de synchronisation entre les médias incorporés dans le document « TéléJounal ». A titre d'exemple, ces synchronisations peuvent concerner les éléments textuels « info » avec la bande audio ou les métadonnées « région » de chaque frame avec la bande audio. De telles synchronisations permettent de déterminer, par exemple, les segments qui correspondent à chaque fragment textuel de l'élément « info » dans la bande audio.

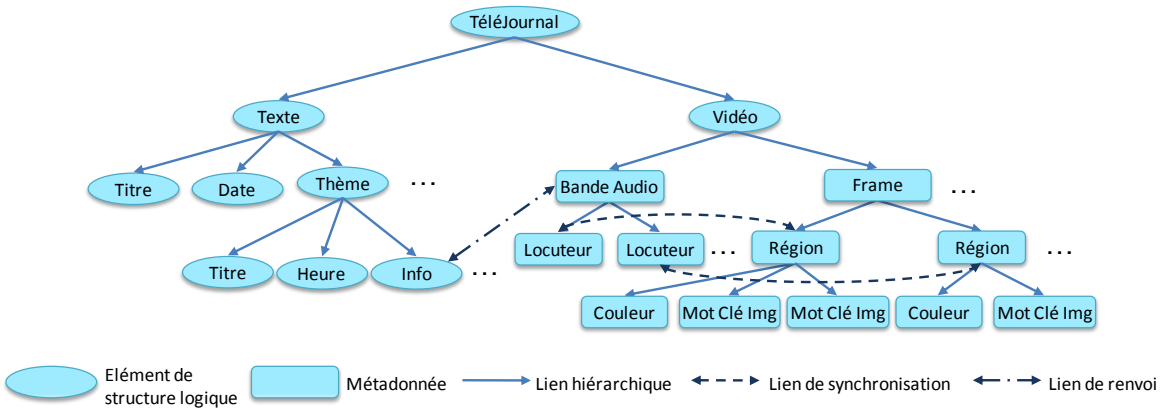


Figure I.5. Structure hypermédia du document « TéléJournal ».

III.2.2.5. Structure spatiale

L'une des principales caractéristiques des documents multimédias, et en particulier des documents audiovisuels et images, par rapport aux documents textuels classiques est leur dimension spatiale. Cette dimension exprime les contraintes d'ordonnancement des différentes parties d'un document lors de sa présentation. Elle permet donc de définir la taille des différentes zones, les superpositions, les juxtapositions, etc.

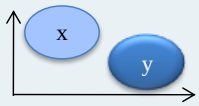
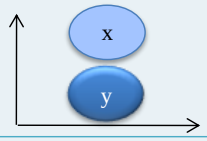
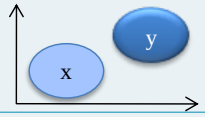
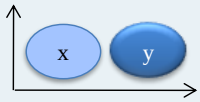
Cette dimension est constituée à partir de relations spatiales. Ces relations sont définies notamment dans les domaines de la représentation de connaissances (Frank 1992) ; (Egenhofer et Franzosa 1991) et le raisonnement spatial (Frank 1996) ; (Egenhofer et al. 1994). La prise en compte de telles relations permet essentiellement la

formulation de requêtes spatiales et la vérification des contraintes de cohérences des bases de données spatiales.

Les propriétés spatiales permettent de situer les objets les uns par rapport aux autres. Ces propriétés peuvent être classées selon deux catégories : les relations cardinales, appelées encore directionnelles, pour décrire l'espace 2D et les relations topologiques pour décrire les voisinages entre les objets (recouvrement, inclusion, etc.).

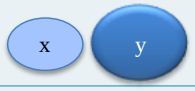
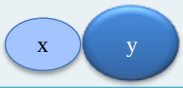
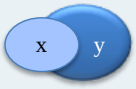
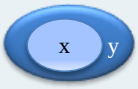
- l'espace vectoriel regroupe les relations cardinales fondées sur des notions géographiques (« Nord », « Sud », « Ouest », « Est »). Ces relations sont d'une grande utilité pour le repérage de certains objets par rapport à d'autres. Elles montrent une orientation particulière de ces objets par rapport à d'autres. Par exemple, elles sont utilisées afin de lier deux points ou deux régions d'une image (Papadias et Theodoridis 1997).
  Le Tableau I.1 présente huit relations cardinales. Une neuvième relation qui traduit l'égalité n'est pas présentée. Selon (Lopez-Ornelas 2005), la relation « égal » entre deux objets est une alternative qui ne pourra jamais se présenter, car elle va à l'encontre de la définition de la segmentation ;

Tableau I.1. Les huit relations cardinales.

Relation cardinale	Représentation graphique	Relation cardinale inverse
Nord-Ouest (x,y)		Sud-Est (y,x)
Nord (x,y)		Sud (y,x)
Nord -Est (x,y)		Sud-Ouest (y,x)
Ouest (x,y)		Est (y,x)

- l'espace topologique exprime les notions de connectivité, d'orientation, d'englobement et des relations de contiguïté entre les objets spatiaux (Egenhofer 1994) ; (Papadias et Theodoridis 1997). Ainsi, les relations topologiques sont décrites par les positions des objets les uns par rapport aux autres comme par exemple les relations « chevauche », « touche », etc. Des exemples de relations topologiques sont présentés dans le Tableau I.2.

Tableau I.2. Exemples de relations topologiques

Relation topologique	Représentation graphique	Définition
Disjonction (x,y)		Deux objets x et y distincts
Adjacence (x,y)		Deux objets x et y se touchant par leurs contours sur un ou plusieurs points
Chevauchement (x,y)		Deux objets x et y se superposant partiellement sur une ou plusieurs parties
Inclusion (x,y)		Deux objets x et y s'emboîtant l'une dans l'autre

(Charhad et Quénot 2004) proposent d'autres types de relations spatiale. Ces relations sont basées sur la distance entre les entités décrites dans la dimension spatiale. Les relations de distance peuvent être par exemple « près », « loin », etc.

La Figure I.6 présente un extrait de la structure spatiale relative à un frame de la vidéo incorporée dans le document « TéléJounal ». Ce frame fait partie d'une scène de conversation entre le présentateur du téléjournal et son invité. Ces deux locuteurs sont présentés sous forme de smiley. La description spatiale de ce frame permet d'identifier les relations entre les objets et les deux personnages figurant dans ce frame ainsi que leurs coordonnées cartésiennes. Ainsi, la Figure I.6 schématise les deux locuteurs au travers des deux smileys, un objet (Table) et les relations spatiales entre ces trois métadonnées. Nous avons utilisé des flèches orientées afin de traduire le sens de lecture des relations.

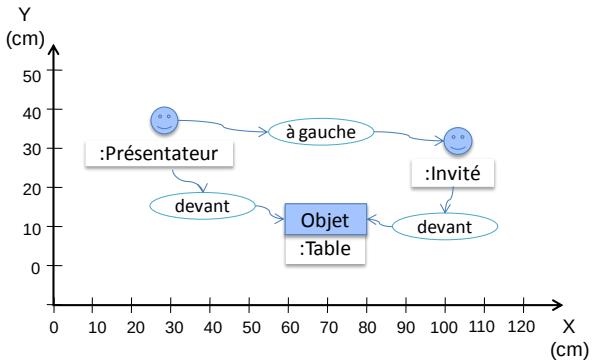


Figure I.6. Extrait de la structure spatiale d'un frame appartenant à la vidéo du document « TéléJournal ».

III.2.2.6. Structure temporelle

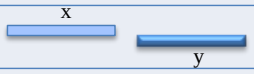
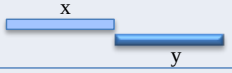
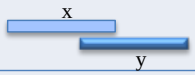
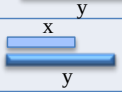
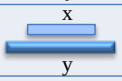
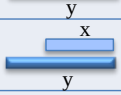
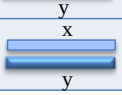
Les documents audio et audiovisuel sont construits autour d'un axe temporel. Cet axe met en jeu les synchronisations et les enchaînements des différents éléments du contenu. La structure temporelle permet de traduire et décrire ces synchronisations et ces

enchaînements. Plusieurs études ont été menées afin de modéliser cette dimension (Allen 1983) ; (Vilain et al. 1986) ; (Vazirgiannis et al. 1998).

L'analyse des relations temporelles relatives aux documents a favorisé l'apparition de plusieurs axes de recherches. (Beigbeder 2004) ; (Metzger et Lallich-Boidin 2004) s'intéressent à la dimension temporelle qui concerne l'histoire et le contexte du document dans différents univers. Le temps de perception du document qui définit le temps de défilement des médias ainsi que leur synchronisation est traité dans (Lalanne et Ingold 2004) ; (Lefèvre et Sèdes 2004). (Nanard 2004) a étudié le temps de transaction et de validité d'un document au travers de la prise en compte de ses versions.

Dans ce qui suit, nous allons nous intéresser aux relations temporelles entre fragments. Ces fragments sont des segments ou des séquences audio ou audiovisuelles. Les relations d'Allen (Allen 1991) permettent de structurer le contenu d'une séquence audiovisuelle en se basant sur les informations temporelles. Allen identifie un ensemble complet de relations temporelles qui peuvent exister entre deux intervalles. Il propose treize relations parmi lesquelles douze sont asymétriques (« Précède », « Rencontre », « Chevauche », « Débute », « Pendant », « Termine » et « Équivalent »). Le Tableau I.3 présente ces relations avec leur représentation graphique.

Tableau I.3 Les relations temporelles d'Allen.

Relation Temporelle	Représentation graphique
Précède (x,y)	
Rencontre (x,y)	
Chevauche (x,y)	
Débute (x,y)	
Pendant (x,y)	
Termine (x,y)	
Equivalent (x,y)	

La Figure I.7 présente une structure temporelle de la vidéo du document « TéléJournal ». Cette structure montre en particulier la succession des séquences, des frames, etc. Cette succession se traduit dans les relations d'Allen par des relations de types « Précède » ou « Rencontre ». Pour des raisons de clarté, nous représentons les éléments qui se succèdent sur un même de temps sans matérialiser les relations. Cette structure assure également l'agencement et la synchronisation entre la bande audio et les différents frames. Par exemple, des relations de synchronisations sont établies entre les métadonnées locuteurs extraites de la bande audio et les métadonnées locuteurs extraites de différents frames et représentées dans la Figure I.7 par des smileys.

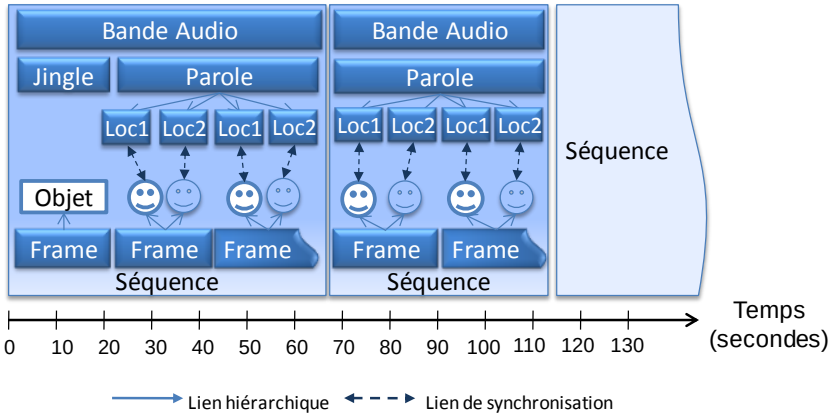


Figure I.7. Extrait de la structure temporelle de la vidéo du document « TéléJournal ».

III.2.2.7. Synthèse

La description structurelle d'un document consiste à identifier chacun des fragments qui le constituent. Cette description peut prendre plusieurs formes et traiter plusieurs aspects. En commençant par le niveau de description le plus utilisé, la structure logique décrit le rôle et la nature de chaque élément du document ainsi que l'ensemble des liens hiérarchiques qui les relient. La structure physique correspond à l'organisation de l'information sous forme de bloc sur un support de présentation. Elle définit ainsi les différentes caractéristiques des blocs qui forment le document. La bonne conception d'une telle structure assure une bonne lisibilité de la structure logique : chaque élément de la structure logique est bien caractérisé au niveau d'un bloc de la structure physique. Ceci amène à un découpage cohérent (pas de chevauchement entre les deux structures) ce qui permet parfois de confondre les deux structures. Cependant, il est nécessaire de distinguer ces deux types d'organisation, car la définition, la manipulation et l'accès aux éléments de ces deux types de structures sont différents et admet un rôle différent. La structure sémantique offre une couche supplémentaire à l'organisation logique. En effet, elle permet de présenter le sens des contenus documentaires. Elle établit une image structurée de l'information contenue dans le document. Les documents multimédias sont aussi caractérisés par une dimension hypermédia. Cette dimension reflète l'organisation inter et intra média. En nous focalisant sur des médias autres que le texte, nous pouvons distinguer des dimensions particulières. En particulier, nous citons la structure spatiale qui exprime les contraintes d'ordonnancement des différentes parties d'une image ou d'un frame appartenant à une séquence vidéo et la structure temporelle qui permet de décrire la synchronisation dans le temps des descripteurs d'un segment audio ou d'une séquence vidéo. Enfin, d'autres structures liées uniquement à des domaines d'études bien spécifiques des documents peuvent être évoquées (linguistique, de discours, etc.). Ces structures peuvent être amenées à coexister.

La Figure I.8 illustre la coexistence des différents types de structures présentées dans un même document. Nous considérons que la structure physique et la structure logique sont des structures de base d'un document du fait qu'elles traduisent les choix de l'auteur. Un document est composé d'une ou plusieurs structures de base et d'un d'ensemble de

blocs de contenu. Les structures de base permettent d'organiser les blocs de contenu. Les interactions, les référencements et les synchronisations qui peuvent avoir lieu entre les blocs (notamment de média différent) d'un ou de plusieurs documents et permettent de définir les structures hypermédia. Un bloc de donnée peut être décrit au travers des métadonnées. Cette description offre une dimension sémantique qui peut être agrégée au document. Au niveau de cette structure sémantique, on peut retrouver zéro ou plusieurs structures spatiales et temporelles.

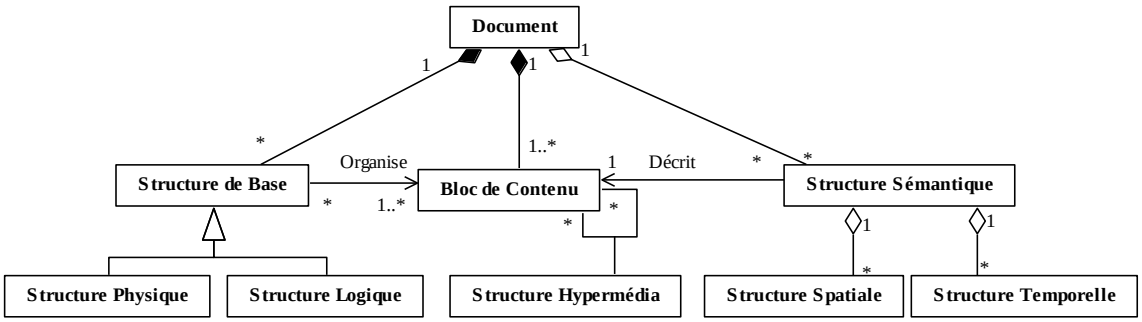


Figure I.8. Modèle de coexistence de structures dans un même document.

III.3. Document structuré et standards

Indépendamment du niveau et de la nature de leur structuration, les documents numériques doivent être décrits et représentés via des standards essentiels à leur exploitation et à leur échange. En effet, un standard est un ensemble de règles qui sont reconnues par tous les types de matériel, de systèmes d'exploitation et les applications associées (traitement de texte, tableur, visualiseur HTML). Ces règles vont assurer le codage des informations.

Les standards assurent trois fonctions :

- la première concerne l'intégrité : les standards permettent de respecter les pensées de l'auteur au travers de la matérialisation ;
- la deuxième est la pérennité : c'est-à-dire la préservation des informations produites par l'auteur pour des usages ultérieurs ;
- la troisième est relative à l'interopérabilité : en effet, un standard doit assurer l'accessibilité à l'information par tous et partout quel que soit l'environnement physique (matériel) ou logique (logiciel) utilisé.

La diversité des contenus documentaires a favorisé l'avènement de plusieurs standards : des standards de présentation et des standards de descriptions des données. Dans ce qui suit, nous présentons les standards les plus cités dans la littérature et les plus utilisés en pratique.

III.3.1. Standards de présentation de données

Les standards de présentation contribuent à la mise en place d'une gestion des structures documentaires afin d'homogénéiser la présentation du contenu et d'améliorer sa lisibilité. Ainsi, ces standards ont pour objet de faciliter la consultation et l'échange des informations via leur structuration. Dans cette section, nous décrivons les principaux standards (SGML, HYTIME, HTML et XML, XHTML, SMIL) qui ont marqué et marquent toujours l'histoire du document électronique.

III.3.1.1. Le standard SGML

SGML[1] (**S**tandard **G**eneralized **M**arkup **L**anguage) permet une structuration de l'information à l'aide de balises. Une balise désigne une marque particulière ajoutée à un texte afin d'en déduire sa structure ou le format dans lequel il sera édité (Goldfarb 1981). En octobre 1986, SGML a été adopté officiellement comme standard international (ISO-8879 1986).

- Principe de SGML

Le standard SGML permet de définir des classes de documents, c'est à dire des documents ayant la même structure logique, et ce, indépendamment de leurs formats d'édition. Cette structure logique est définie dans la DTD. Cette définition, sous la forme d'une arborescence, indique tous les éléments que peut contenir une classe de documents SGML et les contraintes d'organisation. De ce fait, un document SGML comprend :

(1) un ensemble de déclarations où sont précisées les caractéristiques SGML utilisées telles que la version, le jeu de caractères utilisé, etc. (Cf. Figure I.9) Cette partie assure l'adaptation des documents SGML à leurs domaines d'application en choisissant une syntaxe concrète et en activant les fonctionnalités optionnelles. Une déclaration est une partie optionnelle dans un document SGML. En son absence, SGML applique une déclaration par défaut.

```
<!SGML  "ISO 8879:1986 (WWW)"
    CHARSET
          BASESET  "ISO Registration Number 177//CHARSET
                    ISO/IEC 10646-1:1993 UCS-4 with
                    implementation level 3//ESC 2/5 2/15 4/6"
         DESCSET 0       9       UNUSED
                 9       2       9
                 11      2       UNUSED
                 13      1       13
                 14      18      UNUSED
                 32      95      32
                 127     1       UNUSED
                 128     32      UNUSED
                 160     55136   160
                 55296   2048    UNUSED  -- SURROGATES --
                 57344   1056768 57344
CAPACITY        SGMLREF
                TOTALCAP        150000
                GRPCAP          150000
```

[1] http://www.w3.org/MarkUp/SGML/

```
                ENTCAP          150000
SCOPE   DOCUMENT
SYNTAX  PUBLIC "ISO 8879:1986//CAPACITY Reference//EN"
FEATURES
  MINIMIZE    DATATAG  NO    OMITTAG  YES    RANK     NO    SHORTTAG  YES
  LINK        SIMPLE   NO    IMPLICIT NO     EXPLICIT NO
  OTHER       CONCUR   NO    SUBDOC   NO     FORMAL   YES
  APPINFO NONE>
```

Figure I.9. Exemple de déclaration SGML[2].

(2) une DTD « **D**éfinition de **T**ype de **D**ocument » qui décrit la structure logique et exprime l'organisation des différents éléments d'information. Un document SGML n'est dit valide que s'il est associé a une DTD et s'il respecte les contraintes qui y sont définies.

La DTD doit décrire les balises qui limitent les différents composants, ainsi que les règles d'utilisation des balises (l'organisation hiérarchique des différents éléments d'une DTD). L'interprétation d'un document SGML nécessite donc la connaissance de la DTD. Les auteurs de documents utilisent ensuite cette DTD pour rédiger les documents, au moyen d'éditeurs de texte.

La Figure I.10 montre une DTD qui correspond au modèle de structure générique d'une classe de documents de type Article. Cet article est qualifié par l'attribut VERSION (numéro de la version courante) et est composé d'un « TITRE », d'un ou plusieurs « AUTEURS », d'une « DATE », d'un « Résumé » et d'un « Corps » obligatoire. Ce dernier peut être lui-même composé d'une ou de plusieurs « SECTIONS ».

```
<!DOCTYPE Article [
<NOTATION Tiff System "TIFFVIEW.EXE">
<!ELEMENT Article -- (Titre, Auteur, Date, Résumé, Corps)>
<!ATTLIST Article version CDATA "1">
<!ELEMENT Titre -- (#PCDATA)>
<!ELEMENT Auteur -- (Nom*)>
<!ELEMENT Nom -- (#PCDATA)>
<!ELEMENT Date -- (#PCDATA)>
<!ELEMENT Résumé -- (#PCDATA)>
<!ELEMENT Corps -- (section+)>
<!ELEMENT Section -- (#PCDATA)>
<!ATTLIST Section Nom CDATA>
]>
```

Figure I.10. Exemple de DTD d'un document SGML.

(3) un contenu constitue le corps du texte. Il véhicule l'information constituant la « raison d'être » du document. Ce contenu est structuré via des balises qui y ont été insérées pour délimiter la structure logique du texte. Cette structure doit être conforme à la DTD. Généralement, ces noms figurent entre des marques inférieure « < » / supérieure « > » (<nom>) et les balises viennent par paires afin de marquer le début et la fin.

La Figure I.11 est un document SGML balisé suivant la DTD définie précédemment (Cf. Figure I.10), on parle d'instance SGML.

```
<Article Version = "3">
```

[2] http://www.w3.org/TR/REC-html40/sgml/sgmldecl.html

```
        <Titre> ANALYSE MULTIDIMENSIONNELLE DES DOCUMENTS MULTISTRUCTURES
        </Titre>
        <Auteur>
                <Nom> K. Djemal </Nom>
                <Nom> C. Soulé-Dupuy </Nom>
                <Nom> N. Vallès-Parlangeau </Nom>
        </Auteur>
        <Date> 31/04/2009 </Date>
        <Résumé> Avec l'émergence de la numérisation, le besoin d'outils
        de traitement automatique des documents ...
        </Résumé>
        <Corps>
                <Section Nom = "introduction"> Quelle que soit la nature
                des documents circulant dans l'entreprise, l'information
                qui leur est associée ainsi que leur contenu sont
                utilisés par ...
                </Section>
                <Section Nom = "état de l'art"> La problématique de la
                multistructuralité des documents a attiré l'attention ...
                </Section>
                ...
        </Corps>
</Article>
```

Figure I.11. Exemple d'instance SGML.

SGML permet l'échange et la réutilisation de textes électroniques tout en préservant le contenu (information brute) et la structure logique (l'organisation des données). Un des intérêts de ce standard est que la structure logique est indépendante de la présentation de l'information et donc des moyens de restitution. Ainsi, il est possible, à partir du même fichier, de réaliser des présentations visuelles différentes.

De plus, le fait de pouvoir spécifier des règles d'utilisation des balises dans une DTD est un aspect très important de SGML. En effet, ceci permet d'imposer une uniformité aux documents d'un même type rendant plus aisé l'exploitation de l'information contenue dans les documents d'une même classe ou DTD. On note également que le fichier SGML ne contient que du texte ASCII, ce qui permet d'assurer la conservation à long terme des documents.

Finalement, on peut conclure que le standard SGML est bien adapté à la constitution et à la gestion des bases de documents structurés. En effet, la seule connaissance des DTD suffit pour mettre en place des outils pour le traitement automatique des documents.

- Les standards HyTime et DSSSL

Les documents SGML tels qu'ils sont conçus ne peuvent pas être des documents hypertextuels ou des hyperdocuments. Le standard HyTime (**Hy**permedia/**Time**-based Structuring Language) (ISO-10744 1997) réalise l'intégration de documents hypertextuels, à composante temporelle, en permettant d'associer à certaines balises une sémantique spécifique. HyTime est un modèle conceptuel pour l'échange de documents hypermédia par la réalisation d'une méta-DTD qui relie des DTD élémentaires entre elles. Ainsi, un hyperdocument est un ensemble de documents élémentaires reliés par un réseau de liens selon des mécanismes de synchronisation en accord avec la description logique fournie par

SGML. Il est possible, par exemple, de créer un lien vers un point précis dans un fichier sur le web. Hytime est donc une extension compatible avec SGML.

Quant à la structure physique d'un document, elle est prise en charge par le standard DSSSL (**D**ocument **S**tyle **S**emantics and **S**pecification **L**anguage) (ISO-10179 1996). C'est un format de données qui permet de spécifier le style et la présentation d'un document, c'est-à-dire sa structure physique indépendamment du support (papier, CD-ROM, BD documentaire, etc.). Le modèle DSSSL est basé sur la notion de zones contenant des objets divers tels que des graphiques, du texte, des formules, d'autres zones et pouvant se superposer indépendamment de leur contenu. A chaque zone est associé un ensemble de règles de mise en page qui assurent le lien avec la structure logique définie par SGML.

III.3.1.2. Le standard HTML

HTML (**H**yper**T**ext **M**arkup **L**anguage) est le langage le plus connu, car il est largement utilisé pour décrire l'information mise à disposition des utilisateurs du Web. Ce langage, issu de la famille SGML est un langage balisé. Les balises utilisées à ce niveau sont des balises de présentation et de mise en forme. En fait, tous les documents HTML sont conformes à une seule DTD : la DTD HTML (Raggett et al. 1999) consultable sur le site du W3C.

HTML est un langage qui comporte trois avantages principaux : il est facile à apprendre et à comprendre ; les liens hypertextes sont très faciles à mettre en place entre la source et la cible (balise <A …>) et enfin, la faible quantité de balises facilite l'intégration de la DTD HTML dans des logiciels de navigation. Cependant, l'absence de structure logique explicite cause d'énormes problèmes pour le traitement automatique et la recherche d'informations (taux de rappels inacceptables). De plus, le mélange de balises contrôlant l'apparence à celles décrivant la structure du document rend la réutilisation du texte très difficile. A tout ceci, il faut ajouter l'inexistence de contrôle de cohérence vis-à-vis de l'utilisation des balises HTML induisant ainsi une utilisation anarchique des balises.

En conclusion, nous pouvons affirmer que la DTD HTML n'est pas utilisée avec la rigueur qui serait nécessaire à une extraction simple de la structure logique des documents Web. Dans ce contexte, le langage XHTML (eXtended HTML) (Pemberton et al. 2000) permet d'éviter les problèmes que posent HTML. C'est la dernière version de HTML 4.0 qui se rapproche du langage XML (décrit dans la section suivante). En effet, cette recommandation du W3C utilise les mêmes balises que HTML avec une syntaxe qui a subi certaines modifications dans le but de faciliter la transition vers XML. XHTML présente l'avantage d'accroître la pérennité des documents, d'utiliser des outils développés pour XML et de diffuser des documents vers des applications clientes supportées par des matériels de types nouveaux (téléphone WAP, terminaux mobiles, etc.).

III.3.1.3. Le standard XML

Le standard XML (e**X**tensible **M**arkup **L**anguage) (Bray et al. 2000) est un langage de représentation, issu des travaux du W3C, adapté pour l'échange de documentation structurée provenant de sources hétérogènes. Issu de SGML et d'une capitalisation des

acquis réalisés sur ce dernier, il fournit un format de fichier pour représenter les données, un schéma pour décrire la structure des données, et un mécanisme pour étendre et annoter le langage HTML avec des informations sémantiques. XML reprend les caractéristiques essentielles de SGML en laissant de côté celles jugées trop complexes et qui rendaient SGML difficile d'usage.

Dans XML, la séparation de la description structurelle des documents et de la description de leurs présentations physiques offre d'énormes avantages en terme de facilités d'échanges et de production coopérative de documents et surtout une possibilité énorme en terme de traitement automatisé de documents. La puissance de XML réside dans le fait que toutes les informations encodées avec la DTD sont alors très fortement structurées, adaptées à l'applicatif et très facilement manipulables par un outil de visualisation. En effet, l'utilisateur peut définir ses propres balises en fonction de ses besoins offrant une meilleure définition ainsi qu'une meilleure interprétation des documents par les applications qui les restituent. XML apporte également une souplesse au niveau des liens hypertextes dont l'utilisation est plus aisée et qui sont plus complets que ceux de SGML.

Le standard XML est voué à devenir le standard d'échange d'informations sur Internet, mais est également utilisable dans d'autres cadres (par exemple, un document XML peut contenir une base de données Oracle, des tableaux de bord complexes, etc.).

La structure d'un document XML est la suivante :

- un prologue qui est un ensemble de déclarations dont la présence est facultative, mais conseillée ;
- un arbre d'éléments qui forment le contenu proprement dit du document ;
- des commentaires et des instructions de traitement dont la présence est facultative et qui peuvent être, soit dans le prologue, soit dans l'arbre d'éléments.

❑ Principe

XML (Michard 1998) utilise des balises et des attributs comme HTML mais laisse à l'utilisateur l'entière possibilité de définir son propre jeu de balises dans le but de personnaliser la structure des données.

Le standard XML a été conçu pour être utilisé de deux manières distinctes :

- d'une part, on peut utiliser une DTD (comme pour les documents SGML) qui spécifie la structure logique d'une classe de documents (structure générique) et définit les balises à utiliser pour identifier les entités de cette structure. Alors, le document XML faisant appel à cette DTD et s'y conformant est dit document « valide » ;
- d'autre part, un document XML peut être écrit sans DTD, il est alors dit document « bien formé ». Dans ce cas, le document doit respecter la syntaxe du standard XML. Ainsi, il ne peut comporter aucune ambiguïté dans le balisage : tous les éléments doivent posséder une balise ouvrante et fermante, ne peuvent être vides et

les attributs doivent être entre guillemets. Ainsi, un document valide est avant tout bien formé.

La définition des hyperliens dans un document XML se fait à l'aide du langage XLL. L'affichage ou l'impression des données XML impose de définir la façon dont les données s'affichent ou s'impriment, soit par programme, soit en utilisant les feuilles de styles XSL.

- XLL et XSL

XLL (e**X**tensible **L**ink **L**anguage) est le langage de description des liens hypertextes en XML. XLL étend les hyperliens définis par HTML en permettant, entre autres, des liens qui pourront être bidirectionnels ou gérés dans un fichier extérieur au document lui même, des liens vers des cibles sur Internet non balisées au préalable. Des attributs ajoutés aux liens permettent de définir le type de lien (lien vers une définition, lien extérieur, etc.). Ainsi, un lien XML peut être une URL, comme dans HTML ou un XPTR (e**X**tended **P**oin**T**e**R**). Un XPTR (DeRose et al. 2001) ressemble à une URL mais il permet d'exprimer des liens plus complexes car au lieu de pointer sur des documents du Web, il pointe sur des éléments de données au sein d'un fichier XML.

XSL (e**X**tensible **S**tyling **L**angage) (Adler et al. 2001) est le langage utilisé pour la définition des feuilles de style associées aux documents XML. C'est le fichier XSL qui permet de définir les propriétés typographiques et graphiques du document (tel élément XML doit être affiché avec telle police de caractères, de telle couleur, etc.). Une feuille de style XSL se compose de règles de construction décrivant comment les éléments du fichier XML doivent être transformés vers le format de sortie qui est par exemple un document HTML, un texte ASCII.

Les règles de construction se divisent en deux parties :

- la partie « modèle » qui identifie le type des éléments sources ;
- la partie « traitement » qui décrit ce qu'il faut faire avec les éléments qui correspondent à la partie « modèle ».

- Exemple de document XML

Nous présentons, dans la Figure I.12, un exemple de document valide (avec DTD associée interne). Il correspond à la description d'un document de type article scientifique déjà présenté en SGML (Cf. Figure I.11).

```
<?XML VERSION="1.0" ENCODING="ISO-8859-1" STANDALONE="yes" ?>
<!DOCTYPE article [
       < !--   Début de la DTD   -->
       <!ELEMENT article (titre | auteur | date | résumé | corps) >
       <!ATTLIST article version CDATA  #REQUIRED "1">
       <!ELEMENT titre (#PCDATA)>
       <!ELEMENT auteur  (nom*)>
       <!ELEMENT nom (#PCDATA)>
       <!ELEMENT date (#PCDATA)>
       <!ELEMENT résumé (#PCDATA)>
       <!ELEMENT corps (section+)>
       <!ELEMENT section (#PCDATA)>
       <!ATTLIST section nom CDATA #REQUIRED>
```

```
        < !--   Fin de la DTD   --> ]>
< !--   Début de l'instance du document   -->
<Article Version="3">
<titre> ANALYSE MULTIDIMENSIONNELLE DES DOCUMENTS MULTISTRUCTURES
</titre>
<auteur> <nom> K. Djemal </nom> <nom> C. Soulé-Dupuy </nom> <nom> N.
Vallès-Parlangeau </nom> </auteur>
<date> 31/04/2009 </date >
<résumé> Avec l'émergence de la numérisation, le besoin d'outils de
traitement automatique des documents ...</résumé>
<corps>
        <section nom = "introduction"> Quelle que soit la nature des
documents circulant dans l'entreprise, l'information qui leur est
associée ainsi que leur contenu sont utilisés par ... </section>
        <section nom = "état de l'art"> La problématique de la
multistructuralité des documents a attiré l'attention ...</section>
        ...
</corps> </article>
< !--   Fin de l'instance du document   -->
```

Figure I.12. Exemple d'un document XML valide avec DTD interne.

III.3.1.4. Le standard SMIL

SMIL (**S**ynchronized **M**ultimedia **I**ntegration **L**anguage) (Bulterman et al. 2008) assure la description des présentations multimédias synchronisées ou des scénarios multimédias mettant en œuvre les médias de base (audio, vidéo, image et texte) afin de les consulter en temps réel et de façon interactive. SMIL est considéré comme un format d'intégration, c'est-à-dire qu'il ne décrit pas le contenu des objets médias faisant partie d'une présentation multimédia, mais plutôt leur composition temporelle et spatiale ainsi que les hyperliens entre ces objets ce qui correspond à ce qu'on a défini comme structure temporelle (Cf. Section III.2.2.6), spatiale (Cf. Section III.2.2.5) et hypermédia (Cf. Section III.2.2.4). Son principe consiste à construire des compositions séquentielles ou parallèles de média de base. SMIL est basé sur XML. Les auteurs peuvent ainsi créer et éditer facilement des présentations multimédias. Un auteur d'un document SMIL peut :

- décrire le comportement temporel d'une présentation. SMIL utilise les balises <seq> et <par> pour spécifier qu'un ensemble d'objets est joué respectivement en séquence ou en parallèle. La durée d'un objet peut être spécifiée par un délai par rapport à la date de début ou de fin d'un autre objet ;
- construire des médias complexes par des combinaisons de compositions séquentielles et/ou parallèles (respectivement des combinaisons de <seq> et/ou <par>). Les contraintes de temps sont soit implicites, c'est-à-dire calculées à partir de celles des composants, soit spécifiées dans les balises ;
- décrire le placement des objets média sur l'écran pendant la présentation. Sur la fenêtre principale de présentation, SMIL spécifie des régions dont la position et la taille sont exprimées soit en valeur absolue, soit en pourcentage de la taille de la fenêtre principale ;
- associer des hyperliens aux objets média. La désignation de la destination d'un lien est effectuée en terme d'adresse URI (**U**niform **R**esource **I**dentifier).

III.3.1.5. Bilan sur les standards de présentation

SGML fut le premier vrai standard de création de documents électroniques réellement exploitable. Le principe de SGML est celui d'un langage balisé avec une distinction entre la structure logique et la structure physique du document. La définition de la structure logique des documents permet alors la réutilisation des contenus ou la conception de documents volumineux créés par exemple au sein de différents départements d'une organisation ou d'une entreprise. SGML est donc un langage extrêmement puissant, mais en contrepartie très complexe. En effet, la structure d'une classe de documents est décrite dans une DTD qui doit être respectée rigoureusement par les auteurs de documents, ce qui peut éventuellement être un inconvénient pour des non-informaticiens. Il est également difficile de construire un navigateur capable d'afficher des documents SGML. De plus, SGML n'offre pas, en lui-même, des mécanismes de liens bien adaptés à la création d'un hypertexte associant de nombreux documents. Ces liens existent dans le standard complémentaire HyTime, mais celui-ci est beaucoup trop complexe. Ainsi, sa manipulation s'est trouvée restreinte aux spécialistes, et sa complexité a empêché son développement sur le Web.

Le standard HTML, application SGML dédiée au web, pallie ces lacunes, au détriment de la rigueur de la structuration. Un document HTML est caractérisé par le mélange de balises à caractère sémantique et d'un grand nombre de balises et d'attributs ne servant qu'à décrire les caractéristiques visuelles du document. La force de HTML réside dans la simplicité de l'apprentissage du langage, et dans un accès facile aux données sur le Web : la navigation est très facile, la publication rapide. Cependant, les balises sont utilisées de façon orientée présentation, ce qui empêche de connaître ou d'extraire automatiquement la structure logique des documents. Les contraintes inhérentes à HTML freinent le développement à grande échelle des applications d'échange à travers Internet. Ainsi, HTML continue d'être utile pour des documents, échangés sur Internet, qui n'ont pas un cycle de vie très long.

Les concepteurs de XML ont pris les meilleures parties de SGML, profité de l'expérience de HTML pour produire un langage facile d'utilisation, tout en offrant la richesse sémantique de SGML. Ainsi, XML est un sous-ensemble des règles les plus utiles de SGML, en conservant totalement l'esprit. En évitant la lourdeur de SGML, XML est utilisable sans difficulté sur Internet et il supporte une grande variété d'applications. Enfin, il est facile de créer des documents XML, que l'auteur soit confirmé ou non, grâce à la forme « valide » ou « bien formé » des documents. Cependant, même si un document XML peut être directement transcrit en un document HTML à l'aide du langage XSL, ce langage est encore peu utilisé pour la création de documents sur le Web.

Le Tableau I.4.résume l'ensemble des standards présentés dans cette section.

Tableau I.4. Récapitulatif des standards de présentation

Standard	Org-code-année	Nature des documents	Structure documentaire prise en charge	Outils de description	Feuilles de Style
SGML	ISO-8879-1986	Documents textuels et Documents multimédia	Structure logique	Syntaxe abstraite de référence (Langage de balisage SGML)	DSSSL
HTML/ XHTML	W3C 1989	Documents hypertextuel et Documents hypermédia	Structure physique Structure hypermédia	Langage de balisage HTML	DSSSL XSL CSS
XML	W3C 2000	Documents hypermédia	Structure logique et toutes structures représentées sous forme arborescente	Langage de balisage XML (Syntaxe extraite d'ASN1[3])	XSL CSS[4]
SMIL	W3C 1998	Documents Hypermédia	Structure physique Structure temporelle Structure hypermédia	Langage de balisage XML	BLI[5]

III.3.2. Standards de description de données

III.3.2.1. IPTC

L'IPTC (**I**nternational **P**ress and **T**elecommunications **C**ouncil) est un consortium développé dans le but d'échanger des données dans le domaine de la presse. Le format de transmission des documents (textes, images, sons, multimédias) émis par les agences de presse est considéré comme l'un des principaux avantages de ce consortium. L'IPTC repose sur un modèle global de données appelé « IPTC-NAA Information Interchange Model » élaboré en association avec la NAA (**N**ewspaper **A**ssociation of **A**merica). Ce modèle a servi de base à la société Adobe pour définir dans son logiciel Photoshop les informations associées à une image. Ce modèle se base sur 33 métadonnées de type interne appelé « métadonnées IPTC », c'est-à-dire stockées à l'intérieur des fichiers images JPEG, TIFF ou PSD (pour Photoshop). Ce consortium propose plusieurs standards d'annotations de documents de différentes natures : les informations, les événements, le sport, etc.

A titre d'exemple, nous retiendrons les standards suivants :

- NewsML (**News M**arkup **L**anguage) est une spécification des métadonnées IPTC pour la transmission et l'échange des informations d'actualités. Elle est conçue pour l'échange des textes, graphiques, photos, séquences audio, vidéo et animations. NewsML étend le jeu des métadonnées prédéfinies dans IPTC, ainsi que l'utilisation de vocabulaires contrôlés pour spécifier certaines métadonnées propres au domaine des informations (news) ;

[3] Abstract Syntax Notation One
[4] Cascading Style Sheets
[5] Basic Layout Isomorphic

- SportsML (Sports Markup Language) est un format spécialisé pour des scores et des statistiques sportives ;
- ProgramGuideML est un format spécialisé pour les listes des guides de programme à la télévision et à la radio.

III.3.2.2. Le standard RDF

Le W3C travaille également sur la normalisation d'applications génériques permettant de décrire des graphes de documents en explicitant des relations sémantiques. C'est le cas de RDF[6] (**R**esource **D**escription **F**ramework) qui constitue un outil très puissant pour l'indexation et la recherche de documents.

En annotant des documents non structurés et en servant d'interface pour des applications et des documents structurés, RDF permet une certaine interopérabilité entre des applications échangeant de l'information non formalisée et non structurée sur le Web. Il est une des bases du succès du Web sémantique.

Un document traduit en RDF est un ensemble de triplets. Un triplet RDF est une association {sujet, objet, prédicat}. Par exemple, le sujet peut être un document à commenter, l'objet une propriété de ce document (comme son titre) et le prédicat est la valeur de cette propriété. RDF est une structure de données constituée de nœuds et organisée en graphe. Chaque sujet est un URI (Uniform Resource Identifier) ou un nœud anonyme. Chaque prédicat est un URI. Chaque objet est un URI, un littéral ou un nœud anonyme. Un document RDF ainsi formé correspond à un multigraphe orienté étiqueté. Chaque triplet correspond alors à un arc orienté dont le label est le prédicat, le nœud source est le sujet et le nœud cible est l'objet.

III.3.2.3. Le standard Dublin Core

Le Dublin Core, schéma descriptif normalisé né à l'issue d'un meeting sur les métadonnées, entre parmi les standards destinés à améliorer la RI sur Internet et le Web. Il s'agit d'une initiative visant à la consolidation de la normalisation des métadonnées. Nous rappelons à ce niveau que les métadonnées sont un ensemble de rubriques, contenues dans ou associées à un document, donnant des informations sur son contenu. Ce sont ces informations qui sont destinées à être traitées par les moteurs de recherche.

Le Dublin Core a pour objectif d'être assez simple, mais est encore assez peu utilisé sur le Web, alors même qu'il peut parfaitement être exploité avec des technologies existantes, notamment les éléments <META> d'HTML exploitables par les moteurs de recherches comme AltaVista, Excite, Google, etc. En effet, dans les documents HTML, des balises META (META signifie METADATA) permettent de définir un certain nombre d'informations sur le contenu d'une page Web. Leur utilisation faciliterait l'échange d'informations, comme l'ont permis les codes ISBN que l'on emploie pour les livres.

Le Dublin Core propose quinze propriétés descriptives de base (métadonnées). Certaines de ces propriétés sont relatives au contenu de la ressource décrite, et les autres à

[6] http://www.w3.org/RDF/

la propriété intellectuelle de ce contenu et aux caractéristiques physiques de la ressource : « title », « creator », « subject », « description », « publisher », « contributor », « date », « type », « format », « identifier », « source », « language », « relation », « coverage », « rights ». Certaines de ces métadonnées ont été étendues par raffinement des métadonnées de base existantes. A titre d'exemple, la métadonnée date peut être raffinée par les métadonnées : « dateSubmitted » pour préciser la date de soumission d'un document, « dateCopyrighted » pour préciser la date de prise en compte du copyright, etc. D'autres métadonnées ont également été ajoutées aux métadonnées de base du Dublin Core[7] : « abstract », « available », « audience », etc. Ces métadonnées sont compatibles avec plusieurs langages tels que RDF.

III.3.2.4. MPEG-7

Pour décrire des métadonnées dans des documents multimédias, le standard le plus utilisé actuellement est le standard MPEG-7 du groupe MPEG (**M**oving **P**icture **E**xperts **G**roup ou **M**PEG). MPEG-7 (Manjunath et al. 2002) est un standard ISO/IEC développé par le comité MPEG pour la description de contenu de données multimédias supportant un certain degré d'interprétation du sens des informations. Les principaux éléments de MPEG-7 (Martinez 2002) sont :

– les descripteurs (D) qui définissent la syntaxe et la sémantique de chaque caractéristique ou métadonnée d'un document multimédia (par exemple : Video Segment, Ball, Player, GoalKeeper, etc.) ;

– les schémas de description (DS) qui spécifient la structure et la sémantique des relations (par exemple : IsCloseTo, RightOf, SameAs, etc.) qui existent entre composants MPEG-7 i.e entre des descripteurs et/ou des schémas de description ;

– un langage de définition de description (DDL), décrit avec XML schéma, qui définit la syntaxe des outils de description de MPEG-7. Il permet la création de nouveaux schémas de description et de nouveaux descripteurs, mais également autorise l'extension et la modification des schémas de description existants ;

– un système d'outils qui supporte la représentation binaire pour : le stockage, les mécanismes de transmission, la synchronisation des descriptions avec le contenu, la gestion et la protection de la propriété intellectuelle, etc.

La Figure I.13 contient un exemple indicatif de décomposition d'un schéma de description d'une vidéo. Ce schéma décrit l'ensemble des composants d'une vidéo retenus par MPEG-7. La Figure I.13 montre également un ensemble de descripteurs associés au schéma de description présenté. Par exemple, une vidéo est composée d'un ensemble de séquences et une bande sonore. Chaque séquence est, d'une part, composée d'un ensemble de scènes, et d'autre part, décrite par un type, un identifiant, un sujet, etc.

[7] http://dublincore.org/documents/dcmi-terms/

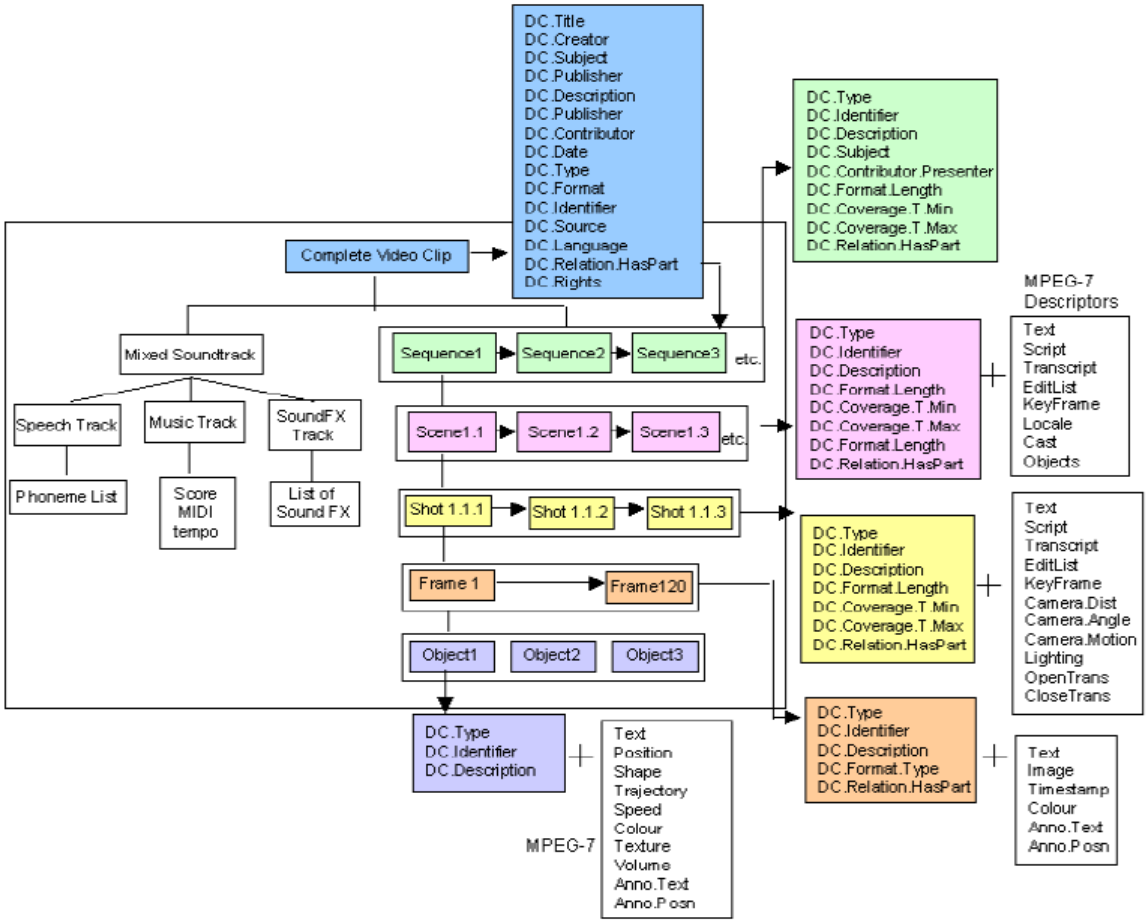


Figure I.13. Un exemple simplifié de décomposition du schéma de description d'une vidéo associé à quelques descripteurs selon MPEG-7.

III.3.2.5. Bilan sur les standards de description de données

Avec l'intégration des éléments de nature complexe tels que les médias dans les documents numériques, il devient nécessaire de définir des standards pour représenter et exploiter par la suite les descriptions de ces éléments complexes. Ces descriptions sont traduites par deux types de métadonnées : descriptives et techniques. Les métadonnées descriptives décrivent le contenu des médias (les sujets abordés, les objets retrouvés, etc.). Les métadonnées techniques décrivent les médias utilisés eux mêmes (par exemple le format du média, les outils nécessaires à son exploitation, etc.). Le Tableau I.5 présente une synthèse des différents standards que nous avons introduits dans cette sous-section. Les standards IPTEC et DC proposent un nombre fixe de métadonnées prédéfinies. Cependant, bien que les standards MPEG-7 et RDF proposent un ensemble de métadonnées, ils permettent d'utiliser aussi des métadonnées non définies *a priori*. Parmi l'ensemble des standards cités, MPEG-7 est le seul qui assure la description des informations techniques au travers des métadonnées techniques.

Tableau I.5. Comparatif des standards de description

Standard	Nombre de métadonnées	Métadonnées descriptives	Métadonnées techniques
IPTC	33	Oui	Non
RDF	Pas fixe	Oui	Non
DC	15 de base	Oui	Non
MPEG-7	Pas fixe	Oui	Oui

IV. Multistructuralité des documents : problématique et applications

La diversité typologique des structures retrouvées dans un même document (Cf. Section III.2.2) nous permet de dire que le document peut avoir plusieurs structures de natures différentes. De plus, un document peut avoir également plusieurs structures de même type. Quel qu'en soit leur rôle, leur nature et leur représentation, les structures qui sont définies sur un même document permettent d'introduire le concept de la multistructuré de documents. Avant d'aborder les problèmes et les applications relatives aux documents multistructurés, nous présentons les définitions d'un document à structures multiples.

IV.1. Définitions

Dans la littérature plusieurs définitions de la multistructuralité ont été proposées. Dans ce qui suit, nous présentons les deux définitions les plus appropriées au contexte général de nos travaux.

Selon une première vision, les différences entre les structures qui décrivent un document sont seulement dues à la décomposition ou au regroupement des parties de contenu de ce document. Dans ce contexte, (Durusau et O'Donnell 2002) et (Tennison et al. 2002) considèrent que le concept de multistructuralité est apparu du fait qu'il est souvent très difficile de réduire la structure d'un document à un arbre unique. Ils supposent que les documents textuels ont souvent plusieurs structures. Ils donnent l'exemple des poèmes qui ont à la fois une structure « poétique » sous forme de strophes et de vers et une structure « textuelle » (ensemble de paragraphes) ou de la Bible qui peut être composée en chapitres et versets ou en sections et paragraphes. La multistructuralité couvre dans ce contexte seulement des structures de type logique et physique.

(Abascal et al. 2004) proposent une définition plus large de la multistructuralité. Ils définissent le document multistructuré comme une entité unique dans laquelle sont englobées des structures différentes. Dans ce cadre, un document multistructuré est décrit par un ensemble de structures mises en correspondance. L'une de ces structures est constitutive du document et toute autre structure doit être rattachée à cette structure pivot (Chatti 2006). La diversité de ces structures est due au cadre d'utilisation du document. Dans ce sens, la multistructuralité est une description d'un document par « un ensemble

d'éléments en relation les uns avec les autres, au cours ou en vue d'un usage » (Abascal et al. 2003).

IV.2. Problématique

Les différents standards évoqués dans ce chapitre sont définis à partir de modèles de données qui ne permettent de représenter qu'une seule structure du document. Or, exploiter le caractère multistructurel des documents nécessite la définition de modèles de données permettant de représenter les différentes structures. Cependant, la coexistence de ces structures dans un même document et notamment dans un document multimédia présente plusieurs enjeux.

Le premier enjeu est celui de l'intégration de toutes les structures du document. Si le document est décomposable en plusieurs entités reliées entre elles de sorte à avoir plusieurs matérialisations d'un même document, chaque entité peut à son tour être décomposée en d'autres entités ayant plusieurs matérialisations possibles. Des structures multiples peuvent donc être définies soit au niveau global du document soit au niveau des entités qui le composent. Dans la Figure I.14, nous présentons un exemple de deux matérialisations différentes d'un ensemble d'entités. La matérialisation 1 traduit l'agencement des entités « A », « B » et « C ». La matérialisation 2 présente l'organisation des entités « A », « B », « C », « D » et « E ».

Assurer la flexibilité de représentation des structures est un vrai challenge. Ces structures doivent pouvoir être définies sur un même niveau du document, mais aussi sur des niveaux différents tout en assurant les liaisons entre-elles.

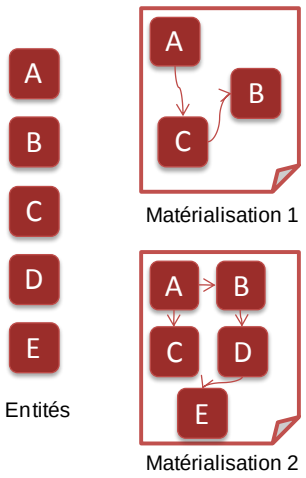


Figure I.14. Deux matérialisations différentes des mêmes entités d'un document.

Le deuxième enjeu concerne le partage du contenu. La définition de plusieurs structures sur un même niveau revient à les articuler sur un même contenu. Ces structures sont appelées dans la littérature des structures concourantes ou encore parallèles. Les éléments de ces structures concourantes peuvent ne pas s'imbriquer les uns dans les autres. Si l'on considère qu'associer un élément de structure à son contenu est une bijection du fait que chaque fragment de contenu admet un élément qui le représente, la définition de

plusieurs structures sur un même contenu transforme cette bijection en surjection : chaque fragment de contenu peut avoir plus qu'un antécédent issu des différentes structures. Cette surjection ne peut être valide que si chaque élément admet une image unique. Or, dans le cas de structures concourantes, si deux éléments issus de deux structures ne peuvent pas s'imbriquer l'un dans l'autre, on ne peut plus associer à ces deux éléments le même fragment de contenu bien qu'ils admettent une partie de contenu commune. Le problème qui se pose dans ce cas est l'entrelacement de contenu entre ces éléments. Ce problème est connu par sa nomination anglaise « overlapping markup » traduite en chevauchement d'éléments.

Pour illustrer ce problème, nous considérons deux structures (une physique et une logique) d'un même extrait de document. Si l'on souhaite fusionner ces structures, on s'aperçoit que l'imbrication du deuxième élément « phrase » de la structure logique avec un élément « ligne » de la structure physique est impossible du fait que le contenu de la phrase s'étale sur deux lignes.

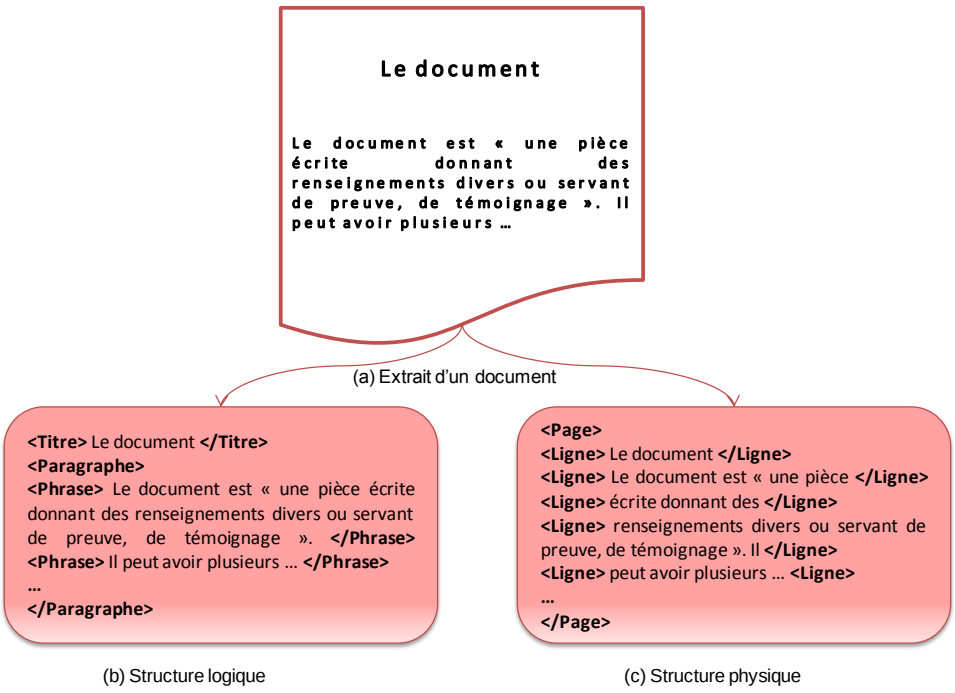


Figure I.15. Un extrait de document structuré de deux façons différentes.

La gestion de la cohérence est un point non négligeable dans la représentation des structures multiples. Ce point s'avère d'autant plus important lorsque les structures sont représentées les une indépendamment des autres et le contenu est dupliqué autant de fois que le nombre de structures définies. En effet, la modification du contenu d'une structure doit se répercuter sur les autres structures. La mise à jour des autres structures est donc indispensable pour garder la cohérence du document. Si nous reprenons l'exemple de la Figure I.15, le changement d'un fragment de contenu relatif à l'une des deux structures, tel que le titre, doit se répercuter sur la deuxième structure sinon le document devient incohérent.

Le dernier enjeu est celui de la restitution. Les informations documentaires doivent pouvoir être restituées de deux façons différentes selon deux objectifs différents. Le premier objectif consiste en la restitution du document selon une vision particulière, un usage et/ou un contexte bien précis. Si chaque matérialisation dépend d'une structure, il est nécessaire de pouvoir restituer le document selon ses différentes structures les unes indépendamment des autres. Dans un deuxième objectif, celui de l'interrogation et la recherche d'information, l'exploitation du caractère multistructurel des documents peut offrir des avantages.

IV.3. Applications de la multistructuralité

Généralement, les différentes structures d'un document sont exploitées indépendamment les unes des autres. Cependant, leur combinaison peut offrir de nouvelles dimensions utiles. L'exploitation du caractère multistructurel des documents peut s'intégrer dans plusieurs applications pour des finalités diverses.

- Gestion des versions

Les documents, une fois écrits sont rarement figés dans le temps. Plusieurs versions d'un même document peuvent être créées. La gestion de ces versions peut être assurée au traves de la multistructuralité. Chaque version est représentée par une structure (n versions $\equiv$ n structures). Gérer ces versions au travers de la multistructuralité permet de gérer le partage de contenu ainsi que les relations entre les versions.

- Restitution orientée contexte

Les systèmes d'information pervasifs se situent au cœur des avènements technologiques. Cependant, l'exploitation des documents dans un contexte en perpétuelle mutation nécessite le développement de mécanismes d'adaptations spécifiques afin de générer des versions d'un même document pour des situations contextuelles différentes. Le document doit être adapté par rapport au contexte de son utilisation, selon le profil de l'utilisateur, sa localisation, l'application et le terminal utilisé. La définition de plusieurs structures pour un même document est une solution appropriée pour assurer cette adaptation. En effet, à partir de chaque structure, il est possible de restituer une version du document orientée contexte.

- La recherche d'informations

La définition de plusieurs structures pour un même document peut améliorer la pertinence des résultats d'un processus de recherche d'informations. En effet, chaque structure apporte des informations supplémentaires qui identifient de façon différente les fragments de contenu. De plus, ces informations peuvent jouer le rôle de paramètres dans des requêtes d'interrogation de documents. Par conséquent, celles-ci deviennent de plus en plus riches et auront des résultats de plus en plus précis. En effet, la combinaison des contraintes sur plusieurs structures permet d'exploiter les corrélations qui existent entre les éléments de ces structures et par conséquent apporter une dimension supplémentaire au document.

V. Conclusion

Nous avons présenté dans ce chapitre le cadre général de nos travaux, ainsi que les concepts de base qui seront utilisés dans notre contribution pour la représentation et la gestion de documents multistructurés. Nous avons détaillé le concept de structure en nous focalisant sur ses différentes typologies, ses différentes représentations et les standards qui la décrivent. Ceci nous a permis d'aborder la problématique de la multistructuralité de documents.

Le constat que nous pouvons faire à ce niveau montre que la définition de plusieurs structures offre une description plus riche dont les informations peuvent être regroupées, recoupées selon les besoins des utilisateurs. Cette richesse laisse espérer un traitement plus précis, plus fin et certainement plus approprié des informations contenues dans les documents. Or, si l'on veut tirer profit de cette richesse, il faut pouvoir représenter simultanément ces structures et gérer le partage de contenu et le chevauchement des éléments lors de la fusion de ces structures.

Les standards et les langages présentés dans ce chapitre ne sont adaptés qu'à la représentation et l'exploitation d'une structure unique du document. Ainsi, nous consacrons le chapitre suivant à la présentation d'un état de l'art sur les documents à structures multiples.

VI. Bibliographie

Abascal, R., Beigbeder, M., Benel, A., Calabretto, S., Chabbat, B., Champin, P. A., Chatti, N., Jouve, D., Prie, Y., et Rumpler, B. (2003). « Modéliser la structuration multiple des documents. » H2PTM, Hermès, Paris, France, 253-258.

Abascal, R., Beigbeder, M., Bénel, A., Calabretto, S., Chabbat, B., Champin, P. A., Chatti, N., Jouve, D., Prié, Y., et Rumpler, B. (2004). « Documents à structures multiples. » SETIT 2004.

Adler, S., Berglund, A., Deach, S., Graham, T., Grosso, P., et Gutentag, E. (2001). "W3C Recommendation Extensible Stylesheet Language (XSL) Version 1.0" <http://www.w3.org/TR/2001/REC-xsl-20011015/>.

Aguiar, F., et Beigbeder, M. (2004). « Construction et utilisation de contextes autour des nœuds d'un hypertexte pour la recherche d'information. » Revue Document numérique, (2004/3), 71-82.

Allen, J. F. (1991). "Time and time again: The many ways to represent time." International Journal of Intelligent Systems, 6(4).

Allen, J. F. (1983). "Maintaining knowledge about temporal intervals." Communication ACM, 26(11), 837-843.

Apers, P. M. G., Blanken, H. M., et Houtsma, M. A. (1997). Multimedia databases in perspective. Springer-Verlag New York, Inc. Secaucus, NJ, USA.

Auffret, G., Carrive, J., Chevet, O., Dechilly, T., Ronfard, R., et Bachimont, B. (1999). "Audiovisual-based hypermedia authoring: using structured representations for efficient access to AV documents." Proceedings of the tenth ACM Conference on Hypertext and hypermedia: returning to our diverse roots: returning to our diverse roots, ACM New York, NY, USA, 169-178.

Bachimont, B. (1998). « Bibliothèques numériques audiovisuelles : Des enjeux scientifiques et techniques. » Revue Document numérique, 2(3), 219-242.

Bachimont, B., et Crozat, S. (2004). « Instrumentation numérique des documents : pour une séparation fonds/forme. » Information-Interaction-Intelligence I3, 4(1), 95-104.

Bechhofer, S., Carr, L., Goble, C., Kampa, S., et Miles-Board, T. (2002). "The semantics of semantic annotation" Lecture notes in computer science, 1152-1167.

Blasselle, B. (1998). Histoire du livre. Gallimard.

Bray, T., Paoli, J., Sperberg-McQueen, C., et Maler, E. (2000). "W3C Recommendation Extensible Markup Language (XML) 1.0 (Second Edition)." <http://www.w3.org/TR/2000/REC-xml-20001006>.

Bringay, S., Barry, C., et Charlet, J. (2004). « Les documents et les annotations du dossier patient hospitalier. » Revue I3 : Information-Interaction-Intelligence, 4(1), 191-211.

Bulterman, D., Jasen, J., Cesar, P., Mullender, S., Hyche, E., DeMeglio, M., Quint, J., et Kawamura, H. (2008). "Synchronized Multimedia Integration Language (SMIL 3.0) W3C Recommendation."

Cabanac, G., Chevalier, M., Chrisment C. et Julien, C. (2010). "Social validation of collective annotations: Definition and experiment". Dans : Journal of American Society for Information Science and Technology, Wiley, Vol. 61 N. 2, 271-287.

Cabanac, G., Chevalier, M., Chrisment C. et Julien, C. (2009). « Activités documentaires des usagers au sein de l'organisation : amélioration par la pratique d'annotation collective ». Dans : Ingénierie des Systèmes d'Information, Hermès Science Publications, Numéro spécial Prise en compte des utilisateurs dans les SI, Vol. 14, N. 3, 97-117.

Charhad, M., et Quénot, G. (2004). "Semantic video content indexing and retrieval using conceptual graphs." Information and Communication Technologies: From Theory to Applications, 2004. Proceedings. 2004 International Conference on, 399-400.

Chatti, N. (2006). « Documents multi-structurés: De la modélisation vers l'exploitation. » Thèse de doctorat, L'institut National Des Sciences Appliquées De Lyon.

Debarbieux, D. (2005). « Modélisation et requêtes des documents semi-structurés : exploitation de la structure de graphe. » Thèse de doctorat.

DeRose, S., Maler, E., et Ron, D. (2001). “W3C XML Pointer Language (XPointer) Version 1.0.” <http://www.w3.org/TR/WD-xptr>.

Dupoirier, G. (1995). Technologie de la GED: techniques et management des documents électroniques. Hermès.

Durusau, P., et O’Donnell, M. B. (2002). “Concurrent markup for XML documents.” Proc. XML Europe.

Egenhofer, M. J. (1994). “Spatial SQL: A query and presentation language.” IEEE Transactions on knowledge and data engineering, 6(1), 86–95.

Egenhofer, M. J., Clementini, E., et Sharma, J. (1994). “Modelling topological spatial relations: Strategies for query processing.” Computers and Graphics, 18(6), 815–822.

Egenhofer, M. J., et Franzosa, R. D. (1991). “Point-Set Topological Spatial Relations.” The International journal of geographical information science and systems, 5(2), 161-174.

Fourel, F. (1998). “Modélisation, indexation et recherche de documents structurés.” Thèse de doctorat, Université Joseph Fourier, Grenoble.

Frank, A. U. (1992). “Qualitative spatial reasoning about distances and directions in geographic space.” Journal of Visual Languages and Computing, 3, 343–343.

Frank, A. U. (1996). “Qualitative spatial reasoning: Cardinal directions as an example.” The International Journal of Geographical Information Science And Systems, 10(3), 269–290.

Fuhr, N., et Großjohann, K. (2001). “XIRQL: A query language for information retrieval in XML documents.” Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, ACM New York, NY, USA, 172-180.

Goldfarb, C. F. (1981). “A generalized approach to document markup.” ACM SIGPLAN Notices, 16(6), 68-73.

ISO-10179. (1996). “DSSSL - Document Style Semantics and Specification Language. ISO/IEC 10179:1996.”

ISO-10744. (1997). “HyTime. ISO 10744:1997 -- Hypermedia/Time-based Structuring Language (HyTime), 2nd Edition.” <http://xml.coverpages.org/hytime.html> (Sep. 1, 2009).

ISO-5127. (2001). “Information and documentation - Vocabulary.”

ISO-8879. (1986). “SSGM - Information processing, Text and Office Systems, Language, International Organization for Standardization (ISO), Geneva, first edition edition.” 15.

Julien, C. (1988). « Bases d'informations généralisées : contribution à l'étude des mécanismes de consultation d'objets multimedia. » IRIT/CERFIA Université Paul Sabatier .

Lalanne, D., et Ingold, R. (2004). « Documents statiques et multimodalité: L'alignement temporel pour structurer des archives multimédias de réunions. » Document numérique, 8(4), 65–89.

Lefèvre, S., et Sèdes, F. (2004). « Indexation de séquences vidéo : Indices liés au temps. » Document numérique, 8(4), 41–48.

Lopez-Ornelas, E. (2005). « Segmentation d'images satellitaires à haute résolution et interaction avec l'information géographique. Application à l'extraction de connaissances. ». Thèse de doctorat, Université Paul Sabatier.

Manjunath, B. S., Salembier, P., et Sikora, T. (2002). Introduction to MPEG-7: multimedia content description interface. Wiley.

Marcoux, Y. (1994). « Les formats normalisés de documents électroniques. » ICO. Intelligence artificielle et sciences cognitives au Québec, 6(1-2), 56-65.

Marshall, C. C. (1998). “Toward an ecology of hypertext annotation.” Proceedings of the ninth ACM conference on Hypertext and hypermedia: links, objects, time and space---structure in hypermedia systems: links, objects, time and space---structure in hypermedia systems, ACM New York, NY, USA, 40-49.

Martinez, J. M. (2002). “Standards-MPEG-7 overview of MPEG-7 description tools, part 2.” IEEE Multimedia, 9(3), 83-93.

Metzger, J. P., et Lallich-Boidin, G. (2004). « Temps et documents numériques. » Document numérique, (2004/4), 11–21.

Michard, A. (1998). XML, langage et applications. Edition Eyrolles.

NISO. (2004). "Understanding Metadata." National Information Standards Organisation (NISO), <http://www.niso.org/publications/press/UnderstandingMetadata.pdf>.

Pédauque, R. T. (2003). « Document : forme, signe et médium, les re-formulations du numérique. » Document de travail STIC-CNRS, RTP 33, Documents et contenu : creation, indexation, navigation.

Pemberton, S., Austin, D., Axelsson, J., Çelik, T., Dominiak, D., et Elenbaas, H. (2000). "XHTML™ 1.0 The Extensible HyperText Markup Language (Second Edition)." <http://www.w3.org/TR/xhtml1/>.

Poullet, L. (1997). « Formaliser la sémantique des documents: Un modèle unificateur. » INFORSID, Toulouse-France, 339-352.

Prié, Y., et Garlatti, S. (2004). « Méta-données et annotations dans le Web sémantique. » Web Sémantique Revue Hors-Série I, 3, 1-24.

Raggett, D., Le Hors, A., et Jacobs, I. (1999). "World Wide Web Consortium - HTML 4.01 Specification." <http://www.w3.org/TR/1999/REC-html401-19991224/>.

Roisin, C. (1999). « Documents structurés multimédia. » Habilitation à diriger des recherches, Institut National Polytechnique de Grenoble.

Roxin, I., et Mercier, D. (2004). Multimédia: les fondamentaux. Introduction à la représentation numérique. Vuibert.

Salazar, F., et Valero, F. (1995). « Analyse automatique de documents vidéo ». Université Paul Sabatier, Toulouse.

Tannier, X. (2006). « Extraction et recherche d'information en langage naturel dans les documents semi-structurés. » Thèse de doctorat, Ecole Nationale Supérieure des Mines de Saint-Etienne.

Tennison, J., Piez, W., et Nicol, G. T. (2002). "LMNL: the Layered Markup and Annotation Language." LMNL, <http://lmnl.net/index.html>.

Vazirgiannis, M., Theodoridis, Y., et Sellis, T. (1998). "Spatio-temporal composition and indexing for large multimedia applications." Multimedia Syst., 6(4), 284-298.

Vilain, M., Kautz, H., et Beek, P. (1986). "Constraint propagation algorithms for temporal reasoning." Readings in qualitative reasoning about physical systems, 377-382.

Zacklad, M., Lewkowicz, M., Boujut, J. F., Darses, F., et Détienne, F. (2003). « Formes et gestion des annotations numériques collectives en ingénierie collaborative. » Actes de la conférence Ingénierie des Connaissances IC, Laval, 207–224.

Chapitre II – Approches de gestion de documents multistructurés

Résumé du chapitre. *Ce chapitre est dédié à la présentation d'un état de l'art sur la gestion de la multistructuralité des documents. Ces approches peuvent être classées en deux catégories : les approches basées sur des langages et les approches basées sur des modèles. Ces approches sont détaillées et discutées selon les différentes problématiques de représentation et d'exploitation des documents à structures multiples.*

Sommaire du Chapitre II.

I. Introduction

Le fait qu'un document puisse être décrit selon plusieurs structures introduit des problématiques de représentation et d'exploitation :

- la gestion des structures concourantes qui découle de la définition de plusieurs structures sur un même contenu. Dans ce cas, il faut assurer le partage de contenu et notamment la gestion du chevauchement d'éléments ;
- la cohérence des informations représentées par l'ensemble des structures. Ce problème se pose notamment lors de la duplication du contenu autant de fois que le nombre de structures définies ;
- la restitution des documents ou des fragments de documents. Cette restitution doit se faire non seulement à partir des différentes structures les unes indépendamment des autres, mais également à partir de la corrélation et les relations qui existent entre ces structures.

Si les deux dernières problématiques concernent l'exploitation des documents multistructurés, elles sont fortement liées à la représentation. La principale problématique liée à la représentation est la gestion des structures concourantes. Ainsi, les problèmes de la cohérence et restitution sont liés à la méthode de gestion des structures concourantes proposée.

Dans ce chapitre, nous nous focalisons sur les approches qui s'intéressent à la représentation et à la manipulation de documents à structures multiples. L'étude de ces approches et la revue de la littérature les concernant, nous ont conduit à les regrouper en deux catégories :

- les approches basées sur des langages : toutes les structures sont décrites et « encapsulées » dans un seul document ;
- les approches basées sur des modèles : ajoutent un niveau d'abstraction supplémentaire en représentant les structures selon un modèle indépendamment de tout langage.

II. Solutions basées sur des langages

Dans les approches basées sur des langages, toutes les structures sont représentées dans un même document. Ces structures partagent ainsi le même contenu. De ce fait, le contenu n'est plus dupliqué et par conséquent le problème de cohérence des informations ne se présente plus. Si le problème de cohérence des informations ne se présente plus, les problèmes de chevauchement des éléments et de restitution doivent être résolus. Cependant, il y a d'autres exigences spécifiques au langage lui-même pour assurer l'exploitation ultérieure des documents : il doit utiliser des notations compatibles avec les langages XML/SGML et il doit être capable de fournir des documents bien formés.

Dans la littérature, deux catégories d'approches ont été proposées : la première regroupe les propositions basées sur l'extension du langage de XML/SGML

(CONCUR/XCONCUR et TEI) et la deuxième regroupe les solutions basées sur d'autres langages (LMNL, MECS/TexMECS et RDFs).

Nous prendrons comme exemple un document qui représente le livre « The SGML Handbook ». Un extrait de ce document est présenté dans la Figure II.1. Dans cette figure, nous présentons également deux structurations possibles de cet extrait de document. Cet extrait sera le support de nos exemples pour les approches basées sur des langages.

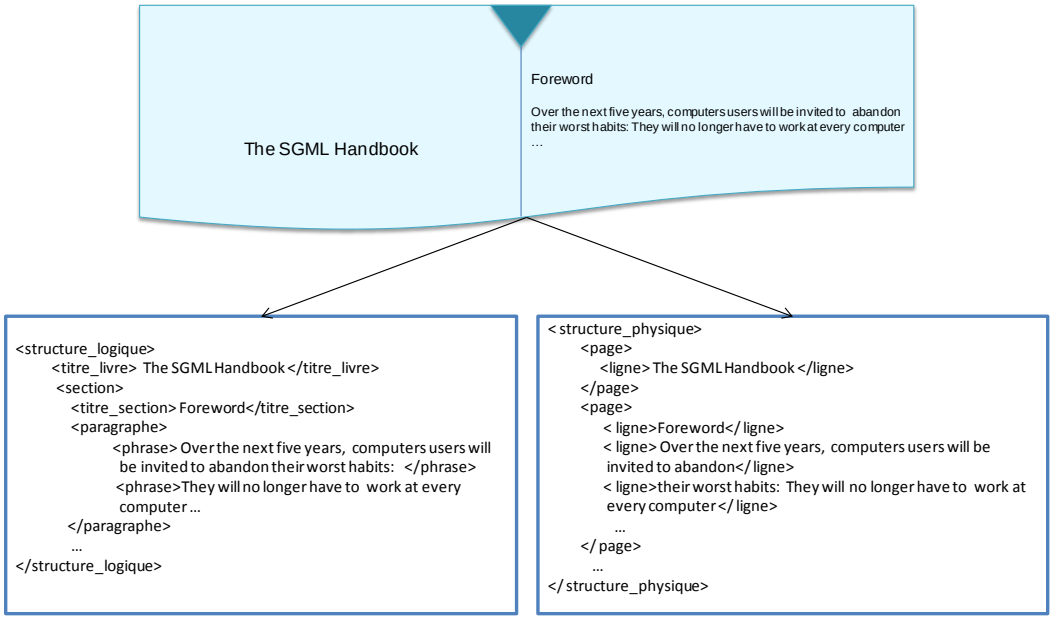


Figure II.1. Structure phyisique et logique pour l'extrait du livre « The SGML Handbook ».

II.1. Extension de SGML/XML

II.1.1. CONCUR/XCONCUR

Le standard SGML admet une fonction appelée « CONCUR » qui permet de gérer les structures concourantes dans un même document (Goldfarb 1990). Par analogie, le standard XML admet la fonction « XCONCUR » (Hilbert et al. 2005). La fonction « CONCUR » est une option figurant dans la déclaration d'un document SGML. Afin de représenter plusieurs structures, cette option doit être activée en modifiant la valeur de l'attribut CONCUR par « YES » au lieu de « NO » (Cf. Figure II.2).

```
<!SGML  "ISO 8879:1986 (WWW)"
…
FEATURES
  MINIMIZE   DATATAG NO     OMITTAG  YES    RANK     NO    SHORTTAG  YES
  LINK       SIMPLE  NO     IMPLICIT NO     EXPLICIT NO
  OTHER      CONCUR  NO YES SUBDOC   NO     FORMAL   YES
  APPINFO NONE>
```

Figure II.2.Extrait de la figure I.9 : option CONCUR.

L'activation de cette option permet de définir autant de DTD que de structures (Barnard et al. 1995) dans un même document SGML. La distinction entre les éléments de

chaque structure est assurée par l'utilisation des espaces de noms. Un préfixe indiquant le nom de la DTD dans laquelle est défini l'élément est ajouté. Ainsi, les documents représentés selon cette méthode sont des documents valides par rapport à chacune des DTD définies.

Les deux DTD suivantes (S1 et S2) (Cf. Figure II.3) définissent deux structures qui peuvent s'appliquer à un même document « livre ». S1 représente la structure logique de ce document et S2 définit sa structure physique. La structure logique se compose d'un titre_livre et d'une ou plusieurs sections. Chaque section se compose à son tour d'un titre et d'un ou plusieurs paragraphes. La structure physique se compose d'une ou plusieurs pages. Chaque page se compose d'une ou plusieurs lignes.

```
<!DOCTYPE S1 [
<!ELEMENT structure_logique (titre_livre, section+) >
<!ELEMENT titre_livre (#PCDATA) >
<!ELEMENT section (titre_section, paragraphe+) >
<!ELEMENT titre_section (#PCDATA) >
<!ELEMENT paragraphe (phrase)+ >
<!ELEMENT phrase (#PCDATA) >
]>
<!DOCTYPE S2 [
<!ELEMENT structure_physique (page)+ >
<!ELEMENT page (ligne)+ >
<!ELEMENT ligne (#PCDATA) >
]>
```

Figure II.3. Deux DTD possibles d'un document « livre ».

La Figure II.4 présente un exemple de représentation SGML d'un livre. Lors de l'exploitation de ce document, seuls les éléments préfixés par le nom de la DTD en question seront pris en compte. Dans l'exemple de la Figure II.4, nous avons préfixé les éléments de la structure logique par l'étiquette « S1 » et les éléments de la structure physique par l'étiquette « S2 ». Par exemple, l'élément page est préfixé par « S2 ». Il appartient alors à la structure physique.

```
<(S1) structure_logique>
<(S2) structure_physique>
       <(S2) page>
              <(S1) titre_livre>
                     <(S2) ligne> The SGML Handbook </(S2) ligne>
              </(S1)titre_livre>
       </(S2) page>
       <(S2) page>
              <(S1) section>
                     <(S1) titre_section>
                            <(S2) ligne>Foreword</(S2) ligne>
                     </(S1) titre_section>
                     <(S1) paragraphe>
                            <(S1) phrase>
                            <(S2) ligne> Over the next five years,
                            computers users will be invited to
                            abandon</(S2) ligne>
                            <(S2) ligne>their worst habits:
                            </(S1) phrase>
                            <(S1) phrase>They will no longer have to
                             work at every computer </(S2) ligne>
```

```
                                …
                        </(S1) paragraphe>
                        …
        </(S2) page>
        <(S2) page>
                </(S1) section>
                …
</(S2) structure physique>
</(S1) structure_logique>
```

Figure II.4. Exemple de document SGML valide par rapport à chacune des deux DTD de la Figure II.3.

Les options « CONCUR » et « XCONCUR » permettent de définir autant de DTD que de structures en assurant leur identification grâce aux espaces de noms. Ce mécanisme implique des notations « encombrantes ». De plus, des parseurs spécifiques doivent être réalisés pour exploiter chacune des structures. Toutefois, un parseur a été créé à partir d'un fichier MuLaX[8] (Multi-Layered XML) afin de construire une représentation analogue à des arbres DOM[9] (Document Object Model) (Hors et al. 2004) où l'on peut référencer plusieurs DTD à la fois (Hilbert et al. 2005). Une extension de l'API SAX[10] (Simple API for XML) a été présentée dans (Schonefeld 2008) afin de supporter les fonctionnalités de XCONCUR.

II.1.2. TEI

TEI (Text Encoding Initiative) est une norme qui assure la représentation des textes sous forme numérique (Burnard 1992). Cette norme a été étendue afin de prendre en compte les multiples hiérarchies d'un même document (Sperberg-McQueen et Burnard 2007). Le principe de base consiste à favoriser l'une des structures et à modifier les autres. Les modifications portent sur les éléments qui se chevauchent avec des éléments de la structure favorisée. Dans ce contexte, trois solutions ont été développées.

- Marquage des limites avec des éléments vides

Cette solution consiste à utiliser des éléments vides appelés « milestones » pour remplacer les éléments qui provoquent un chevauchement. Ces éléments vides doivent marquer le début et la fin de chaque élément remplacé.

Dans la Figure II.5, nous utilisons ce marquage afin de représenter la structure logique et la structure physique du document livre. Nous considérons que la structure logique est la structure principale (privilégiée). Ainsi, les éléments vides sont introduits dans la deuxième structure (physique). L'attribut « n » permet de déterminer le début d'un

[8] Mulax a été présenté comme XCONCUR à la conférence Extreme Markup Languages 2005.

[9] Le DOM est une spécification du W3C qui traduit la structure d'un document comme une arborescence d'objets. Cette spécification définit une API normalisée permettant d'accéder et de manipuler dynamiquement tous les composants d'un document structuré.

[10] SAX est une API qui permet de lire séquentiellement un flux XML et de programmer des traitements en réponse aux événements déclenchés lors de la lecture.

élément et la fin d'un autre. Par exemple, « <page n="2"/> » désigne le début de l'élément « page » numéro 2 et la fin de l'élément « page » numéro 1.

```
<structure_logique>
<structure_physique>
        <page n="1"/>
                <titre_livre>
                        <ligne> The SGML Handbook </ligne>
                </titre livre>
        <page n="2"/>
                <section>
                        <titre_section>
                                <ligne n="1"/>Foreword
                        </titre_section>
                        <paragraphe>
                                <phrase>
                                <ligne n="2"/> Over the next five years,
                                computers users will be invited to abandon
                                <ligne n="3"/>their worst habits:
                                </phrase>
                                <phrase>They will no longer have to
                                 work at every computer <ligne n="4"/>
                                …
                        </paragraphe>
                        …
        <page n="3"/>
                </section>
                …
</structure_physique>
</structure_logique>
```

Figure II.5. Exemple d'utilisation d'éléments vides (extrait du document présenté dans la Figure II.4).

L'utilisation des éléments vides permet d'une part de retrouver les éléments initiaux et d'autre part, de maintenir le document bien formé.

- Fragmentation et reconstitution virtuelle des éléments

La deuxième solution consiste à découper en plusieurs parties le contenu des éléments sur lesquels on observe un chevauchement des éléments. Chaque partie sera représentée par un nouvel élément admettant le même nom et un attribut « n ». La valeur identique de cet attribut permet d'assurer la liaison entre les différentes parties des éléments fragmentés afin de pouvoir le reconstruire ultérieurement.

La Figure II.6 illustre un exemple d'utilisation de ce type de marquage. Dans le premier paragraphe, les éléments « phrase » et « ligne » se chevauchent. Nous avons choisi de découper l'élément « page » afin d'éviter ce chevauchement. Ainsi, nous avons introduit l'élément « <ligne n="ligne4"> » qui assure la liaison entre les deux parties de l'élément fragmenté.

```
<structure_logique>
<structure physique>
        <page>
                <titre_livre>
                        <ligne> The SGML Handbook </ligne>
                </titre_livre>
        </page>
```

```
        <page>
                <section>
                        <titre_section>
                                <ligne>Foreword</ligne>
                        </titre_section>
                        <paragraphe>
                                <phrase>
                                <ligne> Over the next five years,
                                computers users will be invited to
                                abandon</ligne>
                                <ligne n="ligne4">their worst habits:
                                </ligne>
                                </phrase>
                                <phrase>
                                <ligne n="ligne4">They will no longer have
                                to work at every computer </ligne>
                                …
                        </paragraphe>
                        …
        </page>
        <page>
        …
</structure_physique>
</structure_logique>
```

Figure II.6. Exemple de fragmentation d'élément (extrait du document présenté dans la Figure II.4).

- Standoff Markup

La dernière solution consiste à utiliser la technique du « out of line markup » (Sperberg-McQueen et Burnard 2007) appelée aussi « standoff annotation » (McKelvie et al. 1999). Cette technique permet de définir plusieurs hiérarchies fragmentées et stockées séparément et de les relier par des hyperliens. « joint » est un élément virtuel ajouté à la fin du document afin de définir l'ordre des fragments et d'assurer leur reconstruction. Cet élément admet deux attributs : le premier « result » sert à spécifier le nom de l'élément fragmenté et le deuxième « target » sert à retracer le séquencement des sous-éléments au travers de leur identifiant.

Dans la Figure II.7, nous avons utilisé cette technique pour résoudre le problème dû au chevauchement des éléments qui existe entre les deux éléments « ligne » et « phrase ». Nous avons opté pour la fragmentation de l'élément « ligne ». Les deux parties de l'élément fragmenté sont identifiées au travers de l'attribut « id ». Lors de la reconstitution de la structure physique, l'élément virtuel « joint » permet de regrouper ces deux parties via ses deux attributs : « result » qui permet d'identifier l'élément fragmenté (« ligne ») et « target » qui permet de retrouver l'ordre de ces deux parties au travers de leur identifiant.

```
<structure_logique>
<structure physique>
        <page>
                <titre_livre>
                        <ligne> The SGML Handbook </ligne>
                </titre_livre>
        </page>
        <page>
                <section>
                        <titre_section>
```

```
                        <ligne>Foreword</ligne>
                    </titre_section>
                    <paragraphe>
                        <phrase>
                        <ligne> Over the next five years,
                        computers users will be invited to
                        abandon</ligne>
                        <ligne id="part1">their worst habits:
                        </ligne>
                        </phrase>
                        <phrase>
                        <ligne id="part2">They will no longer have
                        to work at every computer </ligne>
                        …
                    </paragraphe>
                    …
        </page>
        <page>
        …
</structure_physique>
</structure_logique>
<joint result = "ligne" targets = "part1 part2">
```

Figure II.7. Exemple d'utilisation de la technique de « standoff markup » (extrait du document présenté dans la Figure II.4).

- Bilan TEI

Afin de gérer le chevauchement des éléments, TEI oblige à privilégier une structure parmi l'ensemble des structures et à imbriquer les autres en fragmentant leurs éléments ou en créant des éléments vides (Sperberg-McQueen et Burnard 2007). Des identifiants sont utilisés dans les éléments modifiés. Ceci permet d'éliminer tout risque de confusion lors de l'étape de reconstitution de chacune des structures. Si ces solutions fournissent toutes les informations pour reconstruire les structures et permettent à toutes les structures d'être manipulées implicitement, la reconstruction automatique des structures nécessite toujours des traitements très lourds. Toutefois, une extension XPath est proposée (Dekhtyar et al. 2005) afin d'interroger les structures multiples d'un document qui admet des éléments vides (milestones). Afin d'offrir une certaine flexibilité au parseur, (Durusau et O'Donnell 2004) proposent de combiner deux techniques : celle utilisée dans CONCUR et celle utilisée dans TEI pour représenter un document multistructuré.

(Durusau et DeRose 2003) proposent un langage basé sur la méthode « milestones » connue sous le nom de « Trojan milestones », qui emploie le même type d'étiquette pour les éléments normaux et les éléments vides (S. DeRose 2004).

II.2. Autres langages

II.2.1. LMNL

Tennison et al., proposent d'utiliser un nouveau langage de balisage (non XML) appelé LMNL (Layered Markup and Annotation Language) (Tennison et Piez 2002). Ce langage s'articule autour de trois concepts de base à savoir les couches appelées « layers », les zones de document appelées « ranges » et les « annotations ». Un document LMNL

n'est pas défini en terme d'éléments comme c'est le cas dans XML, mais de couches (une ou plusieurs) qui se superposent les unes aux autres. La couche la plus basse est une couche de texte qui est constituée d'une chaîne de caractères. Les autres couches sont composées des zones du document (ranges) qui sont étiquetées et qui référencent soit d'autres zones du document (localisées dans une autre couche), soit un ensemble de caractères de la dernière couche. L'utilisation des « ranges » permet d'assurer la gestion des recouvrements entre structures en traitant simultanément l'ensemble des zones du document au lieu de les traiter de façon indépendante.

La Figure II.8 présente un exemple de représentation selon LMNL. Cet exemple présente deux structures concourantes résultant de la définition de deux langues : français et anglais.

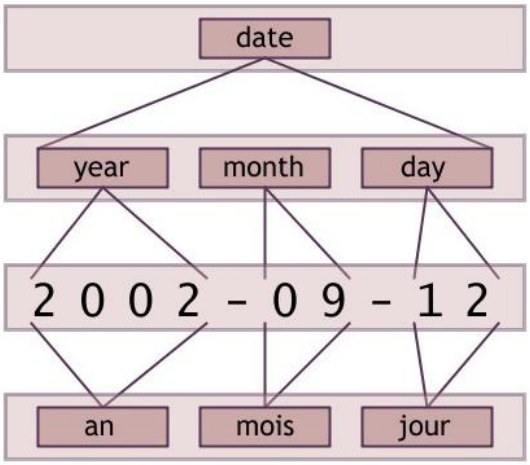


Figure II.8. Deux représentations d'une date selon LMNL (Tennison et al. 2002).

Syntaxiquement, chaque couche est représentée par une expression admettant la forme suivante « [!layer name="…" base="…"] » (Cf. Figure II.9). L'attribut « name » désigne le nom de la couche, l'attribut « base » renseigne sur la nature du contenu de cette couche (des zones de documents ou des caractères). Ces couches sont référencées au niveau de chaque zone de document par une « ~ » suivie de son nom. En plus de cette référence, une zone de document admet une étiquette appelée « Tag ». Il y a trois types d'étiquettes :

- étiquette de début : indique le début d'une zone de document. Cette étiquette admet la forme suivante « '[' TagContent '}' » ;
- étiquette de fin : représente la fin d'une zone de document. Cette étiquette admet la forme suivante « '{' TagContent ']' » ;
- étiquette vide : désigne une zone de document de longueur 0. Cette étiquette admet la forme suivante « '[' TagContent ']' ».

Des exemples des deux premiers types d'étiquettes sont présentés dans la Figure II.9.

```
[!lmnl version="0.1"]
[!layer name="fr" base="#text"]
[!layer name="type" base="#default"]
```

```
[date~type
  }[an~fr}[year}2002{year]{an~fr
  ]-[mois~fr}[month}09{month]{mois~fr
  ]-[jour~fr}[day}12{day]{jour~fr]{date~type]
```

Figure II.9. Représentation syntaxique de l'exemple de la Figure II.8 (Tennison et al. 2002).

LMNL est un nouveau langage de balisage (non XML) qui n'est pas défini en terme d'éléments, mais de « layers » et de « ranges ». L'utilisation des « layers » permet de faciliter la gestion des recouvrements entre structures en partageant des ensembles de « ranges » en commun au lieu de les traiter de façon indépendante. L'inconvénient majeur de ce langage réside dans son non compatibilité avec les applications XML/SGML. En effet, les nouvelles notations nécessitent des parseurs spécifiques qui doivent être développés afin d'assurer d'une part l'annotation et d'autre part l'exploitation.

II.2.2. MECS/TexMecs

MECS (Multi-Element Code System) est un langage développé par Wittgenstein Archive à l'université de Bergen afin d'encoder les structures non hiérarchiques des livres (Huitfeldt 1993). Ce langage partage plusieurs aspects avec le langage SGML. La principale différence entre ces deux langages réside dans leur modèle de données. SGML exige une représentation arborescente des structures alors que MECS ne l'impose pas.

Dans MECS, un élément est représenté par une étiquette de début (start-tags) ayant la forme « <tag/ » et une étiquette de fin (end-tags) ayant la forme « /tag> » (Cf. Figure II.10). Afin de résoudre le problème de chevauchement des éléments, MECS préconise l'usage du principe de la technique milestones : usage d'éléments vides. Ces éléments seront représentés de la façon suivante : « <tag> ». A titre d'exemple, dans la Figure II.10, nous avons utilisé « <phrase/ » pour marquer le début de l'élément « phrase », « /phrase> » pour marquer la fin de ce même élément et nous avons ajouté un élément vide « <ligne> » afin de résoudre le chevauchement qui existe entre les éléments « ligne » et « phrase ». MECS admet d'autres caractéristiques d'encodage de document que nous ne décrivons pas dans ce manuscrit.

```
…
<paragraphe/
        <phrase/
                <ligne/ Over the next five years, computers users will be
                invited to abandon /ligne>
                <ligne/ their worst habits: <ligne>
        /phrase>
        <phrase/
                <ligne>They will no longer have to work at every computer
                /ligne>
        …
/paragraphe>
```

Figure II.10. Représentation d'un extrait du document présenté dans la Figure II.4 avec le language MECS.

TexMECS (Huitfeldt et Sperberg-McQueen 2001) est un langage de balisage développé dans le cadre du projet MLCD (Markup Languages for Complex Documents)

(Huitfeldt et Sperberg-McQueen 2004). Ce langage vise à profiter des avantages de XML, SGML et MECS. En effet, si le document représenté admet une structure hiérarchique claire, TexMECS doit être isomorphe à XML, sinon il doit être isomorphe à MECS.

TexMECS est défini au travers de plusieurs types d'éléments, dont nous citons les plus importants :

- un élément vide marqué par une seule étiquette : « {e id="foo" lang="fr"} ». Les caractères « { » et « } » sont utilisés comme délimiteur syntaxique ;
- un élément classique marqué par une étiquette de début et une étiquette de fin : « {e id="foo" lang="fr"{le contenu}e} » ;
- un élément interrompu sert à représenter les éléments qui présentent un chevauchement : « {e id="foo" lang="fr"{première partie du contenu}-e}… {+e{deuxième partie du contenu}e} ». « }-e} » signale l'interruption de l'élément « e » c'est-à-dire que ce qui vient après n'appartient pas à cet élément. « {+e{ » indique la reprise de l'élément « e » ;
- etc.

```
…
{paragraphe
        {phrase
                {ligne {Over the next five years, computers users will be
                invited to abandon} ligne}
                {ligne {their worst habits:} -ligne}
        phrase}
        {phrase
                {+ligne {They will no longer have to work at every
                computer} ligne}
        …
Paragraphe}
```

Figure II.11. Représentation d'un extrait du document présenté dans la Figure II.4 avec le language TexMECS.

Dans MECS et TextMECS, bien que les structures concourantes soient gérées par des fragments de code incorporés au niveau des éléments, ces structures ne sont pas représentées sous forme d'arbre. De ce fait, un parseur MECS n'offre pas autant de fonctionnalités qu'un parseur SGML.

II.2.3. RDF/RDFs

En se basant sur la richesse de RDF (Lassila et Swick 2000) et RDFs (Brickley et Guha 2004), (Tummarello et al. 2005) présentent une méthode pour gérer le chevauchement d'éléments. L'idée est de considérer les fragments textuels comme une ressource RDF. Ces ressources sont ainsi alignées, en utilisant la propriété « next », pour former une chaîne reflétant l'organisation des différents fragments d'origine. La propriété « printable content » est utilisée afin d'associer la ressource à son fragment textuel. De la même manière, les éléments structurels sont considérés comme des ressources. La même méthode de chaînage est appliquée sur ces éléments structurels. Les relations hiérarchiques ne sont pas directement représentées dans un graphe RDF. Chaque nœud père identifie les

valeurs de la première et de la dernière ressource via les propriétés « first symbol » et « last symbol ». Les ressources de même niveau sont reliées au travers de la propriété « next ».

La Figure II.12 représente un extrait de structure traduit sous forme de ressource RDF. Chaque élément (mot ou ponctuation) correspond à un nœud RDF qui admet une propriété « printable content » référençant le contenu. Par exemple, l'élément « Period » est considéré comme une ressource dont le contenu est identifié par les deux propriétés « first symbol » et « last symbol » ayant respectivement pour valeurs la première et la dernière ressource de la chaîne représentant le contenu.

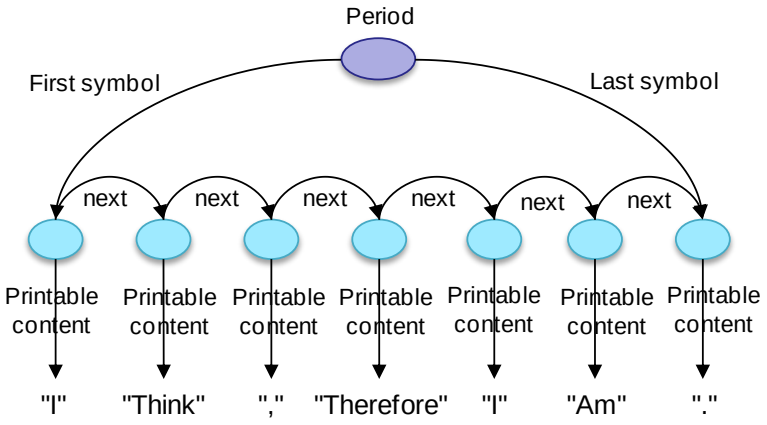


Figure II.12. Extrait d'une structure représentée sous forme de ressource RDF.

La Figure II.13 illustre le cas de deux structures concourantes qui sont marquées par des entrelacements entre leurs éléments. La première structure est constituée d'un ensemble de « Pages » et « Rows » et la deuxième structure est composée de « Chapters » et « Periods ». Les deux structures partagent le même contenu composé de mots (« words ») et de ponctuations (« ponctuations »).

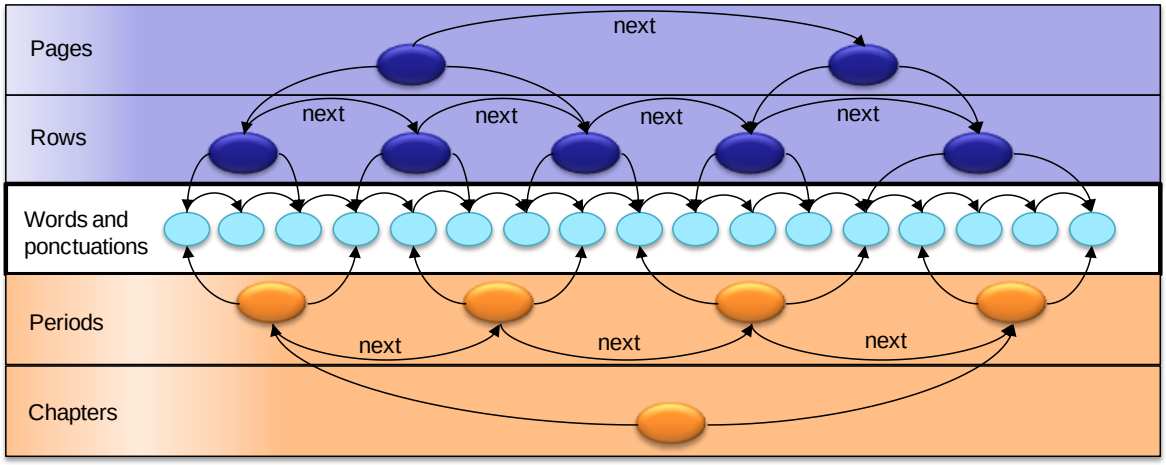


Figure II.13. Deux structures concourantes représentées sous forme de graphe RDF.

II.3. Synthèse des langages

La caractéristique principale des approches basées sur des langages est de représenter l'ensemble des structures dans un même fichier. Ainsi, les structures sont fusionnées les unes avec les autres et par conséquent elles partagent un contenu commun. Le chevauchement d'éléments est géré par fragmentation du contenu relatif à ces éléments. Si des mécanismes spécifiques ont été utilisés dans CONCUR et LMNL, les autres langages ont opté à la fragmentation des éléments eux-mêmes. Afin d'assurer cette fragmentation, trois méthodes se présentent : l'usage des éléments vides (milestones de TEI et MECS), des éléments virtuels (TEI et TexMECS) ou des mécanismes spécifiques de jointure (Standoff Markup de TEI, RDFs). La restitution des documents à partir de chacune de ces structures indépendamment des autres est fortement liée à la méthode utilisée pour la gestion de chevauchements d'éléments. Les méthodes de restitution proposées se basent soit sur des mécanismes à base des espaces de noms soit sur des mécanismes à base de jointures.

D'une façon générale, les approches basées sur des langages sont caractérisées par leur une syntaxe qui sert à structurer les documents de manière précise, concise et sans ambiguïté. En contrepartie, ces approches présentent une double difficulté : d'une part un problème de lisibilité des documents multistructurés pour les utilisateurs et d'autre part un problème de développement de compilateur pour le traitement automatique de ces documents.

III. Solutions basées sur des modèles

Une deuxième catégorie de solutions pour gérer les documents multistructurés consiste à les représenter selon des modèles. Les modèles offrent une représentation indépendante d'un langage donné ce qui favorise l'adaptation du document à toutes les plateformes dans tous les contextes. De tels modèles doivent d'une part décrire les différentes structures du document et d'autre part gérer le chevauchement d'éléments entre ces structures. De plus, un modèle de documents multistructurés doit résoudre les problèmes de cohérence et de restitution. Dans cette section, nous détaillons donc les différents modèles de gestion de la multistructuralité présentés dans la littérature.

III.1. Le modèle MSDM

Le modèle MSDM « **M**ulti**S**tructured **D**ocument **M**odel » (Chatti et al. 2004) est proposé dans l'objectif d'intégrer un ensemble de structures au sein d'un même document toute en assurant l'exploitation de ces structures conjointement. Ce modèle s'appuie sur celui de l'ISDN « **I**nstitut des **S**ciences du **D**ocument **N**umérique » (Abascal et al. 2003). Les deux modèles sont basés sur trois notions :

– une structure de base (SB) est définie afin d'organiser le contenu partagé par plusieurs structures en fragments élémentaires disjoints. Ces fragments servent à reconstituer le contenu original du document ;

– un ensemble de structures documentaires (SD). Chaque SD est une description particulière du contenu ;

– un ensemble de relations de correspondance entre les deux structures : un élément de la première structure (SD) est relié à celui de la deuxième structure (SD ou SB).

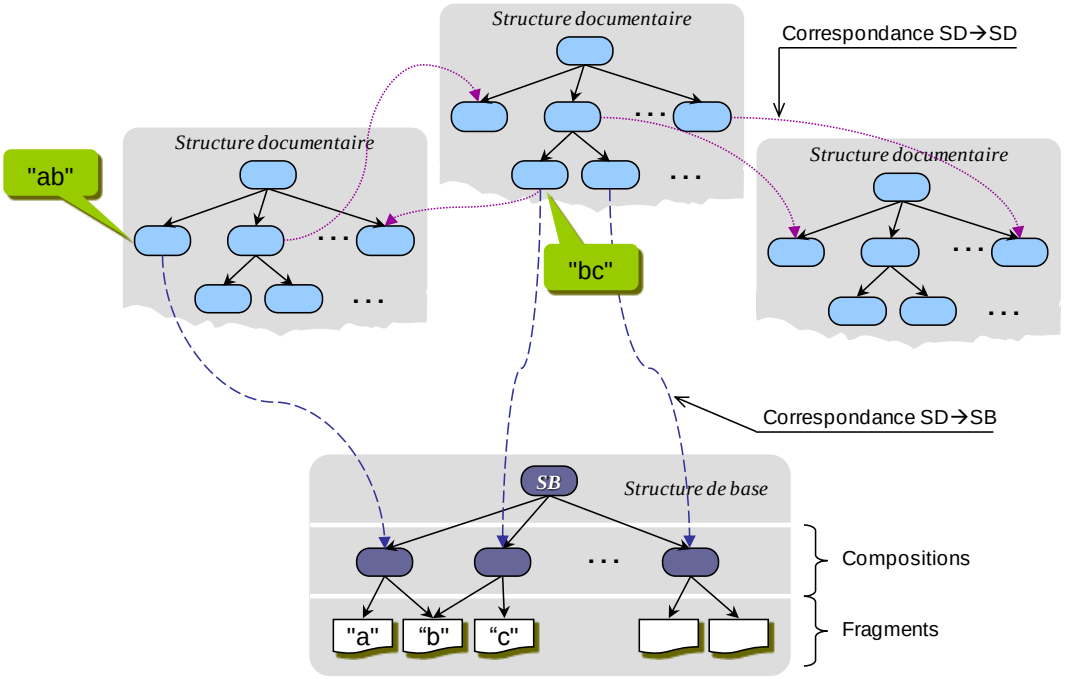


Figure II.14. Illustration du modèle MSDM (Chatti 2006).

La Figure II.14 présente une illustration du modèle MSDM. Cette illustration comprend une structure de base et trois structures documentaires. Les structures documentaires sont d'une part, reliées entre elles par des liens de correspondances de type (SD,SD) et d'autre part associées à la structure de base par des liens de correspondances de type (SD,SB). Une description formelle de ce modèle est détaillée dans (Chatti 2006).

A partir du modèle MSDM, les auteurs ont proposé un formalisme d'encodage des documents multistructurés appelé MultiX (Chatti et al. 2007). Ce formalisme permet de représenter physiquement plusieurs structures dans un même fichier par linéarisation. Chatti et al., utilisent le formalisme XML. Un document MultiX admet trois parties : les structures documentaires (SD), la structure de base (SB) et les correspondances (Cf. Figure II.15). Chaque partie est encapsulée dans un élément dont le nom est préfixé par « msd » et suivi du nom de la partie en question (par exemple <msd:SB>). Les éléments « <msd:SD> » admettent un attribut « name » qui détermine le nom de la structure documentaire traitée au niveau de cet élément.

```
<msd:MSD name="manuscrit" xmlns:msd="http://www.msdm.org/2006/MULTIX/">
        <msd:SD name="nom de la première structure documentaire">
                <!-- ici la première structure documentaire -->
        </msd:SD>
        <msd:SD name="nom de la deuxième structure documentaire">
                <!-- ici la deuxième structure documentaire -->
        </msd:SD>
```

```
        <!-- autres structures documentaires -->
        <msd:SB>
                <!-- la structure de base qui indexe les fragments de
                 contenu élémentaires-->
        </msd:SB>
        <msd:correspondances>
                <!-- définition des relations de correspondances -->
        </msd:correspondances>
</msd:MSD>
```

Figure II.15. La composition de base d'un document MultiX.

Les mécanismes d'exploitation des documents multistructurés proposés dans le cadre de ces travaux consistent en des fonctions d'interrogation. Dans (Chatti et Calabretto 2007), les auteurs proposent la bibliothèque MXQ considérée comme une extension de XQuery. Cette bibliothèque admet deux catégories de fonctions :

- des fonctions exploitant les correspondances SD→SB. Ces fonctions permettent l'exploitation des constituants de la structure de base (fragments de contenu ou éléments) ;
- des fonctions exploitant les correspondances SD→SD. Ces fonctions permettent d'exploiter les liens entre structures. Elles ont pour rôle de retrouver les éléments en correspondance dans les différentes structures.

Toutes les structures représentées s'articulent autour d'une structure de base qui représente le contenu. La restitution de ces différentes structures est assurée au travers des relations de correspondances. Cette structure de base est caractérisée par une organisation « trop » simple (plate). Afin de partager le contenu entre les structures documentaires, la structure de base est fragmentée tant qu'il y a un chevauchement entre deux éléments. Tous les fragments de cette structure sont des feuilles d'une même racine qui ne joue aucun rôle fonctionnel dans la structure. De plus, la mise à jour de cette structure nécessite des mises à jour de toutes les relations (SB,SD). Cette mise à jour peut être la conséquence d'une modification de contenu ou le résultat de la mise à jour d'une des structures SD. Le principal problème qui se pose est celui de la mise à jour des correspondances (SD,SD) entre la structure SD modifiée et les autres structures SD.

III.2. Le modèle Proximal Node

Le modèle « Proximal Node » a été introduit par Navarro et Baeza-Yates (Navarro et Baeza-Yates 1997) dans l'objectif de proposer un langage d'interrogation de documents qui exploite aussi bien le contenu que la ou les structures d'un document. Ce modèle est spécifique aux documents texte. Chaque structure est une arborescence indépendante des autres représentée sous la forme d'une vue. Au travers de ce modèle, (Navarro et Baeza-Yates 1997) ont proposé un langage d'interrogation exploitant les différentes structures associées à un document.

Formellement, le modèle Proximal Node s'articule autour de 7-uplets : $(\tau, \upsilon, C, \mathrm{N}, \mathrm{R}, \mathrm{Constr}, \mathrm{Segm})$, où :

- $\sum = I(\tau)$ est l'ensemble de l'alphabet utilisé ;
- τ: [1..T] → $\sum$ représente la chaîne de caractères du texte global de taille T, c'est-à-dire une suite de T éléments appartenant à l'alphabet $\sum$;
- υ est l'ensemble fini des vues du texte. La vue textuelle est un élément de cet ensemble noté $V_t \in \upsilon$;
- $\mathcal{C} = I(C)$ est un ensemble fini de constructeurs. Le constructeur textuel est un élément de cet ensemble noté $C_t \in \mathcal{C}$;
- $\mathcal{N} = I(N)$est un ensemble fini de nœuds. Les nœuds textuels notés $t_{a,b}$ sont inclus dans cet ensemble tel que $1 \leq a \leq b \leq T$;
- $C: \upsilon \to \wp(\mathcal{C})$ est l'ensemble des constructeurs de chaque vue noté $C(V)$ ou encore C_V avec pour contraintes : $\forall V_1 \neq V_2 \in \upsilon, C_{V_1} \cap C_{V_2} = \emptyset \; et \; C_{V_t} = \{C_t\}$;
- $N: \upsilon \to \wp(\mathcal{N})$ est l'ensemble de nœuds de chaque vue noté $N(V)$ ou encore N_V avec pour contraintes : $\forall V_1 \neq V_2 \in \upsilon, N_{V_1} \cap N_{V_2} = \emptyset \; et \; N_{V_t} = \{t_{a,b} / \; 1 \leq a \leq b \leq T\}$;
- $R : \upsilon \to \wp(\mathcal{N} \times \mathcal{N})$ est la relation binaire qui définit l'arborescence de chaque vue du texte. Pour la vue υ cette relation est notée $R(V)$ ou encore R_V. Cette relation doit vérifier les contraintes suivantes : $\forall V \in \upsilon, \; R_V \subseteq (N_V \times N_V) \; et R(V_t) = \emptyset$;
- Constr: $\mathcal{N} \to \mathcal{C}$ est le constructeur de chaque nœud. Cette fonction est définie de la manière suivante : $\forall V \in \upsilon \; et \; \forall x \in N_V, \text{Constr}(x) \in C_V$. Ce qui implique $\forall a, b / 1 \leq a \leq b \leq T, \text{Constr}(t_{a,b}) = C_t$;
- Segm: $\mathcal{N} \to [1..\text{T}] \times [1..\text{T}]$ détermine le segment associé à chaque nœud de la manière suivante : $\forall x \in \mathcal{N}, \text{Segm}(x) = (a, b) \Rightarrow a \leq b$ et $\text{Segm}(t_{a,b}) = (\text{a}, \text{b})$.

Le modèle Proximal Node présente une dichotomie entre structure et contenu. Une telle approche permet de gérer un contenu commun partagé entre les différentes structures. Les correspondances entre le contenu et les structures se font au travers d'une fonction de calcul de positions (début et fin) de chaque segment. Etant donné que le contenu est une chaîne textuelle, cette fonction utilise le début de texte comme repère et la taille d'un caractère de l'alphabet comme unité de mesure. En plus de son rôle de restitution de document, ce mécanisme permet de gérer le chevauchement éléments. En contrepartie, ce modèle représente chaque structure indépendamment des autres sans assurer la relation entre ces structures. De plus, ce modèle ne gère pas le partage d'éléments (nœuds) entre les structures. Ainsi, la duplication de certains éléments est nécessaire pour construire les différentes vues formant chacune des structures.

Afin d'interroger les documents multistructurés représentés selon le modèle Proximal Node, un langage spécifique est proposé. Ce langage est traduit par une algèbre qui définit trois types d'opérations (Cf. Figure II.16) :

- des opérateurs d'extraction de contenu. Ces opérateurs sont basés sur des fonctions d'appariement « Opers on matches ». Ces fonctions sont les seules à pouvoir accéder aux contenus textuels. Le langage d'appariement défini permet de générer à partir des prédicats de la requête, des ensembles de segments textuels disjoints ;

- des opérateurs d'extraction de structures. Ces opérateurs traitent les prédicats qui intègrent les structures de documents. Les auteurs proposent deux types d'opérateurs : ceux qui s'appliquent aux noms des composantes structurelles (les nœuds) et à leurs types : « Constructor », et ceux qui concernent les noms des hiérarchies et leurs types : « View » ;
- des opérateurs de composition. Ces opérateurs permettent la combinaison des résultats des autres opérateurs définis ci-dessus. Ces derniers seront intégrés dans la requête sous forme d'opérandes. On distingue cinq sous types d'opérateurs :
    - les opérateurs d'inclusion « By including elements » permettent de sélectionner des éléments à partir du premier opérande qui sont inclus dans l'un des deuxièmes opérandes. Ces opérateurs sont « in », « begin » et « end ». Par exemple, « *P* ***in*** *Q* » représente l'ensemble des nœuds de P qui sont inclus dans un nœud de Q,
    - les opérateurs d'inclusion « By included elements » sélectionnent à partir du premier opérande les éléments incluant des éléments du second opérande. Ces opérateurs sont « with », « withbegin » et « withend ». Par exemple, « *P* ***with (k)*** *Q* » représente l'ensemble des nœuds de P qui sont inclus dans au moins k nœuds de Q,
    - les opérateurs structurels permettent de sélectionner des éléments à partir du premier opérande en se basant sur des critères structurels. Les opérateurs proposés sont « parent » et « child ». A titre d'exemple, « [s] *P* ***childs*** *Q* » est l'ensemble des nœuds de P sont des descendants (dans une arborescence particulière) des nœuds de Q,
    - les opérateurs de distance ou encore de positionnement « Positional » permettent de sélectionner à partir du premier opérande des éléments qui admettent une relation de distance avec des éléments du second opérande sous certaines conditions. Les opérateurs de cette catégorie sont « after », « before », « after(k) » et « before(k) ». Par exemple, « *P* ***before*** *Q* (C) » représente l'ensemble des nœuds de P dont le segment de contenu relatif commence après la fin du segment de contenu d'un nœud de Q,
    - les opérateurs de manipulation « set manipulation » permettent de combiner les opérandes deux à deux. Les auteurs considèrent les opérandes comme étant des ensembles. Ce qui leur permet de proposer des opérateurs d'union (« + »), de différence (« - »), d'intersection (« is ») et l'identité (« same ») sous plusieurs critères. Par exemple, « *P* ***same*** *Q* » représente l'ensemble des nœuds de P dont le segment de contenu relatif est le même que celui d'un nœud de Q.

En résumé, grâce à ce langage, trois types d'exploitation de documents sont possibles : recherche plein texte, interrogation selon une même structure, et une exploration combinant toutes les structures. En revanche, l'algèbre proposée ne permet pas de restituer des nœuds de structures différentes dans le résultat d'une même requête. Les auteurs ne proposent pas de mécanisme de calcul de chevauchement d'éléments.

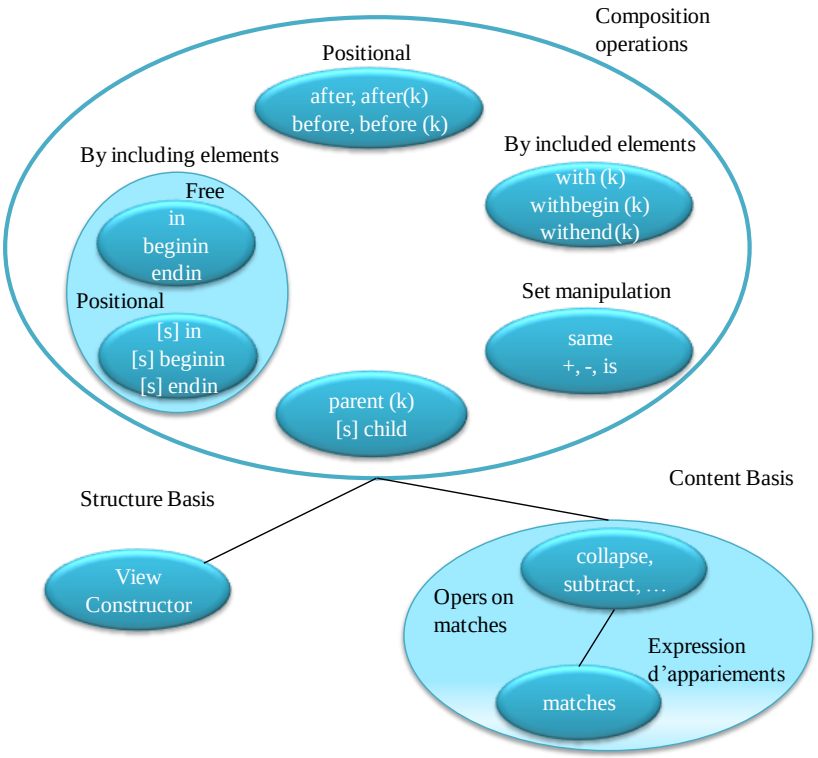


Figure II.16. Les opérations d'interrogation pour le modèle proximal node, classifiées par type.

III.3. Le modèle MSXD

Le modèle MSXD (**M**ulti**S**tructured **X**ML **D**ocuments) (Bruno et Murisasco 2006) a été défini dans l'objectif de représenter les documents multistructurés en prenant en compte les annotations ajoutées par un utilisateur. A l'opposé des deux modèles présentés précédemment qui préconisent le partage de contenu, le modèle MSXD se base sur le principe de la duplication du contenu. Les structures sont donc représentées les unes indépendamment des autres. Les relations entre ces structures sont gérées dans ce qu'ils appellent un « schéma de document multistructuré ». Les représentations structurelles s'appuient sur la notion de Hedge (Murata 1999) (fondation de RelaxNG (Clark et Murata 2001)).

En se focalisant sur les documents textuels, (Bruno et Murisasco 2006) définissent un document multistructuré comme un triplet (V, G, S) où V représente la valeur textuelle, G représente un ensemble de segmentations de V et S représente l'ensemble des structures associées aux segmentations de G. Chaque structure et ses annotations sont définies au travers d'un ensemble de segmentations et décrites avec le langage XML. Le schéma de document multistructuré définit les relations entre ces structures au travers de contraintes faibles exprimées par des relations d'Allen (Allen 1983).

La Figure II.17 illustre un exemple de schéma de document multistructuré d'un poème présenté dans (Bruno et Murisasco 2006). D'une part, ce schéma référence trois structurations possibles de poèmes et un ensemble d'annotations de son contenu ajouté par un utilisateur. Le schéma admet aussi un ensemble de contraintes illustrant les relations entre des fragments de structures différentes. Par exemple, la première contrainte traduit le

fait que les deux éléments « Sonnet » de la structure « poem_physical » et « Poem » de la structure « poem_linguistic » partagent le même contenu.

```
<MsXmlSchema xmlns: a =’http: // sis.univ -tln.fr/annot ’>
<!-- Identification of the structures and annotations -->
<Structures>
        <Structure type="http://sis.univ -tln.fr/msxd/structure/poem/
        physical" alias="poem_physical" grammar ="poem_physical.rnc"/>
        <Structure type="http: //sis.univ -tln.fr/msxd/structure/poem/
        linguistic" alias="poem_linguistic" grammar
         ="poem_linguistic.rnc"/>
        <Structure type="http: //sis.univ -tln.fr/msxd/structure/poem/
        rythmic" alias="poem_rythm" grammar ="poem_rythm.rnc"/>
<Structures>
<Annotations>
        <Annotation type="http: // sis.univ -tln.fr/ annot"
        alias="rythm_annot" grammar ="rythm_annotation.rnc"/>
<Annotations>
<Constrains>
        <!-- RELATIVE CONSTRAINTS BETWEEN STRUCTURES -->
        <!-- Sonnet and poem are equals -->
        <Equal >
                <Fragments type="poem_physical" select ="/Sonnet "/>
                <Fragments type="poem_linguistic" select ="/Poem"/>
        </Equal >
        <Equal >
                <Fragments type="poem_linguistic" select ="/Poem"/>
                <Fragments type="poem_rythm" select ="/Poem"/>
        </Equal >
        <!-- Attributes or elements title and author are equals -->
        <Equal >
                <Fragments type="poem_physical" select ="/Sonnet
                /Head/Title"/>
                <Fragments type="poem_rythm" select ="/Poem/@title "/>
        </Equal >
…
        <!-- A reject begins a verse -->
        <Begins >
                <Fragments type="poem_rythm" select ="Rej"/>
                <Fragments type="poem_physical" select ="Verse"/>
        </Begins >
        <!-- A enjambment must overlap two verses -->
        <Overlaps >
                <Fragments type="poem_rythm" select ="Enj"/>
                <Fragments type="poem_physical" select ="Verse"/>
        </ Overlaps >
        <!-- First Sentence begins just after (meets) Head -->
        <Meets >
                <Fragments type="poem physical" select ="Head"/>
                <Fragments type="poem_linguistic" select ="Sentence
                 [1]"/>
        </Meets >
…
</Constrains>
</MsXmlSchema>
```

Figure II.17. Extrait d’un schéma de document multistructuré (Bruno et Murisasco 2006).

Le modèle MSXD permet de représenter les différentes structures associées à un document les unes indépendamment des autres. Aucune d’entre elles n’est privilégiée. La

duplication du contenu est l'inconvénient majeur de cette solution. Si le problème de chevauchement d'éléments ne se présente plus du fait que le contenu est dupliqué, le problème de cohérence est le problème de base à résoudre. Cette cohérence est assurée par des relations et des contraintes définies dans le schéma de document multistructuré. Or, la création d'un tel schéma requiert des traitements volumineux comme l'analyse simultanée de toutes les structures et l'extraction des contraintes à expliciter.

Afin d'interroger les documents multistructurés représentés selon le modèle MSXD, les auteurs proposent une extension de XQuery dans la mesure où leur modèle est proche de celui utilisé par XQuery et XPath (Fernandez et al. 2002). Ces derniers utilisent le modèle XDM (**X**ML **D**ata **M**odel). Ce modèle est basé sur la notion de séquences et d'item : une séquence est ensemble d'item et un item peut être une valeur atomique ou un nœud. L'extension du modèle XDM consiste à définir un item comme une valeur atomique ou un fragment au lieu d'un nœud. De cette façon, la structure de XQuery n'est pas modifiée. Les extensions ne concernent donc que la sémantique du filtre. Les relations d'Allen sont également utilisées pour gérer le chevauchement d'éléments. Pour la prise en compte de ces relations, des fonctions et des opérateurs ont été ajoutés à XQuery. A titre d'exemple les fonctions « parents » et « children » sont étendues pour retourner respectivement tous les parents et tous les enfants d'un fragment à partir de toutes structures contenant ce fragment.

III.4. Le modèle MCT

Jagadish et al. proposent deux approches. La première appelée « shallow approach » consiste à représenter chaque structure indépendamment des autres sous forme de « petites » arborescences (peu profondes). Ces structures ont l'avantage d'être normalisées, comme par exemple une normalisation selon la forme normale XNF (Arenas et Libkin 2002). En contrepartie, l'interrogation de ces structures nécessite une requête complexe admettant plusieurs jointures. La deuxième approche appelée « deep approach » consiste à représenter l'ensemble des structures dans une même arborescence. Bien que l'interrogation d'une telle arborescence soit beaucoup moins complexe que celle de la première approche, cette arborescence n'est pas normalisée et admet des nœuds redondants. Afin de résoudre ces problématiques, Jagadish et al. proposent le modèle MCT (Jagadish et al. 2004).

MCT (Multi-Colored Trees) est un modèle de représentation de données structurées sous forme d'arbres XML. Chacune des arborescences est identifiable par une couleur unique. Les nœuds d'un même document peuvent avoir une ou plusieurs couleurs qui définissent leur appartenance à une arborescence particulière, l'ensemble des nœuds de même couleur forme une hiérarchie. Chaque nœud admet donc des propriétés additionnelles relatives à la couleur. Ainsi, ce modèle peut être considéré comme une extension du modèle de données XML du fait qu'il reprend les concepts de base de ce modèle en intégrant la notion de coloration. En se basant sur le même principe, celui de coloration, les auteurs proposent un schéma de documents multistructurés inspiré de celui des documents XML (Wiwatwattana et al. 2006).

La Figure II.18 présente un exemple de document « movie » selon le modèle MCT. Trois structures sont présentées en utilisant trois colorations différentes : bleu, rouge et vert. Les nœuds sont représentés par des cercles. Les cercles doubles à deux couleurs différentes représentent les nœuds partagés entre les structures. Par exemple, le nœud « movie-role » est partagé par les structures bleu et rouge.

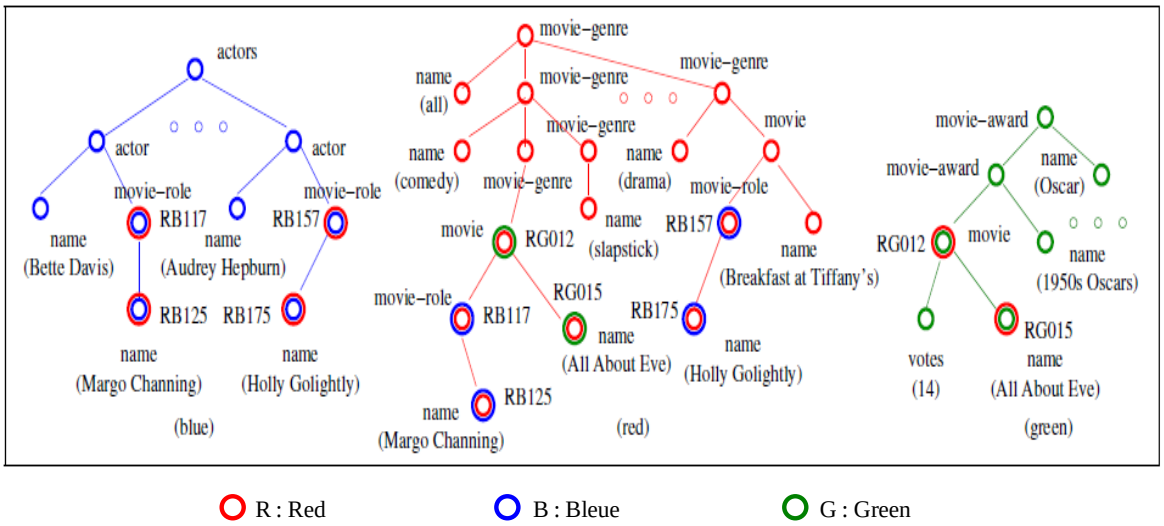


Figure II.18. Exemple de document représenté selon le modèle MCT (Jagadish et al. 2004).

Pour interroger les documents représentés selon le modèle MCT, Jagadish et al. proposent une extension de XQuery appelée MCXQuery (Multi-Colored XQuery). Cette extension consiste en un ensemble de primitives XPath qui permettent de tenir compte de la couleur des nœuds et par conséquent de naviguer dans les arborescences.

A titre d'exemple, la requête qui permet de retrouver les noms des acteurs de films nominés pour un Oscar et dont le titre contient le mot « Eve », s'écrit en MCXQuery de la façon suivante :

```
for     $m in document("mdb.xml")/{red}descendant::movie-genre[{red}
          child::name = "Comedy"]/{red}descendant::movie[contains({red}
          child::name, "Eve")],
        $m in document("mdb.xml")/{green}descendant::movie-award
          [contains({green}child::name,"Oscar")]/{green}descendant::movie
return createColor(black, <m-name> {$m/{red}child::name}</m-name>)
```

La contrainte exercée sur le mot « Eve » est appliquée à la structure rouge alors que la contrainte exercée sur le mot « Oscar » est appliquée à la structure verte.

Le modèle MCT a l'avantage d'être compatible avec XML. La multicoloration des structures peut être une bonne solution afin d'identifier visuellement chacune des structures. En revanche, la couleur est considérée comme une simple étiquette qui joue le même rôle qu'un mécanisme d'espaces de noms. Toute autre étiquette déterminant l'appartenance d'un nœud à une structure peut jouer le même rôle. Même si ce modèle autorise la définition de contenus différents entre structures, son principal objectif était de permettre le partage de données entre ces différentes hiérarchies. Ainsi, les auteurs supposent que le contenu partageable admet une seule segmentation malgré les différentes

structures concourantes qu'il peut avoir. De ce fait, ils ne proposent aucune solution pour gérer le chevauchement d'éléments entre ces structures.

III.5. Le modèle GODDAG

En se basant sur la notion de partage de nœuds et par conséquent de contenu, (Sperberg-McQueen et Huitfeldt 2000) proposent le modèle GODDAG (**G**eneralized **O**rdered-**D**escendant **D**irected **A**cyclic **G**raph). Comme son nom l'indique, un GODDAG est un graphe de nœuds orienté, ordonné et acyclique dans lequel chaque nœud peut être soit un nœud feuille soit un nœud intermédiaire appelé nœud non-terminal. Chaque nœud feuille est étiqueté par une chaîne de caractères et chaque nœud non-terminal est étiqueté par un identifiant générique. Lors de l'étape de construction d'une instance du modèle GODDAG, un nœud de type PCDATA est associé au nœud feuille afin de stocker le contenu du document. Le partage de contenu est assuré via le partage de ces nœuds feuille. Pour gérer le chevauchement entre les différentes structures concourantes, chaque nœud fils peut avoir plusieurs nœuds parents.

Figure II.19 présente deux éléments « a » et « b ». Le contenu du premier élément est « John likes » et le contenu du deuxième élément est « likes Mary ». Ainsi, les deux éléments « a » et « b » se chevauchent. Pour résoudre ce chevauchement, chaque fragment de contenu est rattaché à un élément de type « #PCDATA ».

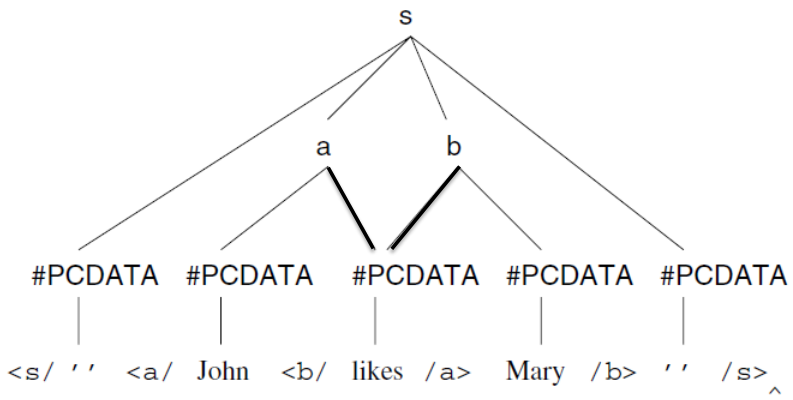


Figure II.19. Exemple de partage de contenu selon le modèle GODDAG (Sperberg-McQueen et Huitfeldt 2000).

L'implémentation de ce modèle passe par l'extension du DOM pour la représentation de documents XML multi-hiérarchiques. Une deuxième solution pour gérer les GODDAG proposée par (Iacob et al. 2004) consiste en l'élaboration d'un compilateur qui traduit les documents multistructurés sous forme de DXD (Document XML distribué : ensemble de documents XML qui partagent la même racine et le même contenu) (Dekhtyar et Iacob 2003) (Dekhtyar et Iacob 2005).

Les langages d'interrogation et de manipulation de structures arborescentes tels que XQuery, XPath, XSLT sont spécifiques à l'interrogation et manipulation du modèle de données XDM (Fernandez et al. 2007) qui ne s'applique qu'aux structures arborescentes.

Aussi, Le Maitre (Le Maitre 2006) propose d'étendre le modèle XDM par ce qu'il appelle les « nœuds retard ». Un « nœud retard » est la représentation virtuelle d'une partie des enfants du nœud père au travers d'une expression XQuery. Ceci permet aux différents nœuds de partager virtuellement leurs nœuds fils. Le principal avantage des « nœuds retard » réside dans la possibilité d'utiliser le langage XQuery classique pour l'interrogation de ces documents.

Le modèle GODDAG assure la coexistence des structures multiples en les représentant sous forme de graphe. Le chevauchement d'éléments entre structures est géré par la fragmentation des données partageables. Même si cette proposition est caractérisée par une compatibilité avec les applications et les outils XML, l'inconvénient majeur est associé à la reconstruction des structures les unes indépendamment des autres. En effet, l'absence de marqueur spécifique à chaque structure rend leur identification difficile.

III.6. Le modèle EMIR²

EMIR² « Extended Model for Image Representation and Retrieval » (Mechkour 1995) est un modèle conçu afin de représenter une image fixe. L'objectif de ce modèle est d'offrir plusieurs descriptions d'une même image. A travers un ensemble de « vues » appelées encore « facettes », ce modèle combine diverses interprétations de l'image permettant d'établir une description complète de son contenu. La Figure II.20 détaille le modèle EMIR² selon BNF (Backus–Normal Form).

```
1. <Image>          ::= <VuePhysique> [ <VueLogique> ]
2. <VuePhysique>    ::= <BitMap>|<GreyScale>|<Colour>|<TrueColour>
3. <VueLogique>     ::= { <VueSymbolique> } <VueStructurelle>
   <VueSpatiale> <VuePerspective>
4. <ObjetImage>     ::= <IdObjetImage> ( [ <VueSpatiale> ], {
   <VueSymbolique> }, [ <VueStructurelle> <VuePerspective> ]) |
   <IdObjetImage>
5. <VueStructurelle>::= {<ObjetImage>}
6. <VueSpatiale>    ::= <IdVueSpatiale> [<ObjetSpatiale>]
   {(<RelationSpatiale> <VueSpatiale>)}|<IdVueSpatiale>
7. <RelationSpatiale>::= <RelationTopologique>|<RelationVectorielle>|
   <RelationMétrique>
8. <ObjectSpatiale> ::= <Point>|<Segment>|<Polygone>|
   {<ObjectSpatiale>}
9. <RelationTopologique>    ::= Dans|Disjoint|Chevauche|Touche|Croise
10.  <RelationVectorielle>  ::= Nord|Sud|Est|West
11.  <RelationMétrique>     ::= Proche|Loin
12.  <VuePerspective>       ::= <Couleur>|<Luminosité>|<Texture>
13.  <VueSymbolique>::= <Attribut>|<Classe>
14.  <Attribut>     ::= <NomAttribut> :<Domaine>|Set(<Attribut>)|Liste
   (<Attribut>|[<Tuple>]
15.  :<Domaine>     ::= <Entier>|<Réel>|<Chaine>|<Booléen>
```

Figure II.20. Modèle de EMIR² selon BNF (Backus–Normal Form).

Les facettes proposées dans le modèle EMIR² (Cf. Figure II.20) sont classifiées en deux niveaux de description :

– le niveau physique permet de décrire des caractéristiques de bas niveau de l'image. A ce niveau, l'image est définie par une matrice de pixels. Des objets peuvent être identifiés dans cette matrice par des régions ;

– le niveau logique rassemble toutes les facettes décrivant le contenu de l'image :

- la facette structurelle définit l'ensemble des objets de l'image qui sont considérés par l'indexeur comme étant les plus pertinents pour la description de cette image. Chaque objet de l'image peut être un objet simple ou complexe. Les objets complexes peuvent être décrits par d'autres facettes de ce même niveau logique,
- la facette spatiale décrit la forme des objets ainsi que les relations spatiales entre ses objets. Ces relations peuvent être des relations topologiques (dans, disjoint, chevauche, touche, croise, etc.), des relations vectorielles (nord, sud, est ou ouest) ou des relations métriques (loin, proche, etc.),
- la facette perspective inclut tous les attributs visuels de l'image et/ou des objets de l'image. Elle décrit l'apparence des composantes de l'image telles qu'elles sont perçues par l'observateur. Le modèle EMIR² considère essentiellement trois attributs visuels : la couleur, la luminosité et la texture,
- la facette symbolique associe une description sémantique à l'image et/ou aux objets de l'image. Elle est composée par des attributs de nature entiers, réels, chaîne de caractères ou booléens.

Afin de décrire les documents vidéo, (Charhad et Quénot 2004) proposent d'étendre le modèle EMIR² par un ensemble de facettes spécifiques. Les auteurs ont classifié les facettes proposées en deux catégories :

– les facettes génériques regroupent l'ensemble des facettes qui décrivent les caractéristiques communes dans un document vidéo indépendamment de sa décomposition en média (images, bande audio et texte), telle que par exemple la nature temporelle de la vidéo. Deux types de facettes ont été définis :

- les facettes temporelles qui représentent l'ensemble des relations temporelles reliant les éléments d'informations dans le document vidéo,
- les facettes événementielles qui décrivent les différents événements contenus dans un document vidéo. Un événement est considéré comme une ou plusieurs actions. Une action est un fait qui survient à un moment défini d'une séquence vidéo ;

– les facettes spécifiques permettent une description du contenu vidéo par média. En effet, la décomposition d'un document vidéo en un ensemble de média (image, audio ou texte) offre une description spécifique orientée média. La représentation spécifique d'un document vidéo contient les facettes suivantes :

- la facette sémantique associe une description sémantique du contenu visuel, du contenu audio ou au contenu textuel. Cette description est souvent définie par des métadonnées traduites en concepts. Chaque facette sémantique peut être composée de trois autres facettes. Une sous-facette visuelle pour décrire les différents frames (image) constituant la séquence vidéo, une sous-facette audio pour interpréter le contenu audio et une sous-facette texte pour traiter toutes les informations textuelles qui appartiennent à ce document,
- la facette signal assure la description des caractéristiques de bas niveau afin de générer des descriptions sémantiques telles que les caractéristiques des couleurs dans le contenu d'un frame. La facette signal regroupe plusieurs sous-facettes, notamment lorsqu'il s'agit de la description du niveau visuel. Chaque sous-facette présente une caractéristique spécifique telle que la couleur, la texture ou les positions spatiales des objets visuels.

La Figure II.21 illustre l'ensemble des facettes d'un document vidéo selon (Charhad 2005).

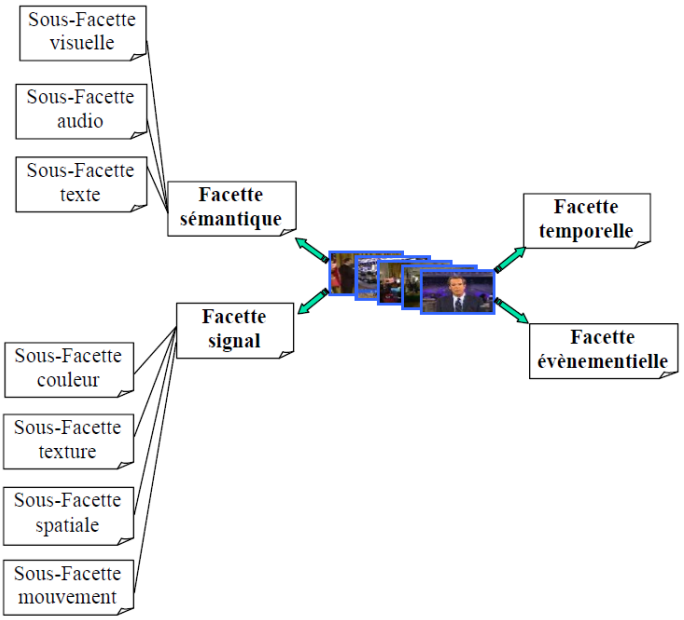


Figure II.21. Modélisation multifacette d'un document vidéo (Charhad 2005).

Le modèle EMIR² est l'un des premiers modèles conçus pour la description d'un document non textuel. Ce modèle permet de représenter une image au travers d'un ensemble de vues. Ces vues traitent différents aspects de l'image. L'extension du modèle EMIR² découle essentiellement du besoin de décrire les documents vidéo. Si l'on considère que le document vidéo est un ensemble d'images synchronisées avec une bande audio, l'extension du modèle EMIR² doit traiter l'aspect dynamique des documents vidéo.

L'inconvénient majeur de l'ensemble de ces deux approches réside dans l'aspect statique des structures proposées. En effet, elles sont figées et proposent une liste d'attributs fixes, ce qui engendre un manque de flexibilité dans la représentation des

documents. Avec de telles approches, nous ne pouvons pas exploiter que les descripteurs identifiés dans la structure de base proposée. Cela implique d'autre part, que tous les descripteurs doivent être renseignés même avec des valeurs nulles et d'autre part que l'on ne peut rajouter aucun nouveau descripteur.

L'objectif de ces modèles est d'offrir plusieurs descriptions d'une image ou d'une séquence vidéo afin de faciliter leur exploitation au sein d'un processus de recherche d'informations classique. Ainsi, le partage de contenu entre ces descriptions n'est pas abordé. De ce fait, les problèmes de chevauchement d'éléments, de cohérence et de restitution ne sont pas traités.

III.7. Le modèle de Fourel

Fourel et al. ont défini un modèle afin de représenter les composantes documentaires exploitables durant un processus de recherche d'information (Fourel et Mulhem 1996). Ce modèle se base sur la notion de vue définie dans le modèle EMIR². Cependant, les auteurs ont défini d'autres types de vues (Cf. Figure II.22) : la vue structurelle, la vue physique, la vue attribut interne, la vue attribut externe, la vue position, la vue relation et la vue sémantique notées respectivement *Str*, *Phy*, *EA*, *IA*, *Pos*, *Rel* et *Sem*. La vue structurelle est considérée comme une structure de base de ce modèle puisqu'elle représente la structure logique du document. Le modèle de Fourel représente les relations d'interdépendance entre les vues d'un même document. L'interdépendance est définie par six types de relations permettant l'association d'un élément de la vue structurelle avec un élément appartenant à l'une des autres vues. Ces relations sont notées : R_{Phy}, R_{EA}, R_{IA}, R_{Pos}, R_{Rel} et R_{Sem} dans la Figure II.22.

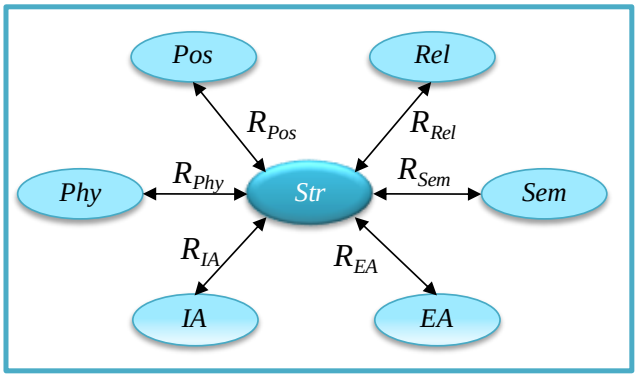


Figure II.22. Les vues et leurs inter-relations.

Les sept vues du modèle de Fourel sont :

- la vue structurelle (Str) représente la structure logique du document. De ce fait, les auteurs la considèrent comme étant le noyau de leur modèle. Cette vue établit des relations avec le reste des vues ;
- la vue physique (Phy) est une vue liée à la nature des différentes parties du contenu. Selon cette nature (texte, image, etc.), la vue physique doit représenter des informations qui peuvent être utiles dans la recherche du document ;

- la vue attribut externe (AE) regroupe des métadonnées qui n'ont pas de rapport direct avec le contenu du document multimédia d'où la nomination « externe ». Par exemple, il est possible de retrouver dans cette vue l'auteur du document, la date de sa création, etc. ;
- la vue attribut interne (AI) regroupe les métadonnées qui décrivent le contenu lui-même du document ou d'une partie du document ;
- la vue position (Pos) reflète la disposition spatiale des différents composants du document multimédia les uns par rapport aux autres ;
- la vue relation (Rel) représente les différents liens internes et externes qui peuvent exister dans les documents multimédias ;
- la vue sémantique (Sem) est définie pour décrire la sémantique du contenu multimédia des documents.

Dans le modèle de Fourel, l'ensemble des vues s'articule autour de la vue structurelle qui représente la structure logique. Cette vue admet une structuration unique et indépendantes des autres vues. De plus, ces dernières ne représentent pas des structures concourantes à la structure logique. Ainsi, la problématique de chevauchement d'éléments n'est pas abordée. De ce fait, les problèmes de cohérence et de restitution ne se posent pas également. Si les éléments de ces vues peuvent varier d'un document à un autre, la nature des vues définies est statique. De plus, ce modèle tel qu'il est conçu ne permet pas de définir deux vues de même nature.

III.8. Le modèle de Mbarki

Dans le but de prendre en compte et de représenter plusieurs descriptions d'une même composante multimédia rattachée à une structure logique d'un document, Mbarki et al. proposent un modèle (Mbarki 2008) qui assure la représentation des documents multimédias (Cf. Figure II.23). Ce modèle offre :

- une description structurelle qui permet de traduire la structure logique d'un document. Elle décrit ses éléments et ses attributs. Cette description est unique pour un même document ;
- une description des métadonnées qui permet de décrire une composante multimédia de plusieurs façons. Ainsi, plusieurs descriptions peuvent être rattachées à un même élément de la structure logique.

Dans la Figure II.23, les deux descriptions sont présentées. Chaque description admet un niveau spécifique et un autre générique. Dans la description structurelle, le niveau spécifique est dédié à la représentation de la structure logique spécifique à un document et le niveau générique est dédié à la représentation d'une structure logique générique qui regroupe plusieurs structures logiques similaires. La description des métadonnées est traduite par une structure spécifique des métadonnées qui représente les métadonnées spécifiques à une composante multimédia et une structure générique de métadonnées qui regroupe les structures de métadonnées spécifiques similaires.

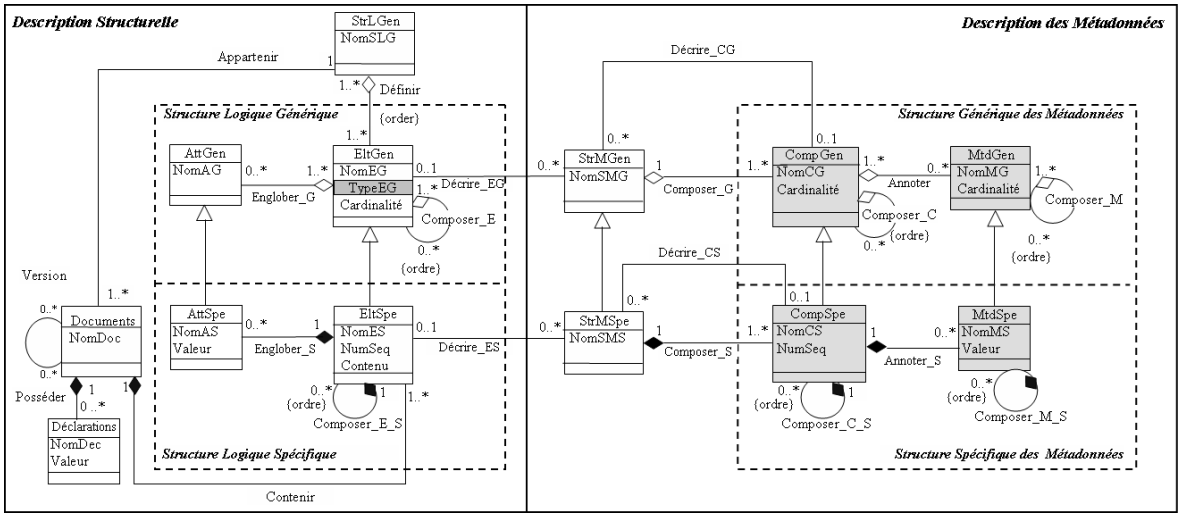


Figure II.23. Modèle de Mbarki et al.

Ce modèle a servi de base pour la définition d'un entrepôt de documents multimédias. Cet entrepôt permet la centralisation des informations documentaires afin de faciliter l'accès et les analyses qui peuvent être menées sur les documents qu'il contient. Ainsi, les auteurs proposent deux types d'exploitations des documents multimédias à savoir :

- la recherche d'informations qui consiste à retrouver des granules documentaires ou des passages de documents en réponse à une requête utilisateur. Etant données la complexité et la richesse des documents multimédia, Mbarki et al. basent leurs recherches sur les granules ce qui permettra d'obtenir des résultats plus précis ;
- l'analyse multidimensionnelle qui consiste à analyser les informations documentaires de l'entrepôt selon plusieurs dimensions. Elle se base sur la création de vues temporaires pour visualiser d'une manière graphique et synthétique le contenu d'une grande masse documentaire en présentant un sujet d'analyse selon plusieurs axes. Le choix de ces paramètres (sujet et axes d'analyses) est assez souple et dynamique.

Ce modèle est caractérisé par sa flexibilité dans la définition des structures de métadonnées qui peuvent être rattachées à une même composante. Ces structures s'articulent autour d'une seule structure de base : la structure logique. Cependant, la définition de structures multiples se limite aux composantes du document et non pas à la globalité du document.

III.9. Graphe d'annotation

Les graphes d'annotations (Bird et Liberman 2001) ont été définis dans l'objectif de spécifier un formalisme de représentation de plusieurs annotations sur une même source de données. Ils ont été appliqués dans les traitements et les analyses de langues en prenant en compte simultanément les domaines linguistiques : phonétique, morphologie, syntaxe, prosodie, etc. Cette proposition s'appuie sur le partage de granules de données. Ces

données sont de nature textuelle. Ainsi, les nœuds du graphe sont composés de fragments textuels. Ces nœuds sont reliés entre eux par des arcs étiquetés. Les étiquettes indiquent le type de marqueur spécifique à une structure. Ceci permet d'identifier chaque structure indépendamment des autres. A partir de chaque graphe, il est possible d'extraire un fichier XML correspondant aux différentes structures représentées. Le modèle physique d'un tel fichier étant une hiérarchie de balises associées à un contenu textuel, il préconise l'usage des attributs de type ID IDREF afin de conserver une structure sous forme de graphe.

Figure II.24 illustre un exemple d'instanciation des graphes d'annotation. Deux structures sont présentées. La première structure est dédiée à un découpage en lignes (structure physique) alors que la deuxième est consacrée à un découpage en phrase (structure logique).

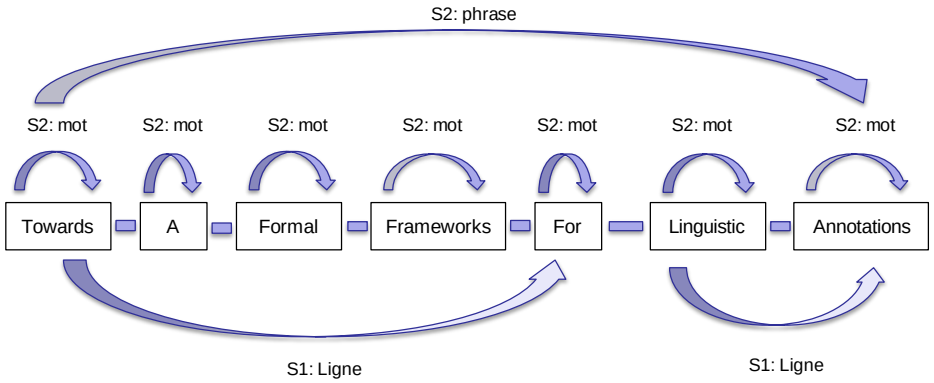


Figure II.24. Exemple d'un extrait de document représenté par des graphes d'annotation.

Les graphes d'annotation imposent un modèle formel propriétaire pour la représentation des annotations multiples sur un même flux de données. Le partage des fragments stockés dans les nœuds du graphe peut être considéré comme le principal avantage de cette représentation. Cependant, ce partage de nœuds nécessite la fragmentation du contenu associé aux nœuds qui se chevauchent, jusqu'au plus petit fragment commun entre les différentes structures. Ceci engendre une décomposition des nœuds eux-mêmes. De plus dans le cas d'une mise à jour de structure (ajout ou modification), ceci implique de reconstruire tous les liens des nœuds impactés par la fragmentation. Enfin il est difficile de représenté ce graphe en XML avec des outils standards.

III.10. Synthèse des modèles

La représentation des structures au travers des modèles s'articule autour de deux aspects : la fusion et de la dépendance des structures. Certains modèles préconisent la représentation indépendante des structures. Dans ce cas, il est nécessaire d'établir les relations entre ces structures afin de pouvoir les exploiter simultanément.

Autour de ces différents aspects de représentation émergent des solutions au problème de chevauchement d'éléments. Si dans le cas de la fusion de structures, ce

problème est résolu par fragmentation du contenu, dans le cas où les structures sont représentées les unes indépendamment des autres, deux cas se présentent :

- rattachement du contenu aux structures. Ceci revient à la duplication du contenu. Si le problème de chevauchement ne se présente plus (chaque structure admet un contenu qui lui propre), il devient nécessaire d'assurer la cohérence du contenu dupliqué entre les différentes structures ;
- dissociation du contenu et des structures. Le contenu est donc représenté dans une nouvelle dimension. Cette dimension peut être une structure ou une vue. La gestion de chevauchement est assurée par fragmentation ou par indexation du contenu représenté dans la nouvelle dimension.

IV. Synthèse

Si la représentation de plusieurs structures au travers d'un langage impose leur fusion, les modèles offrent plus de flexibilité et de souplesse dans la représentation de ces structures. En exploitant cette flexibilité et cette souplesse, plusieurs représentations des structures multiples sont proposées : partage de contenu, partage de nœuds, relations entres structures.

Quelque soit l'approche utilisée, langage ou modèle, la représentation des structures multiples est contrainte par un ensemble de problèmes : le partage de contenu, la cohérence des données et la restitution.

Un comparatif des travaux étudiés dans ce chapitre est présenté dans le Tableau II.1 en fonction des problématiques recensées.

- Partage de contenu :
  - modèle de données : détermine la nature du modèle qui décrit les objets d'une structure et leurs relations, par exemple le modèle en arbre. Dans ce critère, nous décrivons le modèle de représentation d'un document multistructuré,
  - gestion de chevauchements d'éléments : nous spécifions la méthode de gestion de chevauchement d'éléments adoptée,
  - structure privilégiée : ce critère permet de déterminer si une structure est privilégiée et est considérée comme structure de base,
  - partage de nœuds : ce critère renseigne sur la possibilité de partager des nœuds entre les différentes structures,
  - nature du contenu : indique la nature des documents supportés par le formalisme ou le modèle correspondant ;

- Cohérence de données :
  - gestion de relations inter structures : nous examinons si les relations entre structures sont prises en compte ;

- Restitution
  - compatibilité XML/SGML : une proposition est compatible avec XML/SGML si elle utilise la même syntaxe et obéit aux règles de construction de ce langage ou bien s'elle manipule en entrée/sortie des documents traduits dans ce langage,
  - exploitation : nous nous intéressons aux techniques d'exploitation correspondant à la solution proposée.

Tableau II.1.Comparatif des travaux sur la multistructuralité.

	Modèle de données	Structure privilégiée	Partage de nœuds	Gestion des chevauchements d'éléments	Gestion des relations inter structures	Nature du contenu	Compatibilité XML/SGML	Exploitation
Concour/ XConcure	Arbre	Non	Oui	Fragmentation	Non	Mono média : Texte	Syntaxe SGML/XML	Pas de langage spécifique présenté
TEI	Structure complexe sous forme arborescente	Oui	Oui	Fragmentation	Non	Mono média : Texte	Syntaxe XML	Pas de langage spécifique présenté
LMNL	States sous forme arborescente	Non	Oui	Indexation	Non	Mono média : Texte	Non	Extension de XPath
MECS/ TexMECS	GODDAG	Non	Oui	Fragmentation	Non	Mono média : Texte	Non	Pas de langage spécifique présenté
RDF	Modèle de données RDF	Oui	Non	Fragmentation	Non	Multimédia	Syntaxe XML	Pas de langage spécifique présenté
MSDM	Graphe propriétaire	Oui	Les nœuds de la structure de base	Fragmentation	Oui	Multimédia : Texte et Image	Compatible XML (entrée/sortie)	Extension de XQuery
Proximal nodes	Arbre	Oui	Non	Indexation	Non	Mono média : Texte	Compatible XML (entrée/sortie)	Extension de XQuery
MSXD	Arbre	Non	Non	Duplication	Oui	Mono média : Texte	Syntaxe XML (document XML par structure)	Extension de XQuery
Arbres XML Colorées	Arbre	Non	Oui	Fragmentation	Via le partage de nœuds	Mono média : Texte	Compatible XML (entrée/sortie)	Extension de XQuery
GODDAG	GODDAG	Non	Les nœuds racines	Fragmentation	Non	Mono média : Texte	Compatible XML (entrée/sortie)	Extension de XPath
EMIR² + Extension de Chahad	Graphe conceptuel	Non	Non	Non	Oui	Mono média : Image/Vidéo	Non	Extension d'un modèle de RI
Modèle de Forel	Graphe conceptuel	Oui	Les nœuds de la structure de logique (pivot)	Non	Oui	Multimédia	Non	Pas de langage spécifique présenté
Modèle de Mbarki	Arbre	Oui	Les nœuds de la structure de logique (pivot)	Non	Non	Multimédia	Oui (entrée/sortie)	Moteur SQL
Graphe d'annotation	Graphe propriétaire	Non	Oui	Fragmentation	Non	Mono média : Texte	Syntaxe XML (pour sérialisation)	XPath

V. Conclusion

Dans ce chapitre, nous avons présenté un aperçu des différentes approches de gestions de documents multistructurés. Ces travaux sont destinés à décrire des architectures et des mécanismes permettant d'intégrer les différentes structures définies au sein d'un même document. Ces travaux s'articulent autour de deux catégories d'approches : la première est basée sur des langages et la seconde s'appuie sur des modèles. Si les modèles sont indépendants vis-à-vis d'un langage particulier, les langages doivent être basés sur des modèles afin de représenter les documents. De tels modèles servent à définir d'une part, l'organisation des éléments de contenus et d'autre part, les éléments de syntaxes autorisés et la grammaire du langage.

Si nous nous focalisons sur les propositions basées sur des modèles, nous distinguons ceux qui se limitent à la représentation des structures multiples du document et ceux qui se limitent à la représentation des structures multiples d'une entité d'un document. Or, il est évident que la combinaison de ces deux aspects apporte plus de flexibilité et de possibilités d'exploitation. Dans la littérature, seul le modèle MSXD permet d'associer des structures multiples au document ainsi qu'à ses entités. En revanche, vu que chaque structure est représentée indépendamment des autres, les structures relatives à une entité seront associées uniquement à une seule structure, celle qui agrège l'entité décrite, et pas les autres.

La gestion de chevauchement d'éléments entre structures est gérée, dans la majorité de ces propositions, par une approche de fragmentation. En effet, le contenu du document est découpé tant qu'il y a un chevauchement entre les éléments de ses différentes structures. Ce qui permet d'avoir des fragments de tailles différentes qui peuvent aller dans certains cas jusqu'à la taille de l'unité de mesure (caractère par exemple pour du texte). Ceci induit une lourde gestion des différents fragments ainsi que de leurs relations et par conséquent un manque de flexibilité dans l'exploitation et les manipulations possibles des documents.

Tous les travaux de modélisation présentés supposent que chaque document est caractérisé par sa (ses) propre(s) structure(s) spécifique(s). Or, il est fréquent de constater que des documents qui décrivent le même type d'information (curriculum vitae, films documentaires, un type d'émission radio particulier, etc.) possèdent des structures similaires ou sont décrits par un même ensemble de métadonnées. Il serait alors intéressant de pouvoir repérer ces similarités pour déduire des classes génériques de documents et ne pas se contenter de rester à un niveau spécifique. L'utilisation de ces classes génériques permettra d'une part de jouer le rôle d'un schéma de document et d'autre part de focaliser les éventuelles recherches d'informations sur une collection (classe de documents) bien déterminée.

VI. Bibliographie

Abascal, R., Beigbeder, M., Benel, A., Calabretto, S., Chabbat, B., Champin, P. A., Chatti, N., Jouve, D., Prie, Y., et Rumpler, B. (2003). « Modéliser la structuration multiple des documents. » H2PTM, Hermès, Paris, France, 253-258.

Allen, J. F. (1983). "Maintaining knowledge about temporal intervals." Communication ACM, 26(11), 837-843.

Arenas, M., et Libkin, L. (2002). "A Normal Form for XML Documents."

Barnard, D., Burnard, L., Gaspart, J., Price, L., Sperberg-McQueen, C., et Varile, G. (1995). "Hierarchical encoding of text: Technical problems and SGML solutions." Computers and the Humanities, 29(3), 211-231.

Bird, S., et Liberman, M. (2001). "A formal framework for linguistic annotation." Speech Communication, 33(1-2), 23-60.

Brickley, D., et Guha, R. V. (2004). "Rdf vocabulary description language 1.0: Rdf schema." World Wide Web Consortium recommendation .

Bruno, E., et Murisasco, E. (2006). "MSXD: A Model and a Schema for Concurrent Structures Defined over the Same Textual Data." Database and Expert Systems Applications, 172-181.

Burnard, L. (1992). "The Text Encoding Initiative: A progress report." New Directions in English Language Corpora: Methodology, Results, Software Developments, 97.

Charhad, M. (2005). « Modèles de Documents Vidéo basés sur le Formalisme des Graphes Conceptuels pour l'Indexation et la Recherche par le Contenu Sémantique. » Thèse de doctorat, UNIVERSITÉ JOSEPH FOURIER.

Charhad, M., et Quénot, G. (2004). "Semantic video content indexing and retrieval using conceptual graphs." Information and Communication Technologies: From Theory to Applications, 2004. Proceedings. 2004 International Conference on, 399-400.

Chatti, N. (2006). « Documents multi-structurés: De la modélisation vers l'exploitation. » Thèse de doctorat, L'institut National Des Sciences Appliquées De Lyon.

Chatti, N., et Calabretto, S. (2007). « Adaptation de XML et XQuery pour la représentation et l'interrogation des documents multi-structurés. » Saint-Etienne, 109-124.

Chatti, N., Calabretto, S., et Pinon, J. M. (2004). « Vers un environnement de gestion de documents à structures multiples ».

Chatti, N., Calabretto, S., Pinon, J. M., et Kaouk, S. (2007). "MultiX: an XML-based formalism to encode multi-structured documents." Proceedings of Extreme Markup Languages 2007, 2007.

Clark, J., et Murata, M. (2001). "RELAX NG Specification. OASIS Committee Specification." WWW: http://www.relaxng.org/spec-20011203.html .

Dekhtyar, A., et Iacob, I. (2003). "A Framework for Management of Concurrent XML Markup." Conceptual Modeling for Novel Application Domains, <http://www.springerlink.com/content/apx60p65eubpm8la> (Mai. 26, 2009).

Dekhtyar, A., et Iacob, I. E. (2005). "A framework for management of concurrent XML markup." Data Knowl. Eng., 52(2), 185-208.

Dekhtyar, A., Iacob, I. E., et Methuku, S. (2005). "Searching Multi-hierarchical XML Documents: The Case of Fragmentation." Database and Expert Systems Applications, 576-585.

DeRose, S. (2004). "Markup overlap: A review and a horse." Extreme Markup Languages, Citeseer.

Durusau, P., et DeRose, S. J. (2003). "OSIS: A Users' Guide to the Open Scripture Information Standard." Bible Technologies Group.

Durusau, P., et O'Donnell, M. B. (2004). "Tabling the overlap discussion." Extreme Markup Languages 2004.

Fernandez, M., Malhotra, A., Marsh, J., Nagy, M., et Walsh, N. (2002). "XQuery 1.0 and XPath 2.0 data model." W3C Working Draft, 15.

Fernandez, M., Malhotra, A., Marsh, J., Nagy, M., et Walsh, N. (2007). “XQuery 1.0 and XPath 2.0 Data Model (XDM).”

Fourel, F., et Mulhem, P. (1996). “Modelling multimedia structured documents: a retrieval oriented approach.” Proceedings of the 7th International Workshop on Database and Expert Systems Applications, IEEE Computer Society, 179-184.

Goldfarb, C. F. (1990). The SGML handbook. Oxford University Press, Inc., 664.

Hilbert, M., Schonefeld, O., et Witt, A. (2005). “Making CONCUR work.” Extreme Markup Languages.

Hors, A. L., Byrne, S., Champion, M., Nicol, G., Robie, J., Le Hégaret, P., et Wood, L. (2004). “Document object model (DOM) level 3 core specification.” W3C Recommendation. http-J/www. xv3. org/TR/DOMrLevel-3-Core.

Huitfeldt, C. (1993). “MECS-A Multi-Element Code System.” ACH-ALLC, 16-19.

Huitfeldt, C., et Sperberg-McQueen, C. M. (2001). “TexMECS: an experimental markup meta-language for complex documents.”

Huitfeldt, C., et Sperberg-McQueen, C. M. (2004). Markup Languages for Complex Documents–an Interim Project Report.

Iacob, I. E., Dekhtyar, A., et Kaneko, K. (2004). “Parsing concurrent XML.” Proceedings of the 6th annual ACM international workshop on Web information and data management, ACM, Washington DC, USA, 23-30.

Jagadish, H. V., Lakshmanan, L. V. S., Scannapieco, M., di Roma, U., Sapienza, L., Srivastava, D., et Wiwatwattana, N. (2004). “Colorful xml: One hierarchy isn’t enough.” ACM New York, NY, USA, 251-262.

Lassila, O., et Swick, R. (2000). “Resource Description Framework.” IEEE Intelligent Systems, 15(6), 67-69.

Le Maitre, J. (2006). “Describing multistructured XML documents by means of delay nodes.” Proceedings of the 2006 ACM symposium on Document engineering, ACM New York, NY, USA, 155-164.

Mbarki, M. (2008). « Gestion de l’hétérogénéité documentaire : le cas d’un Entrepôt de documents multimédia. » Thèse de doctorat, Université Paul Sabatier.

McKelvie, D., Brew, C., et Thompson, H. S. (1999). “Using SGML as a Basis for Data-Intensive Natural Language Processing.” COMPUTERS AND THE HUMANITIES, 31, 367-388.

Mechkour, M. (1995). “EMIR²: An Extended Model for Image Representation and Retrieval.” DEXA, 395-404.

Murata, M. (1999). “Hedge automata: a formal model for XML schemata.” <http://www.xml.gr.jp/relax/hedge_nice.html>.

Navarro, G., et Baeza-Yates, R. (1997). “Proximal nodes: a model to query document databases by content and structure.” ACM Transactions on Information Systems (TOIS), 15(4), 400-435.

Schonefeld, O. (2008). “A Simple API for XCONCUR Processing concurrent markup using an event-centric API.”

Sperberg-McQueen, C. M., et Burnard, L. (2007). TEI P5: Guidelines for Electronic Text Encoding and Interchange. Oxford, Providence, Charlottesville, and Bergen.

Sperberg-McQueen, C. M., et Huitfeldt, C. (2000). “GODDAG: A Data Structure for Overlapping Hierarchies.” LECTURE NOTES IN COMPUTER SCIENCE, 139-160.

Tennison, J., et Piez, W. (2002). “The Layered Markup and Annotation Language (LMNL).” Extreme Markup, Montreal.

Tennison, J., Piez, W., et Nicol , G. T. (2002). “LMNL: the Layered Markup and Annotation Language.” LMNL, <http://lmnl.net/index.html>.

Tummarello, G., Morbidoni, C., et Pierazzo, E. (2005). “Toward textual encoding based on RDF.” Proceedings ELPUB'2005.

Whitmer, R., Wood, L., et Le Hégaret, P. (2004). “W3C. Document Object Model (DOM), DOM level 1, 2 and 3 specifications.”

Wiwatwattana, N., Jagadish, H. V., Lakshmanan, L. V. S., et Srivastava, D. (2006). “Making Designer Schemas with Colors.” Proceedings of the 22nd International Conference on Data Engineering, IEEE Computer Society Washington, DC, USA, 15.

DEUXIEME PARTIE :
NOTRE PROPOSITION : MODELISATION, INTEGRATION ET EXPLOITATION DE DOCUMENTS A STRUCTURES MULTIPLES

Chapitre III – Modélisation de documents à structures multiples

Résumé du chapitre. *Ce chapitre présente le modèle de documents multistructurés que nous avons proposé. Ce modèle appelé « MVDM » (**M**ulti**V**iew **D**ocment **M**odel) est organisé autour du concept de vue. Une vue correspond à une organisation particulière d'un document. Elle traduit l'une des structures d'un document à structures multiples. Le modèle comprend deux niveaux de description. Le niveau spécifique décrit les documents eux-mêmes au travers de leurs structures spécifiques. Le niveau générique décrit des structures génériques en tant que classes de structures spécifiques similaires ou identiques.*

Sommaire du Chapitre III.

I. Introduction

Les documents multimédias sont des objets complexes et hétérogènes, fédérant des entités documentaires de nature différente, répondant à des représentations différentes. La coexistence de plusieurs médias engendre de nouvelles dimensions au sein d'un même document. Ainsi, la représentation des structures multiples d'un document multimédia impose des enjeux supplémentaires par rapport à la représentation des structures multiples d'un document monomédia. En plus des enjeux de partage de contenu, de restitution multiple des documents et de corrélation des structures concourantes, il est nécessaire d'assurer la coexistence de ces structures définies sur des niveaux différents du document multimédia (niveau global ou au niveau d'un contenu monomédia intégré dans le document).

L'objectif de ce chapitre est donc de présenter un modèle qui permette de représenter les différentes structures qui peuvent être perçues dans un document mono ou multimédia. Si les différentes approches de gestion de la multistructuralité présentées dans l'état de l'art (Cf. Chapitre II) permettent la représentation des structures multiples d'un même document monomédia, ils n'abordent pas les problématiques de représentation de plusieurs structures issues de plusieurs médias retrouvés dans un même document.

Le modèle proposé doit pouvoir décrire des structures de même type ou de type différent. Il doit tenir compte ainsi de la diversité typologique des relations qui lient les différentes entités d'un document. Dans les approches de l'état de l'art, on ne prend en compte qu'une seule relation, non typée, entre deux entités la relation hiérarchique.

L'étude des approches de gestion des structures multiples de la littérature nous a permis de distinguer deux catégories d'approches. Les approches de la première catégorie sont basées sur les langages et celles de la deuxième catégorie sont basées sur des modèles. Nos travaux se positionnent tout naturellement dans la deuxième catégorie d'approches. La clarté de représentation des structures multiples et l'adaptation à tous les contextes sont les principaux avantages de cette approche. En effet, la modélisation des différents concepts liés à un document nous offre :

- une vision concise et précise de sa composition en terme de nœuds, de relations entre nœuds et de structures qui en découlent ;
- une représentation indépendante vis-à-vis d'un langage donné ce qui favorise l'adaptation du document à toutes les plateformes dans tous les contextes.

Nous présentons dans ce chapitre, dans un premier temps, notre démarche de modélisation de documents multistructurés en l'illustrant par des exemples. Ensuite, nous traitons la notion de classification de documents au travers de la modélisation de la structure générique d'une collection de documents structurellement similaires. Le modèle MVDM est dans un premier temps décrit selon un formalisme UML et ensuite de façon formelle.

II. Modélisation spécifique d'un document à structures multiples

II.1. Objectif

Un document, quelle que soit sa nature (monomédia ou multimédia), peut être décrit par une ou plusieurs structures. Ces structures sont soit de natures différentes (une structure logique et une structure sémantique), soit de même nature (par exemple deux structures sémantiques définies par deux personnes différentes). Nous proposons donc d'englober la notion de structure dans une notion plus large qui est celle de ***vue***.

Une vue peut avoir deux significations (Ribière et al. 2002). Elle peut désigner un angle de vision en se focalisant sur un certain aspect de l'entité étudiée (si l'entité étudiée est un avion, une vue peut se focaliser sur l'aspect sécurité), comme elle peut indiquer une opinion en considérant une interprétation particulière de l'entité étudiée (une vue peut traduire l'interprétation d'un expert du domaine sur l'aspect sécurité de l'avion). Dans les deux cas, le terme « vue » présente une utilisation ou une dimension d'interprétation qui donne un sens aux différentes entités étudiées.

Dans notre cas, nous définissons une vue comme étant une représentation qui traduit une organisation du document. Cette représentation peut se focaliser sur un certain aspect du document tel que par exemple l'aspect temporel, comme elle peut offrir une interprétation particulière du document due à un usage particulier. Si l'organisation d'un document est déduite à partir de sa structure, la définition des vues va nous permettre de matérialiser des structures non seulement de même type, mais également de types différents.

La représentation des vues est basée sur le concept de fragmentation du document qu'elle décrit en entités ou nœuds structurants. Ces nœuds doivent donc permettre d'identifier, de décrire et de caractériser chaque entité du document. De fait, chaque entité ou nœud peut être décrit séparément et lui-même fragmenté.

Les relations entre les entités (nœuds) offre la possibilité de reconstruire l'intégralité du document en lui donnant une forme. La définition de plus d'une relation entre deux entités permet de matérialiser plus d'une organisation possible pour un même document. La diversité typologique des relations permet de définir des organisations de natures différentes. A ce niveau, et contrairement à ce qui est présenté dans la littérature, nous proposons de représenter des relations typées, hiérarchiques et non hiérarchiques (Djemal et al. 2007).

Une vue traduit ainsi une forme du document au travers de la mise en correspondance de ses nœuds. Cette mise en correspondance est concrétisée par les relations.

Les entités composant un document peuvent être des fragments textuels facilement gérables, mais également des composantes multimédias plus complexes à gérer. De telles entités, considérées comme des documents en soient, doivent être fragmentées à leur tour

formant ainsi des structures à associer à des nœuds. En conséquence, il est possible de définir des structures multiples au niveau du document lui-même et des structures multiples associées à un nœud de ce même document.

La Figure III.1 illustre une possibilité de coexistence de ces deux types de structures. Nous présentons deux structures $StrGl_1$ et $StrGl_2$ globales au document qui représentent deux fragmentations globales différentes d'un même contenu. Une partie de ce contenu désigne une image. Cette image est décrite par deux autres structures $StrAs_1$ et $StrAs_2$.

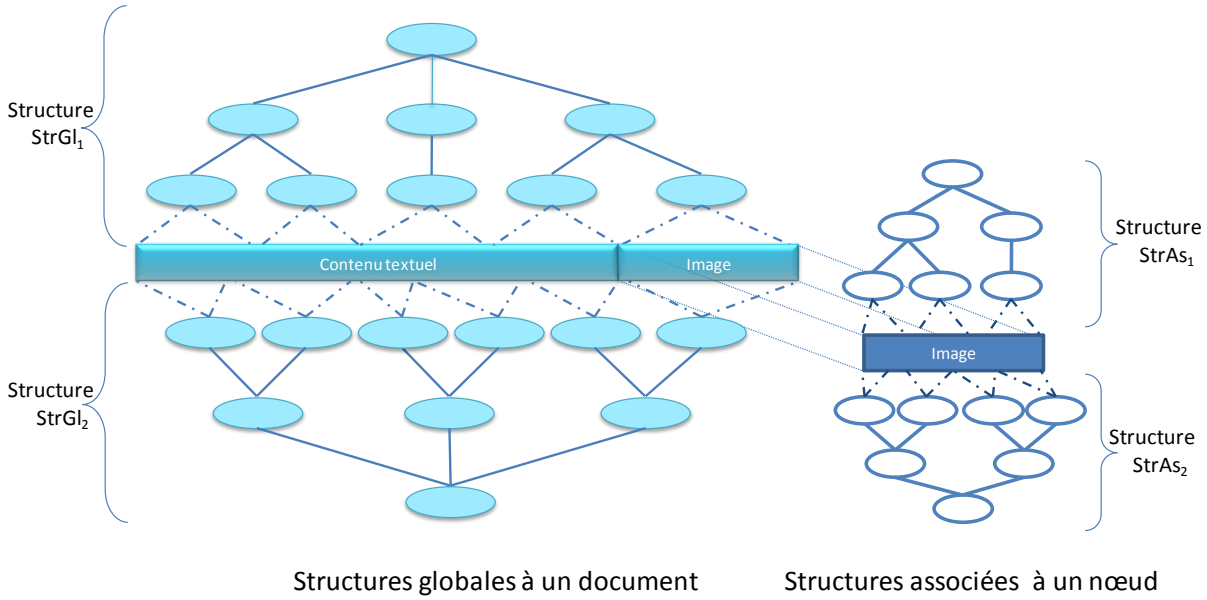


Figure III.1. Illustration de la coexistence de structures à différents niveaux du document.

Dans la section suivante, nous détaillons notre modélisation spécifique de documents à structures multiples. Nous décrivons les différentes métaclasses utilisées ainsi que leur rôle dans la représentation d'un document.

II.2. Modèle spécifique et description des différentes métaclasses

Nous proposons une modélisation basée sur une technique de fragmentation « virtuelle » qui nous permet de découper un document en nœuds structurants. Cette fragmentation est dite virtuelle du fait que le contenu du document n'est pas réellement fragmenté. Il est stocké sous forme d'un bloc de données et référencé par les différents éléments de structures et en conséquence par les différentes structures documentaires. Ainsi, un élément et un contenu sont associés par des index qui déterminent la localisation de chaque fragment de contenu dans le bloc de donnée global. Ces index sont traduits par des attributs rattachés à chaque élément décrit. ***Cette indexation permet la gestion d'un contenu commun autour duquel s'articulent plusieurs structures. Ceci nous permet d'une part d'éviter la redondance de stockage et d'autre part d'assurer la gestion de chevauchements entre les nœuds de structures concourantes***. En nous basant sur les notions de fragmentation virtuelle, d'indexation, de relations et de vue, nous avons conçu un modèle spécifique de documents multistructurés (Cf. Figure III.2).

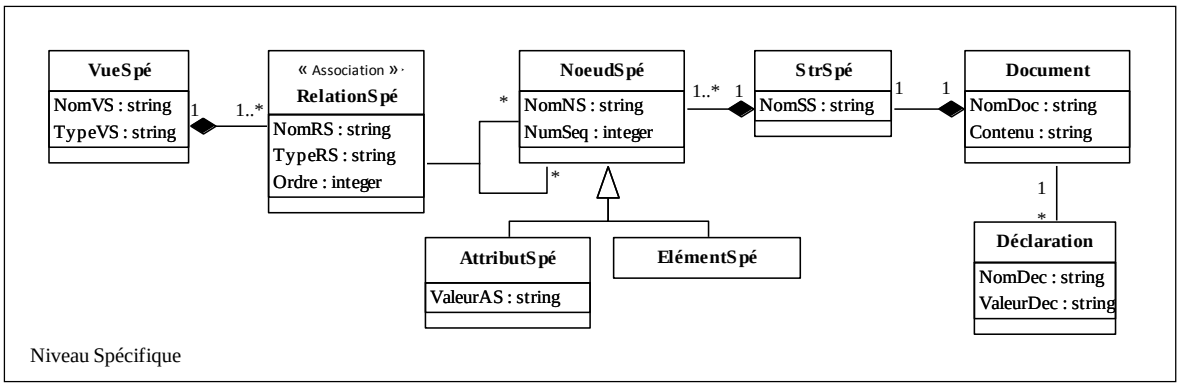


Figure III.2. Modèle spécifique de documents multistructurés en UML.

Le niveau spécifique de notre modèle est décrit par dix métaclasses. Au travers de ces métaclasses et leurs associations, nous modélisons l'ensemble des concepts que nous avons définis pour représenter les structures multiples liées à un document.

Nous présentons une métaclasse « Document » afin d'identifier chaque document de la collection. Un document est caractérisé par son nom et par son contenu. Ce contenu peut être soit une suite de caractères (document textuel) soit une référence vers un média soit des combinaisons des deux (document multimédia). Un tel contenu n'étant consulté qu'au travers des vues qui l'organisent, il est stocké intégralement au niveau de cette métaclasse et référencé par les nœuds spécifiques grâce à des index (attributs spécifiques associés à un nœud élément).

La métaclasse « Déclaration » assure le stockage des informations complémentaires au document, comme sa version par exemple. Dans cette métaclasse, nous retenons le nom et la valeur de chaque déclaration.

Les entités composant le document sont représentées par des nœuds. Ces entités seront traduites au niveau d'une métaclasse appelée « NœudSpé ». Chaque nœud spécifique est caractérisé par son nom et par son type. Nous avons recensé deux types de nœuds spécifiques : les éléments et les attributs. Ces derniers admettent des caractéristiques spécifiques. De ce fait, nous avons conçu deux métaclasses : « ElémentSpé » et « AttributSpé ». Ces métaclasses sont rattachées à la métaclasse « NœudSpé » via un lien d'héritage.

La métaclasse « ElémentSpé » représente les éléments. Ces éléments font référence au contenu via des index. Ces index peuvent être des marques de début et de fin dans le cas d'un fragment textuel ou segment sonore comme elles peuvent être des marques de positions dans le cas d'une image. Quelle que soit leur nature, ces index seront matérialisés par des attributs spécifiques rattachés à chaque élément décrit. Un élément pouvant être décomposé en sous- éléments (image en zones par exemple), nous avons créé une association réflexive sur la métaclasse « NoeudSpé ». Cette association permet de traduire les relations entre nœuds et par conséquent entre éléments ou entre éléments et attributs.

Afin de décrire les attributs nous avons conçu une métaclasse « AttributSpé ». Les attributs apportent des informations complémentaires relatives à l'élément auxquels ils sont associés. Ainsi, les instances de cette métaclasse sont toujours associées aux instances de la métaclasse « ElémentSpé ». Ils viennent en quelque sorte les qualifier. Un attribut est toujours un couple (*nom, valeur*). Le nom est représenté par l'attribut « NomNS » de la métaclasse « NœudSpé » alors que la valeur est décrite par l'attribut « ValeurAS » de la métaclasse « AttributSpé ».

« RelationSpé » est une métaclasse d'association. Elle décrit donc l'ensemble des associations qui peuvent exister entre deux nœuds spécifiques indépendamment de leur type (élément ou attribut). Chaque instance de cette métaclasse est caractérisée par son numéro d'ordre, son type et son nom. Le numéro d'ordre détermine l'ordre du nœud fils parmi l'ensemble des fils du nœud père. Chaque relation admet son propre type. Ces types peuvent varier d'une simple composition qui caractérise les dimensions logique ou physique d'un document, à une description qui montre la dimension sémantique, ou à un référencement qui spécifie la dimension hypertexte, ou encore à une relation temporelle ou spatiale qui détermine les dimensions spatiotemporelles, etc. (Djemal 2007a). Pour certains types de relations, et en particulier pour les relations spatio-temporelles, il est nécessaire de déterminer le nom de la relation. En effet, pour un même type de relations, on peut avoir des relations de natures différentes (Cf. chapitre 1 section II.1). Par exemple, Allen (Allen 1991) propose treize relations différentes de types temporelles.

La métaclasse « VueSpé » permet, au travers de ses instances, de déterminer une structuration particulière d'un document. Une vue ne pouvant pas être reconstruite à partir des nœuds du fait que ces derniers sont partagés entre les vues d'un même document, elle sera reconstruite à partir des différentes relations qui existent entre les nœuds. Chaque instance de « VueSpé » est caractérisée par son nom et son type. Le type de la vue spécifique peut être déduit des relations qui composent la vue.

Enfin, nous avons défini une métaclasse appelée « StrSpé » qui nous permet de représenter l'ensemble des structures spécifiques. Chaque instance de cette métaclasse offre une vision détaillée d'un document en terme d'entités qui le composent. Cette structure est composée de l'ensemble des nœuds spécifiques du document. Chaque structure spécifique est caractérisée par un nom.

II.3. Exemples

Dans cette section, nous présentons des exemples d'instanciation du modèle spécifique de documents multistructurés. Ces exemples illustrent notamment la représentation des différents médias selon notre modèle.

- Document texte

Dans la Figure III.3, nous présentons une instanciation à partir d'un document textuel. Nous montrons une décomposition de ce document en trois paragraphes. Cette instanciation est traduite au travers d'un diagramme objet. Pour chaque paragraphe, nous présentons une instance de « NœudSpé ». Ce nœud est de type « ElémentSpé », un objet de

ce type est donc créé. Pour chaque paragraphe, nous avons associé des marques de début et de fin en utilisant comme unité de mesure le caractère. Ces marques sont traduites par des instances de « NœudSpé » de type « AttributSpé ». Par exemple, le premier paragraphe commence à la position 1 et se termine à la position 300. Ceci est traduit par deux instances de la métaclasse « AttributSpé » ayant comme valeurAS : « 1 » et « 300 ». Les relations entre les différentes instances de « NœudSpé » sont traduites par des instances de « RelationSpé ».

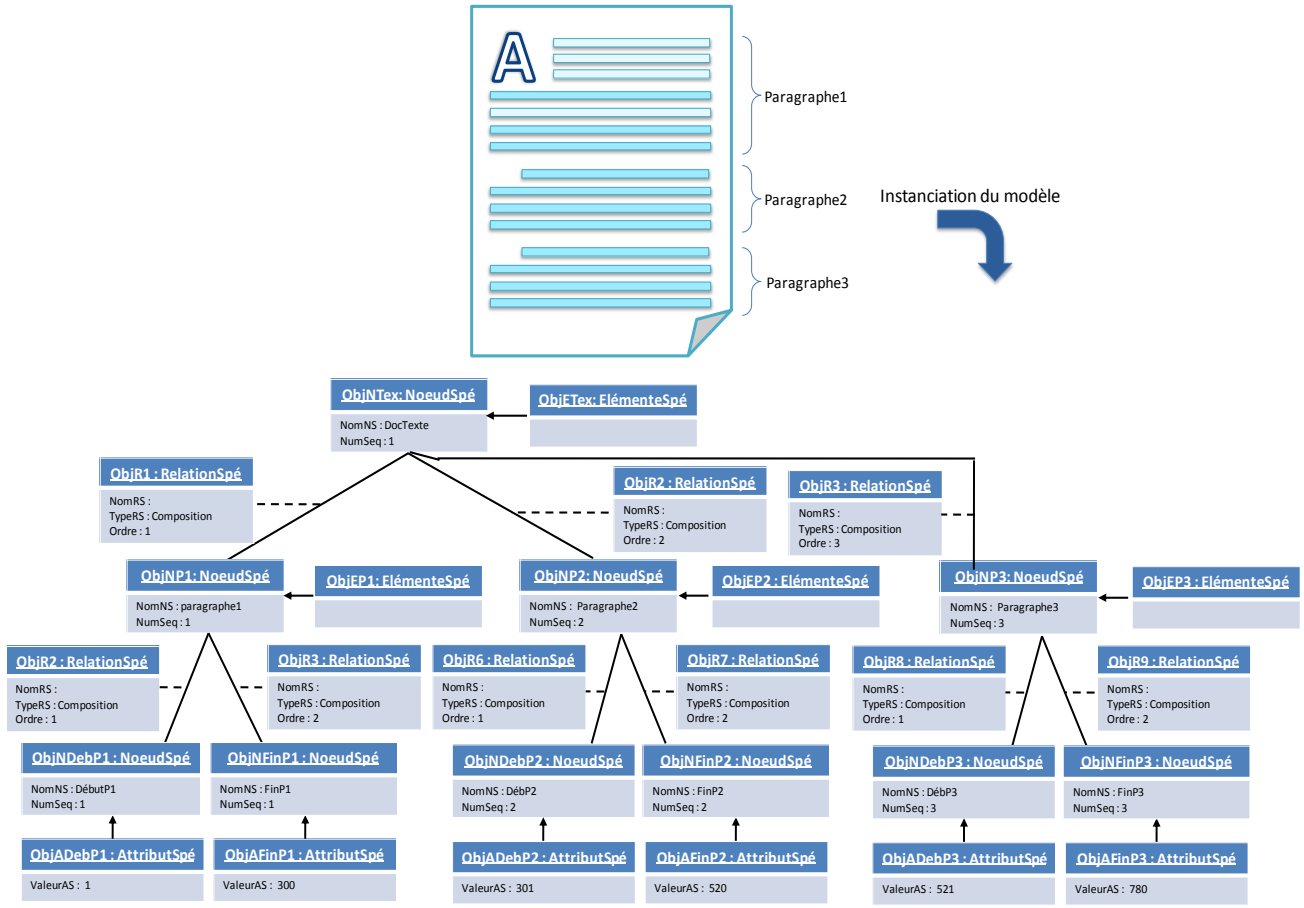


Figure III.3. Extrait d'un exemple d'instanciation du modèle par un document textuel.

- Document image

La Figure III.4 présente un exemple d'instanciation du modèle avec un document image. Nous montrons en particulier le découpage de cette image en un ensemble de zones. Nous considérons ici qu'une zone peut être localisée à partir des coordonnées des quatre points formant les angles de cette zone. Cette caractéristique peut varier selon la nature de la zone à localiser. Les coordonnées d'un point (x,y) en centimètres expriment la position du point par rapport à l'origine du repère. Chaque zone est une instance de la métaclasse « NœudSpé » de type « ElémentSpé ». Les coordonnées de la zone sont des instances de la métaclasse « AttributSpé », rattachées à l'instance de « NoeudSpé » représentant la zone. Dans la Figure III.4, nous considérons que le point P1 est l'origine du repère matérialisé par deux instances de la métaclasse « AttributSpé ». Se pose ici le problème du partage de

contenu. En effet, certains points délimitent plusieurs zones : P2 est commun à deux zones (1,3), P4 est commun à trois zones (1,2,3). Nous constatons dans l'instanciation du modèle, que ces points commun ne sont pas dupliqués mais réellement partagés par les nœuds qui représentent les zones. Les attributs spécifiques représentant les coordonnées de P2 sont rattachés à la fois au nœud spécifique Zone1 et au nœud spécifique Zone3.

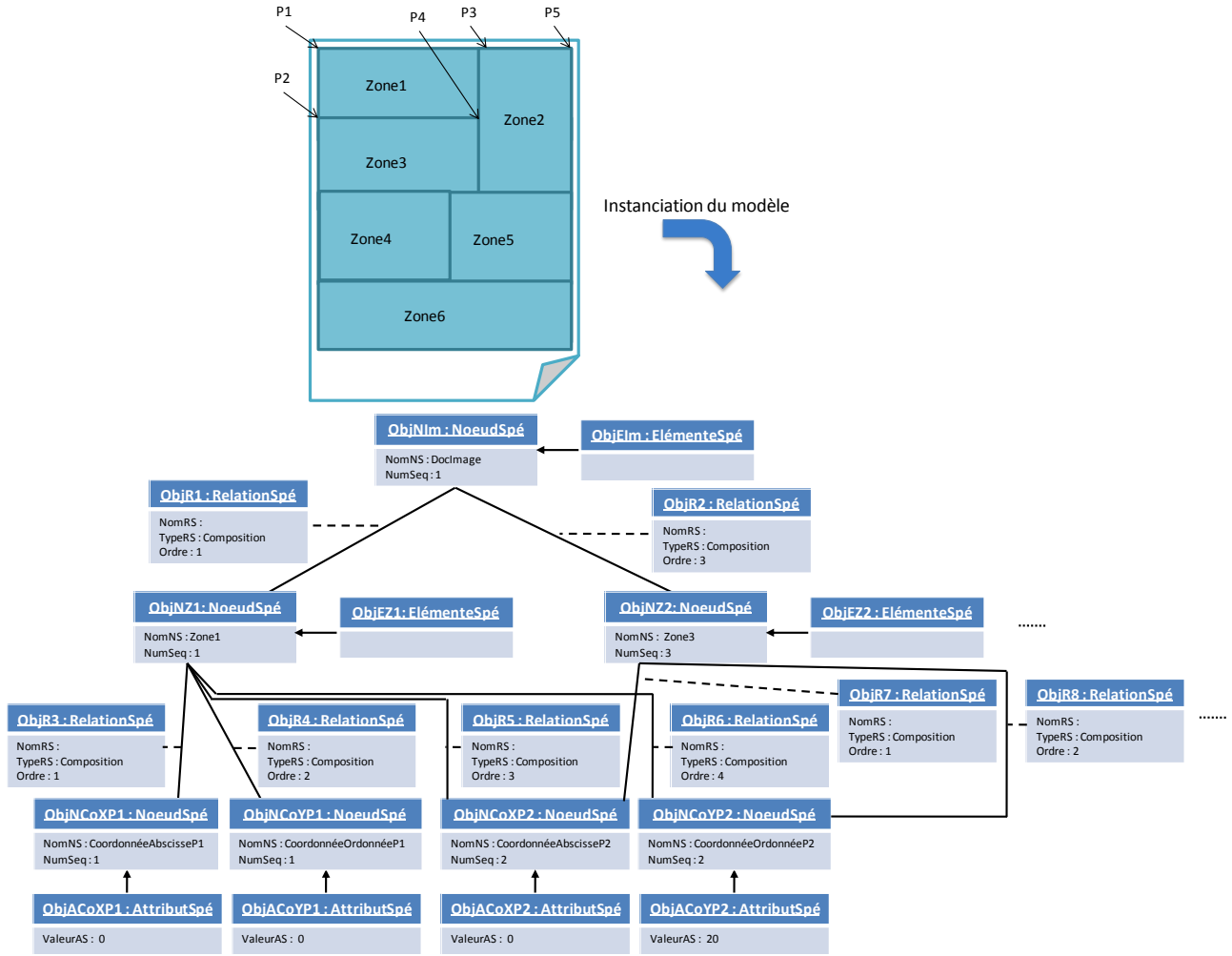


Figure III.4. Extrait d'un exemple d'instanciation du modèle par un document image.

- Document audio

Dans la Figure III.5, nous présentons un exemple de séquence audio découpée en trois segments. Une instance de la métaclasse « NœudSpé » est créée pour chaque segment. Des marques de début et de fin traduites sous forme d'instances de la métaclasse « AttributSpé » sont associées à chaque instance de la métaclasse « NœudSpé » correspondant à un segment. En utilisant comme unité la seconde, ces marques expriment le début et la fin de chaque segment par rapport au début de la séquence audio qui est considéré comme origine. Par exemple, le segment 2 commence à la 256ème seconde et se termine à la 820ème seconde.

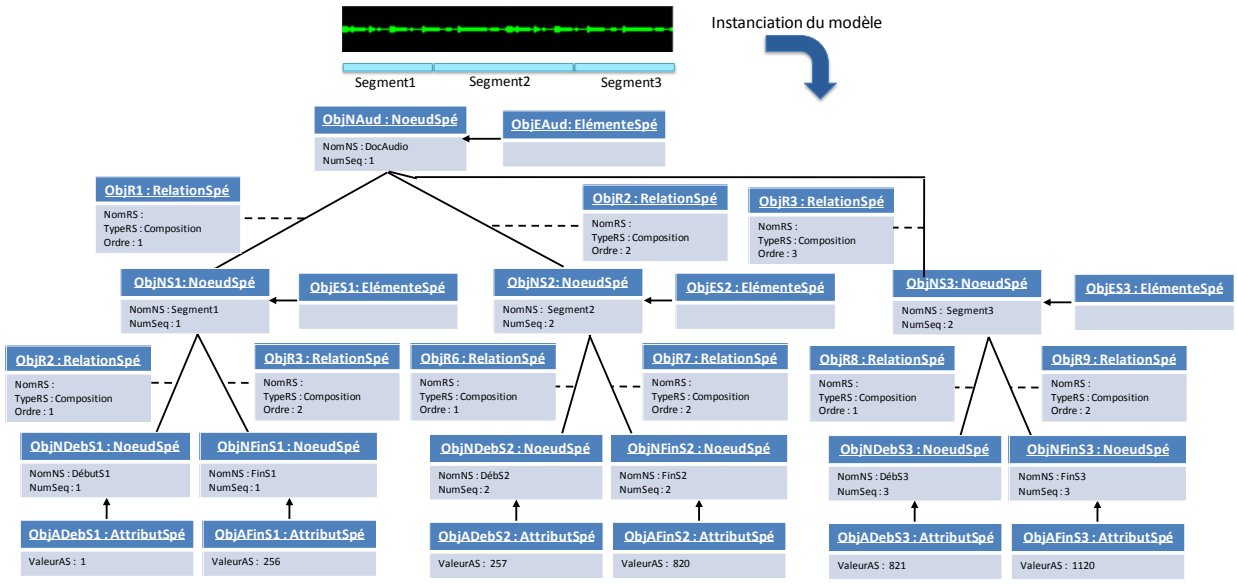


Figure III.5. Extrait d'un exemple d'instanciation du modèle par un document image.

II.4. Représentation de structures à différents niveaux du document

L'émergence des structures multiples pour un même document est le résultat de la définition de n structures soit au niveau du document lui-même, soit au niveau des entités qui le composent. Si nous prenons l'exemple d'un document multimédia : une page web sportive qui traite des éliminatoires du Championnat d'Europe de football 2008. Ce document admet deux structures au niveau global et deux structures au niveau d'une entité : séquence audio.

La Figure III.6 montre deux structures différentes du document choisi. La première structure (a) montre l'aspect physique (présentation) du document alors que la deuxième structure (b) détaille la dimension logique de ce même document. La structure physique montre un découpage en sections, en colonnes et en lignes. La structure logique comprend deux parties : la première partie correspond à un découpage en actualités et la deuxième partie à un découpage en thèmes.

```
<Page_Sportive>
    <Section>
        <Colonne>
            <Ligne>Sport.fr<\Ligne>
        <\Colonne>
    <\Section>
    <Section>
        <Colonne>
            <Ligne>Coupe d'europe<\Ligne>
            <Ligne>L'Equipe de France, impressionnante face à l'Italie, poursuit<\Ligne>
            <Ligne>sa campagne de qualification à l'Euro 2008 avec un déplacement<\Ligne>
            <Ligne>périlleux en Ecosse.<\Ligne>
            <Ligne src='/images/zidane.png'><\Ligne>
            <Ligne>Zidane<\Ligne>
            <Ligne>Zinedine Zidane et Nathalie Baye font partie des nombreuses<\Ligne>
            <Ligne> personnalits qui ont t promues le 1er janvier 2009 au<\Ligne>
            <Ligne> grade d'officier ou de chevalier de la ..<\Ligne>
            <Ligne src='/audio/squence.mp3'<\Ligne>
            ...
        <\Colonne>
        <Colonne>
            <Ligne>FootBall<\Ligne>
            <Ligne>Foot - FIFA : C. Ronald
            <Ligne>Volleyball<\Ligne>
            <Ligne>Le tirage au sort intgr
            <Ligne>Coupes de France Senior
            <Ligne>hier soir au Palais des
            <Ligne> par M. Serge Deloutre,
            <Ligne>Vice-président de la LN
            ...
        <\Colonne>
    <\Section>
    ...
<\Page_Sportive>
```

(a) Structure physique

```
<Page_Sportive>
    <Titre>Sport.fr<\Titre>
    <Actualités>
        <Titre>Coupe d'europe<\Titre>
        <Contenu>
            <Texte>L'Equipe de France, impressionnante face à l'Italie, poursuit
             sa campagne de qualification à l'Euro 2008 avec un déplacement
             périlleux en Ecosse.
            <\Texte>
            <Image src='/images/zidane.png'><\Image>
        <\Contenu>
        <Titre>Zidane<\Titre>
        <Contenu>
            <Texte>Zinedine Zidane et Nathalie Baye font partie des nombreuses
             personnalités qui ont été promues le 1er janvier 2009 au grade
             d'officier ou de chevalier de la ...
            <\Texte>
            <Audio src='/audio/squence.mp3'><\Audio>
        <\Contenu>
        ...
    <\Actualités>
    <Thèmes>
        <Titre>FootBall<\Titre>
        <Contenu>
            <Texte>Foot - FIFA : C. Ronaldo, joueur de l'année 2008<\Texte>
        <\Contenu>
        <Titre>Volleyball<\Titre>
        <Contenu>
            <Texte>Le tirage au sort intgral des matchs de 1/8mes de Finale des
             Coupes de France Senior Fminine et Masculine de Volley-Ball a eu lieu
             hier soir au Palais des Victoires de Cannes. Le tirage a été effectué
             par M. Serge Deloutre, Président de la FFVB, et M. Alain Griguer,
             Vice-président de la LNV.
            <\Texte>
        <\Contenu>
        ...
    <\Thèmes>
<\Page_Sportive>
```

(b) Structure logique

Figure III.6. Structure physique et structure logique relatives document page web spotive.

En se focalisons sur l'élément « Audio » de l'exemple de la Figure III.6, nous constatons qu'il admet des informations supplémentaires (Cf. Figure III.7-a). Ces informations sont fournies par différents utilisateurs du document pour décrire plus précisément son contenu. Le contenu de l'élément « Audio » peut être décrit au travers d'une décomposition des segments en thèmes (Cf. Figure III.7-b). Il peut être également présenté grâce à une deuxième description basée sur la décomposition des segments en locuteurs (Cf. Figure III.7-c).

Figure III.7. Deux descriptions en thèmes et en locuteurs de l'élément audio de la page web sportive.

II.4.1. Représentation des structures multiples au niveau global du document

Notre approche de modélisation des documents multistructurés est basée sur une dichotomie entre structure(s) et contenu. Cette dichotomie assure le partage de contenu commun. Le référencement de ce contenu par les différents éléments feuilles des différentes structures résout le problème de chevauchement d'éléments entre ces structures. La notion de vue adoptée dans notre modélisation permet de représenter n structures simultanément. La Figure III.8 présente l'instanciation du modèle MVDM pour l'exemple de la Figure III.6. Les éléments sont schématisés par des ovales. Le contenu est représenté indépendamment des deux structures. Etant donné que ce contenu est textuel, le fragment de contenu associé à chaque élément feuille est référencé par des marques de début de fin. Selon le modèle MVDM, ces marques doivent être représentées sous forme d'attributs. Cependant, pour des raisons de clarté et de lisibilité, nous les avons présentées sous forme de couples (début_fin) associés aux relations entre les éléments feuilles des deux structures et le contenu.

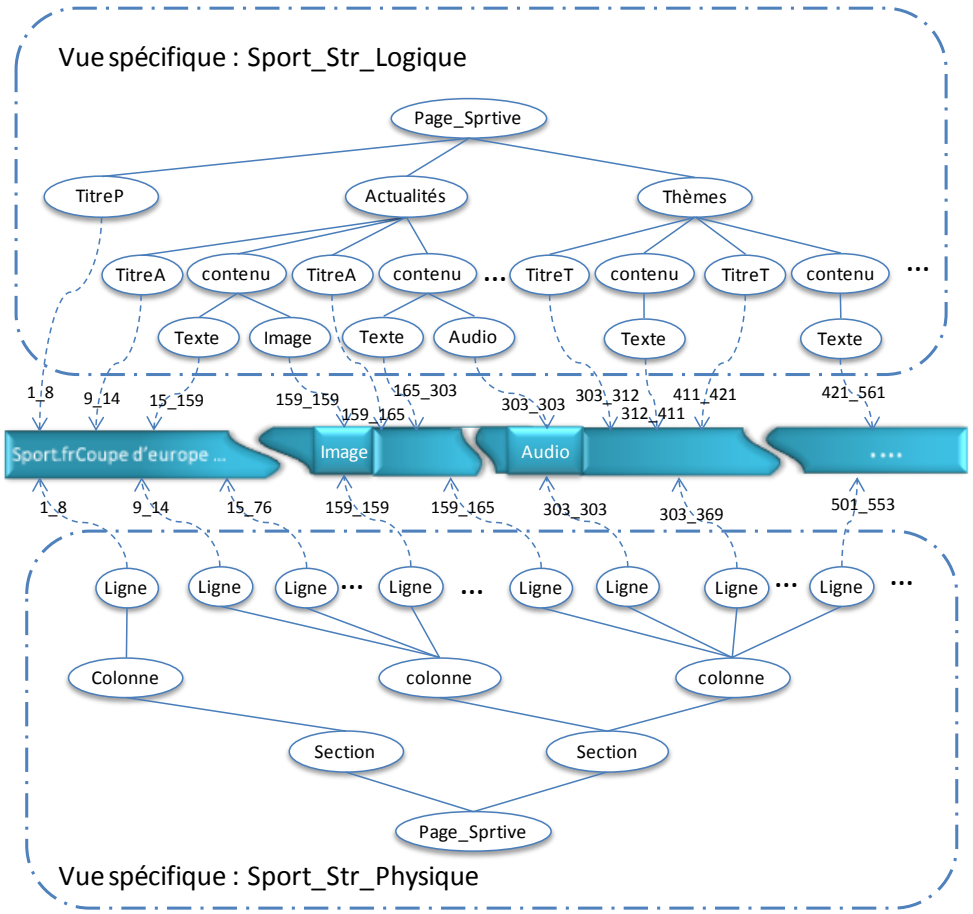


Figure III.8. Instanciation du modèle : cas de structures multiples globales à un même document.

II.4.2. Représentation des structures multiples associées à un nœud d'un document

Les structures associées à un nœud sont le résultat de descriptions des entités complexes d'un document. Une entité peut avoir plus d'une description, plus qu'une décomposition et par conséquent plus qu'une structure. Dans ce cas de figure, les modèles de la littérature (Cf. Chapitre II) proposent de recréer entièrement une nouvelle structure globale. Ceci engendre une description redondante des parties communes entre structures. Selon notre approche de modélisation, le partage de nœuds entre structures et de partie de contenu entre nœuds feuilles permettra d'éviter cette redondance. Les vues telles que nous les avons conçues peuvent être définies à n'importe quel niveau du document et peuvent se chevaucher dans la mesure où elles peuvent être imbriquées les unes dans les autres.

Si nous reprenons les deux descriptions relatives à l'élément audio (Cf. Figure III.7), chacune des descriptions est représentée par une vue. Ces vues partagent obligatoirement l'élément audio décrit. Elles partagent également d'autres éléments tels que « Séquence », « Musique », etc. (Cf. Figure III.9). Les différents éléments sont associés aux segments qu'ils décrivent au travers des marques de début et de fin traduites par des nœuds de types attributs. Pour des raisons de clarté, nous représentons ces marques dans la Figure III.9 par

des couples (début_fin) associés aux relations entre les éléments des deux structures et le contenu.

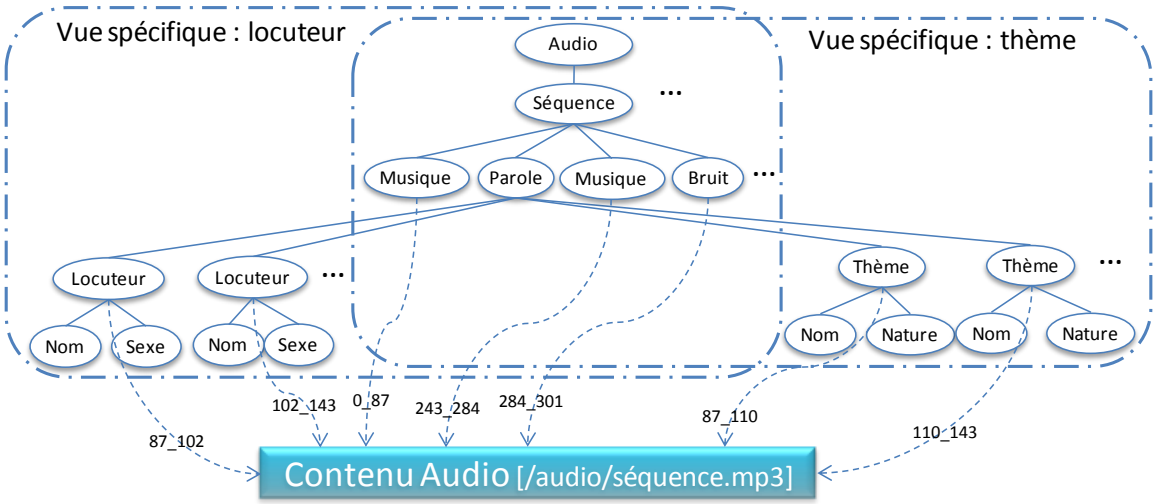


Figure III.9. Instanciation du modèle : cas de structures multiples associées à une même entité.

II.5. Du partage du contenu au partage des nœuds

Dans cette section, nous allons nous intéresser à une des problématiques relatives aux définitions des structures multiples : à savoir le chevauchement d'éléments. Le chevauchement est dû au partage de contenu et de nœuds entre les différentes structures d'un même document.

II.5.1. Partage de contenu entre nœuds de structures différentes

Lorsqu'on parle de multistructuralité, on y associe directement la gestion de partage de contenu. Le partage de contenu est le résultat de la définition de plusieurs structures concourantes. Nous rappelons que deux structures peuvent être concourantes seulement si elles sont définies à un même niveau : soit au niveau global du document, soit au niveau d'une entité (nœud) du document. La problématique majeure issue du partage de contenu est la gestion de chevauchement d'éléments. Ce chevauchement est dû aux différences de segmentations découlant des structures associées au contenu.

Le problème du chevauchement d'éléments est résolu dans le modèle MVDM au travers du mécanisme d'indexation proposé. Le contenu est stocké en tant que tel et chaque structure de document y fait référence. La définition de marqueurs de position, tels que des marqueurs de début et de fin dans le cas d'un contenu textuel, permet d'associer chaque élément à un fragment de contenu.

La Figure III.10 présente un extrait d'instanciation du modèle avec un exemple de chevauchement d'éléments. Nous prenons le cas d'une séquence audio segmentée de deux façons : la première en locuteur et la deuxième en thème. Les éléments locuteur et thème décrivent le même contenu, celui de la séquence audio. Ces éléments présentent des décompositions sémantiquement distinctes, ce qui explique le chevauchement de ces deux

éléments. Dans la Figure III.10, nous présentons un élément thème qui est défini sur l'intervalle temporel (en secondes) [20..44,5] et un élément locuteur qui est défini sur l'intervalle temporel (en secondes) [30..75]. Nous avons schématisé ces deux éléments par des rectangles en pointillés à coté de la séquence audio afin de montrer leur chevauchement.

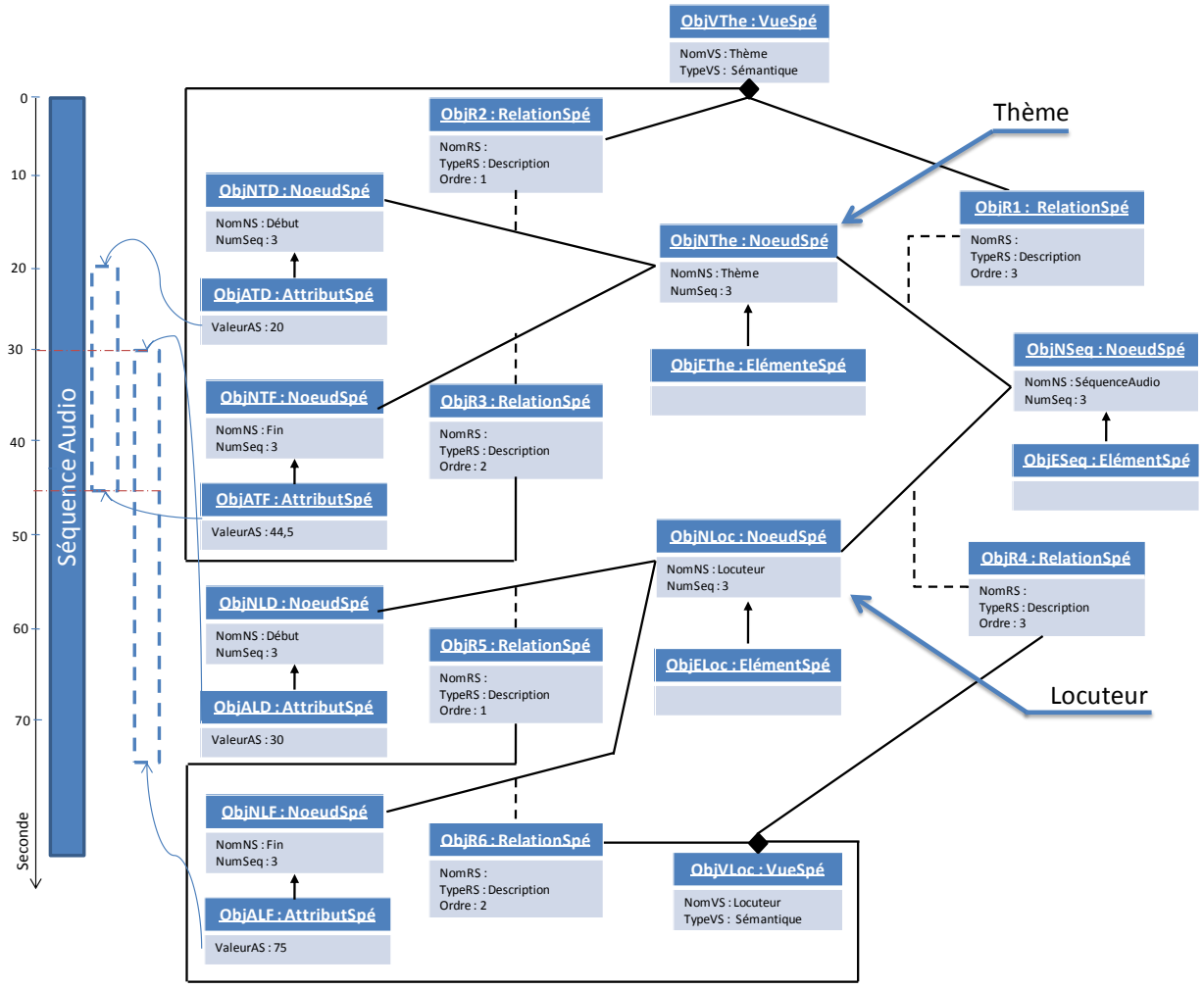


Figure III.10. Cas de partage de contenu.

II.5.2. Partage de nœuds entre structures

Lorsque deux structures documentaires partagent des nœuds, ces nœuds peuvent être soit partagés, soit dupliqués entre les deux vues représentant les deux structures documentaires. Le partage de nœuds permet d'une part, d'éviter la redondance et d'autre part, de lever l'ambigüité lors de la restitution et l'exploitation des documents. Par exemple, dans le cas du non partage de nœuds entre vues, lorsqu'un utilisateur formule une requête en appliquant une contrainte sur un nœud qui est dupliqué dans n structures différentes, le système doit parcourir les n nœuds dupliqués et par conséquent les n structures afin de trouver le résultat.

Dans le cas d'une mise à jour des nœuds communs, le partage de nœuds permet de ne mettre à jour qu'un seul nœud. On gagne ici tout d'abord en cohérence, les risques d'erreurs et d'oublis sont inexistants et le gain de temps est évident.

La Figure III.11 présente un extrait de résumé d'un article scientifique. Ce résumé figure dans des actes et des pages web dédiés sous des formats différents. Ainsi, pour chacun de ces deux formats, une structure physique est créée. Nous nous focalisons, dans cet exemple, sur la taille des caractères utilisés dans les deux formats. Ces tailles sont spécifiées dans des attributs relatifs à l'élément « Résumé ». Dans la Figure III.11, nous présentons les objets relatifs à l'élément commun « Résumé » : l'objet « ObjNRes », « ObjERes », « ObjNRD », « ObjARD », « ObjNRF » et « ObjARF ». En effet, les deux structures documentaires partagent l'élément résumé et les index qui référencent son contenu. Dans la Figure III.11, nous présentons également les objets différents entre les deux structures documentaires : les objets « ObjNTai1 », « ObjATai1 », « ObjR1 » et « ObjVFAc » pour la structure physique relative au format des actes ; et les objets « ObjNTai2 », « ObjATai2 », « ObjR2 » et « ObjVFPw » pour la structure physique relative au format des pages web.

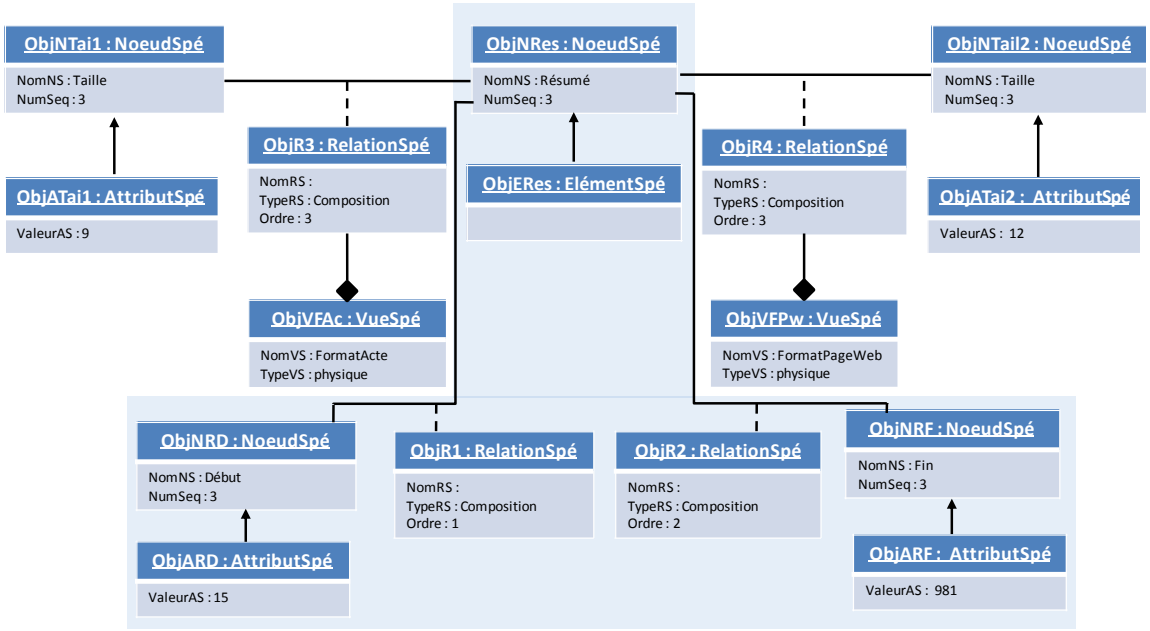


Figure III.11. Cas de partage de nœuds.

III. Modélisation d'une collection de documents multistructurés

III.1. Objectif et intérêt

Dans la réalité, les collections de documents sont très hétérogènes. Cette hétérogénéité pose en général des problèmes d'identification des granules documentaires, d'évaluation des similarités entre granules ou types de granules et de restitution de ces granules à partir d'un document ou d'une collection de documents.

L'originalité de nos travaux réside dans la possibilité de gérer les documents par sous-collections (curriculum vitae, films documentaires, un type d'émission radio particulier, etc.) en se basant sur leurs similarités structurelles. Notre objectif consiste à regrouper, sous forme de classes, les documents ayant des structures similaires sans pour autant perdre les informations qui leur sont spécifiques (nom spécifique de chaque fragment de document, etc.).

Le regroupement des documents passe par un processus de classification. La notion de classification des documents selon la similarité de leurs structures documentaires permet d'optimiser la recherche ultérieure de tel ou tel type de documents. Ces documents étant multistructurés, ils seront classés selon une seule structure ou selon l'ensemble de leurs structures. Ainsi, chaque classe sera caractérisée par une « *vue générique* » qui représente une collection de « *vues spécifiques* » ou par une « *structure générique* » qui représente une collection de « *structures spécifiques* ». Une « *vue spécifique* » caractérise une organisation particulière d'un document selon une structure et ; une « *structure spécifique* » présente l'organisation d'un document multistructuré.

La liste des structures et des vues génériques peut être utilisée pour simplifier l'accès et la recherche dans les documents. Ceci permettra d'offrir plusieurs points d'accès au contenu des documents en vue d'en faciliter leur exploitation. Ainsi, les utilisateurs peuvent avoir un accès direct à une collection particulière qui répond à leurs besoins. Par exemple, il est possible de faire des recherches seulement dans les flashs d'information annotés dans le cadre d'un corpus particulier et non pas dans tous les flashs d'information ou dans tous les documents audio de la collection. Considérée comme étant un schéma de documents multistructurés, une structure générique permet de plus de simplifier l'accès au contenu de ces documents en combinant des critères sur l'ensemble des nœuds génériques qui la composent.

La Figure III.12 illustre le rattachement spécifique/générique entre les vues et les structures ainsi que les agrégations des vues spécifiques/structures spécifiques et des vues génériques/structures génériques. Nous considérons que « VS11 » et « VS12 » sont deux vues spécifiques (une vue logique et une vue physique par exemple). Ces deux vues sont agrégées dans une même structure spécifique SS1 d'un document nommé « CV1 ». « VS21 » et « VS22 » sont deux vues spécifiques (une vue logique et une vue physique également) agrégées dans une même structure spécifique SS2 d'un document nommé « CV2 ». Si « VS11 » et « VS21 » sont représentées par la vue générique « VG1 » et « VS12 » et « VS22 » sont représentées par la vue générique « VG2 », nous pouvons dire que ces deux vues génériques sont agrégées au sein d'une même structure générique qui désigne la structure générique des CVs.

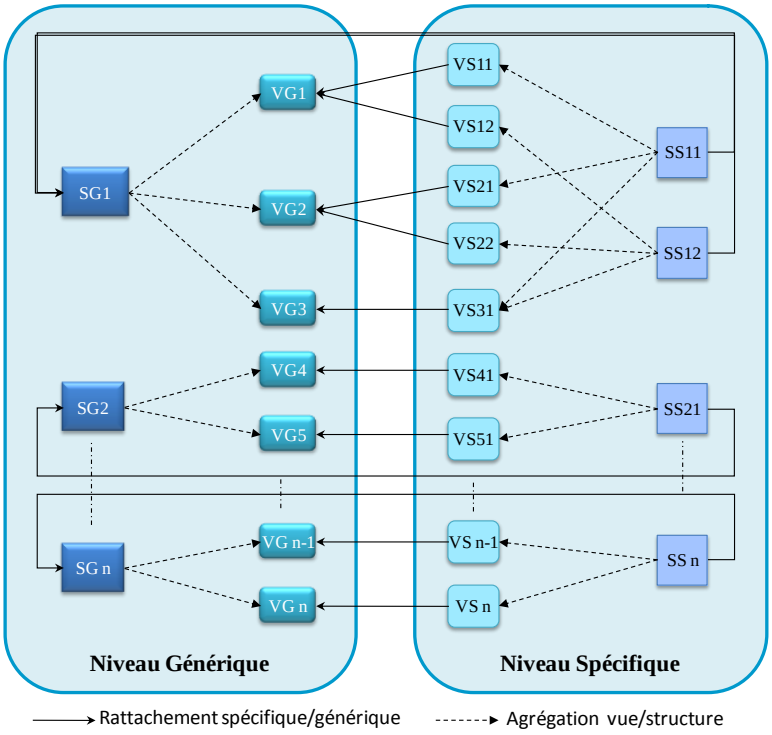


Figure III.12. Classification par vues et structures génériques.

III.2. Modèle générique et description des métaclasses associées

L'idée d'un niveau générique se base sur le regroupement des documents sous forme de classes. Une classe de documents est considérée comme un ensemble homogène et cohérent d'un point de vue structurel. Comme pour la partie spécifique, la partie générique est composée des métaclasses « StrGén », « NœudGén », « RelationGén » et « VueGén ». La Figure III.13 détaille ce modèle en utilisant le formalisme UML.

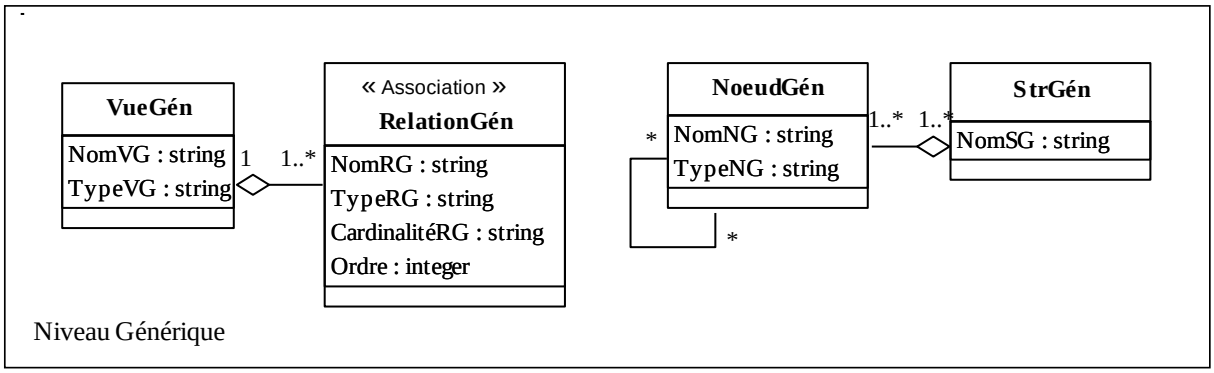


Figure III.13. Modèle générique de documents multistructurés en UML.

Un nœud générique regroupe un ensemble de nœuds spécifiques afin d'en faciliter l'accès. La métaclasse « *NœudGén* » représente l'ensemble des nœuds génériques. Ces nœuds sont modélisés indépendamment de toute structure. De ce fait, ils peuvent appartenir à une ou plusieurs structures génériques. Chaque nœud générique est caractérisé par son nom et par son type (élément ou attribut). Nous n'avons pas choisi de proposer des

métaclasses pour chaque type de nœuds, comme au niveau spécifique, car ces derniers n'admettent pas de caractéristiques particulières.

Les relations génériques sont regroupées dans la métaclasse « *RelationGén* ». Cette dernière représente une métaclasse d'association. Elle décrit donc l'ensemble des associations qui peuvent exister entre deux nœuds génériques indépendamment de leur type (élément ou attribut). Chaque instance de cette métaclasse générique est caractérisée par son nom, son type, son ordre. Mais également, cette métaclasse définit la cardinalité du lien qu'elle représente. La cardinalité désigne le nombre d'occurrences des relations spécifiques représentées par la relation générique décrites dans cette métaclasse dans une même vue spécifique. Si aucune cardinalité n'est précisée (« »), la relation spécifique doit apparaître une et une seule fois dans la vue spécifique. Avec la cardinalité « * », la relation spécifique peut ne pas apparaître ou apparaître plusieurs fois dans une même vue spécifique. La cardinalité « + » indique que la relation spécifique doit apparaître soit une fois, soit plusieurs fois dans une même vue spécifique. Enfin, avec cette cardinalité « ? », la relation spécifique peut apparaître une seule fois ou ne pas apparaître du tout.

Une vue générique est un regroupement d'un ensemble de vues spécifiques structurellement similaires. Elle modélise ainsi une organisation particulière d'une sous-collection de documents. Ceci permet à une vue générique de représenter un schéma d'un ensemble de vues spécifiques. Elle est considérée alors comme une classe de vues spécifiques en se basant sur une de leurs structures documentaires. La métaclasse « *VueGén* » permet de représenter une vue générique. Chaque instance de « *VueGén* » est caractérisée par son nom et son type.

Enfin, la métaclasse « *StrGén* » permet de représenter l'ensemble des structures génériques. Chaque instance de cette métaclasse offre une « vision globale » d'une classe de documents à partir de ses vues génériques. Cette structure est composée d'un ensemble de nœuds génériques reliés par des relations génériques qui dépondent des vues génériques. Chaque structure générique est caractérisée par un nom.

III.3. Exemple de représentation d'une collection de documents

La Figure III.14 présente un exemple de deux structures documentaires : l'une pour une émission de divertissement et l'autre pour une interview. Ces deux structures documentaires sont représentées par deux vues génériques agrégées dans une même structure générique. Les deux vues génériques correspondent à deux modèles de descriptions d'une séquence audio. La différence entre ces deux vues génériques réside dans la manière de segmenter la séquence audio. La première segmentation découpe la séquence audio selon les locuteurs qui interviennent alors que la deuxième segmentation favorise un découpage en thèmes abordés. Nous associons deux vues spécifiques à chacune des deux séquences. Les vues spécifiques d'une même séquence audio correspondent aux deux descriptions de la séquence telles que nous les avons présentées dans la Figure III.7.

Supposons que nous voulions extraire la liste des locuteurs qui interviennent sur le thème du « Sport » évoqué dans les deux documents de la Figure III.14. Dans ces

documents, les entités « Locuteur » et « Loc » désignent toutes l'entité sémantique « Locuteur » ; alors que l'entité sémantique « thème » appartient à une deuxième vue. Ce cas de figure simule l'interrogation de documents en se basant sur leur multistructuralité (deux conditions sur deux vues différentes du document).

Au travers de cet exemple, nous montrons l'intérêt d'avoir des vues génériques au lieu d'un schéma unique qui regroupe un ensemble de structures. L'intérêt se matérialise à deux niveaux :

- la conservation de l'organisation spécifique du document. Cette organisation risque d'être modifiée par l'ajout de champs vides dans le cas l'utilisation d'un schéma unique. Selon notre approche, nous ne rattachons que les nœuds spécifiques qui admettent des nœuds pères dans le niveau générique. Par exemple, le nœud générique « Bruit » n'est représenté dans aucune vue spécifique agrégée de la structure spécifique « Interview » ;

- la conservation du nom des nœuds spécifiques (ceux donnés par le créateur de document) quand ils sont différents de ceux des nœuds génériques. Par exemple, le nœud spécifique « Loc » de la vue spécifique « Locuteur » sera rattaché au nœud générique « Locuteur ».

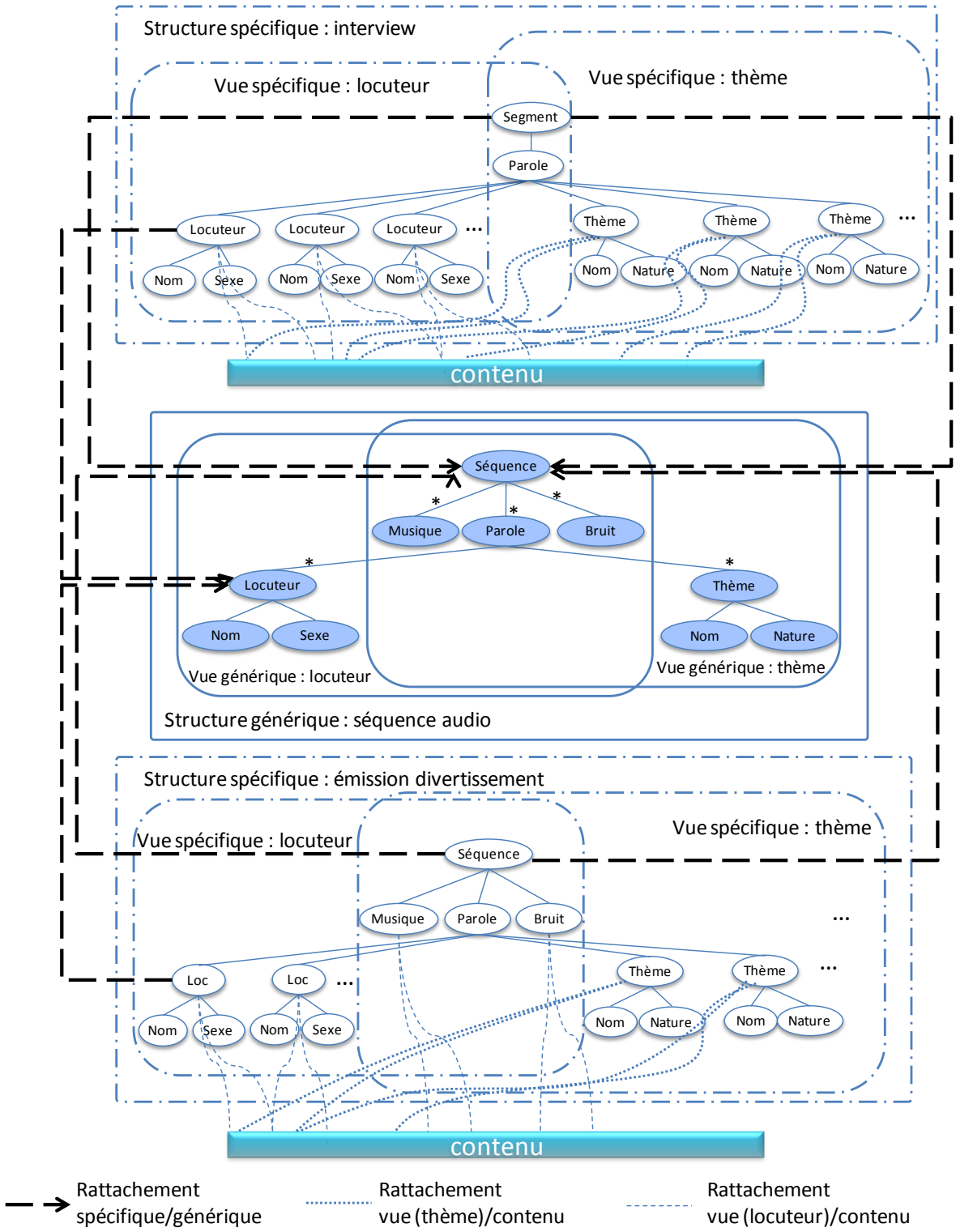


Figure III.14. Illustration du « regroupement » de structures similaires

Remarques : Par souci de clarté, nous n'avons pas présenté dans la Figure III.14 tous les liens qui relient les fragments génériques aux fragments spécifiques. Nous n'avons représenté que les liens entre les nœuds racines de chaque vue générique aux nœuds racines des vues spécifiques et les nœuds spécifiques qui admettent des noms différents à leurs nœuds génériques.

IV. Modèle de représentation de documents multistructurés

Depuis sa standardisation, l'**U**nified **M**odeling **L**anguage « UML » (RTF 1999) s'impose progressivement comme étant un langage de modélisation objet de systèmes, qu'ils soient logiciels, matériels conceptuels, ou organisationnels. UML est un langage semi-formel qui permet de représenter des modèles en utilisant des outils graphiques « simples » pour le concepteur, l'informaticien et lisibles pour l'utilisateur. En contrepartie, ces avantages sont offerts au détriment de la précision de définition des règles et des contraintes. En se basant sur des outils mathématiques, les langages formels neutralisent tous les risques d'ambiguïté et d'incertitude liées aux langages semi-formels.

Ainsi, il s'avère intéressant de commencer par une modélisation semi-formelle en UML, et ensuite de la compléter par une modélisation formelle en vue de tirer profit de leurs avantages respectifs. Dans ce qui suit, nous détaillons notre modèle de documents multistructurés « MVDM » (**M**ulti **V**iew **D**ocument **M**odel) dans un premier temps en utilisant UML et ensuite en nous basant sur des notations formelles.

IV.1. Modélisation UML

Le constat que l'on a pu faire au travers des modèles de la littérature montre un manque de flexibilité dans la gestion et la manipulation de documents multistructurés en grande partie lié au fait qu'ils se focalisent sur les structures multiples définies au niveau global du document. Or, les documents numériques et en particulier les documents multimédias peuvent admettre, en plus des structures multiples définies au niveau global du document, des structures multiples associées aux différentes entités qui le composent. Si l'on souhaite pouvoir stocker et manipuler, sous plusieurs facettes et plusieurs points de vue, un seul ou un ensemble de documents multistructurés, il s'avère important de pouvoir modéliser tous les concepts liés d'une part au document lui-même (ses structures et son contenu) et d'autre part à une collection de documents. Ainsi, nous proposons un modèle « MVDM » (Djemal et al. 2008a) (Cf. Figure III.15) qui intègre deux niveaux de descriptions :

- le niveau générique est décrit par les métaclasses : « StrGén », « NœudGén », « RelationGén » et « VueGén ». Nous rappelons que ce niveau permet la classification d'une collection de documents (Cf. Section III). La structure générique « StrGén » représente la structure globale d'une classe de documents multistructurés. Elle est définie par un ensemble de nœuds génériques « NœudGén ». Les relations génériques « RelationGén » caractérisent le lien qui existe entre-deux nœuds génériques selon une vue particulière. Une vue générique « VueGén » référence une « sous-structure » de la structure générique. Chaque sous-structure désigne une représentation particulière d'une classe de documents ;

- chaque métaclasse représentée au niveau générique admet son équivalent au niveau spécifique. Nous rappelons que ce niveau nous permet la description et la prise en compte des caractéristiques propres à chaque document appartenant à la collection (Cf. Section II). Ainsi, le niveau spécifique est décrit par les métaclasses : « StrSpè », « NœudSpé », « RelationSpé » et « VueSpé ». Le niveau spécifique contient également deux autres métaclasses propres à un document : la métaclasse « Document » qui désigne un document spécifique et la métaclasse « Déclaration » qui permet de garder l'information concernant les caractéristiques des documents telles que par exemple leurs versions. Dans le niveau spécifique, nous détaillons aussi les types de nœuds spécifiques afin de représenter leurs propres caractéristiques. Une métaclasse est conçue pour chaque type de nœuds.

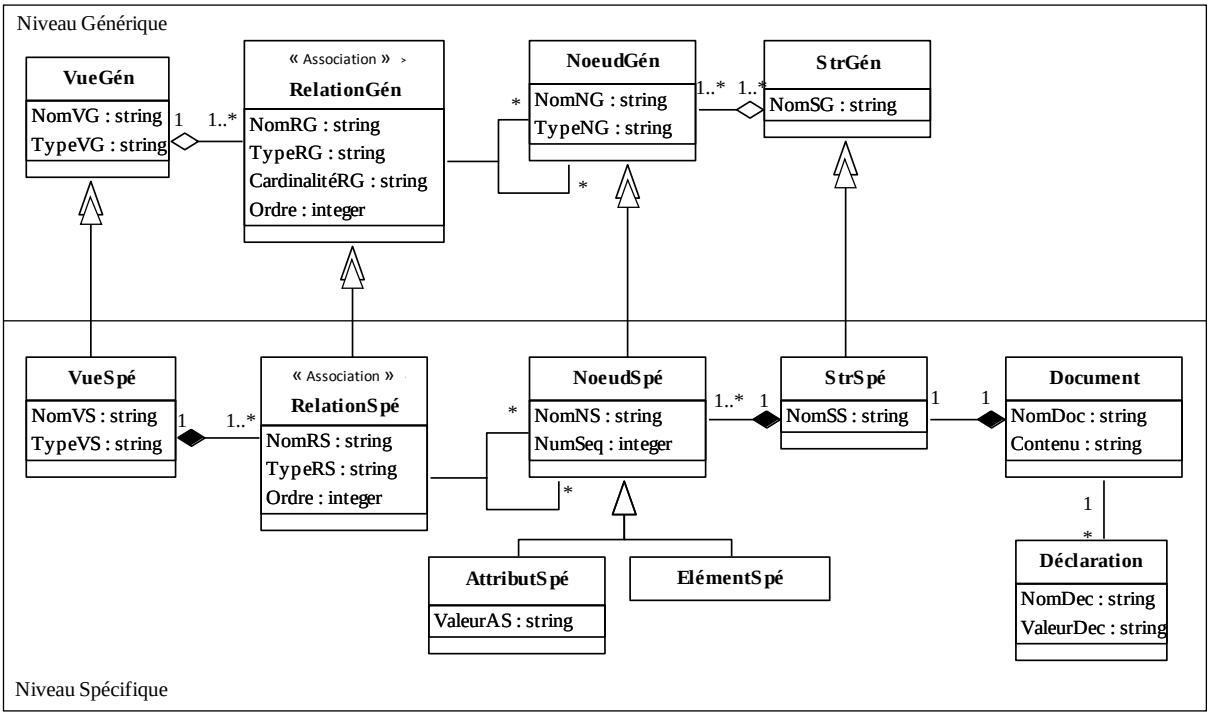


Figure III.15. Le modèle MVDM en UML.

La relation entre les deux niveaux (spécifique et générique) est assurée par un lien particulier que nous définissons comme lien de conformité. UML ne possédant pas ce type de relation, nous avons opté pour un nouveau stéréotype : « Å » (Cf. Figure III.15).

Lien de conformité : nécessité

La relation entre le niveau spécifique et le niveau générique pourrait être décrite par des liens d'héritage du fait que les classes spécifiques sont d'une part, totalement cohérentes avec les classes génériques, et d'autre part, elles peuvent être enrichies par des informations spécifiques. Toutefois, ces classes spécifiques ne représentent pas des sous-classes mais plutôt des sortes d'instances de classes génériques. Ainsi, le lien de conformité traduit la notion d'instanciation qui caractérise la relation entre une classe spécifique et une classe générique.

Lien de conformité : caractéristiques

Ce lien de conformité défini doit avoir, en plus de l'aspect d'instantiation, les caractéristiques suivantes :

- une classe spécifique hérite de la classe générique. La classe spécifique possède toutes les caractéristiques de la classe dérivée ;
- homomorphisme : ce lien assure l'homomorphisme entre le niveau spécifique et le niveau générique. En effet, cette relation garantit que tout fragment spécifique doit être rattaché à un fragment générique.

Gérer les relations entre nœuds génériques et spécifiques ainsi que les différents liens génériques/spécifiques donnent une certaine complexité à notre modèle. Une telle complexité justifie l'utilisation des meilleures techniques disponibles afin de garantir une meilleure qualité de spécification et de conception, et de maintenir cette qualité lors de l'implantation. Pour ce faire, nous détaillons dans la section suivante notre modélisation formelle.

IV.2. Modélisation formelle de documents multistructurés

Nous présentons dans cette section la description formelle de notre approche de modélisation (Djemal et al. 2008b). En plus de la formalisation du lien de conformité, nous optons pour une spécification formelle afin d'éliciter les agrégations entre structures et vues. Si la structure est composée de nœuds, elle doit avoir une ou plusieurs vues afin d'offrir une ou plusieurs formes pour un même document. Une telle agrégation entre structures et vues ne peut pas être représentée en UML du fait de la confusion qui peut avoir lieu entre les relations structure/nœuds et structure/vues.

Le modèle MVDM s'articule autour de deux concepts fondamentaux : les objets et les règles. Les objets « O_i » représentent l'ensemble des composantes d'un document multistructuré ou de la classe de documents à laquelle ils appartiennent. Les règles représentent l'ensemble des perceptions qui organisent les différents objets O_i.

IV.2.1. Ensembles d'objets

Le Modèle MVDM est défini au travers de douze ensembles d'objets : O = {Doc, Dec, SS, NS, ES, AS, VS, RS, SG, NG, RG, VG}.

Nous définissons chaque objet $O_i \in O$ comme un couple (A_i,M_i) où :

– $A_i = \{A_{i1}, A_{i2}, ..., A_{ip}\}$ est l'ensemble d'attributs représentant la partie statique de chaque objet O_i ;

– $M_i = \{M_{i1}, M_{i2}, ..., M_{iq}\}$ est l'ensemble des méthodes représentant la partie dynamique de chaque objet O_i, chaque méthode utilise une liste d'arguments et éventuellement retourne une valeur.

Nous décrivons dans ce qui suit les quatorze ensembles d'objets :

– Doc représente l'ensemble des documents spécifiques. Chaque document est caractérisé par son nom et son contenu. Par exemple, si le document est de type textuel, le contenu consiste en une suite de chaînes de caractères. Cette chaîne de caractères est un segment défini sur [0..LongFrag] ;

– Dec est l'ensemble des déclarations d'un document. Cet objet permet de garder des informations complémentaires concernant un document, telle que sa taille par exemple ;

– SS est l'ensemble des structures spécifiques. Chaque structure spécifique $ss_i \in$ SS est caractérisée par un nom. La structure spécifique représente l'ensemble des composants d'un document. Ainsi, cette structure est composée de l'ensemble des nœuds spécifiques du document ;

– NS est l'ensemble des nœuds spécifiques. Chaque nœud spécifique $ns_i \in$ NS est caractérisé par son nom, son numéro séquentiel (c'est le numéro de séquence du nœud spécifique ns_i dans la composition du nœud générique ng_i) et son type. Les nœuds spécifiques peuvent être de deux types de nœuds : des éléments ou des attributs ;

– AS décrit l'ensemble des attributs d'élément. Les attributs sont également des nœuds. L'ensemble des attributs d'élément $as_i \in$ AS sont caractérisés particulièrement par leur valeur ;

– ES représente l'ensemble des éléments spécifiques d'un document. Chaque élément spécifique $es_i \in$ ES est un nœud spécifique $ns_i \in$ NS mais il présente des caractéristiques propres. Le contenu est l'une des propriétés d'un élément spécifiques. Ce contenu doit être référencé au travers d'index. Ces index seront traduits par des attributs. Par exemple, si l'élément référence un fragment textuel, un attribut début et un attribut fin seront utilisés pour marquer le début et la fin de ce fragment ;

– RS est l'ensemble des relations spécifiques. Ces relations caractérisent les liens qui existent entre les nœuds spécifiques. Pour chaque relation spécifique $rs_i \in$ RS, nous gardons son nom, son type et son ordre ;

– VS est l'ensemble des vues spécifiques. Chaque vue spécifique $vs_i \in$ VS est caractérisée par son nom et son type. Ce type peut être déduit des relations qui définissent la vue ;

– SG est l'ensemble des structures génériques. Chaque structure générique $sg_i \in$ SG est caractérisée par son nom et référence une structure commune à un ensemble de structures spécifiques ;

– NG est l'ensemble des nœuds génériques. Un nœud générique $ng_i \in$ NG est caractérisé par son nom et son type qui décrit la nature du nœud : élément ou attribut ;

– RG est l'ensemble des relations génériques. Ces relations caractérisent les liens qui existent entre les nœuds génériques. Ainsi, nous stockons le nom de ce lien, son type (composition, synchronisation, agencement, etc.), son ordre et sa cardinalité (« » pour désigner un et un seul, « ? » pour désigner zéro ou un, « + » pour désigner un ou plusieurs et « * » pour désigner zéro ou plusieurs) ;

– VG est l'ensemble des vues génériques. Une vue générique $vg_i \in$ VG est caractérisée par son nom et par son type. Une vue générique référence une sous-structure de la structure générique. Cette sous-structure peut-être logique, physique, sémantique, temporelle, spatiale, etc.

Certains objets présentent des méthodes spécifiques. Ces méthodes sont traduites au travers des fonctions.

Pour retrouver les fils d'un nœud spécifique $ns_i \in$ NS dans le cadre d'une structure spécifique ss_i, nous avons défini la fonction (1) F_{SS^P} Cette fonction est une méthode d'un nœud spécifique. Dans ce cadre, il est tout à fait possible de définir la fonction inverse $F_{SS^{P-1}}$ qui renvoie l'ensemble des parents d'un nœud spécifique.

$$F_{SS}^{\ P} : NS \rightarrow NS^K \quad ns_i \mapsto F_{SS}(ns_i) = \{ns_1, ..., ns_K\} - \{ns_i\} \qquad \textbf{(1)}$$

La fonction (2) $F_{VS^{PQ}}$ permet de retrouver les fils d'un nœud spécifique dans le cadre d'une vue particulière (Q) rattachée à un document (P).

$$F_{VS}^{\ PQ} : NS \rightarrow NS^K \quad ns_i \mapsto F_{VS}(ns_i) = \{ns_1, ..., ns_K\} - \{ns_i\} \qquad \textbf{(2)}$$

Les fonctions F_{SG^P} et $F_{VG^{PQ}}$ sont similaires aux fonctions (1) et (2) pour un nœud générique.

$$F_{SG}^{\ P} : NG \rightarrow NG^K \quad ng_i \mapsto F_{SG}(ng_i) = \{ng_1, ..., ng_K\} - \{ng_i\} \qquad \textbf{(3)}$$

$$F_{VG}^{\ PQ} : NG \rightarrow NG^K \quad ng_i \mapsto F_{VG}(ng_i) = \{ng_1, ..., ng_K\} - \{ng_i\} \qquad \textbf{(4)}$$

La fonction (5) F_{G^P} permet de retrouver le nœud générique représentant un nœud spécifique. À ce niveau, nous pouvons définir aussi la fonction $F_{G^{P-1}}$ qui permet de déterminer les nœuds spécifiques rattachés à un nœud générique.

$$F_G^{\ P} : NS \rightarrow NG \quad ns_i \mapsto F_G(ns_i) = \{ng_i\} \qquad \textbf{(5)}$$

IV.2.2. Ensembles de règles

Afin d'organiser l'ensemble des objets que nous avons définis, nous élaborons un ensemble de règles. Ces règles ont pour objectif de déterminer la contribution de chaque objet à la construction des autres. Nous décrivons dans ce qui suit trois catégories de règles.

IV.2.2.1. Règles spécifiques

Les règles spécifiques concernent le niveau spécifique. A ce niveau, nous traitons la spécificité d'un document particulier « P ». Ainsi, notre première règle (6) traduit la

décomposition d'un document P en une structure spécifique $SS(Doc^P)$ et un ensemble de déclarations $Dec(Doc^P)$.

$$Doc^P \equiv \langle SS(Doc^P), Dec(Doc^P) \rangle \quad \textbf{(6)}$$

La règle (7) présente les éléments « formant » une structure spécifique P. Cette structure est composée d'un ensemble de nœuds spécifiques $NS(SS^P)$ et d'un ensemble de relations spécifiques $RS(SS^P)$. Les relations entre les objets de cet ensemble sont assurées par la fonction γ_{SS}^{N}. Par exemple, si nous considérons un document « k » qui est composé de trois nœuds ns_1, ns_2 et ns_3 tels que ns_1 est composé de ns_2 et ns_3 respectivement selon les deux relations rs_1 et rs_2, on aura alors $\gamma_{SS}^{k}(rs_1)=ns_1ns_2$ et $\gamma_{SS}^{k}(rs_2)=ns_1ns_3$.

$$SS^P \equiv \langle NS(SS^P), RS(SS^P), \gamma_{SS^P} \rangle \quad \textbf{(7)}$$

La règle (8) présente l'organisation d'une vue Q spécifique d'un document P en nœuds et relations. La fonction γ_{VS}^{PQ} traduit les relations entre les nœuds d'une vue Q associée à un document P.

$$VS^{PQ} \equiv \langle NS(VS^{PQ}), RS(VS^{PQ}), \gamma_{VS^{PQ}} \rangle \quad \textbf{(8)}$$

La règle (9) montre la nature d'un nœud spécifique. Chaque nœud spécifique peut être un élément ou un attribut.

$$NS \equiv \langle ES \vee AS \rangle / ES \cap AS = \varnothing \quad \textbf{(9)}$$

La règle (10) met en œuvre les relations entre vues spécifiques et structures spécifiques associées à un même document P. La structure représentée par une vue Q est considérée comme une partie de la structure globale du document. Ainsi, cette sous-structure est incluse dans la structure spécifique du document. Cette inclusion implique que tous les nœuds appartenant à une vue sont inclus dans l'ensemble des nœuds formant la structure spécifique et toutes les relations de cette vue sont incluses dans l'ensemble des relations de la structure spécifique.

$$\forall P \text{ et } \forall Q, VS^{PQ} \subseteq SS / NS(VS^{PQ}) \subseteq NS(SS^P) \text{ et } RS(VS^{PQ}) \subseteq RS(SS^P) \quad \textbf{(10)}$$

La règle (11) traduit la composition de chaque structure spécifique en terme de vues spécifiques. Chaque structure spécifique comprend une ou plusieurs vues spécifiques qui partagent les nœuds et les relations telles qu'il est présenté dans la règle (10).

$$SS^P \equiv \{VS^{PQ}\} / \forall Q, NS(VS^{PQ}) \subseteq NS(SS^P) \text{ et } RS(VS^{PQ}) \subseteq RS(SS^P) \quad \textbf{(11)}$$

Selon notre approche, il est possible que deux vues spécifiques partagent des nœuds spécifiques. Ce partage est vérifié par la règle (12).

$$\forall A \text{ et } B \text{ deux vues d'un même document } P, \quad NS(VS^{PA}) \cap NS(VS^{PB}) = NS(VS^{PA} \times VS^{PB}) \quad \textbf{(12)}$$

IV.2.2.2. Règles génériques

Les règles génériques concernent le niveau générique de notre modèle. La règle (13) présente l'organisation d'une vue générique Q appartenant à une structure générique P. La fonction Υ_{VG}^{PQ} traduit les relations entre les nœuds génériques d'une vue Q.

$$VG^{PQ} \equiv \left\langle NG\left(VG^{PQ}\right), RG\left(VG^{PQ}\right), \gamma_{VG^{PQ}} \right\rangle \quad \textbf{(13)}$$

La règle (14) présente la composition de la structure générique. Cette dernière rassemble toutes les sous-structures génériques encapsulées dans des vues génériques formant ainsi un graphe. Cette structure représente à un ensemble de structures spécifiques. Ainsi, elle est composée par des nœuds génériques et des relations génériques.

$$SG^{P} \equiv \left\langle NG\left(SG^{P}\right), RG\left(SG^{P}\right), \gamma_{SG^{P}} \right\rangle \quad \textbf{(14)}$$

La règle (15) met en œuvre la relation entre une vue générique Q et la structure générique relative P. La structure représentée par une vue Q est incluse dans la structure générique de sorte que tous les nœuds constituants cette vue sont inclus dans l'ensemble des nœuds formant la structure générique et toutes les relations de cette vue sont incluses dans l'ensemble des relations de la structure générique.

$$\begin{aligned} &\forall P \, et \, \forall Q, VG^{PQ} \subseteq SG \,/\, NG(VG^{PQ}) \subseteq NG(SG^{P}) \\ &\qquad et \; RG(VG^{PQ}) \subseteq RG(SG^{P}) \end{aligned} \quad \textbf{(15)}$$

La règle (16) traduit la composition de chaque structure générique en terme de vues génériques. Une structure générique est une organisation de vues génériques qui partagent des nœuds génériques et des relations génériques (Cf. règle 16).

$$SG^{P} \equiv \left\{ VG^{PQ} \right\} / \forall Q, NG(VG^{PQ}) \subseteq NG(SG^{P}) \, et \; RG(VG^{PQ}) \subseteq RG(SG^{P}) \quad \textbf{(16)}$$

IV.2.2.3. Règles génériques/spécifiques : lien de conformité

Les règles génériques/spécifiques découlent de la classification de documents proposée et par conséquent du lien de conformité que nous avons défini (Cf. section IV.1).

Un graphe G^P assure la correspondance entre le niveau générique et le niveau spécifique. La fonction Υ_{G}^{P} rattache un nœud spécifique à son représentant générique au travers d'une relation $RSG(G)^P$ (Cf. règle 17).

$$G^{P} \equiv \left\langle NG\left(G^{P}\right), NS\left(G^{P}\right), RSG(G^{P}), \gamma_{G^{P}} \right\rangle \quad \textbf{(17)}$$

La règle (18) traduit le fait que chaque nœud spécifique doit être rattaché à un nœud générique.

$$\forall ns_i \in NS, \; \exists \; ng_i \in NG \quad / F_{G^{P}}(ns_i) = \left\{ ng_i \right\} \quad \textbf{(18)}$$

IV.3. Synthèse

Dans cette section, nous avons décrit le modèle MVDM selon deux approches de modélisations : modélisation UML et modélisation formelle. Si la première modélisation offre une vision claire et facilement lisible, elle ne peut pas traduire certains aspects tels

que la liaison entre les deux niveaux spécifique et générique, la composition d'une structure en terme de nœuds et de vues. De ce fait, nous avons proposé dans un premier temps, un nouveau lien que nous avons appelé lien de conformité. Ce lien nous a permis de relier les deux niveaux générique et spécifique afin de traduire que le niveau spécifique ce n'est qu'une instance du niveau générique. Dans un deuxième temps, nous avons décrit le modèle MVDM selon une approche formelle. Ceci nous a permis d'éliciter d'une part, la composition d'une vue spécifique (Cf. règle 8), d'une vue générique (Cf. règle 13), d'une structure spécifique (Cf. règle 7) et d'une structure générique (Cf. règle 14) en terme de nœuds et de relations ; et d'autre part, l'agrégation d'un ensemble de vues spécifiques dans une structure spécifique (Cf. règles 10 & 11) et l'agrégation d'un ensemble de vues génériques dans une structure générique (Cf. règles 15 & 16). Cet ensemble de règles ont permis de lever l'ambigüité entre la composition d'une structure (générique ou spécifique) en terme de nœuds et relations ; et le rôle d'agrégat de vues qui est associé à cette structure.

V. Conclusion

Notre approche de gestion de la multistructuralité documentaire passe par la proposition d'un modèle de représentation. Ce modèle est basé sur la notion de vue. Une vue permet de représenter les différentes structurations possibles d'un document : structure physique, structure logique, etc. Nous avons élargi ce concept de vue de façon à pouvoir prendre en compte plusieurs structures de même type : n structures physiques, n structures logiques, etc.

L'originalité de notre modèle réside dans :

- la définition de vues indépendantes malgré l'éventuel partage de nœuds entre ces vues ;
- la séparation entre structures (représentées sous forme de vues) et contenu (gardé intact) ;
- la prise en compte conjointement des structures multiples définies sur le niveau global du document et des structures multiples associées à une des entités de ce document, tout en assurant la dichotomie entre ces deux types de structures ;
- dichotomie entre niveau générique (classification structurelle d'une sous-collection de documents) et niveau spécifique (préservation de la (des) structure(s) d'un document).

Ceci offre une certaine richesse et une grande flexibilité dans la définition des structures et la description du contenu des documents. En effet, la définition de vues indépendantes permet de reconstruire un document selon un aspect particulier. Notre approche présente également l'avantage de stocker le contenu d'un document comme un bloc de données et chaque vue de ce document y fait référence. Cette solution élimine les redondances dues au stockage multiple du contenu appartenant aux différentes structures et par conséquent aux différentes vues. Elle permet aussi la gestion des éventuels chevauchements dûs aux définitions de plusieurs structures. Le niveau générique du

modèle « MDVM » permet de regrouper sous forme de classes les documents ayant des structures similaires sans pour autant perdre les informations spécifiques de ces documents. La classification selon des vues génériques d'un côté et selon des structures génériques d'un autre côté implique plusieurs points d'accès aux documents.

Nous avons atteint les objectifs fixés, c'est-à-dire :

- la modélisation indépendante de tout type de structure ; nous pouvons représenter tout type de structure arborescente ou non et au-delà de cela n versions d'un même type de structure pour un document ;
- la modélisation indépendante d'une structure pivot ; aucune des structures n'est privilégiée malgré le partage de nœuds ;
- la modélisation indépendante du type de document. Les attributs et les types d'éléments ne sont pas fixés *a priori*.

Nous avons décrit le modèle MVDM selon deux approches complémentaire. En nous basant sur le langage UML, nous avons détaillé les concepts de base du modèle proposé (structure, vue, nœud, relation) sous forme de métaclasses. Nous avons pu traduire les propriétés de ces concepts (attributs des métaclasses) et leurs organisations (relations entre métaclasses). Nous avons complété notre modélisation semi-formelle par une modélisation formelle. Au travers d'un ensemble de règles, nous avons détaillé ce qu'est un document, une structure spécifique, une vue spécifique, une structure générique et une vue générique. Ces règles nous ont permis notamment d'expliciter les agrégations entre structures spécifiques (resp. génériques) et vues spécifiques (génériques).

L'objectif de ce chapitre était de décrire le modèle « MVDM » que nous avons proposé et de mettre en exergue la démarche qui nous a amenés à la concrétisation de ce modèle. Au travers de ses deux niveaux, le modèle « MVDM » montre comment gérer les documents et leurs classes associées, mais ne détaille pas la démarche à suivre afin d'assurer la classification. Le chapitre suivant sera donc consacré à la description de la démarche de classification de documents sur laquelle sera basée l'instanciation du niveau générique du modèle proposé. Le chapitre suivant présente également des techniques de recherche et de restitution basées sur la richesse et la flexibilité du modèle « MVDM ».

VI. Bibliographie

Allen, J. F. (1991). “Time and time again: The many ways to represent time.” International Journal of Intelligent Systems, 6(4).

Djemal, K., Mbarki, M., et Vallès-Parlangeau, N. (2007). « Une approche multi-vues pour la gestion des documents multistructurés. » Document numérique, Entreposage de documents et données semi-structurées, 10(2), 37–61.

Djemal, K. (2007a). “A Multi-Views Repository for Multi-Structured Documents.” 9th International Conference on Enterprise Information Systems (ICEIS), 544-548.

Djemal, K., Soule-Dupuy, C., et Valles-Parlangeau, N. (2008a). “Modeling and Exploitation of Multistructured Documents.” Information and Communication Technologies: From Theory to Applications, 2008. ICTTA 2008. 3rd International Conference on, 1-6.

Djemal, K., Soule-Dupuy, C., et Valles-Parlangeau, N. (2008b). “Formal modeling of multistructured documents.” Research Challenges in Information Science, 2008. RCIS 2008. Second International Conference on, Marrakech, 227-236.

Ribière, M., Dieng-Kuntz, R., et Sophia-Antipolis, I. (2002). “AViewpoint Model for Cooperative Building of an Ontology.” U. Priss, D.

RTF, U. (1999). “OMG Unified Modeling Language Specification, Version 1.3, UML RTF proposed final revision.” OMG, June, 4(83), 486.

Chapitre IV – Document multistructuré : de l'intégration à la restitution

Résumé du chapitre. *Après avoir détaillé les spécificités du modèle « MVDM » dans le chapitre précédent, ce chapitre se focalise sur son exploitation. Il présente d'une part la démarche d'intégration des documents multistructurés et d'autre part la démarche de restitution de ces documents. La démarche d'intégration repose sur deux aspects : la dématérialisation et la classification de documents. La dématérialisation garantit l'extraction de l'organisation et du contenu du document. La démarche de classification permet d'associer la structure d'un document à sa classe structurelle. Ceci revient à calculer une distance dite « structurelle » qui tiendra compte à la fois de l'organisation des éléments (position hiérarchique et ordre), du coût d'adaptation des représentants des classes ainsi que de la représentativité des sous-graphes. Le chapitre s'achève sur la présentation de deux techniques de restitution des documents stockées selon le modèle « MVDM » à savoir l'interrogation et l'analyse multidimensionnelle.*

Sommaire du Chapitre IV.

I. Introduction

Les bases documentaires classiques retournent souvent de longues listes de documents que l'utilisateur ou le système doit parcourir s'il veut pouvoir trouver les unités d'information susceptibles de répondre à ses besoins. Afin d'optimiser cette tâche, nous évoquons deux catégories de solutions. Ces deux catégories correspondent à deux volets différents qui s'appliquent à deux niveaux distincts dans le processus de gestion des bases documentaires. Le premier volet cherche à réduire le nombre de documents retournés sans perdre les plus pertinents. Le deuxième volet cherche à créer des index sur le contenu du document lui même afin de restituer les fragments pertinents.

La classification de documents offre une alternative à la première catégorie de solutions par la génération de classes (ou « clusters »). Il s'agit d'un processus d'organisation de documents visant à l'optimisation du processus de restitution. Concrètement, la classification permet de focaliser les processus d'interrogation et de recherche sur un sous-ensemble de documents jugés similaires. Le modèle MVDM présenté dans le chapitre III assure la représentation de chaque document intégré selon une ou plusieurs vues spécifiques, mais aussi il permet de décrire, pour chaque vue la classe de vues génériques qui lui correspond. La question qui se pose à ce niveau est la suivante : comment rattacher une vue spécifique au document à la classe de vues la plus représentative de ses caractéristiques ?

Il s'avère que la structuration de documents apporte une dimension supplémentaire à l'indexation des documents. Les éléments de structure donnent un accès direct à des fragments de contenu et par conséquent à l'information recherchée. Si l'on considère que le document est par essence multistructuré, la prise en compte de ces différentes structures permet d'affiner l'exploitation de ces documents. La combinaison de contraintes sur plusieurs structures assure un résultat plus précis et permet de mieux localiser les fragments documentaires pertinents. Il faut donc trouver des techniques qui permettent d'exploiter cette richesse.

Nous présentons dans un premier temps la démarche d'intégration de nouveaux documents (Cf. section II). Dans cette section, nous nous intéressons en particulier à l'indexation et à la fragmentation du document à intégrer, ainsi qu'à sa classification. La section III présente des techniques d'exploitation des documents multistructurés permises grâce au modèle MVDM.

II. Démarche d'intégration de documents multistructurés

Dans le chapitre précédent, nous avons présenté le modèle MVDM dont les principes sont : le partage de nœuds et le partage de contenu entre les différentes vues spécifiques du document et la dichotomie entre une description générique et une description spécifique des documents. Dans cette section, nous allons détailler la démarche d'intégration des documents multistructurés tout en prenant en compte toutes les spécificités du modèle

MVDM. Nous essayons surtout de répondre aux questions suivantes : quelles sont les techniques à utiliser afin d'intégrer un document selon le modèle MVDM : extraction des vues spécifiques, leurs agrégations dans des structures spécifiques et génération des vues génériques représentatives des vues spécifiques extraites ? Quelle démarche faut-il suivre et sur quels critères faut-il se baser afin d'assurer une bonne discrimination des classes ?

Nous allons dans ce qui suit présenter une démarche d'intégration des documents multistructurés qui respecte notre modélisation. Cette démarche passe par deux étapes : (1) dématérialisation des documents et instanciation du niveau spécifique du modèle (Cf. Section II.1), et (2) classification des documents et instanciation du niveau générique du modèle (Cf. Section II.2).

II.1. Dématérialisation des documents et instanciation du niveau spécifique du modèle

Le niveau spécifique du modèle MVDM est caractérisé par un ensemble de vues permettant de représenter les différentes structures d'un document. L'intégration d'un document multistructuré selon des vues nécessite des parseurs spécifiques à chaque langage d'encodage. Or, la complexité de ces langages d'encodage a fait que, dans la pratique, ces derniers sont rarement utilisés pour représenter des corpus de documents multistructurés (Cf. chapitre 2). De ce fait, chaque structure d'un document multistructuré est représentée dans un fichier à part. Ainsi, notre démarche de dématérialisation de documents se base sur l'hypothèse suivante : un document multistructuré est représenté par un ensemble de fichiers. Chaque fichier est considéré comme une structuration différente de ce même document.

Le document passe, en premier lieu, par une étape de dématérialisation qui assure l'extraction de la structure et du contenu. Cette extraction consiste à spécifier, pour chaque document à intégrer dans la base, son organisation sous forme d'une arborescence ordonnée et étiquetée ainsi que son contenu. Les étiquettes d'une arborescence représentent les différentes balises et éventuellement les différents noms d'attributs extraits du document. A l'issue de cette phase, deux vues sont générées : une vue spécifique qui représente intégralement l'organisation du document et une vue générique qui matérialise la représentation générique d'une vue spécifique au niveau générique.

La génération d'une vue spécifique engendre l'instanciation des métaclasses *« VueSpé »*, *« NœudSpé »* et *« RelationSpé »* du modèle MVDM. Nous rappelons ici que le contenu du document est stocké au niveau de la métaclasse *« Document »* dans le champ *« contenu »*.

La représentation générique de la vue spécifique sera utilisée dans la phase de classification. Cette représentation est matérialisée par un ensemble d'instances des métaclasses *« VueGén »*, *« NœudGén »* et *« RelationGén »* du modèle MVDM. La création de la représentation générique de la vue spécifique repose sur une sélection des éléments et des attributs représentatifs de l'organisation de la vue spécifique mais aussi à la spécification de la multiplicité des occurrences de ces représentants et leurs imbrications.

En effet, un élément ou attribut peut avoir plusieurs instances dans un même document (par exemple un article peut avoir plusieurs auteurs). Au niveau générique, on ne matérialise qu'un représentant unique de ces éléments et attributs. Ces traitements sont résumés par l'algorithme « *GénérationVueGénérique* » (Cf. Annexe). Cet algorithme est basé essentiellement sur l'appel récursif de la fonction de traitement des fils. Les appels récursifs reviennent aux traitements des sous-arborescences de chaque nœud. Cette fonction s'exécute selon l'algorithme « *traiterFils* » (Cf. Annexe).

Avant de commencer la démarche de classification, il est intéressant de vérifier l'unicité des noms de nœuds génériques d'une même vue générique. En effet, dans certains cas, une même vue générique peut contenir des nœuds ayant des étiquettes identiques. Par exemple, dans un document audio, chacun des nœuds « Locuteur » et « Thème » peut être composé par un nœud fils « Nom ». La vérification de l'unicité consiste à lever cette ambiguïté qui peut être une source d'erreur dans la phase de classification. Ainsi, nous proposons de renommer les nœuds ayant la même étiquette. Plus précisément, nous préfixons chacune des étiquettes ayant un homonyme par l'étiquette de son père dans l'arborescence. La vérification d'unicité est décrite par l'algorithme « *vérifierUnicité* » (Cf. Annexe).

Selon le modèle MVDM, la nouvelle vue spécifique doit être agrégée à une structure spécifique. Cette agrégation se réalise au travers du partage de nœuds. Le choix de la structure spécifique dépend du contenu du document à intégrer. Il s'agit ici de vérifier si ce contenu existe déjà dans la base sous la forme d'une autre vue spécifique. Une question (à laquelle nous ne répondrons pas dans cette thèse) est de savoir comment déterminer de façon automatique que deux vues spécifiques partagent le même contenu et par conséquent elles décrivent le même document ? En effet, la comparaison du contenu si elle est possible dans le cas des documents textuels, elle ne retourne des résultats satisfaisants que dans le cas d'identité absolue entre les deux contenus comparés. Cette comparaison est d'autant plus complexe lorsqu'il s'agit de documents multimédias. En admettant que cette tâche est assurée par l'utilisateur, le gestionnaire de vues assure l'agrégation de ces derniers au sein d'une structure spécifique (existante ou à créer). Ainsi, deux cas se présentent :

Cas 1. le contenu associé à la vue spécifique du document à intégrer dans la base est jugé, par l'utilisateur, identique à celui d'une vue spécifique existante. Ce cas de figure correspond à l'ajout d'une structure à un document multistructuré qui existe déjà dans la base. Le gestionnaire de vue agrège alors la nouvelle vue spécifique à la structure spécifique sélectionnée par l'utilisateur ;

Cas 2. lorsque l'utilisateur juge qu'il s'agit d'une vue spécifique propre à un nouveau document, le gestionnaire de vue propose la création d'une nouvelle structure et vue spécifique. Cette structure spécifique symbolise un nouveau document multistructuré. Cependant, ce document n'admet qu'une seule vue spécifique à ce niveau de traitement.

La Figure IV.1 détaille la démarche d'intégration d'un nouveau document. Le générateur de vue se charge de la dématérialisation du nouveau document D_i. Il génère une vue spécifique $VS_{i?}$ sous forme d'instances des métaclasses « *VueSpé* », « *RelationSpé* »,

« *NoeudSpé* », « *Document* » et « *Déclaration* ». Ayant la liste des structures spécifiques existantes, l'utilisateur doit décider de rattacher la vue spécifique $VS_{i?}$ à une nouvelle structure spécifique j (SS_j) ou à une structure spécifique existante k (SS_k). Le gestionnaire de vue se charge d'agréger $VS_{i?}$ à la structure spécifique (j ou k selon le choix effectué). Il permet de gérer le partage de nœuds spécifiques et de contenu entre les vues. Le contenu du document sera stocké une seule fois indépendamment de l'ensemble des vues du document.

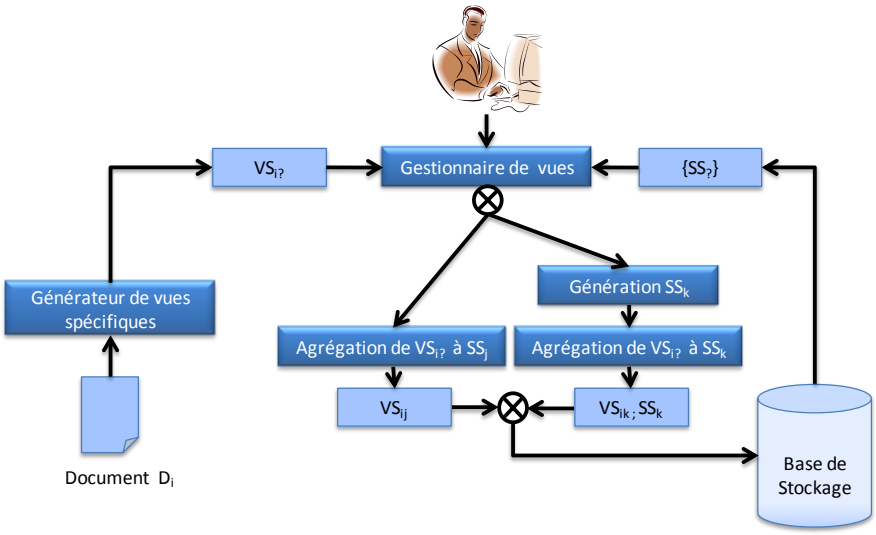


Figure IV.1. Démarche d'instanciation du modèle - niveau spécifique.

Remarque importante : Dans ce qui suit, par soucis de simplification, le terme *vue spécifique* est utilisé pour parler de *représentation générique d'une vue spécifique*.

II.2. Classification de vues et instanciation du niveau générique du modèle

Les systèmes de recherche d'information classiques retournent souvent de longues listes de fragments retrouvés à partir de tous les documents de la base. L'utilisateur ou le système doit parcourir ces listes s'il veut pouvoir trouver les unités d'information susceptibles de répondre à ses besoins. La classification de documents offre une alternative à ce type de restitution de résultats par la génération de classes (ou « clusters »). Il s'agit d'un processus d'organisation de documents visant l'optimisation du processus de restitution (Soulé-Dupuy 2001).

Comme toute approche de classification, la catégorisation de documents peut être accomplie en mode supervisé appelé en anglais *« classification »*, ou non supervisé appelé en anglais *« clustering »*. En mode supervisé, les catégories (ou classes) de documents sont connues *a priori*. Ces classes ont en général un sens pour leur concepteur. La tâche du classifieur se limite dans ce cas à associer le document à la classe « la plus proche ». Afin d'assurer cette classification, une méthode d'apprentissage est nécessaire. Il existe plusieurs méthodes d'apprentissage non supervisées, à savoir : k-means (MacQueen 1966), arbre de décisions (Breiman 1998), réseaux de neurones (Schütze et al. 1995), machine à

support de vecteurs « SVM » (Joachims 1997), etc. En mode non supervisé, les classes ne sont pas connues *a priori*. C'est le classifieur qui se charge de déterminer les classes de documents et d'affecter un nouveau document à une classe. Quelle que soit la méthode, les classes sont fondées sur la structure propre de l'ensemble des documents à classer. Elles n'ont une signification que par rapport aux critères de classification choisis.

La première étape de construction d'un classifieur repose sur le choix de paramètres discriminants qui permettront d'assurer la dichotomie entre classe. Cette étape est cruciale dans la qualité du classifieur. Nous pouvons distinguer, en ce qui concerne le document, deux types de paramètres peuvent être pris en compte :

- paramètres structurels : le rattachement d'un document à une classe est induit par des données factuelles extraites de sa structure ;
- contenu (sémantique, mot clés, etc.) : le critère de classification porte sur la partie non structurée du document. Ce type de classification est généralement réalisé automatiquement en tenant compte de la similarité entre contenus de documents.

Le dernier paramètre repose sur une représentation plate du document. Ainsi, ce dernier est considéré comme un monolithe ou comme un « sac de mots ». (Soulé-Dupuy 2001) ; (Doucet et Ahonen-Myka 2002) exploitent la richesse du modèle vectoriel qui est défini comme une perspective d'amélioration de performance des systèmes de recherche d'informations. Ils représentent un document sous la forme d'un vecteur dans lequel les éléments sont soit des balises (éléments de structures), soit des mots du texte (contenu sémantique) soit une combinaison des deux. (Yi et Sundaresan 2000) représentent la structure du document en utilisant le modèle vectoriel. Dans ce cas, les éléments d'un vecteur sont soit des mots (contenu), soit d'autres vecteurs (représentation récursive de la structure).

Certes, la prise en compte du contenu dans les démarches de classification améliore les résultats en termes d'homogénéité sémantique des classes et de pertinence des documents classés. Cependant, elle requiert une phase d'indexation qui complexifie ce processus. Ainsi, plusieurs approches se limitent à des paramètres structurels pour la classification surtout que généralement les éléments de structure intègrent une forme de sémantique.

La seconde étape est la construction des classes. Comment arriver à regrouper ou répartir des documents dans des classes représentatives et homogènes ? Nous parcourons ici quelques processus classiques de construction des classes :

- Estimation statistique :

La construction d'une classe dépend d'un ensemble de paramètres statistiques. (Diligenti et al. 2001) proposent un modèle de classification basé sur les Chaînes de Markov Caché. Ils considèrent que l'arborescence d'un document est générée par une chaîne (arbre) d'états cachés. (Piwowarski et al. 2002) (Denoyer et Gallinari 2004) proposent de modéliser un document à l'aide de Réseaux Bayésiens. Ils modélisent les dépendances statistiques entre nœuds d'une même structure via un modèle génératif. Afin

d'améliorer la classification, (Vu et al. 2003) transforment ce modèle en un modèle discriminant en utilisant la méthode du noyau Fisher. L'inconvénient majeur de ces approches reste sans doute la phase d'apprentissage nécessaire avant d'appliquer les algorithmes de classification ;

- Sous-arbres fréquents :

(Termier et al. 2002) ; (Costa et al. 2004) ; (Kutty et al. 2008) ; (Saleem 2008) proposent d'utiliser la notion d'arbres fréquents afin de classifier les structures. L'objectif de leurs approches est de trouver les sous-arbres qui sont inclus dans au moins n arbres d'une collection. (Vercoustre et al. 2006) proposent de transformer l'arborescence d'un document en un ensemble de chemins. Ainsi, ils calculent la fréquence de ces chemins afin d'assurer la classification des documents. Bien que cette méthode de classification permette d'avoir des classes ayant des documents qui partagent des sous-arborescences communes, ces documents peuvent avoir des arborescences assez distinctes. De plus, le coût des calculs nécessaires à ce type d'approche augmente d'une façon exponentielle en fonction de la longueur des sous-arborescences et du nombre de documents ;

- Calcul de distance :

Cette technique est basée sur le calcul de similitude entre documents. Dans la littérature, deux types de distances ont été proposés : la distance d'édition et la distance d'alignement. Les algorithmes de calcul de la distance d'édition sont apparus pour la comparaison de chaînes de caractères (Wagner et Fischer 1974). (Shasha et Zhang 1997) définissent la distance d'édition entre deux arborescences comme étant la somme des coûts des séquences d'opérations d'édition qui transforment une arborescence en une autre. Une opération peut être un ajout, une suppression ou une modification (Zhang et al. 1996). Dans ces travaux, les calculs de distance sont appliqués à des arborescences. Si ces algorithmes intègrent dans le calcul de distance le niveau du nœud dans l'arborescence, ils ne tiennent pas compte de l'ordre des nœuds à un même niveau. La distance d'alignement (Romany et Bonhomme 2000) peut être considérée comme un cas particulier de la distance d'édition. Pour pouvoir calculer la distance d'alignement, il est nécessaire que les deux arbres concernés soient isomorphes. Ainsi, des nœuds « espaces » étiquetés par λ, sont ajoutés aux deux arborescences. La superposition de ces dernières permet d'avoir des couples de nœuds alignés. La distance d'alignement est la somme des coûts de chaque couple possédant des étiquettes différentes (Jiang et al. 1995). Contrairement aux techniques utilisées pour le calcul de distance d'édition, cette technique peut être appliquée à des graphes. En contrepartie, les coûts tels qu'ils sont présentés, ne tiennent pas compte de l'importance du nœud modifié dans un graphe. Cette importance est issue de l'influence du changement d'un nœud sur les nœuds qui en dépendent.

Dans un autre objectif, celui de la gestion des versions, (Cobena et al. 2002a) ont proposé un outil intitulé XyDiff qui permet la détection des changements structurels dans un document XML. Cet outil est basé sur un algorithme de calcul de distance entre structures (Cobena et al. 2002b). Contrairement à XyDiff, l'outil X-Diff (Wang et al. 2003) est proposé dans le but d'obtenir de résultats optimums. Il est performant pour les

documents de taille petite ou moyenne. Cependant, en raison de la complexité de l'algorithme, son temps d'exécution est assez long quand les deux documents à comparer sont assez complexes. La deuxième limite de cette approche concerne son incapacité à prendre en compte l'ordre des éléments frères de même niveau.

Lorsque dans la plupart des travaux de classification, la notion de représentant de classe n'est pas explicitée, (Tekli et al. 2007) et (Bertino et al. 2002) considèrent la DTD comme un représentant d'une classe de documents XML. Afin de représenter plus de documents conformes à une même DTD dans une même classe, (Dalamagas et al. 2006) proposent de transformer les documents XML au travers de moulinettes XSLT. Or, ces transformations se limitent à réordonner des éléments afin que le document soit valide par rapport à une DTD. Ainsi, ils ne peuvent classifier que des documents strictement identiques. De plus, l'organisation initiale du document sera perdue suite à ces transformations.

En se focalisant sur des problématiques d'entreposage de documents, ce problème de classification a été abordé au sein de notre équipe dans le cadre de la thèse de K. Khrouf (Khrouf 2004) et celle de M. Mbarki (Mbarki 2008). K. Khrouf a proposé un modèle générique qui permet de représenter les structures logiques des documents ainsi que leur contenu. Afin d'assurer la classification des documents, ces structures logiques sont regroupées selon des structures génériques. Une extension de ce modèle est présentée dans les travaux de thèse de M. Mbarki. Cette extension consiste en la prise en compte de la sémantique du document qui est traduite par l'ensemble des structures de métadonnées. Une structure générique de métadonnées est ajoutée au modèle pour représenter ces structures spécifiques de métadonnées. Quelle que soit leur nature, les structures génériques dans ces travaux sont représentées sous forme arborescente. Or, une telle « forme » ne peut pas couvrir que des structures spécifiques qui admettent les mêmes hiérarchies ou des hiérarchies complémentaires.

II.2.1. Démarche d'instanciation du niveau générique du modèle

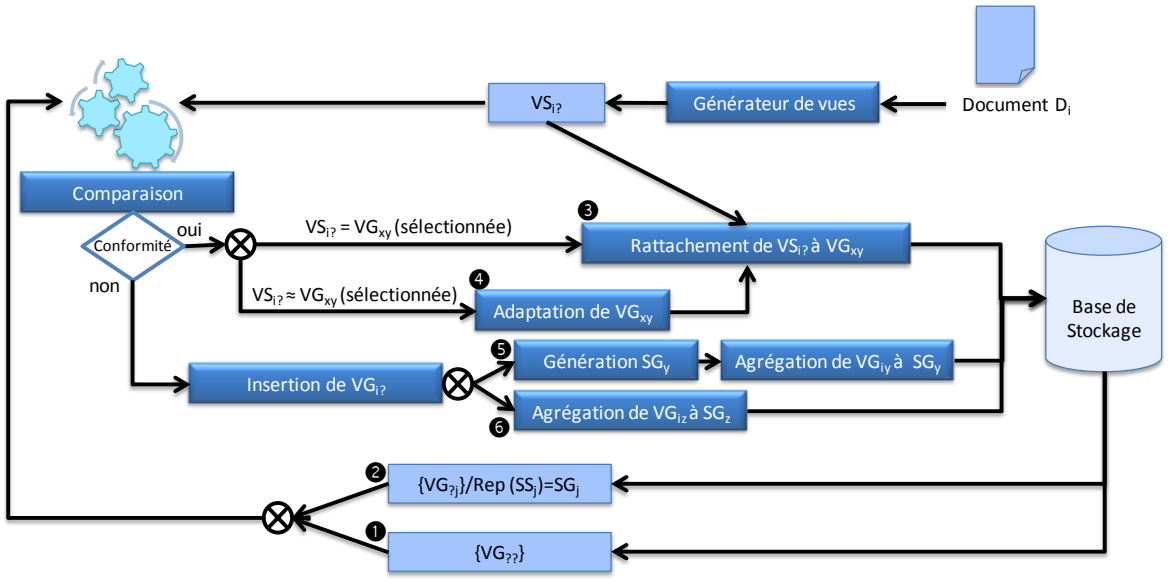
Pour notre cas, afin de choisir la vue générique à laquelle la nouvelle vue spécifique doit être rattachée, nous proposons une démarche de classification hiérarchique ascendante non supervisée. Le premier document inséré dans la base sert de premier représentant. Les classes sont ensuite construites par agrégation des individus (documents) structurellement proches. L'organisation d'une vue générique traduit l'organisation du représentant d'une classe. Ces représentants, contrairement à ce qui est réalisé dans les autres approches (Tekli et al. 2007) (Bertino et al. 2002) et (Dalamagas et al. 2006), peuvent être sous forme de graphe. Une telle représentation est le résultat d'une superposition des arborescences structurellement identiques ou « proches » rattachées à une même classe.

La démarche classification a pour but d'organiser des individus en les regroupant en classes homogènes en fonction de leurs caractéristiques. Au niveau de la classification que nous voulons désormais réaliser, les individus sont représentés par les vues spécifiques et les classes sont matérialisées par les vues génériques. La description d'une vue spécifique est matérialisée au travers de l'organisation de ses nœuds. Nous considérons que cette

propriété joue un rôle principal dans la caractérisation d'une vue. De ce fait, le regroupement des individus est établi à partir du calcul d'une distance appelée « structurelle ». Cette distance joue le rôle de critère de ressemblance appelé encore critère d'agrégation.

La description générique offerte par le modèle MVDM traduit cette classification au travers des vues spécifiques rattachées à des vues génériques. La construction du niveau générique du modèle passe par la génération de nouvelles instances des métaclasses « StrGén », « VueGén », « NoeudGén » et « RelationGén ». La classification revient alors à rattacher les instances des métaclasses du modèle spécifique (telles que la vue, la structure, les nœuds et les relations) aux instances des métaclasses du modèle générique. La Figure IV.2 expose la démarche d'instanciation du niveau générique et de rattachement des nouvelles instances du modèle spécifique à sa classe de vues.

Le choix d'une vue générique peut se faire soit parmi toute les vues génériques existantes (Cf. Figure IV.2.1), soit parmi un sous ensemble de vues génériques agrégées à une structure générique déterminée (Cf. Figure IV.2. 2). Ce dernier cas est relatif à une vue spécifique agrégée à une structure spécifique définie ; dans ce cas cette vue spécifique doit être rattachée à une des vues génériques agrégées dans la structure générique représentative de la structure spécifique en question.



$\{VG_{??}\}$ est l'ensemble des vues génériques existantes
$\{VG_{?j}\}/Rep\ (SS_j)=SG_j$ est l'ensemble des vues génériques agrégées à la structure générique SG_j représentative de la structure spécifique SS_j
$VS_{i?}$ est la vue spécifique du document D_i

VG_{xy} est la vue générique la plus proche de $VS_{i?}$
$VG_{i?}$ est la nouvelle vue générique qui représente $VS_{i?}$
SG_y est une nouvelle structure générique
SG_z est une structure générique existante

Figure IV.2. Démarche d'instanciation du modèle - niveau générique.

Lors du rattachement d'une vue spécifique à une vue générique, trois cas peuvent se présenter (Cf. Figure IV.2). La vue spécifique à intégrer peut-être :

- totalement incluse dans une vue générique existante. Dans ce cas, nous rattachons la vue spécifique du document à cette vue générique (Cf. Figure IV.2.3) ;

- partiellement différente, mais relativement proche, d'une vue générique existante. Dans ce cas, nous adaptons cette vue générique afin qu'elle puisse représenter la nouvelle vue spécifique (Cf. Figure IV.2.4). Ensuite nous rattachons la vue spécifique du document à la vue générique adaptée ;
- totalement différente de toutes les vues génériques existantes dans la base (pas de classification possible selon les classes existantes). Dans ce cas, nous créons une nouvelle vue générique de la nouvelle vue spécifique. Cette nouvelle vue générique sera soit agrégée à une nouvelle structure générique (Cf. Figure IV.2.5), soit agrégée à une structure générique existante (Cf. Figure IV.2.6).

Afin de pouvoir prendre de telles décisions, il est impératif de pouvoir comparer deux vues et mesurer leur degré de ressemblance. Cette comparaison revient à calculer la distance entre deux vues. Ainsi, avant de détailler notre démarche de classification, nous présentons dans la section suivante comment calculer cette distance.

II.2.2. Comparaison de vues : calcul d'une distance structurelle

Pour comparer deux vues, il faut pouvoir mesurer leur degré de similarité. Ce degré de similarité est calculé à partir d'une distance dite distance structurelle (Djemal et al. 2010b).

Notre démarche de comparaison de vues passe par trois étapes (Cf. Figure IV.3) :

- la pondération des relations de chacune des deux vues à comparer. A la fin de cette étape, une matrice de représentation est générée pour chaque vue ;
- alignement de vues : cette étape consiste à ajouter des nœuds virtuels à chacune des deux vues de façon à avoir deux représentations virtuellement similaires en terme de nœuds. A ce niveau du processus de comparaison, l'adaptation des vues revient à l'adaptation de leur matrice de représentation ;
- le calcul de similarité entre les deux vues doit déterminer la distance entre ces deux vues. Plus précisément, cette étape consiste à calculer un degré de similarité *Sim* basé sur le calcul de la distance d'alignement des différentes relations des deux vues.

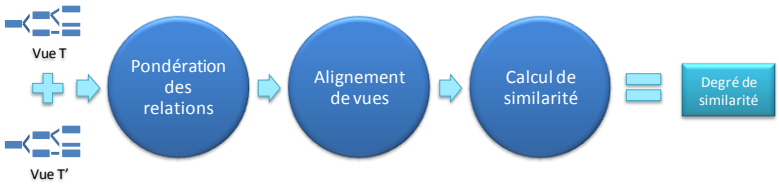


Figure IV.3. Démarche de comparaison de vues.

II.2.2.1. Concepts de base

Soit $\mathbb{G} = (\mathbb{V}, \mathbb{E})$ un graphe ordonné, orienté et étiqueté, avec $\mathbb{V} = \{v_1, \dots, v_n\}$ l'ensemble des nœuds de $\mathbb{G}$ et $\mathbb{E} = \{\mathcal{E}_1, \dots, \mathcal{E}_m\}$ l'ensemble des arcs de $\mathbb{G}$:

- v_0 est un nœud unique n'appartenant à aucun graphe, appelé nœud vide ;

- $fils[v]$ est la liste des fils du nœud v. Le nombre de fils du nœud v est noté nf_v ;
- $anc[v]$ est l'ensemble des ancêtres du nœud v dans $\mathbb{G}$;
- $chm[v]$ est la liste des nœuds qui composent le chemin allant de la racine r vers v de $\mathbb{G}$, v inclus ($chm[v] = \{v\} + anc\ [v]$) ;
- $pere[v]$ est la liste des pères du nœud v. Le nombre de pères du nœud v est noté np_v ;
- $desc[v]$ est l'ensemble des descendants du nœud v dans $\mathbb{G}$. Ce sont des nœuds ayant v comme ancêtre, v non inclus ;
- $départ\ (\mathcal{E})$ est le nœud origine ou de départ de l'arc $\mathcal{E}$;
- $arrivée\ (\mathcal{E})$ est le nœud extrémité ou d'arrivée de l'arc $\mathcal{E}$;
- $etiq[v]$ est l'étiquette associée au nœud v. Les étiquettes sont les noms des balises ;
- $card(\mathbb{G}')$ est le nombre de nœuds appartenant à $\mathbb{G}'/\mathbb{G}' \subseteq \mathbb{G}$ ($\mathbb{G}$' peut être un graphe ou une partie du graphe) ;
- représentation matricielle : en mathématique, une matrice d'adjacence relative à un graphe G à n sommets est une matrice carrée de dimension n*n. Chaque élément non-diagonal a_{ij} traduit l'existence d'une relation entre les deux sommets i et j.
  Nous nous sommes basés sur le principe d'une matrice d'adjacence pour représenter nos graphes. Cependant, les éléments (a_{ij}) de notre matrice ne renseignent pas seulement sur l'existence d'une relation $\mathcal{E}$ entre deux sommets, mais aussi sur le poids $\mathcal{P}(\varepsilon)$ de cette relation. Ainsi, la matrice $\mathcal{M}_p$ est une matrice représentative d'un graphe G telle que :
  $$(a_{ij})_{\mathcal{M}_p} = \begin{cases} \mathcal{P}(\varepsilon), & si\ \exists\ \mathcal{E} \in \mathbb{E}\ /\ \mathcal{E}\ relie\ v_i\ et\ v_j\ \ avec\ v_i, v_j\ \in \mathbb{V} \\ 0, & sinon. \end{cases}$$

La transformation des graphes pondérés en matrices permet de minimiser les traitements nécessaires aux calculs de distance entre ces graphes. En effet, après la transformation de chacun des graphes pondérés en matrices, la distance se calcule à partir de la différence des deux matrices. La transformation d'un graphe en matrice nécessite un seul parcours de ce dernier. En contrepartie, le calcul de distance entre deux graphes pondérés directement et sans passer par les matrices nécessite au minimum 1 parcours du premier graphe et n-1 parcours du deuxième afin de rechercher pour chacun des arcs du premier graphe son correspondant dans le deuxième graphe. Ceci dit, cette représentation peut générer plusieurs termes vides (matrices creuses). Ces matrices présentent un double problème : le premier est relatif à l'espace mémoire nécessaire pour charger chacune de ces matrices et le deuxième est relatif aux traitements et opérations utilisant ces matrices qui peuvent être de plus en plus lents et admettent des résultats qui tendent vers zéro. Ce deuxième problème ne se pose pas dans nos travaux du fait qu'on utilise des opérations élémentaires telles que différence, addition ou multiplication terme à terme.

II.2.2.2. Pondérations des relations

Une vue générique telle que nous l'avons conçue, peut avoir trois formes : elle peut être arborescente, sous forme de graphe acyclique, ou encore de graphe cyclique (au moins un cycle). Afin de démontrer la faisabilité de notre démarche de pondération, nous traitons un exemple de chaque forme de vue générique (Cf. Figure IV.4).

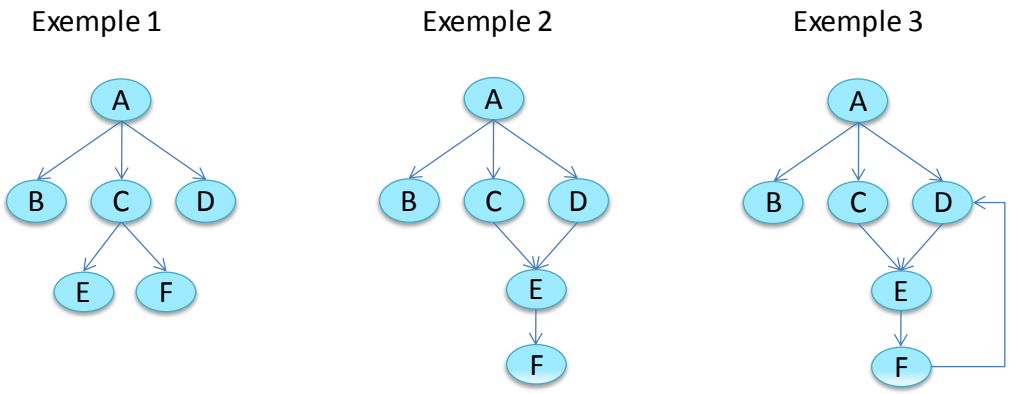


Figure IV.4. Trois exemples de vues génériques.

Nous avons opté pour une classification basée sur des paramètres structurels. Le premier paramètre, presque évident, concerne l'organisation des relations de la vue. Nous entendons par organisation de la vue, à la fois la hiérarchie des nœuds, mais aussi l'ordre des nœuds sur un même niveau hiérarchique. Le second paramètre, un peu plus fin, prend en compte le coût d'adaptation de la vue générique à la vue spécifique. Le troisième paramètre traduit la représentativité des chemins pour l'ensemble des structures rattachées à la classe. Ces trois paramètres permettent de calculer une pondération globale pour chaque relation.

A. Pondération structurelle

- Définition

La pondération structurelle consiste à attribuer des poids aux relations d'une vue générique de manière à tenir compte d'une part de la profondeur (niveau du ou des nœuds pères par rapport à la racine) et d'autre part de l'ordre (ordre des nœuds fils par rapport à leur(s) nœud(s) père(s)).

- Technique utilisée et formalisation

Soit $\mathcal{P}_{Str}$ une fonction déterminant le poids de chaque relation ($\mathcal{E}$) par rapport à sa position « relative » dans un graphe :

$$\mathcal{P}_{Str} : \mathbb{E} \rightarrow]0..1[$$

$$\varepsilon \mapsto \mathcal{P}_{Str}(\varepsilon) = \begin{cases} \frac{\beta}{N^{\alpha}}, & si\ départ(\mathcal{E}) = v_r\ ; \\ \frac{\sum_{i=1}^{k} \mathcal{P}_{Str}(\mathcal{E}_i)}{k} + \frac{\beta}{N^{\alpha}}, & sinon. \end{cases}$$

Où

- v_r est le nœud racine du graphe avec $v_r \in \mathbb{V} / anc[v_r] = \emptyset$;
- $\mathcal{E}_i \in \mathbb{E} / \forall i \in [1..k]$; $arrivée(\mathcal{E}_i) = départ(\mathcal{E})$;
- α : niveau moyen du nœud v_j dans le graphe tel que $v_j = départ(\mathcal{E})$;
- β : numéro d'ordre du nœud v_{j+1} dans l'ensemble des fils du nœud v_j tel que $v_j = départ\ (\mathcal{E})$ et $v_{j+1} = arrivée(\mathcal{E})$;

- k : nombre de nœuds père du nœud v_{j+1} tel que $v_{j+1} = arrivée(\mathcal{E})$;
- N : nombre maximum de fils pour un même père défini par 10^x, $\forall$ x $\in$ [1..n] ;
- $\mathcal{P}_{Str}$ permet de gérer toute vue de document avec la contrainte que $card(fils(v)) < N$ pour tout nœud v non feuille. Par exemple, si N est égal à 100, un nœud peut avoir jusqu'à 99 fils. Nous utiliserons une valeur de N = 10.

La formule de pondération structurelle se base sur un calcul récursif des poids. Le poids d'une relation est déterminé à partir des poids des précédentes relations des k nœuds pères. Dans le cas d'un arbre, k=1, dans le cas d'un graphe, un nœud peut avoir plusieurs pères. La première partie de la formule permet de prendre en compte le poids des précédentes relations avec les k nœuds pères. La deuxième partie de la formule permet de se focaliser sur la position (ordre et profondeur) de la relation pondérée.

Dans un poids structurel, le nombre de décimales détermine la profondeur (β) de la relation pondérée et la valeur de la dernière décimale renseigne sur l'ordre de la relation (α). Les valeurs des autres décimales traduisent les positions d'ordre des précédentes relations dans chaque niveau. Par exemple, le poids structurel 0,43 signifie qu'il s'agit d'une relation de niveau 2. Cette relation est la troisième relation par rapport au nœud de départ, et ce nœud est le 4ème fils de son père (Cf. Figure IV.5).

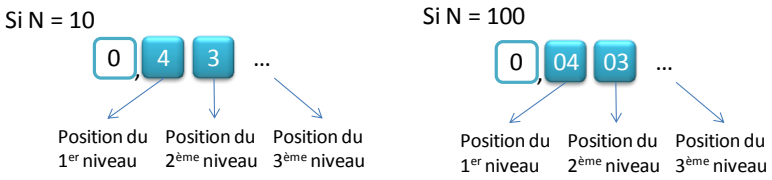


Figure IV.5. Exemple de poids structurels avec deux valeurs différentes de « N ».

Le calcul des poids structurels est exécuté selon l'algorithme « PondérationStructurelle » (Cf. Annexe).

❑ Représentation matricielle

Une matrice est générée afin de traduire les positions des différentes relations d'une vue. De ce fait, les éléments a_{ij} ne renseignent pas que sur l'existence d'une relation entre deux sommets, mais aussi sur la position relative de cette relation. La matrice $\mathcal{M}_{Str}$ est une matrice représentative d'une vue telle que :

$$(a_{ij})_{\mathcal{M}_{Str}} = \begin{cases} \mathcal{P}_{Str}(\varepsilon), & si\ \exists\ \mathcal{E} \in \mathbb{E} \,/\, \mathcal{E}\ relie\ v_i\ et\ v_j\ \ avec\ v_i, v_j\ \in \mathbb{V}\ ; \\ 0, & sinon. \end{cases}$$

❑ Exemple

Dans la figure suivante, nous présentons la pondération structurelle des trois exemples de la Figure IV.4. Pour chaque vue, nous détaillons également sa représentation matricielle.

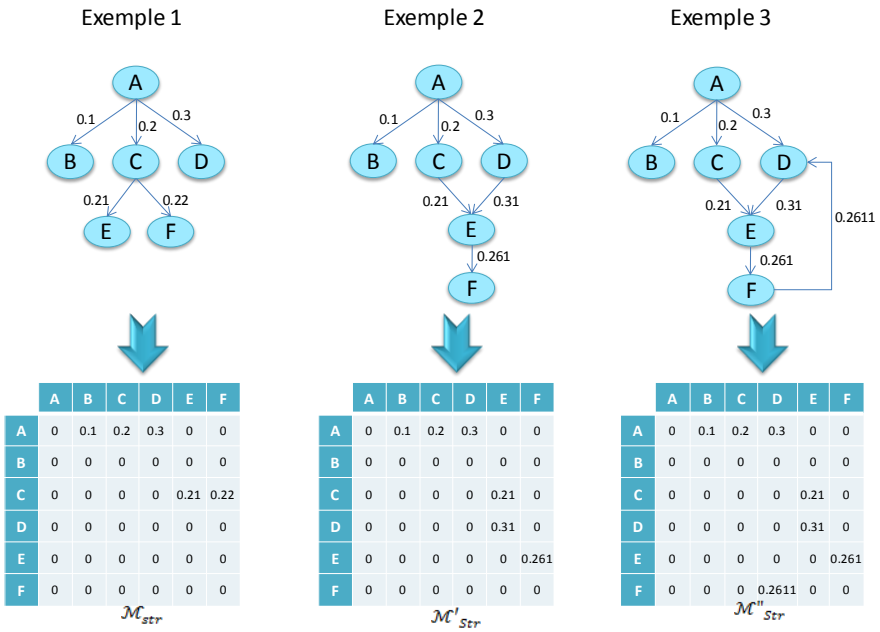


Figure IV.6. Pondérations structurelles (exemples de vues présentées dans la Figure IV.4).

B. Pondération d'adaptation

- Définition

Deux vues peuvent être considérées comme proches sans pour autant être identiques. Ainsi, la vue générique devra être adaptée à la vue spécifique à rattacher ; les nœuds et/ou relations manquants seront intégrés à la vue générique de départ. Si on prend l'exemple d'une modification de chemin pour un sous-graphe, plus ce sous-graphe comportera de nœuds et relations, plus la modification sur la vue sera importante. Cette remarque est cruciale pour des cas où nous sommes à la limite entre l'adaptation de la vue et la création d'une nouvelle classe. Si l'adaptation coûte trop cher, alors, mieux vaudra créer une classe plutôt que de surcharger une classe existante. Aussi, il nous semble important de pondérer la ressemblance purement structurelle par le coût d'adaptation de la vue générique.

- Technique utilisée et formalisation

La pondération d'adaptation consiste à attribuer des poids aux relations d'une vue générique en se basant sur l'appartenance de ces relations à des chemins. Plus il y aura de chemins dépendant d'une relation, et plus ce coût d'adaptation sera fort. Dans le cas d'une arborescence, un nœud dépend de son nœud père. Ainsi, plus la position du nœud à modifier se rapproche de la racine, plus le coût d'adaptation est élevé. De ce fait, pour une arborescence contenant n feuilles (n chemins), le poids de la première relation à partir de chaque nœud feuille est égal à 1/n. Par voie de conséquence, la somme des poids des relations dépendant directement de la racine est égale à 1.

Les vues génériques, telles que nous les avons conçues, peuvent représenter des arborescences, mais elles peuvent aussi représenter des graphes. Chaque vue générique est le résultat de la superposition de l'ensemble des arborescences qui traduisent les

représentations génériques des vues spécifiques rattachées. De ce fait, nous optons pour des vues génériques sous forme arborescente pour calculer les poids d'adaptation. La transformation de la vue générique à pondérer est donc nécessaire. Afin de réaliser cette transformation, nous proposons les deux hypothèses suivantes :

Hypothèse 1 : les relations (arcs) qui introduisent un cycle ne sont pas représentées dans la nouvelle arborescence. Ces relations seront prises en compte lors de l'affectation finale des poids d'adaptation. Dans ce cas, nous attribuons le poids le plus faible à ces relations qui ne sont ajoutées dans le graphe que dans le but d'assurer la représentativité de toutes les arborescences qui sont rattachées ;

Hypothèse 2 : les nœuds qui admettent plus qu'un père, et les nœuds qui en dépendent, sont dupliqués autant de fois que le nombre de leurs pères. Ainsi, des poids temporaires sont attribués aux nouvelles relations issues des nœuds dupliqués. Après le calcul et lors de l'affectation finale des poids, les nœuds dupliqués seront regroupés à nouveau. Le poids d'adaptation final de chacune des relations relatives aux nœuds regroupés sera égal à la somme des poids temporaires de ses relations.

Au travers de ces deux hypothèses, nous réduisons le coût des relations introduisant des cycles du fait qu'elles ne sont introduites dans la vue générique que pour représenter une arborescence rattachée et nous augmentons le coût de modification (ajout ou suppression) des relations qui dépendent d'un nœud à plusieurs pères puisque ces relations ont plus de « chance » d'appartenir à un chemin.

La fonction $\mathcal{P}_{Adap}$ est une fonction déterminant la pondération d'adaptation de chaque relation ($\mathcal{E}$).

$\mathcal{P}_{Adap} : \mathbb{E} \rightarrow]0..1[$

$$\varepsilon \mapsto \mathcal{P}_{Adap}(\varepsilon) = \begin{cases} 1/m, & si\ \mathcal{E} = \mathcal{E}_f; \\ \sum_{j=1}^{t} \mathcal{P}_{Adap}(\mathcal{E}_j), & sinon. \end{cases}$$

Où

- m : représente le nombre total de nœuds feuilles $v_f \in \mathbb{V} / fils[v_f] = \emptyset$;
- $\mathcal{E}_f \in \mathbb{E} / v_f = arrivée(\mathcal{E}_f)$; avec v_f (un des nœuds feuilles) $\in \mathbb{V} / fils[v_f] = \emptyset$;
- $\mathcal{E}_j \in \mathbb{E} / \forall j \in [1..t]$; $arrivée(\mathcal{E}) = départ(\mathcal{E}_j)$.

Remarque : nous attribuons le poids minimal aux relations générant un cycle (Cf. Figure IV.7.Exemple 3).

$$\mathcal{P}_{Adap}(\varepsilon) = Min(\mathcal{P}_{Adap}(\varepsilon"))/ \forall\ arrivée(\mathcal{E}) \in chm(arrivée(\mathcal{E}))\ et\ \mathcal{E}" \in \mathbb{E}.$$

Le calcul des poids d'adaptation est exécuté selon l'algorithme « PondérationAdaptation » (Cf. Annexe). Cet algorithme est basé essentiellement sur l'appel récursif de la fonction de calcul de coûts traduite par algorithme « CalculerCoûts » (Cf. Annexe).

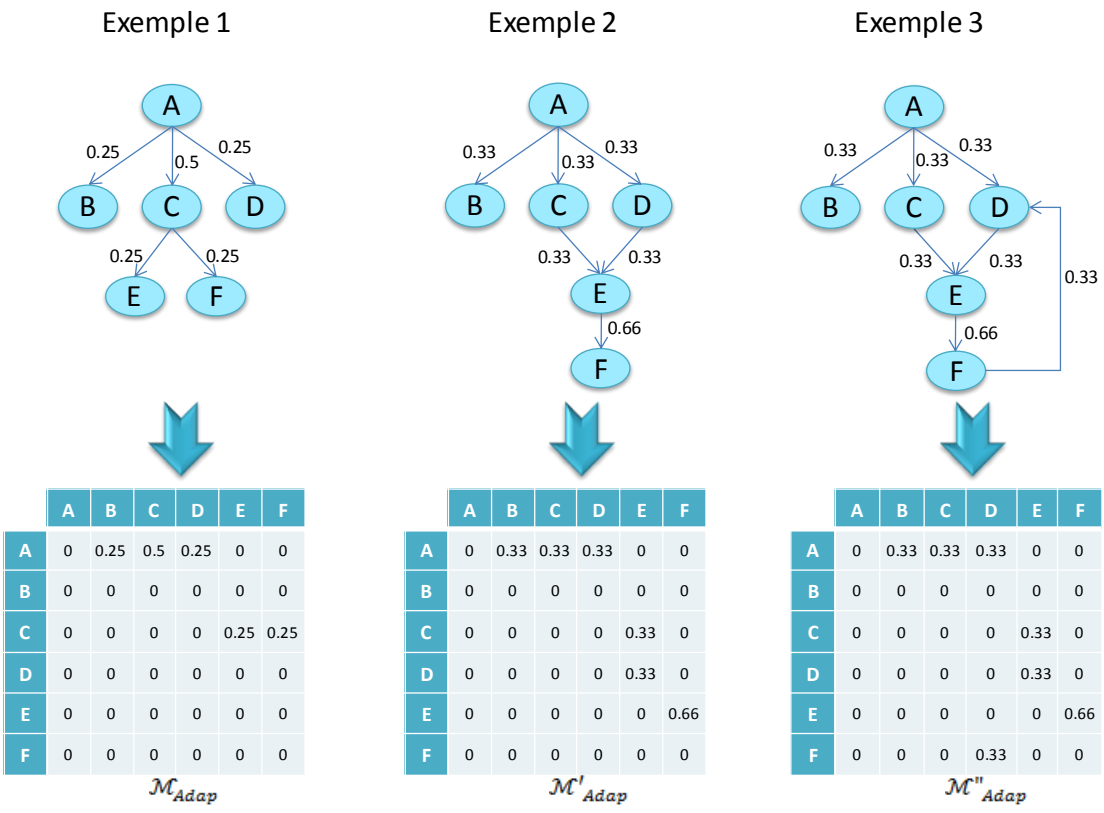
- Représentation matricielle

Selon le même principe que celui utilisé par la pondération structurelle, nous générons une représentation matricielle des différentes relations d'une vue. Chaque élément a_{ij} renseigne sur la pondération d'adaptation de la relation entre les deux nœuds i et j. Ainsi, la matrice $\mathcal{M}_{Adap}$ est une matrice représentative d'une vue telle que :

$$(a_{ij})_{\mathcal{M}_{Adap}} = \begin{cases} \mathcal{P}_{Adap}(\varepsilon), & si\ \exists\ \mathcal{E} \in \mathbb{E}\ /\ \mathcal{E}\ relie\ v_i\ et\ v_j\ \ avec\ v_i, v_j\ \in \mathbb{V}\ ; \\ 0, & sinon. \end{cases}$$

- Exemple

Dans la figure suivante, nous présentons la pondération d'adaptation des trois exemples de la Figure IV.7. Pour chaque exemple, nous présentons la matrice de représentation associée.



	A	B	C	D	E	F
A	0	0.25	0.5	0.25	0	0
B	0	0	0	0	0	0
C	0	0	0	0	0.25	0.25
D	0	0	0	0	0	0
E	0	0	0	0	0	0
F	0	0	0	0	0	0

	A	B	C	D	E	F
A	0	0.33	0.33	0.33	0	0
B	0	0	0	0	0	0
C	0	0	0	0	0.33	0
D	0	0	0	0	0.33	0
E	0	0	0	0	0	0.66
F	0	0	0	0	0	0

	A	B	C	D	E	F
A	0	0.33	0.33	0.33	0	0
B	0	0	0	0	0	0
C	0	0	0	0	0.33	0
D	0	0	0	0	0.33	0
E	0	0	0	0	0	0.66
F	0	0	0	0.33	0	0

Figure IV.7. Pondérations d'adaptation (exemples de vues présentées dans la Figure IV.4).

C. Pondération de représentativité

- Définition

La pondération par rapport à la représentativité consiste à attribuer des poids aux relations d'une vue générique en fonction des relations spécifiques qui lui sont rattachées par rapport au nombre de relations possibles rattachées à la vue générique en question.

- Technique utilisée et formalisation

Une vue générique évolue au fur et à mesure des rattachements de vues spécifiques. Certains sous-graphes apparaissent dans de nombreuses vues spécifiques alors que d'autres n'apparaissent que peu de vues spécifiques. L'idée ici est de favoriser les relations génériques les plus représentées au niveau spécifique. Ceci nous amène à calculer pour chaque relation générique le rapport entre le nombre de relations rattachées et le nombre de relations qui auraient dû être rattachées si toutes les vues spécifiques représentées par la vue générique en question étaient identiques.

La fonction $\mathcal{P}_{Rep}$ est une fonction qui permet de calculer la pondération de représentativité de chaque relation ($\mathcal{E}$).

$$\mathcal{P}_{Rep} : \mathbb{E} \rightarrow [0..1]$$

$$\varepsilon \mapsto \mathcal{P}_{Rep}(\varepsilon) = \frac{nbrRelationsExistantes}{nbrRelationsPossibles}$$

Où

- $nbrRelationsExistantes$: représente le nombre de relations spécifiques rattachées à la relation générique $\mathcal{E}$;
- $nbrRelationsPossibles$: représente le nombre total de relations spécifiques qui peuvent être rattachés à la relation générique $\mathcal{E}$. Ce nombre est égal au nombre de vues spécifiques rattachées à la vue générique traitée.

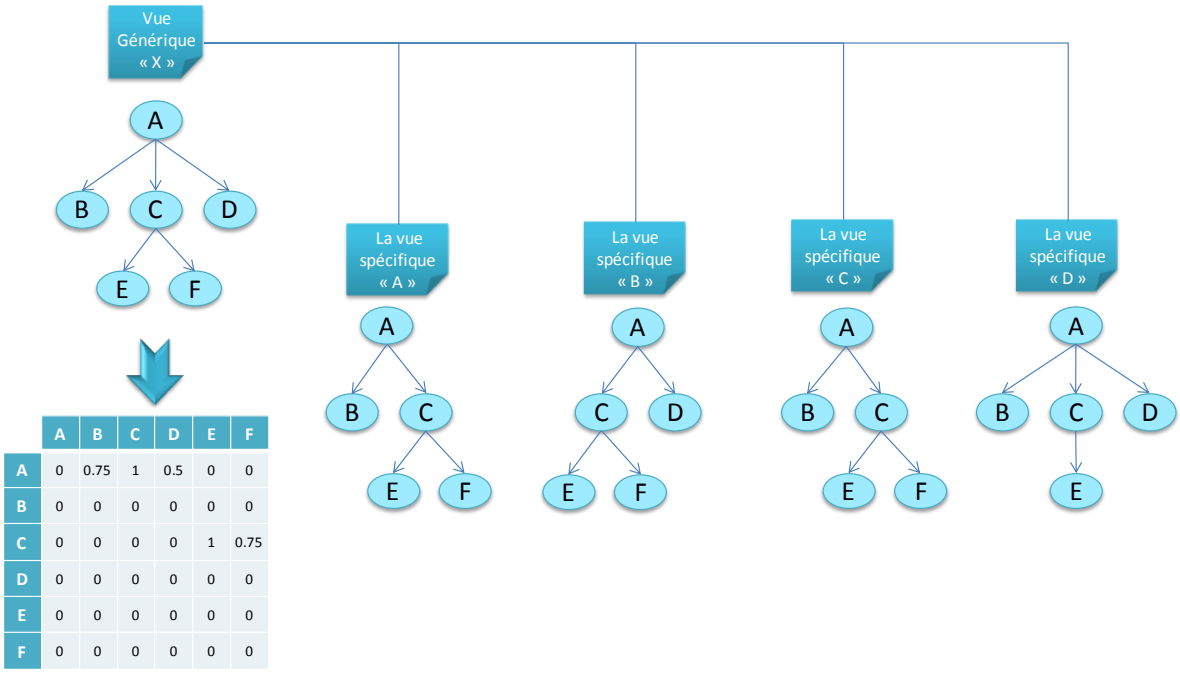
- Représentation matricielle

Enfin, les différents poids sont représentés au sein d'une matrice. Chaque élément a_{ij} de cette matrice renseigne sur la pondération de représentativité de la relation entre les deux nœuds i et j. Ainsi, la matrice $\mathcal{M}_{Rep}$ est une matrice représentative d'un vue telle que :

$$(a_{ij})_{\mathcal{M}_{Rep}} = \begin{cases} \mathcal{P}_{Rep}(\varepsilon), & si \ \exists \ \mathcal{E} \in \mathbb{E} \ / \ \mathcal{E} \ relie \ v_i \ et \ v_j \ \ avec \ v_i, v_j \ \in \mathbb{V} \ ; \\ 0, & sinon. \end{cases}$$

- Exemple

La Figure IV.8 représente un exemple de pondération de représentativité. Nous supposons que la vue générique de l'exemple (1) de la Figure IV.4 admet quatre vues spécifiques. Bien que cette vue générique représente l'ensemble de ces vues spécifiques, aucune de ces dernières n'est identique au représentant de la classe (vue générique). Par exemple, la relation entre les nœuds « A » et « D » n'est présente que dans les vues spécifiques « 1 » et « 2 », d'où son poids égal à 2/4=0,5. Si une relation apparaît dans toutes les vues, alors son poids sera de 1 (cas de la relation entre « C » et « E »).



	A	B	C	D	E	F
A	0	0.75	1	0.5	0	0
B	0	0	0	0	0	0
C	0	0	0	0	1	0.75
D	0	0	0	0	0	0
E	0	0	0	0	0	0
F	0	0	0	0	0	0

Figure IV.8. Pondération de représentativité de l'exemple (1) de la Figure IV.4.

D. Poids final

- Définition

Le poids final d'une relation est le poids qui va nous servir pour le calcul de la distance entre deux vues. Il est calculé à partir des trois pondérations $\mathcal{P}_{str}, \mathcal{P}_{Adap}$ et $\mathcal{P}_{Rep}$. La combinaison de ces trois pondérations par une multiplication permet de traduire l'influence de chacune d'entre elles sur le poids final. Ce poids tient compte donc, de la structure, du coût d'adaptation et de la représentativité de la vue générique.

- Technique utilisée et formalisation

La fonction $\mathcal{P}_f$ est une fonction déterminant le poids final de chaque relation ($\mathcal{E}$).

$\mathcal{P}_f : \mathbb{E} \rightarrow]0..1[$

$$\varepsilon \mapsto \mathcal{P}_f(\varepsilon) = \mathcal{P}_{Str}(\varepsilon) * \mathcal{P}_{Adap}(\varepsilon) * \mathcal{P}_{Rep}(\varepsilon)$$

- Représentation matricielle

La matrice $\mathcal{M}_f$ finale des poids est le résultat de la multiplication terme à terme des éléments des trois matrices $\mathcal{M}_{Str}$, $\mathcal{M}_{Adap}$ et $\mathcal{M}_{Rep}$. Ainsi, la matrice $\mathcal{M}_f$ est la matrice de représentation d'un vue telle que :

$$(a_{ij})_{\mathcal{M}_f} = (a_{ij})_{\mathcal{M}_{Str}} * (a_{ij})_{\mathcal{M}_{Adap}} * (a_{ij})_{\mathcal{M}_{Rep}}$$

❑ Exemple

La Figure IV.9 présente la matrice des poids finaux de l'exemple (1) de la Figure IV.4. Pour la pondération de représentativité, nous reprenons les valeurs de l'exemple de la Figure IV.8.

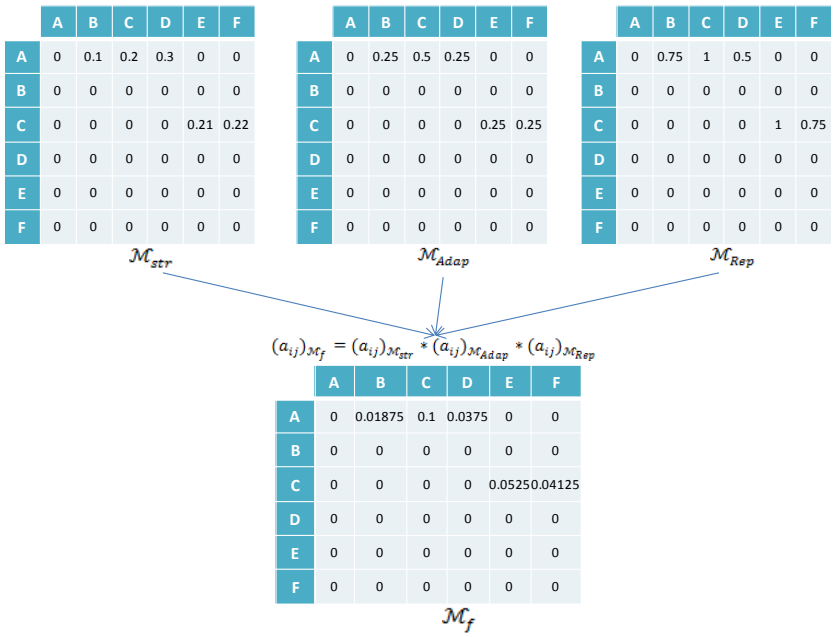


Figure IV.9. Calcul du poids final de l'exemple (1) de la Figure IV.4.

E. Conclusion

Nous avons présenté, dans cette section, une distance entre structures dont l'originalité réside dans la combinaison de trois pondérations : une pondération structurelle permettant de traduire l'organisation d'une vue (hiérarchie et ordre) ; une pondération d'adaptation permettant d'évaluer le coût de modification d'un nœud et une pondération de représentativité permettant de favoriser les relations les plus représentées. Une étude expérimentale doit être menée pour valider le choix de la combinaison des trois poids et l'influence de chacun d'entre eux sur le résultat de la classification.

II.2.2.3. Alignement de vues

❑ Définition

L'objectif est de faire évoluer artificiellement les deux vues à comparer afin de mesurer les transformations à effectuer sur chacune d'elles pour obtenir deux compositions identiques (si on voulait obtenir deux vues égales). Pour ceci, les adaptations effectuées consistent en l'ajout de nœuds vide à chacune de ces vues dans le but de les équilibrer. Cette adaptation est dite virtuelle du fait qu'aucune relation à partir ou vers les nœuds ajoutés n'est établie réellement.

Dans l'étape précédente, les vues pondérées sont représentées sous forme matricielle. Ainsi, l'ajout des nœuds vides se fait directement au niveau des matrices de représentation

en ajoutant une ligne et une colonne pour chaque nœud vide intégré. Les relations à partir et vers ces nœuds sont pondérés par un zéro, ce qui signifie qu'il ne s'agit pas de relations réelles.

- Technique utilisée et formalisation

Soient,

$\mathbb{G} = (\mathbb{V}, \mathbb{E})$ un graphe ordonné, orienté et étiqueté, représentant une vue générique, avec : $\mathbb{V} = \{v_1, \dots, v_n\}$ l'ensemble des nœuds de $\mathbb{G}$ et $\mathbb{E} = \{\mathcal{E}_1, \dots, \mathcal{E}_m\}$ l'ensemble des arcs de $\mathbb{G}$,

et,

$\mathbb{G}' = (\mathbb{V}', \mathbb{E}')$ un graphe ordonné, orienté et étiqueté, représentant une vue générique, avec : $\mathbb{V}' = \{v'_1, \dots, v'_n\}$ l'ensemble des nœuds de $\mathbb{G}'$ et $\mathbb{E}' = \{\mathcal{E}'_1, \dots, \mathcal{E}'_m\}$ l'ensemble des arcs de $\mathbb{G}'$,

Soit $alignNode$ la fonction d'alignement permettant d'associer un nœud d'une arborescence à un autre nœud d'une deuxième arborescence ayant la même étiquette.

$$\forall\, v \in \mathbb{V}\ et\ \forall\, v' \in \mathbb{V}' \cup \{v_0\} \Rightarrow \begin{cases} alignNode(v) = v', si\ seulement\ si\ etiq(v) = etiq(v') \\ alignNode(v) = v_0, sinon \end{cases}$$

$$\forall\, v' \in \mathbb{V}'\ et\ \forall\, v \in \mathbb{V} \cup \{v_0\} \Rightarrow \begin{cases} alignNode(v') = v, si\ seulement\ si\ etiq(v') = etiq(v) \\ alignNode(v') = v_0, sinon \end{cases}$$

$\forall\ v \in \mathbb{V}\ ou\ \mathbb{V}'$; Le poids de la relation qui lie les nœuds v et v_0 est égal à 0.

II.2.2.4. Mesure de similarité entre deux vues

- Définition

La distance entre deux vues permet de déterminer le degré de similarité entre ces deux vues. Ce degré de similarité sera utilisé tout au long des différentes phases de notre démarche de classification (Cf. Section II.2.3).

- Technique utilisée et formalisation

Pour mesurer la similarité entre deux vues, il est nécessaire préalablement de déterminer la distance d'alignement d'une relation d'une vue avec son image dans l'autre vue. La distance d'alignement entre $\mathcal{E}$ et $\mathcal{E}'$, $D_n(\mathcal{E}, \mathcal{E}')$, correspond à la valeur absolue de la différence des poids de $\mathcal{E}$ et $\mathcal{E}'$ (sachant que $\mathcal{E}'$ est l'alignement de $\mathcal{E}$).

Soit $alignRelation$ la fonction d'alignement permettant d'associer une relation d'un graphe à une relation d'un deuxième graphe dont les nœuds de départ et d'arrivée ont les mêmes étiquettes.

Soient $\mathcal{E} \in \mathbb{E}\ et\ alignRelation(\mathcal{E}) = \mathcal{E}'$,

$D_n(\mathcal{E}, \mathcal{E}') = \left|\mathcal{P}_f(\mathcal{E}) - \mathcal{P}_f(\mathcal{E}')\right|$,

$\mathcal{P}_f(\mathcal{E})$ est le poids final d'une relation $\mathcal{E}$ (Cf. Section II.2.2.2)

Le degré de similarité *Sim* est calculé en fonction de la distance d'alignement de tous les nœuds par rapport à l'ensemble des poids des deux vues.

$$sim(\mathbb{G}, \mathbb{G}') = 1 - \frac{\sum D_n(\varepsilon, \varepsilon')}{\sum \mathcal{P}_f(\varepsilon) + \sum \mathcal{P}_f(\epsilon')}$$

II.2.2.5. Exemple

Dans la Figure IV.10, nous présentons deux exemples de calcul de degré de similarité à partir des trois structures S_1, S_2 et S_3. S_1 est considérée comme la vue générique. Les deux autres structures S_2 et S_3 présentent des différences sur un seul nœud respectivement à la racine et sur une feuille. Chacune de ces deux structures est comparée à la première. $\mathcal{M}_f(S_1)$, $\mathcal{M}_f(S_2)$ et $\mathcal{M}_f(S_3)$ sont les matrices de poids final, ensuite nous adaptons ces matrices deux à deux : $\mathcal{M}_f(S_1)$ par rapport à $\mathcal{M}_f(S_2)$, et $\mathcal{M}_f(S_2)$ par rapport à $\mathcal{M}_f(S_1)$ afin de calculer la distance entre S_1 et S_2 ; $\mathcal{M}_f(S_1)$ par rapport à $\mathcal{M}_f(S_3)$ et $\mathcal{M}_f(S_3)$ par rapport à $\mathcal{M}_f(S_1)$ afin de calculer la distance entre S_1 et S_3. Le résultat des calculs est le suivant : $sim(S_1, S_2)$ = 0,603 et $sim(S_1, S_3)$ = 0,824.

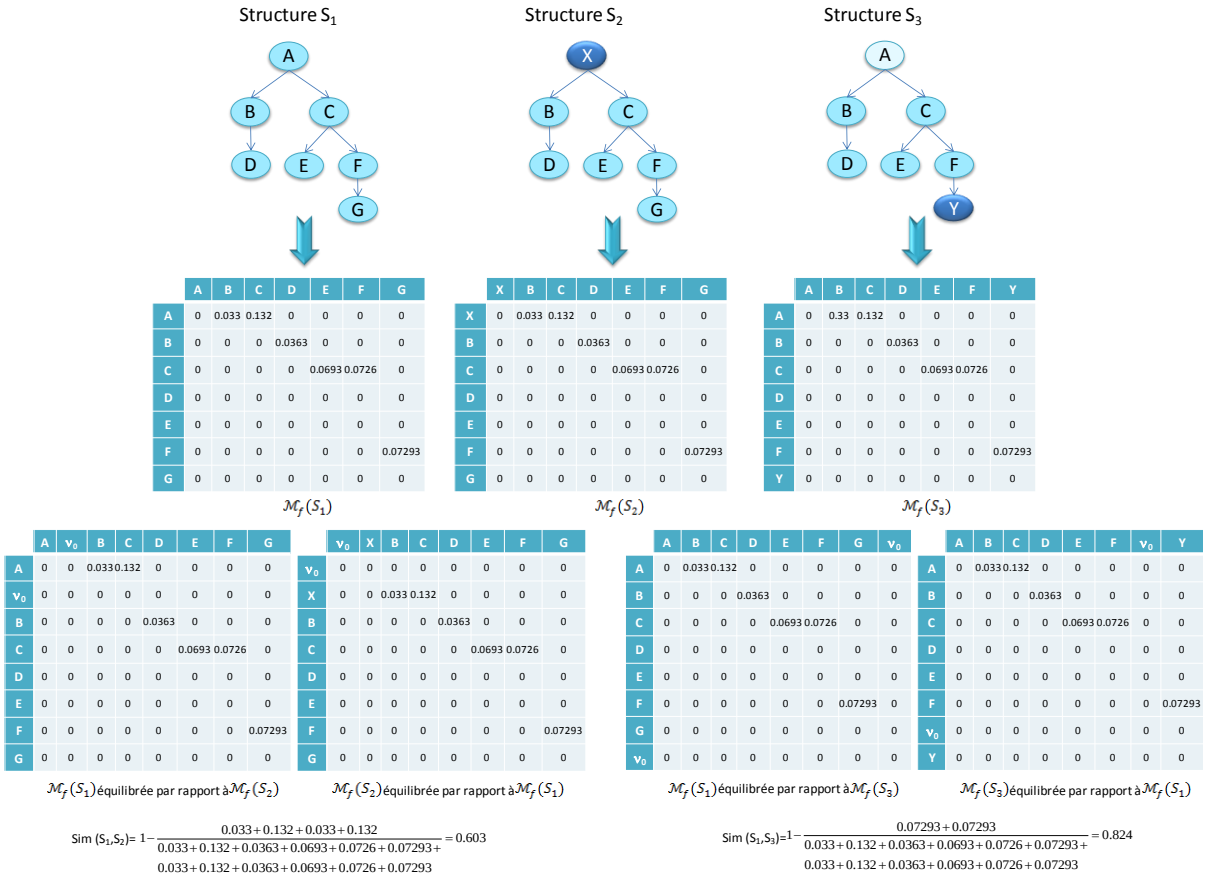


Figure IV.10. Deux exemples de calcul de similarité avec comparaison des résultats.

$sim(S_1, S_3)$ est plus élevée que $sim(S_1, S_2)$. Ceci montre que le calcul pénalise, comme nous le souhaitions, les différences détectées entre les nœuds situés dans la partie

haute des arborescences qui induisent des coûts d'adaptation élevés.. La question corollaire sera celle du réglage du seuil de similarité.

II.2.3. Démarche globale de classification

La démarche de classification repose sur deux phases : la première phase consiste à rattacher la nouvelle vue spécifique à une vue générique et la deuxième phase consiste à conserver de la représentativité des classes suite aux modifications apportées à leur représentant (vues génériques). Dans ce qui suit, nous détaillons les deux phases de notre démarche de classification.

II.2.4. Agrégation d'individus : affectation des vues aux classes

L'objectif de cette première étape est d'agréger l'individu intégré dans la classe la plus proche. La Figure IV.11 présente la démarche globale d'agrégation d'un individu. Afin de réduire le nombre de comparaisons, nous sélectionnons parmi l'ensemble des vues génériques, un sous-ensemble de vues. Dans un second temps, nous effectuons des comparaisons entre le représentant générique de la vue spécifique et les vues génériques sélectionnées. A l'issue de cette phase, nous retenons la vue générique la plus proche. Si son degré de similarité est supérieur à un seuil d'agrégation $S_{Agrég}$, la vue spécifique sera rattachée à la vue générique sélectionnée.

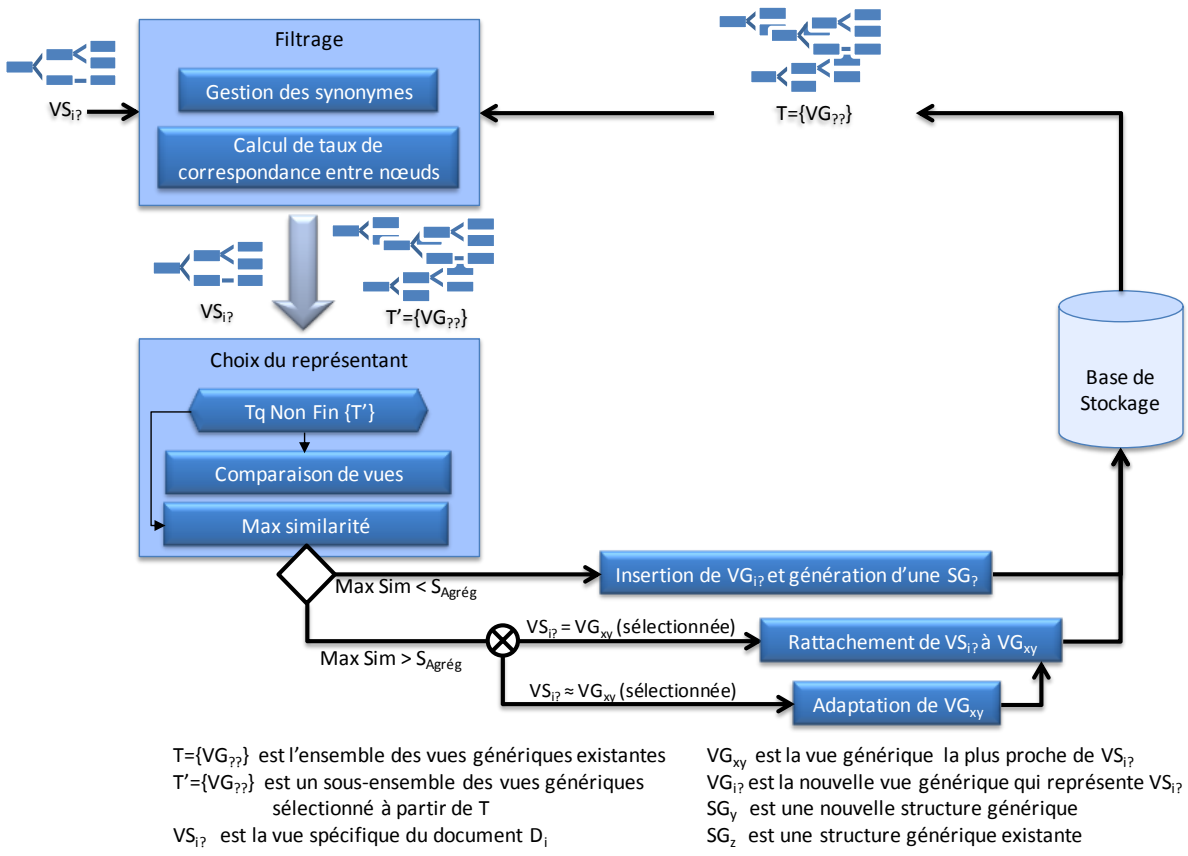


Figure IV.11. Démarche d'agrégation d'un individu.

II.2.4.1. Filtrage

Le filtrage consiste à sélectionner l'ensemble des vues génériques de la base auxquelles la vue spécifique du document à intégrer est susceptible d'être rattachée. Cette sélection est basée sur le calcul du taux de correspondance entre la vue du document à intégrer et chacune des vues génériques de la base. Les vues génériques pour lesquelles le taux de correspondance est supérieur à un seuil S_f (déterminé par expérimentation) sont sélectionnées pour les étapes suivantes de la classification.

Le calcul du taux de correspondance entre nœuds s'appuie sur une fonction d'alignement qui associe à chacun des nœuds d'une vue un nœud d'une seconde vue possédant la même étiquette ou une étiquette jugée « équivalente ». Nous proposons d'unifier les étiquettes des nœuds via la gestion des synonymes.

A. Gestion des synonymes

Dans certains cas, des documents ont la même vue spécifique, mais la dénomination de certains nœuds diffère. Par exemple, nous avons deux transcriptions de deux documents audio qui admettent deux vues spécifiques quasiment identiques à un nœud près : dans la première nous trouvons un nœud « Thème » et dans la seconde un nœud « Sujet ». Ces deux nœuds admettent des noms d'étiquettes dits synonymes. La gestion des synonymes permet de lever cette ambigüité et par conséquent d'optimiser notre démarche de classification.

Ainsi, nous définissons la fonction $syn()$ qui recherche tous les synonymes possibles d'un nœud, en utilisant soit des dictionnaires de synonymie (Miller 1995), soit des ontologies de domaines (Hernandez 2005).

Deux étiquettes de deux nœuds v et v' sont considérées équivalentes s'elles vérifient la contrainte suivante :

$$\forall\, v\ et\ v' \in \mathbb{V}\,;\, etiq(v) \Leftrightarrow etiq(v')\ si\ et\ seulement\ si$$
$$syn\big(etiq(v)\big) \cap syn\big(etiq(v')\big) \neq \emptyset$$

Dans le cas ou les deux étiquettes sont considérées comme synonymes, l'étiquettes de la vue générique représentative du document à intégrer est remplacée par celle de la vue générique.

B. Calcul du taux de correspondances entre nœuds

- Définition

Le calcul du taux de correspondance permet d'évaluer le pourcentage de nœuds communs entre les deux vues comparées. Ainsi, il serait possible de déterminer l'ensemble des vues génériques les plus approchantes de la vue spécifique du document.

- Technique utilisée et formalisation

Pour réaliser cette étape, nous définissons une « fonction d'alignement ». Cette fonction associe à chacun des nœuds d'une vue un nœud d'une seconde vue possédant la même étiquette lorsque ce nœud existe, ou le nœud vide dans le cas contraire.

Au sein d'une vue, chaque nœud possède une étiquette unique et deux nœuds différents ne peuvent posséder la même étiquette. Cette unicité est garantie par l'analyse lexicale de la phase d'extraction. Cette contrainte implique qu'un nœud d'une vue ne peut être aligné qu'à un seul nœud d'une autre vue ou au nœud vide.

Soient,

$\mathbb{G} = (\mathbb{V}, \mathbb{E})$ un graphe ordonné, orienté et étiqueté, représentant une vue générique, avec : $\mathbb{V} = \{v_1, \dots, v_n\}$ l'ensemble des nœuds de $\mathbb{G}$ et $\mathbb{E} = \{\mathcal{E}_1, \dots, \mathcal{E}_m\}$ l'ensemble des arcs de $\mathbb{G}$,

et,

$\mathbb{G}' = (\mathbb{V}', \mathbb{E}')$ un graphe ordonné, orienté et étiqueté, représentant une vue générique, avec : $\mathbb{V}' = \{v'_1, \dots, v'_n\}$ l'ensemble des nœuds de $\mathbb{G}'$ et $\mathbb{E}' = \{\mathcal{E}'_1, \dots, \mathcal{E}'_m\}$ l'ensemble des arcs de $\mathbb{G}'$.

Soit $alignNode$ la fonction d'alignement permettant d'associer un nœud d'une arborescence à un autre nœud d'une deuxième arborescence ayant la même étiquette.

$$\forall\, v \in \mathbb{V}\ et\ \ \forall\, v' \in \mathbb{V}' \cup \{v_0\} \Rightarrow \begin{cases} alignNode(v) = v', si\ seulement\ si\ etiq(v) = etiq(v') \\ alignNode(v) = v_{0\,}, sinon \end{cases}$$

$$\forall\, v' \in \mathbb{V}'\ et\ \ \forall\, v \in \mathbb{V} \cup \{v_0\} \Rightarrow \begin{cases} alignNode(v') = v, si\ seulement\ si\ etiq(v') = etiq(v) \\ alignNode(v') = v_{0\,}, sinon \end{cases}$$

Soient $NC_m(\mathbb{G})$ l'ensemble des nœuds de $\mathbb{G}$ qui ont une image non nulle ($v' \neq v_0$) dans $\mathbb{G}'$ et $ND_m(\mathbb{G})$ l'ensemble des nœuds de $\mathbb{G}$ qui sont mis en relation avec le nœud vide $v_{0\,}$ (selon le même principe, nous définissons $NC_m(\mathbb{G}')$ et $ND_m(\mathbb{G}')$).

$NC_m(\mathbb{G}) = \{v \in \mathbb{V} / alignNode(v) \neq v_0\}$; $NC_m(\mathbb{G}') = \{v' \in \mathbb{V}' / alignNode(v') \neq v_0\}$

$ND_m(\mathbb{G}) = \{v \in \mathbb{V} / alignNode(v) = v_0\}$; $ND_m(\mathbb{G}') = \{v' \in \mathbb{V}' / alignNode(v') = v_0\}$

Ces ensembles vérifient les égalités suivantes :

$$\mathbb{V} = NC_m(\mathbb{G}) \cup ND_m(\mathbb{G})$$

$$\mathbb{V}' = NC_m(\mathbb{G}') \cup ND_m(\mathbb{G}')$$

La sélection des vues génériques de la base, à comparer à la vue spécifique du document à intégré, est déterminée grâce à un taux de correspondance T_c :

$$T_c = \left(\frac{card(NC_m(\mathbb{G}))}{card(\mathbb{V})} + \frac{card(NC_m(\mathbb{G}'))}{card(\mathbb{V}')} \right) / 2$$

Ce taux évalue le pourcentage de nœuds communs entre les deux vues comparées. En effet, ce taux est composé de deux quotients : le premier quotient calcule le pourcentage de nœuds communs par rapport à la première vue générique (extraite de la vue spécifique du document) et le second détermine le pourcentage de nœuds communs par rapport à la deuxième vue générique (une des vues génériques stockées).

II.2.4.2. Choix du représentant

- Définition

Cette étape consiste à reprendre les vues génériques sélectionnées dans l'étape précédente afin d'en choisir éventuellement une : la plus approchante de la vue extraite du document.

- Principe

En se basant sur les degrés de similarité calculés (selon la méthode présentée dans la section II.2.2), le système retient la vue générique qui admet le degré de similarité le plus élevé, puis compare ce degré de similarité à un seuil d'agrégation $S_{Agrég}$ (fixé par expérimentations) en dessous duquel les vues seront jugées trop différentes. Selon cette comparaison, la décision d'adapter une vue générique existante ou d'intégrer la nouvelle vue générique (représentative de la vue spécifique du document) est prise. Si le degré de similarité est strictement inférieur à $S_{Agrég}$, la nouvelle vue générique sera intégrée dans la base. Dans le cas contraire et si $S_{Agrég}$ inférieur à 1, la fusion des deux vues est exigée. Dans ce cas, la vue générique issue de la base doit être adaptée afin de pouvoir représenter la nouvelle vue spécifique (celle du nouveau document).

- Exemple

La Figure IV.12 présente une illustration du choix d'une vue générique. Après le calcul de distances entre chaque vue générique de la base et la vue spécifique du document à intégrer, le système élimine les vues génériques jugées trop différentes (cas de la vue générique « G »). Ensuite, il sélectionne la vue générique la plus représentative ; dans cet exemple, la vue générique « A » sera choisie.

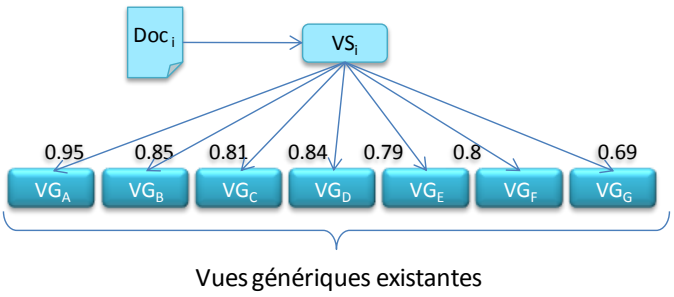


Figure IV.12. Exemple de sélection d'une vue générique.

II.2.5. Conservation de la représentativité des classes

- Objectif

Cette étape à pour objectif de garder une dispersion minimale au sein de chaque classe. Cette dispersion est susceptible d'être élargie suite à l'adaptation du représentant d'une classe (une vue générique). Une grande dispersion traduit le regroupement d'individus éloignés, hétérogénéité dans une même classe et par conséquent des individus mal placés.

- Principe

Lors de l'adaptation du représentant d'une classe, la conservation d'une inertie qui dépasse le seuil S_{Disp} est nécessaire. Cette inertie dite intra classe se traduit par l'écart type des degrés de similarité de tous les individus par rapport à leur représentant. En d'autres termes, cette inertie permet de s'assurer qu'après les adaptations qu'elle vient de subir, la vue générique adaptée reste représentative de toutes les vues spécifiques qui y sont rattachées. L'adaptation d'une vue générique est effectuée suite au rattachement d'un individu à une classe. Cet individu peut être soit une nouvelle vue spécifique intégrée soit, une vue spécifique reclassée.

- Démarche

La phase de conservation de la représentativité des classes (Cf. Figure IV.13) se déclenche uniquement lors de l'adaptation d'un représentant d'une classe. Dans un premier temps il faut calculer l'inertie de la classe adaptée. Afin de réaliser cette étape, il faut calculer de nouveau les degrés de similarité entre la vue générique adaptée et l'ensemble des vues spécifiques des documents qu'elle représente. Ensuite, il faut calculer l'écart type de ces degrés de similarité noté σ afin d'évaluer leur dispersion. Le calcul de dispersion revient au calcul de l'inertie intra classe. Cette dispersion ne doit pas dépasser pas un seuil S_{Disp}. Dans le cas où la dispersion est plus grande que ce seuil, on désagrège les individus les plus distants un par un jusqu'à une dispersion au dessous du seuil S_{Disp}.

Les individus désagrégés seront rattachés à de nouvelles classes. Au niveau de cette étape, nous reprenons la démarche de choix d'un représentant, présentée lors de la phase d'agrégation d'individus. Nous rappelons ici que pour chaque individu à classer, il faut comparer sa vue spécifique avec les vues génériques existantes. La vue générique la plus similaire est retenue. A ce niveau, trois cas se présentent (Cf. Figure IV.13) :

- la similarité maximale est inférieure au seuil d'agrégation, dans ce cas une nouvelle vue générique est crée à partir de la vue spécifique (l'individu) à reclasser ;
- la similarité maximale est égale à 1. Ceci signifie que l'individu a trouvé une classe qui le représente exactement. Dans ce cas, le système rattache cette vue spécifique (l'individu) à la vue générique sélectionnée ;
- la similarité maximale est supérieure au seuil d'agrégation, mais inférieure à 1. Dans ce cas, il faut rattacher cette vue spécifique (l'individu) à la vue générique sélectionnée après l'avoir adaptée. Cette adaptation déclenche de nouveau la phase de conservation de représentativité des classes.

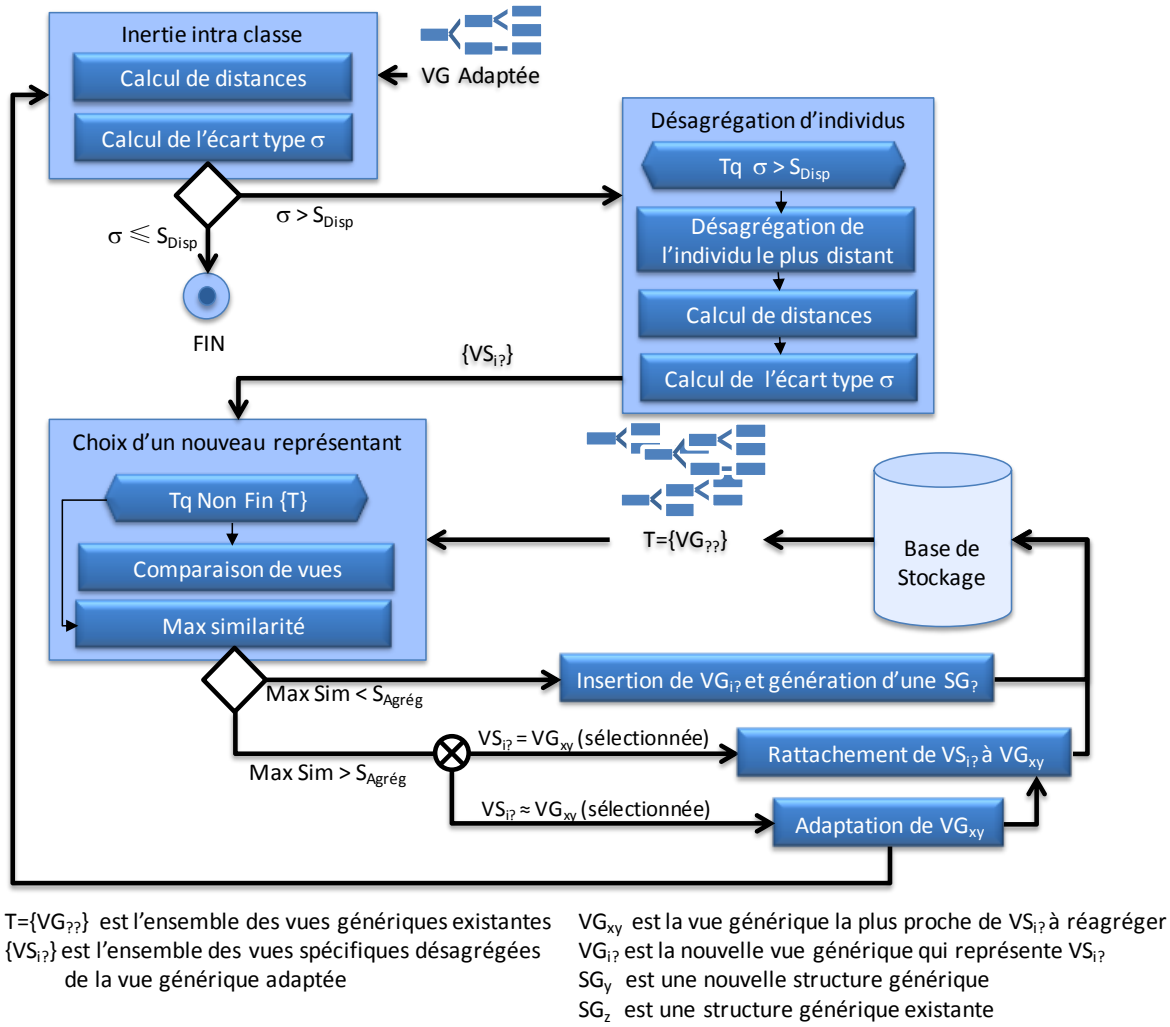


Figure IV.13. Démarche de conservation de représentativité des classes.

❑ Exemple

Dans la Figure IV.14, nous montrons l'impact de l'adaptation d'une vue générique suite à l'ajout d'une nouvelle vue spécifique « VS_{A5} ». Cette adaptation a augmenté les distances entre le représentant de la classe (la vue générique) « VG_A » et les différentes vues spécifiques rattachées. En particulier VS_{A1} s'éloigne avec un degré de similarité égal à 0,69. Ceci va se répercuter sur l'écart type qui devient égal à 0,098. Si nous considérons que la dispersion ne doit pas dépasser 0,06 (c'est à dire $S_{Disp} = 0{,}06$), cette vue générique doit être perturbé. L'individu VS_{A1} jugé le plus distant est désagrégé de cette classe. Dans cet exemple, nous présentons deux cas possibles :

- dans le premier cas, le degré de similarité entre VS_{A1} et VG_B (vue générique jugée la plus similaire) est supérieur au seuil $S_{Agrég}$. VS_{A1} est alors rattachée à VG_B et par conséquent le représentant de cette classe est adapté. Cette adaptation, déclenche le calcul de l'inertie intra classe. Le nouvel écart type est égal à 0,058 inférieur à S_{Disp}. Ceci vérifie la condition d'arrêt ;

- dans le deuxième cas, le degré de similarité entre VS_{A1} et VG_B (vue générique jugée la plus similaire) est inférieur au seuil $S_{Agrég}$. Une nouvelle vue générique VG_N est générée afin de représenter l'individu désagrégé VS_{A1}.

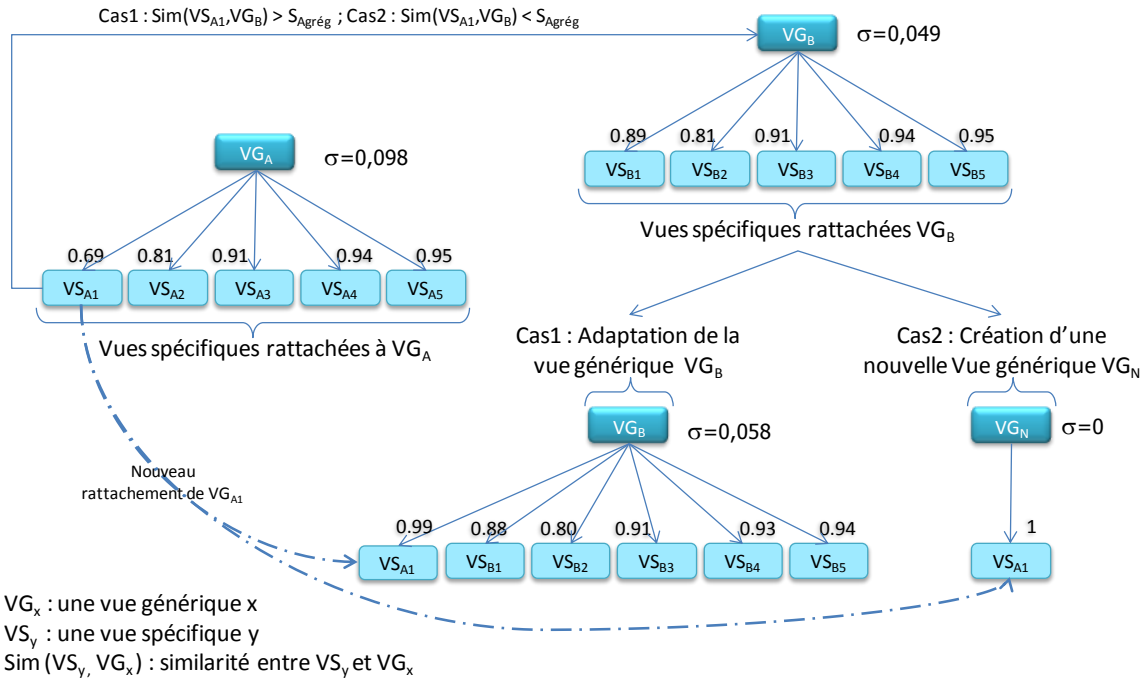


Figure IV.14. Exemple de conservation de la représentativité de classes.

III. Recherche et restitution de documents

Dans la section précédente, nous avons présenté les techniques permettant l'intégration des documents multistructurés issus de sources disséminées et hétérogènes. Nous souhaitons maintenant proposer des techniques permettant de manipuler aussi bien la ou les structures que le contenu des documents intégrés (Djemal 2007b). A cette fin, nous proposons deux techniques complémentaires :

- la recherche de documents : consiste à retrouver de données factuelles et des éléments répondant à un ou plusieurs critères relatifs à une ou plusieurs structures de documents ;
- l'analyse multidimensionnelle : consiste à analyser les informations documentaires de la base selon des axes d'analyse (dimensions) et un sujet (fait) non prédéfinies.

Par la suite, nous illustrons nos propos à partir de l'exemple de structure générique présentée dans Figure IV.15.

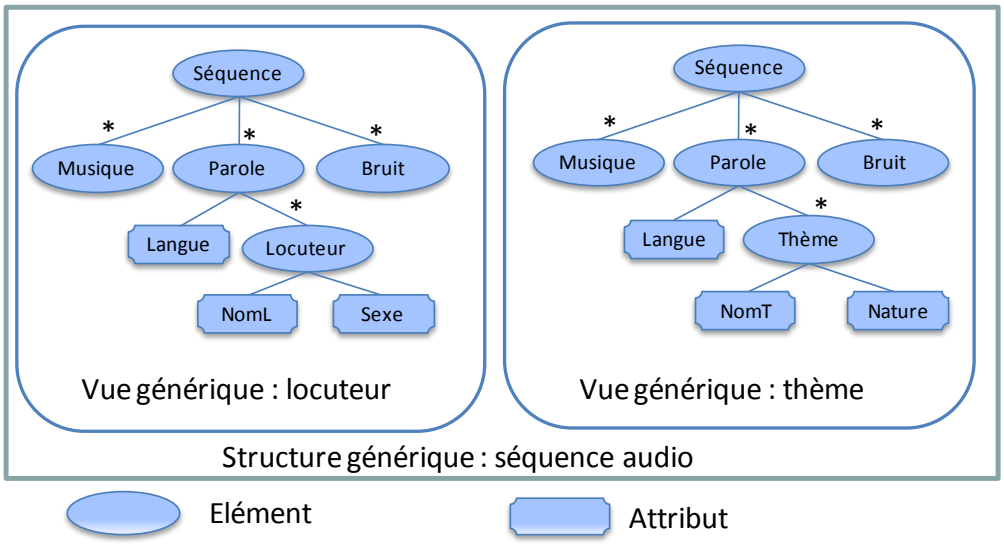


Figure IV.15. Exemple de vues génériques associées à une même structure générique.

III.1. Recherche de documents multistructurés

Nous décrivons, dans cette section, les techniques de recherche de documents ou de fragments de documents dans une collection de documents multistructurés. Cette recherche consiste à retrouver des fragments de documents en se basant sur les différentes structures. Selon le modèle « MVDM », ces fragments sont représentés au travers des nœuds spécifiques. La structure générique est le point de départ facilitant l'accès aux nœuds spécifiques. Afin d'affiner le résultat, des contraintes peuvent être définies sur les nœuds génériques.

III.1.1. Démarche de recherche de documents multistructurés

Afin de rechercher de documents ou des fragments de documents en se basant sur les différentes vues relatives à une structure générique, nous proposons une démarche d'interrogation basée d'une part sur la génération de requêtes SQL et d'autre part sur l'ajustement des résultats de ces requêtes via une gestion des chevauchements.

La recherche de fragments de documents nécessite la construction de requêtes complexes. La complexité de ces requêtes est double. D'une part, elle est due aux calculs de profondeurs de chacun des nœuds utilisés dans la requête. D'autre part, elle est due aux jointures nécessaires pour remonter dans l'arborescence. Ainsi, afin de déterminer le nombre de jointures à utiliser dans sa requête, le système doit chercher à chaque fois le niveau de chaque nœud dans l'arborescence et la vue générique utilisée pour remonter dans cette arborescence.

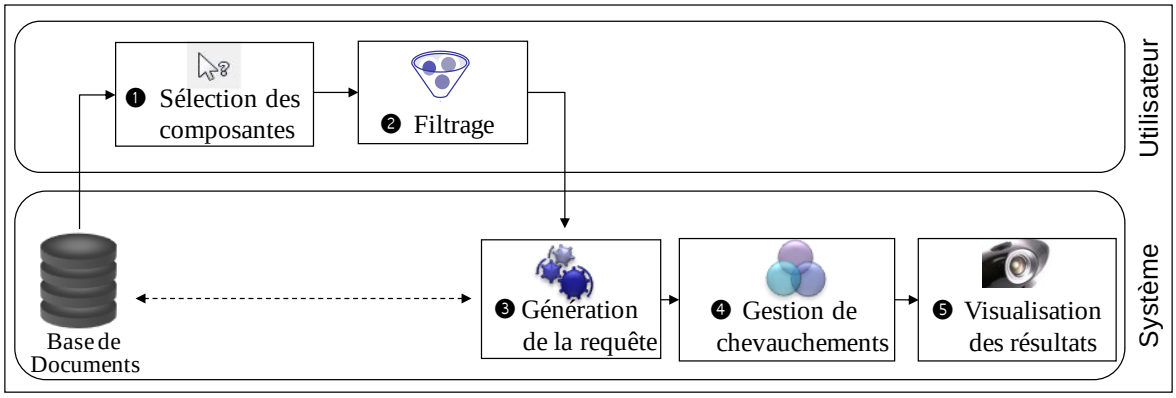


Figure IV.16. Démarche de recherche de documents multistructurés.

Le processus proposé pour rechercher de fragments de documents multistructurés peut-être schématisé comme l'indique la Figure IV.16. Ce processus se déroule en cinq phases :

❶ Sélections des composants : la première étape doit permettre à l'utilisateur de choisir les composantes de sa requête. Ces composantes consistent en un ensemble de nœuds qui appartient à différentes vues relatives à une même structure générique.

❷ Filtrage : la deuxième étape, celle de filtrage, doit permettre à l'utilisateur de spécifier des valeurs précises afin de composer sa requête. Nous proposons deux types de filtre. Lorsque la valeur de la composante sélectionnée est sous forme numérique, nous proposons un filtrage qui permet de fixer des critères de sélection, en utilisant des opérateurs classiques de comparaison arithmétique (<, >, =, <>, <=, >=). Pour les valeurs textuelles, nous proposons d'utiliser les techniques de filtrage par mots-clés éventuellement liés par les opérateurs logiques (+ : et, - : pas, | : ou) ;

❸ Génération de la requête : une fois que les composantes de la requête sont sélectionnées et les conditions de filtrages sont spécifiées, le système génère la requête en question. Il calcule le niveau de chaque nœud dans l'arborescence et il détermine les jointures nécessaires ;

❹ Gestion de chevauchements : cette étape a pour objectif d'ajuster des résultats à présenter à l'utilisateur. Cet ajustement est possible lorsqu'il existe deux nœuds définis sur un même contenu. En effet, le système sauvegarde au préalable une liste « NœGénChe » pour chaque structure générique. Cette liste représente la liste des nœuds génériques représentant des nœuds spécifiques dont le contenu est susceptible d'être chevauché. Le calcul de chevauchement s'effectue entre tous les nœuds de type élément qui appartiennent à une même ligne de la vue « Jointure » et qui figurent, eux ou un de leurs nœuds descendants, sur la liste « NœGénChe ». Les nœuds de type attribut seront pris en compte également dans le calcul de chevauchement. Ils seront représentés par leur nœud père.

❺ Visualisation des résultats : la dernière étape, celle de visualisation, consiste à afficher à l'utilisateur le résultat de sa requête. Ces résultats sont des données factuelles et/ou des éléments.

III.1.2. Exemple

La démarche de recherche est illustrée ici au travers de la requête suivante : « rechercher tout les segments de paroles où « *Julien Courbet* » aborde les thèmes d'« *économie* ». ».

Pour spécifier sa requête, l'utilisateur doit sélectionner les nœuds « Parole », « NomL » et « NomT » en se basant sur la structure générique « Séquence Audio » (Cf. Figure IV.15). Ensuite, il doit définir ces règles de filtrage ; dans notre exemple « NomL = *'Julien Courbet'* » et « NomT = *'économie'* ». A partir de ces informations, le système génère la requête suivante :

```
SELECT n1.sondoc.numdoc, ref(n1), ref(n2), ref(n3)
From noeudspe n1, noeudspe n2, noeudspe n3, relationspe r1, relationspe
r2, relationspe r3, relationspe r4
WHERE n1.herite.nomng='Parole'
AND n2.herite.nomng='NomT' AND n2.Valeur = 'économie'
AND n3.herite.nomng='NomL' AND n3.Valeur = 'Julien Courbet'
AND n1.sondoc.appartient.nomsg='Sequence Audio'
AND n1.numns = r1.noeudpere.numns AND n1.numns = r2.noeudpere.numns
AND r1.noeudfils.numns= r3.noeudpere.numns
AND r2.noeudfils.numns= r4.noeudpere.numns
AND n2.numns = r2.noeudfils.numns AND n3.numns = r4.noeudfils.numns
AND r1.herite.savuegen.nomvg='Locuteur'
AND r2.herite.savuegen.nomvg='Thème'
AND r3.herite.savuegen.nomvg='Locuteur'
AND r4.herite.savuegen.nomvg='Thème';
```

La gestion de chevauchement s'applique sur les résultats de cette requête. Dans la Figure IV.17, nous présentons un extrait des résultats de la requête avant et après la gestion de chevauchement. Nous montrons également le calcul de ce chevauchement pour la première ligne de la table résultat. Nous remarquons ainsi que le fragment parole qui répond à la requête se trouve exactement dans le segment audio qui débute à 89 secondes et finit à la 120 secondes (au lieu de 67_134).

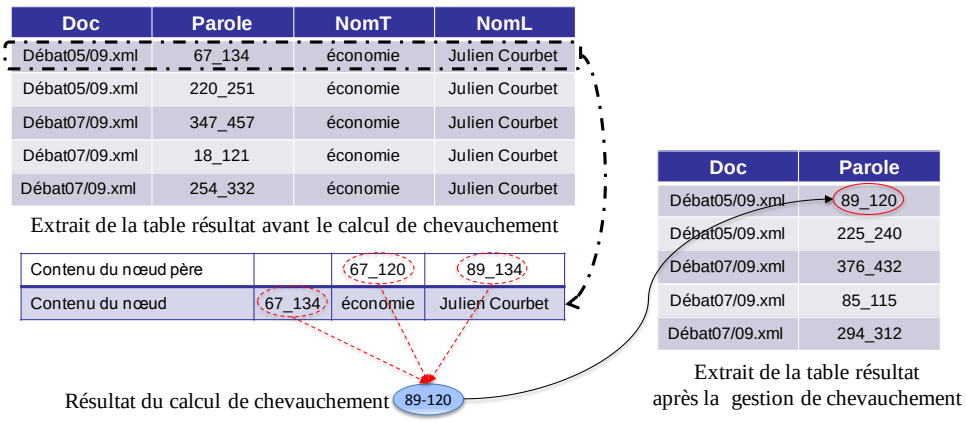


Figure IV.17. Calcul du chevauchement dans le cas d'une recherche de fragments de documents.

III.2. Restitution multidimensionnelle

Nous décrivons, dans cette section, la technique d'analyse multidimensionnelle que nous souhaitons appliquer aux informations documentaires intégrées dans une base de documents organisée selon le modèle « MVDM ». Cette technique d'analyse multidimensionnelle consiste en une structuration des données selon plusieurs axes d'analyse pouvant représenter des notions variées. Ces données peuvent être, selon le modèle proposé, des éléments spécifiques ou des attributs (Djemal et al. 2009a) (Djemal et al. 2010a).

Les approches d'analyse multidimensionnelle existantes se basent sur la structure du document. Dans cette finalité, (Khrouf et Soulé-Dupuy 2004) définissent une structure générique qui représente des documents structurellement similaires. En se focalisant sur les documents XML, (Golfarelli et al. 2001) et (Pokorny 2001) se basent sur la DTD de ces documents lorsque (Tseng et Chou 2006) utilisent le schéma XML. Toutes ces approches s'appuient sur une seule structure du document. Or, en ce qui nous concerne, un document peut avoir plusieurs structures.

Afin d'analyser d'une manière multidimensionnelle le contenu de la base de documents, une première opération consiste à construire et alimenter des magasins. Il s'agit d'un extrait d'informations organisé de manière adéquate à des fins décisionnelles. Les données extraites sont alors adaptées à un usage particulier.

Dans le cadre de nos travaux, nous avons adopté les tables multidimensionnelles afin de visualiser le contenu des magasins générés (Gyssens et Lakshmanan 1997) puisque la représentation sous forme de tableau est la vision la plus simple et la plus intuitive à laquelle les utilisateurs sont habitués.

La démarche proposée pour restituer les informations contenues dans la base d'une manière multidimensionnelle, peut-être schématisée comme l'indique la Figure IV.18. Ce processus se compose de trois phases :

1) construction des schémas des magasins : cette phase nécessite l'intervention de l'utilisateur pour préciser le sujet (fait) et les axes d'analyse (dimensions) ;
2) génération des magasins de documents : au cours de cette phase, le magasin doit être généré de manière automatique et transparente vis-à-vis de l'utilisateur. Ainsi, il est nécessaire d'accéder à la base afin de récupérer les valeurs et instancier les magasins ;
3) visualisation des tables multidimensionnelles : une fois le magasin construit, cette phase permet de visualiser automatiquement son contenu selon une table multidimensionnelle.

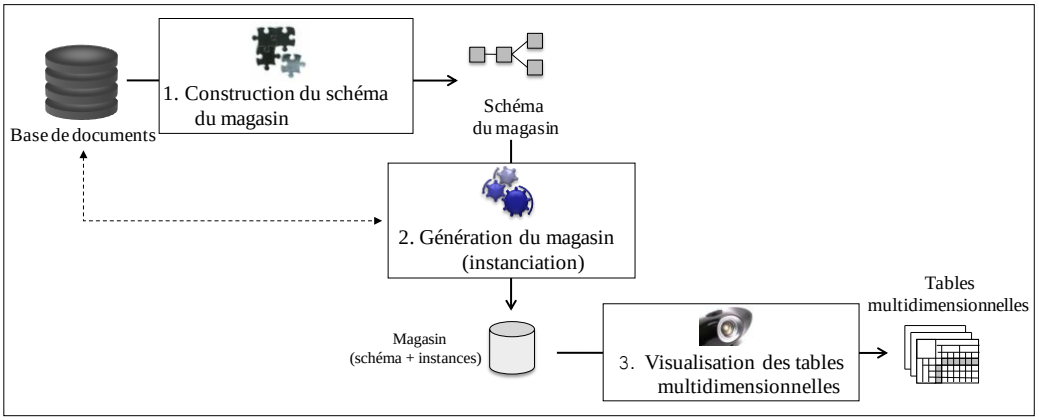


Figure IV.18. Démarche d'analyse multidimensionnelle.

III.2.1. Démarche de construction des schémas des magasins

La première phase du processus d'analyse multidimensionnelle consiste à générer, à partir de la base, le schéma du magasin de documents désiré.

La construction des schémas des magasins (Cf. Figure IV.19) se compose de quatre étapes, à savoir : ❶ choix du type d'analyse, ❷ sélection des composants d'analyse (Fait/Dimension), ❸ filtrage et ❹ visualisation du schéma du magasin.

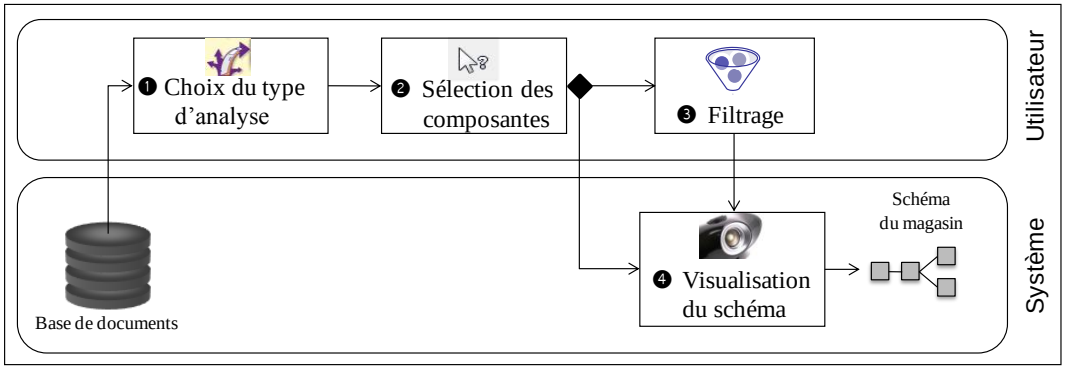


Figure IV.19. Démarche de construction des schémas des magasins.

❶ La première étape doit permettre à l'utilisateur de choisir un type d'analyse. En se basant sur le modèle « MVDM », cinq types d'analyses sont envisageables :

- analyse par *structure générique* : cette analyse s'applique à un ensemble de documents rattachés à une même structure générique. Dans ce cas, les composantes d'analyse seront choisies indépendamment des vues associées,
- analyse par *vue générique* : cette analyse admet le même principe que la précédente. Cependant, les analyses se focalisent sur une seule vue associée à une structure générique d'un ensemble de documents,
- analyse par *nœuds génériques* : les deux premières propositions peuvent être, dans certains cas, restrictives puisqu'il est possible qu'un nœud générique soit utilisé par plusieurs structures génériques. Cette proposition consiste alors à analyser les

documents par nœuds génériques pouvant appartenir à plusieurs structures génériques,

- analyse par *structure spécifique* : ce quatrième type consiste à analyser le contenu d'un et un seul document en se basant sur sa structure spécifique. A cette fin, il est nécessaire de se référer à sa structure générique afin de déterminer son schéma,
- analyse par *vue spécifique* : ce dernier type d'analyse consiste à analyser le contenu d'un document en se focalisant sur une et une seule vue spécifique.

Ces différents types d'analyse permettront à l'utilisateur de se focaliser sur une ou plusieurs structures, sur un domaine bien défini ou même sur un document, selon ses besoins ;

❷ Au cours de la deuxième étape, l'utilisateur doit sélectionner les composants d'analyse, à savoir un fait (sujet d'analyse) et ses dimensions (axes d'analyse). L'utilisateur doit indiquer également l'ordre des dimensions et la fonction d'agrégation pour la mesure (indicateur d'analyse) du fait (Compte, Somme, Maximum, Minimum, Moyenne, contenu). Dans le cas d'une analyse par nœuds génériques, la sélection de composants se fait au travers des listes, puisque ce type d'analyse nécessite l'utilisation de plusieurs arborescences. Pour les autres types d'analyse, la sélection de composants se fait à partir de la structure générique ou de la vue générique choisie au préalable par l'utilisateur ;

❸ La troisième étape, celle de filtrage, doit permettre à l'utilisateur de spécifier des valeurs précises sur les dimensions et le fait afin d'affiner ses analyses. Nous distinguons deux types de filtrage :

- pour une dimension, l'utilisateur doit choisir, parmi toutes ses valeurs, celles qu'il veut intégrer dans le magasin,

- pour le fait, nous proposons deux types de filtre. Lorsque la valeur du fait est sous forme numérique, nous proposons un filtrage qui permet de fixer des critères de sélection, en utilisant des opérateurs classiques de comparaison arithmétique (<, >, =, <>, <=, >=). Pour les valeurs textuelles, nous proposons d'utiliser les techniques de filtrage par mots-clés éventuellement liés par les opérateurs logiques (+ : et, - : pas, | : ou) ;

❹ La dernière étape, celle de visualisation, consiste à afficher à l'utilisateur le schéma du magasin de documents selon une représentation graphique qui illustre les choix d'analyse avant de générer les magasins (base des tables multidimensionnelles).

III.2.2. Démarche de génération des magasins de documents

Cette phase consiste à générer le magasin d'une manière *automatique* afin de récupérer les informations de la base. Cette génération se déroule en deux étapes (Cf. Figure IV.20) à savoir :

❶ génération d'une vue pour chaque composant d'analyse (que ce soit une dimension ou un fait) ;

❷ jointure et regroupement des différentes vues générées lors de la première étape.

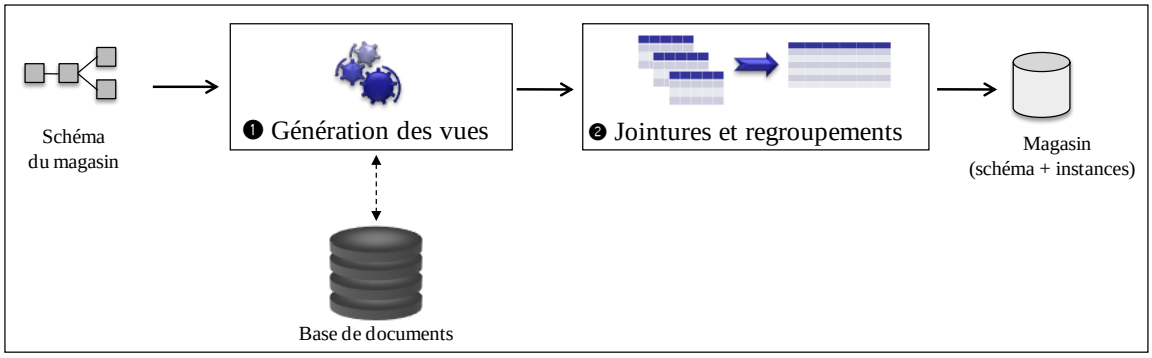


Figure IV.20. Démarche de génération de magasins.

III.2.2.1. Génération d'une vue pour chaque composant d'analyse

Pour chaque dimension, nous devons générer une vue. Les vues que nous manipulons dans cette section sont les vues utilisées dans les bases de données et non pas les vues modélisées dans le modèle « MVDM ». Une vue en base de données représente une synthèse d'une requête d'interrogation de la base. Ainsi, elle peut être considérée comme une table virtuelle définie par une requête.

Le nom de la vue créée aura la forme suivante : « Dim_n » ; où n désigne le numéro d'ordre de la dimension. Cette vue englobe également un bloc d'attributs « Anc_x » et un champ « Nœud ». Le nombre d'attributs « Anc_x » dépend du nombre de dimensions qui seront utilisées pour effectuer l'analyse multidimensionnelle. Ces attributs correspondent aux numéros (identifiants) des premiers ancêtres communs entre le paramètre en cours et chacun des autres paramètres. Le dernier champ « Nœud » de la vue correspond aux nœuds spécifiques qui héritent du nœud générique sélectionné comme paramètre d'analyse.

Pour le nœud jouant le rôle du fait, le système doit générer une vue dont le nom est « Fact », englobant aussi un bloc d'attributs, « Anc_x » et un champ « Nœud ».

D'autres contraintes doivent être ajoutées aux vues générées :

- si le type d'analyse est par vue générique (resp. structure générique), tous les nœuds spécifiques doivent appartenir aux vues spécifiques (resp. structures spécifiques) des documents rattachés à la vue générique (resp. structure générique) choisie ;

- si le type d'analyse est par vue spécifique (resp. structure spécifique), tous les nœuds spécifiques doivent appartenir à la vue spécifique (resp. la structure spécifique) du document choisi par l'utilisateur ;

- si l'utilisateur exige des valeurs spécifiques pour une dimension ou pour un fait (opération de filtrage), des conditions doivent être ajoutées pour ne prendre en compte que ces valeurs.

Dans le cas d'une analyse par nœud générique, il faut déterminer dans un premier temps les structures génériques contenant tous les nœuds fixés comme composants d'analyse. Pour chaque structure générique déterminée, il faut générer une vue selon l'approche que nous venons de détailler. Enfin, une opération d'union entre les vues générées sera établie.

La vue d'une dimension admet la forme générique suivante :

```
CREATE VIEW Dim_i (Doc, Anc_1, {Anc_2, Anc_3}, nœud) AS
SELECT  n.Doc, n.Anc_1, {n.Anc_2, n.Anc_3}, n
FROM    V1Dim_i n
{UNION
…
UNION
SELECT  n.Doc, n.Anc_1, {n.Anc_2, n.Anc_3}, n
FROM    VmDim_i n};
```

III.2.2.2. Jointure et regroupement des différentes vues générées

Cette étape consiste à joindre et regrouper toutes les vues générées lors de l'étape précédente. Ainsi, une nouvelle vue est établie par une jointure sur les attributs « Anc_x » de toutes les vues créées. Nous rappelons qu'à ce niveau, nous manipulons encore que des nœuds spécifiques.

La nouvelle vue aura alors la forme suivante :

```
CREATE VIEW Jointure (nœud_d1, {nœud_d2, nœud_d3}, nœud_f) AS
SELECT  d1.nœud, {d2.nœud, d3.nœud}, f.nœud
FROM    Dim_1 d1, {Dim_2 d2, Dim_3 d3}, Fact f
WHERE   d1.Anc_1 = f.Anc_1     {AND    d1.Anc_2 = d2.Anc_2}
{AND    d1.Anc_3 = d3.Anc_2}   {AND    d2.Anc_1 = f.Anc_2}
{AND    d2.Anc_3 = d3.Anc_3}   {AND    d3.Anc_1 = f.Anc_3};
```

La définition de plusieurs structures sur un même contenu introduit un ou des chevauchements d'éléments. Dans un premier temps, nous consultons la liste des nœuds génériques « NœGénChe » pouvant impliquer des chevauchements. Cette liste est construite durant l'intégration d'une structure générique. Pour chaque ligne de la vue « Jointure », si les nœuds figurent sur la liste « NœGénChe », nous procédons aux calculs de chevauchement dans le but de déterminer la partie du contenu commune. Ce processus est appliqué aux nœuds de la liste « NœGénChe », ainsi que leurs descendants. Seul les nœuds de type élément sont concernés par ce calcul de chevauchement. Si des nœuds de type attribut sont choisis, le processus est appliqué à leurs éléments pères.

Soit la table « JointureT » une table résultat qui contient les valeurs des nœuds de type attribut et des index sur le contenu des nœuds de type élément. Une fois la table « JointureT » créée, il est nécessaire d'effectuer une opération de regroupement afin d'appliquer la fonction d'agrégation choisie par l'utilisateur tout en prenant en compte la fonction de filtrage imposée sur le fait. Cette dernière vue générée représente le contenu du magasin. Elle aura la forme suivante :

```
CREATE VIEW Vue (j.nœud_d1, {j.nœud_d2, j.nœud_d3}, j.nœud_f) AS
SELECT          j.nœud_d1, {j.nœud_d2, j.nœud_d3,} Fonction(j.nœud_f)
FROM            JointureT j
GROUP BY        j.nœud_d1 {, j.nœud_d2, j.nœud_d3}
{HAVING Fonction(j.nœud_f) = X}
{HAVING Fonction(j.nœud_f) > X}
{HAVING Fonction(j.nœud_f) < X}
{HAVING Fonction(j.nœud_f) <> X}
{...};
```

III.2.3. Démarche de visualisation des tables multidimensionnelles

Une fois que le magasin de documents a été généré, la dernière phase, celle de visualisation, sera déclenchée. Elle consiste à afficher le contenu de la dernière vue générée par le système sous forme de table multidimensionnelle assez simple à manipuler et à interpréter. En effet, ces tables permettent de mieux apprécier le contenu des magasins de documents. Elles organisent les données en les classant suivant les dimensions choisies par l'utilisateur. Ainsi, les colonnes représentent la première dimension, les lignes représentent la deuxième dimension et les plans représentent la troisième dimension. Les valeurs des mesures des faits sont représentées à l'intérieur des tables sous forme d'interrelation entre les différentes valeurs des dimensions.

Etant donné que chaque plan de la table multidimensionnelle correspond à une seule valeur de la troisième dimension, le passage de la dernière vue générée par le système en une table multidimensionnelle se fait par génération de vues en effectuant une sélection sur chacune des valeurs de la troisième dimension. Chacune des nouvelles vues contient trois colonnes : (1) la première dimension, (2) la deuxième dimension et (3) le fait.

A partir de chacune de ces vues, le système doit :

- récupérer toutes les valeurs possibles de la première dimension. Ces valeurs seront affichées dans les colonnes du plan correspondant ;

- récupérer toutes les valeurs possibles de la deuxième dimension. Ces valeurs seront affichées dans les lignes du plan correspondant ;

- restituer pour chaque couple (une colonne i et une ligne j) la mesure correspondante à partir de la troisième colonne de la vue (le fait). Cette mesure sera affichée dans la case correspondante (intersection entre i et j).

III.2.4. Exemple

Afin d'illustrer cette démarche de restitution multidimensionnelle des documents multistructurés, nous reprenons l'exemple de la Figure IV.15. Cet exemple présente une structure générique basée sur deux vues génériques : « locuteur » et « thème ». A partir de ces vues, nous choisissons trois dimensions : « NomL », « NomT » et « Langue » ; et un fait : « Parole ».

Pour la première dimension « NomL », le système doit générer la vue suivante :

```
CREATE VIEW Dim_1 ("Doc", "Anc_1", "Anc_2", "Anc_3", "NomL") AS
SELECT n.sondoc.numdoc, r1.noeudpere.numns, r1.noeudpere.numns,
r1.noeudpere.numns, ref(n)
From noeudspe n, relationspe r1, relationspe r2
WHERE n.herite.nomng='NomL'
AND n.sondoc.appartient.nomsg='séquence_audio'
AND n.numns=r2.noeudfils.numns
AND r2.herite.savuegen.nomvg='locuteur'
AND r2.noeudpere.numns= r1.noeudfils.numns
AND r1.herite.savuegen.nomvg='Locuteur';
```

Pour la deuxième dimension « NomT », le système doit générer la vue suivante :

```
CREATE VIEW Dim_2 ("Doc", "Anc_1", "Anc_2", "Anc_3", "NomT") AS
SELECT n.sondoc.numdoc, r1.noeudpere.numns, r1.noeudpere.numns,
r1.noeudpere.numns, ref(n)
From noeudspe n, relationspe r1, relationspe r2
WHERE n.herite.nomng='NomT'
AND n.sondoc.appartient.nomsg='séquence_audio'
AND n.numns=r2.noeudfils.numns
AND r2.herite.savuegen.nomvg='Thème'
AND r2.noeudpere.numns= r1.noeudfils.numns
AND r1.herite.savuegen.nomvg='Thème';
```

Pour la troisième dimension « Langue », le système doit générer la vue suivante :

```
CREATE VIEW Dim_3 ("Doc", "Anc_1", "Anc_2", "Anc_3", "Langue") AS
SELECT n.sondoc.numdoc, r1.noeudpere.numns, r1.noeudpere.numns,
r1.noeudpere.numns, ref(n)
From noeudspe n, relationspe r1
WHERE n.herite.nomng='Langue'
AND n.sondoc.appartient.nomsg='séquence_audio'
AND n.numns=r1.noeudfils.numns AND r1.herite.savuegen.nomvg='Thème';
```

Pour le fait « Parole », le système doit générer la vue suivante :

```
CREATE VIEW Fact ("Doc", "Anc_1", "Anc_2", "Anc_3", "Parole") AS
SELECT n.sondoc.numdoc, n.numns, n.numns, n.numns, ref(n)
From noeudspe n
WHERE n.herite.nomng='Parole'
AND n.sondoc.appartient.nomsg='séquence_audio'
AND n.numns=r1.noeudfils.numns AND r1.herite.savuegen.nomvg='Thème';
```

Une fois ces quatre vues créées, le système génère une nouvelle vue « Jointure » :

```
CREATE VIEW Jointure ("NomL", "NomT", "Langue", "Parole") AS
SELECT Dim_1.NomL, Dim_2.NomT, Dim_3.Langue, Fact.Parole
From Dim_1, Dim_2, Dim_3, Fact
WHERE   Dim_1.Anc_1 = Dim_2.Anc_1      AND Dim_1.Anc_2 = Dim_3.Anc_1
AND     Dim1.Anc_3 = Fact.Anc_1        AND Dim_2.Anc_2 = Dim_3.Anc_2
AND     Dim_2.Anc_3 = Fact.Anc_2       AND Dim_3.Anc_3 = Fact.Anc_3;
```

Les nœuds « Thème » et « Locuteur » sont marqués sur la liste « NœGénChe ». Le calcul de chevauchent concerne donc ces nœuds ainsi que leurs nœuds parents. Dans notre exemple, le nœud concerné est « Parole ». Les nœuds « NomL », « NomT » et « Langue » sont de type attribut ; ils sont pris en compte dans le calcul de chevauchement au travers de leur nœud père. Par conséquent, la valeur finale du fait est le résultat de l'intersection du contenu des trois nœuds « Thème », « Locuteur » et « Parole ». Dans la Figure IV.21, nous présentons le calcul de chevauchement effectué sur la première ligne de la vue « jointure ». Le contenu étant une séquence audio, il sera traduit par des marques de début et de fin exprimées en seconde.

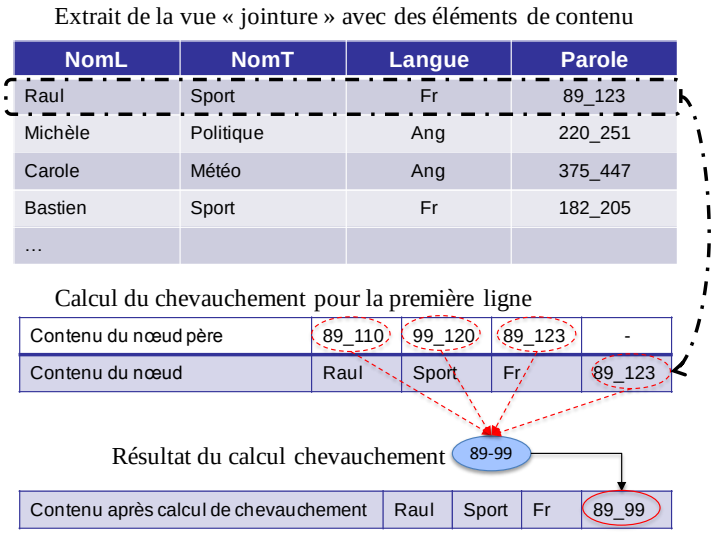


Figure IV.21. Gestion du chevauchement dans le cas d'une analyse multidimensionnelle.

Après la gestion du chevauchement, une nouvelle table « JointureT » est créée. Une opération de regroupement est effectuée sur cette table afin d'appliquer la fonction d'agrégation. Ainsi, une nouvelle vue qui représente le contenu du magasin est établie. A partir de cette dernière vue, le système génère une table multidimensionnelle telle que présentée dans la Figure IV.22.

Cette démarche nous permet plus de précision dans la localisation de fragments pertinents. Cette précision est assurée d'une part par l'ajout de nouveaux paramètres d'analyse et d'autre part, par la gestion de chevauchement entre les nœuds définis sur un même contenu. Par exemple, si nous ne considérons qu'une seule structure au niveau de l'exemple présenté dans la section précédente, nous aurons alors moins de nœuds qui structurent le contenu et par conséquent moins de paramètres d'analyse. Dans ce cas, les analyses possibles ne peuvent plus intégrer des paramètres tenant compte des thèmes et des locuteurs en même temps. La gestion du chevauchement permet d'ajuster le résultat suivant le contenu des deux nœuds qui se chevauchent. Dans l'exemple de la Figure IV.21, nous montrons comment la mesure du premier fait passe de (89_123) à (89_99).

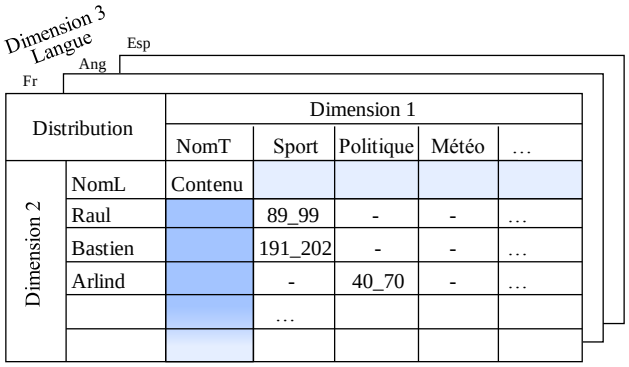


Figure IV.22. Tables multidimensionnelles.

IV. Conclusion

Notre approche de gestion de la multistructuralité repose sur une méthodologie de modélisation présentée dans le chapitre précédent. L'exploitation d'une telle modélisation nécessite la définition de règles d'instanciation du modèle et de techniques d'exploitation spécifiques.

Dans un premier temps, nous avons abordé les problématiques liées à l'instanciation du modèle « MVDM ». Cette instanciation repose sur une démarche d'intégration de documents multistructurés. Ce processus comprend deux phases importantes :

(1) l'instanciation du niveau spécifique à partir de la dématérialisation du document. Cette dématérialisation est basée sur l'extraction de la structure et du contenu du nouveau document ;

(2) l'instanciation du niveau générique basée sur une démarche de classification. Cette démarche est assurée via le rattachement de sa vue spécifique à la vue générique adéquate (vue générique existante la plus proche, éventuellement adaptée, ou nouvelle vue générique).

La phase d'extraction permet de déterminer la vue spécifique d'un nouveau document. L'originalité de cette phase réside dans la définition de méthodes d'extraction des structures indépendamment de leur typologie et la spécification d'une fonction d'alignement et la vérification de l'unicité des étiquettes des différentes structures extraites. Cette contrainte assure, lors de l'application de la fonction d'alignement, qu'un nœud d'une vue ne peut être aligné qu'à un seul nœud d'une autre vue, ou au nœud vide.

L'originalité de la phase de classification réside dans :

- l'exploitation des paramètres structurels comme critère de ressemblance de documents. La similarité entre deux documents est évaluée à partir de la distance qui sépare l'organisation de leurs vues spécifiques ;
- la spécification d'un représentant pour chaque classe. Ce représentant est structuré sous forme de graphe afin de couvrir le maximum d'arborescences structurellement « proches ». Un tel représentant permet de faciliter l'accès et par conséquent la restitution des différents documents regroupés au sein d'une même classe ;
- la conservation de la représentativité des classes qui permet de reclasser des vues spécifiques jugées mal classées soit parce que leurs classes d'origines ont subi plusieurs adaptations, soit parce que lors de l'intégration de cette vue spécifique la nouvelle classe n'était pas encore créée.

La similarité entre deux vues est calculée à partir d'une distance structurelle dont l'originalité réside dans les différents paramètres qu'elle intègre. Cette « distance structurelle » repose sur la combinaison de plusieurs pondérations, dont certaines sont issues ou adaptées à partir de la littérature sur le sujet :

- une pondération structurelle permettant de traduire l'organisation structurelle (hiérarchie et ordre) ;

- une pondération permettant d'évaluer le coût d'adaptation : le coût de modification (ajout ou suppression) d'un nœud est d'autant plus élevé que celui-ci a de nœuds dépendants ;
- une pondération par rapport à la représentativité d'une relation.

Nous avons proposé deux démarches de restitutions documentaires (recherche et analyse multidimensionnelle). La multistructuralité, dans ces cas, offre à l'utilisateur des résultats plus pertinents et plus précis, grâce à :

- l'ajout de nouveaux paramètres issus des structures multiples définies sur le document ;
- la gestion de chevauchement entre les nœuds définis sur un même contenu. Ceci nous permet de mieux localiser l'information pertinente.

La démarche de recherche de documents que nous avons proposée est basée sur l'intégration de techniques d'interrogations et de recherche d'informations garantissant l'exploitation d'une ou plusieurs structures du document et de son contenu conjointement.

L'approche que nous avons choisie pour effectuer des analyses multidimensionnelles sur des informations issues des documents se base sur les structures. A partir de ces structures, les nœuds et le contenu sont transposés en sujet d'analyse (fait) et en axes d'analyse (dimensions). Les nœuds choisis peuvent appartenir à une seule structure du document comme ils peuvent dépendre de plusieurs structures du même document.

L'ensemble des propositions décrites dans ce chapitre ont été mises en œuvre dans un prototype de gestion de documents multistructurés. Ce prototype va nous permettre de tester, de valider et de mesurer les performances de nos approches de modélisation et de classification.

V. Bibliographie

Bertino, E., Guerrini, G., Mesiti, M., Rivara, I., et Tavella, C. (2002). "Measuring the Structural Similarity among XML Documents and DTDs". http://www.disi.unig.

Breiman, L. (1998). Classification and Regression Trees. Chapman & Hall/CRC.

Cobena, G., Abiteboul, S., et Marian, A. (2002a). XyDiff, tools for detecting changes in XML documents.

Cobena, G., Abiteboul, S., et Marian, A. (2002b). "Detecting Changes in XML Documents". Proceedings of international conference on data engineering, IEEE Computer Society Press; 1998, 41-52.

Costa, G., Manco, G., Ortale, R., et Tagarelli, A. (2004). "A Tree-Based Approach to Clustering XML Documents by Structure". Lecture notes in computer science, 137-148.

Dalamagas, T., Cheng, T., Winkel, K. J., et Sellis, T. (2006). "A methodology for clustering XML documents by structure". Information Systems, 31(3), 187-228.

Denoyer, L., et Gallinari, P. (2004). "Bayesian network model for semi-structured document classification". Information Processing and Management, 40(5), 807-827.

Diligenti, M., Gori, M., Maggini, M., et Scarselli, F. (2001). "Classification of HTML documents by Hidden Tree-Markov Models". Proceedings of ICDAR, 849–853.

Djemal, K. (2007b). « Vers une exploitation de documents multi-structurés ». Congrès de l'INFormatique des Organisations et Systèmes d'Information et de Décision (INFORSID'07), Perros-Guirec, 37-52.

Djemal, K., Soulé-Dupuy, C., et Vallès-Parlangeau, N. (2009a). « Analyse multidimensionnelle des documents multistructurés ». Colloque Veille Stratégique Scientifique et Technologique (VSST 2009), Nancy.

Djemal, K., Soulé-Dupuy, C., et Vallès-Parlangeau, N. (2010a). "Multistructured documents: from modelling to multidimensional analyses". SciWatch Journal.

Djemal, K., Soulé-Dupuy, C., et Vallès-Parlangeau, N. (2010b). "Classification de documents : calcul d'une distance structurelle". Extraction et la Gestion des Connaissances EGC2010, Hammamet, Tunisie, Du 26/01/2010-29/01/2010, 609-614.

Doucet, A., et Ahonen-Myka, H. (2002). "Naive Clustering of a Large XML Document Collection". Proceedings of the First Workshop of the Initiative for the Evaluation of XML Retrieval (INEX), 81–87.

Golfarelli, M., Rizzi, S., et Vrdoljak, B. (2001). "Data warehouse design from XML sources". Proceedings of the 4th ACM international workshop on Data warehousing and OLAP, ACM Press New York, NY, USA, 40-47.

Gyssens, M., et Lakshmanan, L. V. S. (1997). "A Foundation for Multi-Dimensional Databases". Proceedings of the international conference on very large data bases, Institute of electrical & electronics engineers (IEEE), 106-115.

Hernandez, N. (2005). « Ontologies pour l'aide à l'exploration d'une collection de documents ». Ingénierie des Systèmes d'Information, 10(1), 11–31.

Jiang, T., Wang, L., et Zhang, K. (1995). "Alignment of trees—an alternative to tree edit". Theoretical Computer Science, 143(1), 137-148.

Joachims, T. (1997). Text Categorization with Support Vector Machines: Learning with Many Relevant Features. Springer.

Khrouf, K., et Soulé-Dupuy, C. (2004). "A Textual Warehouse Approach: A Web Data Repository". Intelligent Agents for Data Mining and Information Retrieval, Idea Group Publishing, 101-124.

Khrouf, K. (2004). « Entrepôts de documents : De l'alimentation à l'exploitation ». Thèse de doctorat, Université Paul Sabatier.

Kutty, S., Tran, T., Nayak, R., et Li, Y. (2008). "Clustering XML Documents Using Closed Frequent Subtrees: A Structural Similarity Approach". Lecture Notes In Computer Science, 183-194.

MacQueen, J. B. (1966). "Some Methods for Classification and Analysis of Multivariate Observations." Western Management Science Inst Univ of CALIFORNIA LOS ANGELES, 281-297.

Mbarki, M. (2008). « Gestion de l'hétérogénéité documentaire : le cas d'un Entrepôt de documents multimédia ». Thèse de doctorat, Université Paul Sabatier.

Miller, G. A. (1995). "WordNet: a lexical database for English". Communications of the ACM, 38(11), 39-41.

Piwowarski, B., Denoyer, L., et Gallinari, P. (2002). « Un modèle pour la recherche d'information sur des documents structurés ». Actes des 6 èmes Journées internationales d'Analyse statistique de Données Textuelles (JADT'02), 605–616.

Pokorny, J. (2001). "Modelling stars using XML". Proceedings of the 4th ACM international workshop on Data warehousing and OLAP, ACM Press New York, NY, USA, 24-31.

Romany, L., et Bonhomme, P. (2000). "Parallel Alignment of Structured Documents". Parallel Text Processing, Kluwer Academic Publishers, Dordrecht, Boston, London, 201-218.

Saleem, K. (2008). "Schema matching and integration in large scale snario". Université Montpellier II - Sciences et Techniques du Languedoc.

Schütze, H., Hull, D. A., et Pedersen, J. O. (1995). "A comparison of classifiers and document representations for the routing problem". Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval, ACM New York, NY, USA, 229-237.

Shasha, D., et Zhang, K. (1997). "Approximate Tree Pattern Matching". Pattern Matching Algorithms, Oxford University Press, 341–371.

Soulé-Dupuy, C. (2001). « Bases d'informations textuelles : des modèles aux applications ». Habilitation à diriger des recherches, Université Paul Sabatier.

Tekli, J., Chbeir, R., et Yetongnon, K. (2007). "Structural Similarity Evaluation Between XML Documents and DTDs". Web Information Systems Engineering – WISE 2007, 196-211.

Termier, A., Rousset, M. C., et Sebag, M. (2002). "TreeFinder: a First Step towards XML Data Mining". Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'02), IEEE Computer Society Washington, DC, USA, 450.

Tseng, F. S. C., et Chou, A. Y. H. (2006). "The concept of document warehousing for multi-dimensional modeling of textual-based business intelligence". Decision Support Systems, 42(2), 727-744.

Vercoustre, A. M., Fegas, M., Lechevallier, Y., Despeyroux, T., et Rocquencourt, I. (2006). « Classification de documents XML à partir d'une représentation linéaire des arbres de ces documents ». Paris, France, 433–444.

Vu, T. H., Denoyer, L., et Gallinari, P. (2003). « Un modèle statistique pour la classification de documents structurés ». Journees francophones d'Extraction et de Gestion des Connaissances (EGC 2003), Lyon, France, Jan.

Wagner, R. A., et Fischer, M. J. (1974). "The String-to-String Correction Problem". J. ACM, 21(1), 168-173.

Wang, Y., DeWitt, D. J., et Cai, J. Y. (2003). "X-Diff: an effective change detection algorithm for XML documents." Data Engineering, 2003. Proceedings. 19th International Conference on, 519-530.

Yi, J., et Sundaresan, N. (2000). "A classifier for semi-structured documents". Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM New York, NY, USA, 340-344.

Zhang, K., Wang, J. T. L., et Shasha, D. (1996). "On the Editing Distance Between Undirected Acyclic Graphs". International Journal of Foundations of Computer Science, 7(1), 43-58.

Chapitre V – Implantation et expérimentation

Résumé du chapitre. *Ce chapitre présente l'implantation des propositions de ce mémoire de thèse au sein du prototype MDOCREP. Ce prototype est constitué d'une application Java qui interagit avec le SGBD objet relationnel Oracle 10g2. Au travers de ce prototype, nous validons notre modélisation et nous montrons la faisabilité des démarches d'intégration et de restitution multidimensionnelle proposées. Ce chapitre présente également un ensemble d'expérimentations qui va permettre d'évaluer la démarche de classification.*

Sommaire du Chapitre V.

I. Introduction

Afin de valider les propositions présentées dans ce mémoire de thèse, nous avons réalisé un outil d'aide à l'intégration et à la restitution de documents multistructurés, intitulé « MDOCREP » (**M**ultistructured **DOC**ument **REP**ository). L'intégration est basée sur l'instanciation du modèle « MVDM » (Cf. Chapitre III - Section IV). Cette intégration suit la démarche de classification décrite dans le chapitre IV – Section II. La restitution des documents intégrés repose sur une adaptation de techniques de recherche d'information et d'analyse multidimensionnelle (Cf. Chapitre IV – Section III).

Plus précisément, MDOCREP assure les fonctionnalités suivantes :

- supporter une construction incrémentale d'une base de documents à partir de corpus filtrés et sélectionnés (jugés pertinents) récupérés de sources disséminées et hétérogènes ;
- assister l'utilisateur et éventuellement le décideur dans la restitution des documents intégrés selon une requête définie graphiquement ;
- automatiser une grande partie des tâches qui permettent la manipulation de la base de documents et de les rendre plus abordables ;
- supporter une ergonomie adaptée à la résolution des tâches complexes avec multifenêtrage, navigation et manipulation directe et intuitive de ses différents composants.

Dans ce chapitre, nous décrivons dans un premier temps l'architecture de l'outil MDOCREP, ainsi que ses différents modules. Nous présentons dans un deuxième temps les expérimentations réalisées et les résultats obtenus. Au travers de ces expérimentations, nous validons la démarche de classification de vues ainsi que la démarche de restitution multidimensionnelle présentées dans le chapitre IV.

II. Architecture de MDOCREP

Le prototype implanté MDOCREP repose sur une architecture client-serveur schématisée dans la Figure V.1. Cette architecture est basée d'une part, sur le SGBD « Oracle 10g2 » pour le stockage des structures et des contenus des documents selon le modèle « MVDM » et d'autre part, sur une interface client « Java 1.5 » afin d'assurer les différents traitements d'intégration et de restitution des documents.

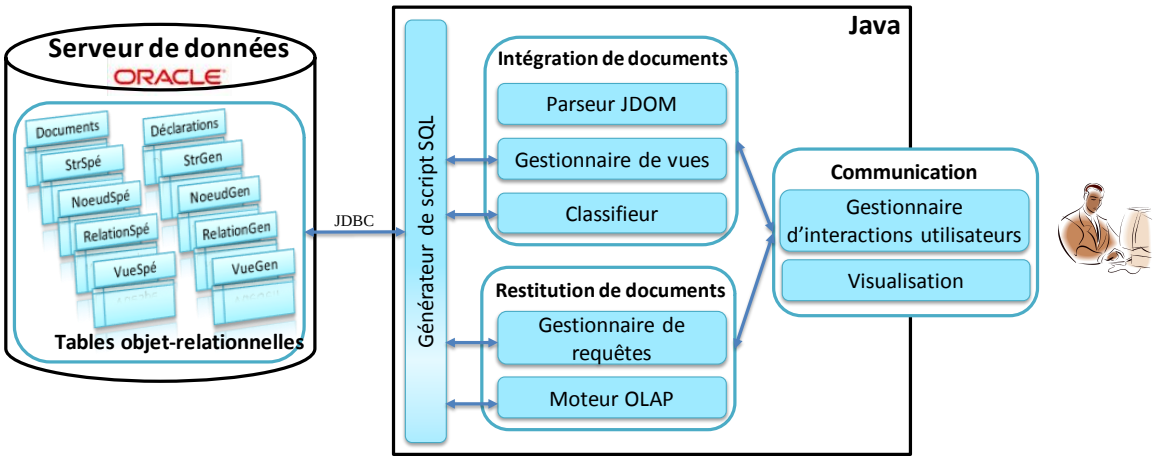


Figure V.1. Architecture générale du prototype MDOCREP.

II.1. Serveur de données

Le module serveur est une base de données (BD) objet relationnelle reposant sur le SGBD Oracle 10g2 ; il implante le modèle objet « MVDM » sous forme de tables objet-relationnelles. Ces tables admettent des attributs de types classiques (entier « number », chaine de caractère « varchar2 », etc.).

Le serveur de données communique avec l'application Java au travers de l'API JDBC. Il permet d'exécuter des scripts SQL générés à partir d'un générateur de scripts SQL. Ce dernier interagit avec les modules d'intégration et de restitution de documents afin de paramétrer les requêtes.

II.2. Intégration de documents

Le module d'intégration de documents est décomposé en trois sous-modules : le parseur JDOM, le gestionnaire de vues et le classifieur.

- Parseur JDOM

Afin d'extraire la structure des documents XML à intégrer dans la base, nous avons utilisé une API JDOM. Bien que JDOM interagit avec les normes existantes, telles que l'API SAX (Simple API for XML) et l'API DOM (Document Object Model), elle n'est pas considéré comme une extension à ces API. Elle a l'avantage d'offrir un robuste et léger moyen de manipulation (lecture et d'écriture) de données XML sans utiliser des notations complexes et sans surcharger la mémoire comme c'est le cas pour les autres API. Au travers de l'API JDOM, nous identifions les éléments et les attributs ainsi que le contenu de chaque document à intégrer dans la base.

- Gestionnaire de vues

Ce sous-module assure plusieurs tâches. Il permet de :

- intégrer une nouvelle vue spécifique (resp. générique) en assurant le partage de nœuds dans le cas de l'intégration d'une nouvelle vue pour un document existant,

- intégrer une nouvelle structure spécifique (resp. générique),
- agréger une vue spécifique (resp. générique) dans une structure spécifique (resp. générique),
- rattacher une vue (resp. structure) spécifique à une vue (resp. structure) générique,
- adapter une vue générique existante en lui ajoutant des nœuds génériques et des relations génériques afin qu'elle puisse représenter de nouvelles vues spécifiques.

- Classifieur

C'est au niveau de ce sous module que les démarches de classification et de calcul de distance entre vues présentées dans le chapitre IV sont exécutées. Le classifieur permet donc de :

- calculer la distance entre deux vues,
- trouver la vue générique la plus représentative de la vue spécifique à intégrer,
- conserver la représentativité des classes en reclassant les vues spécifiques dans les classes de vues les plus appropriées.

II.3. Restitution de documents

Le prototype MDOCREP propose deux techniques d'exploitation des documents multistructurés intégrées. Ces deux techniques sont présentées au travers de deux sous-modules :

- le gestionnaire de requêtes qui permet d'analyser les paramètres et les conditions choisis par l'utilisateur et de générer des requêtes selon les critères d'interrogation des documents multistructurés ;

- le moteur OLAP qui a comme rôle d'assister l'utilisateur (décideur) dans l'élaboration de magasins de documents. Cette assistance se traduit par la construction du schéma du magasin, la génération du magasin et la visualisation du contenu du magasin sous forme de table multidimensionnelle.

II.4. Communication

Le module de communication est composé de deux sous-modules :

- Gestionnaire d'interactions utilisateurs

Il s'agit des boites de dialogues, des menus et des interfaces permettant à l'utilisateur d'interagir avec les différents modules de l'outil. Ces composantes supportent une ergonomie adaptée à la résolution des tâches complexes avec multifenêtrage, navigation et manipulation directe et intuitive des différentes collections de documents. Ces composantes se caractérisent par les critères suivants :

- flexibles : elles peuvent être adaptées aux différentes utilisations possibles,
- incrémentales : l'exploitation se fait étape par étape, en visionnant si nécessaire ou si souhaité les résultats intermédiaires de chaque étape,

- uniformes : les modes d'interaction sont identiques pour les différentes fonctionnalités permettant d'améliorer l'utilisabilité de l'outil.

- Visualisation

MDOCREP permet d'automatiser une grande partie des tâches de manipulation et de les rendre plus abordables à tout utilisateur. Il permet de visualiser les organisations des vues afin de choisir les paramètres d'exploitation. Chaque type de nœud représenté dans une vue est représenté par un graphisme différent :

- les ovales pour les éléments ;
- les rectangles pour les attributs d'éléments.

Chaque fragment est précédé par sa cardinalité lorsque celle-ci est différente de un (? : zéro ou une occurrence, + : une ou plusieurs occurrences, * : zéro ou plusieurs occurrences).

Dans les analyses multidimensionnelles (Cf. Section V), nous utilisons les rectangles rouges pour désigner les dimensions et les rectangles orange pour désigner les faits.

III. Classification des vues

En utilisant le prototype MDOCREP, nous avons validé notre démarche de classification (Chapitre IV. Section II.2) au travers d'un ensemble d'expérimentations. Nous rappelons que cette démarche de classification comprend deux phases :

- la première consiste à rattacher une vue spécifique à sa classe ;
- la deuxième consiste à conserver la représentativité des classes notamment après leur adaptation.

Nous rappelons également que cette démarche de classification est basée sur une distance dite « structurelle » calculée à partir de trois pondérations :

- pondération structurelle : « *Str* ». Cette pondération permet de traduire l'organisation d'une vue (hiérarchie et ordre).
- pondération d'adaptation « *Adapt* » permettant d'évaluer le coût de modification d'un nœud ;
- pondération de représentativité « *Rep* » permettant de favoriser les relations les plus représentées.

Dans cette section, nous avons deux objectifs étroitement liés : l'évaluation de la démarche de classification et l'évaluation de l'impact des différentes pondérations sur la distance proposée. Dans ce qui suit nous présentons le corpus utilisé puis les expériences réalisées et enfin les résultats et les analyses de ces expériences.

III.1. Description du corpus

L'évaluation proposée est basée sur un corpus de 78 notices descriptives de livres au format XML issues de la bibliothèque de l'Université Toulouse 1 Capitole. Chaque

document comprend en moyenne 126 nœuds (80 éléments et 46 attributs) répartis sur six niveaux maximum. Chaque nœud comprend au maximum 10 nœuds fils.

Nous avons choisi un corpus relativement petit afin de pouvoir examiner chacun des documents. L'étude de ce corpus a montré que tous les documents partagent un ensemble de nœuds (éléments et attributs) constituant des sous-arborescences communes. Ces nœuds concernent la description du livre (Cf. Figure V.2) : « Description », « LanguageInfo », « TitleAndResponsibility », « Work », « TitleGroup », « Title », « IntellectualResponsibility », « PersonalName », « PublicationGroup », « Publication », « Publisher », etc. Les différences entre les documents de ce corpus sont constatées au niveau de l'ordre de certains nœuds communs, mais également dans l'intégration de sous-arborescences spécifiques telles que celles relatives à :

- l'origine et l'historique dans le cas des livres anciens « Origin », « Description », « Date », etc. ;
- la localisation physique du livre « PhysicalLocation », « Room », etc. ;
- la localisation de la version électronique dans le cas où elle existe « ElectronicLocation », « AccessMethod », « URL », etc.

En conclusion, la classification manuelle de ces documents basée sur leur similarité structurelle donne entre 7 et 11 classes différentes en fonction du degré de similarité souhaité. En effet, nous avons comparé les DTDs de ces documents. Nous avons constaté qu'il y a 7 DTDs non complémentaires marquées par plusieurs sous-arborescences (parties de DTD) différentes. Nous avons constaté également que 4 DTDs peuvent être couvertes par d'autres DTDs. Cependant, cette couverture est partielle : chacune de ces quatre DTDs ne représente qu'une partie de la DTD la plus proche.

La Figure V.2 présente un exemple de document issu de ce corpus de test.

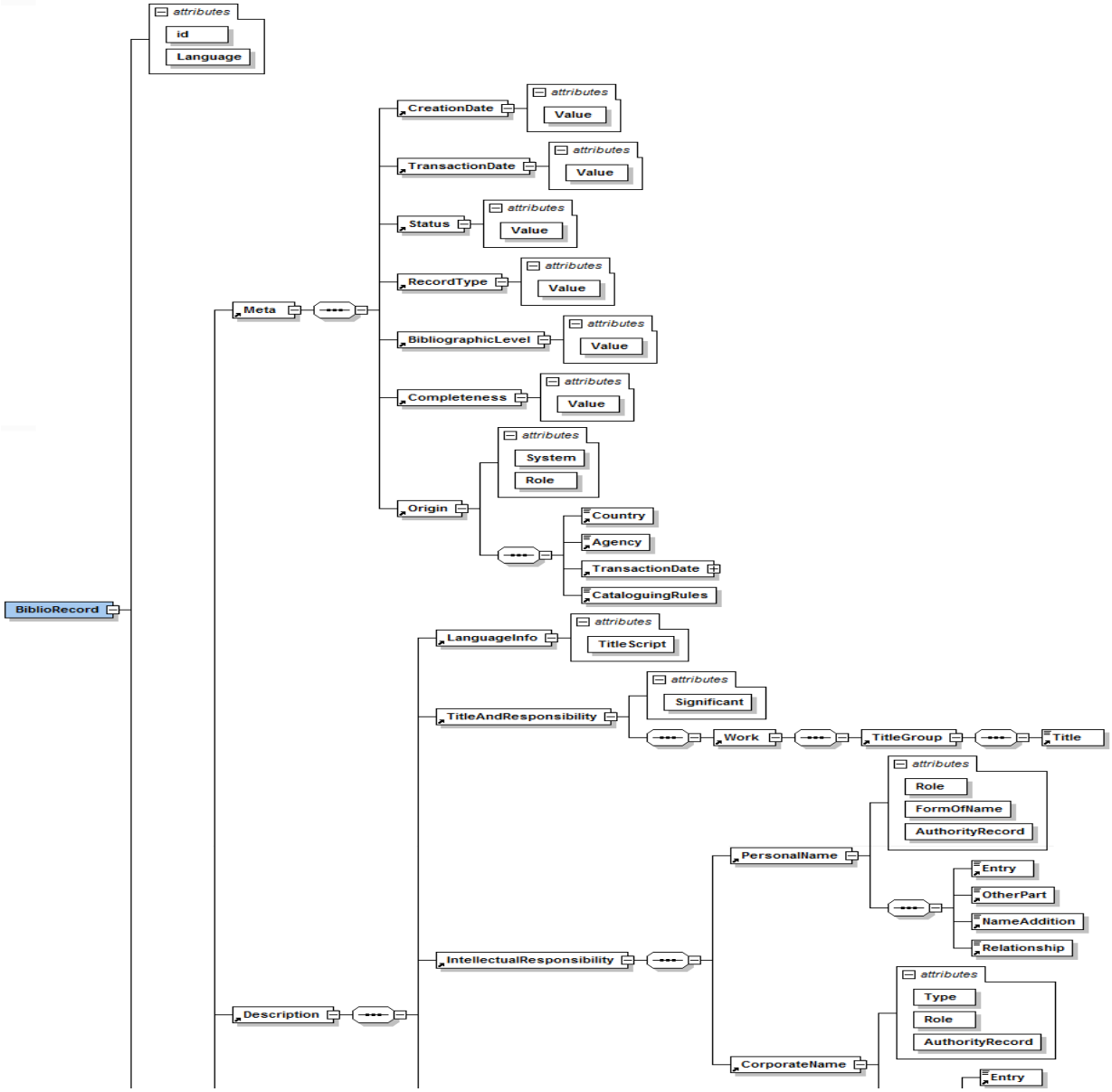
```
<?xml version="1.0" encoding="UTF-8"?>
<BiblioRecord Language="fr" id="PPN060789255">
	<Meta>
		<CreationDate Value="20020507" />
		<TransactionDate Value="" />
		<Status Value="c" />
		<RecordType Value="a" />
		<BibliographicLevel Value="m" />
		<Completeness Value="0" />
		<Origin Role="Issuer" System="">
			<Country>FR</Country>
			<Agency>Abes</Agency>
			<TransactionDate Value="20040330" />
			<CataloguingRules>AFNOR</CataloguingRules>
		</Origin>
	</Meta>
	<Description>
		<LanguageInfo TitleScript="ba" />
		<TitleAndResponsibility Significant="True">
			<Work>
				<TitleGroup>
					<Title>Timbres de los
gloriosissimos patriarchas fundadores de las sagradas religiones, y de
algunos de sus mas esclarecidos hijos...Primera parte...</Title>
```

```
                                </TitleGroup>
                        </Work>
                </TitleAndResponsibility>
                <IntellectualResponsibility>
                        <PersonalName Role="Primary" FormOfName="Surname"
AuthorityRecord="060789786">
                                <Entry>Castillo</Entry>
                                <OtherPart>Diego del</OtherPart>
                                <NameAddition>O.J.</NameAddition>
                                <Relationship>070</Relationship>
                        </PersonalName>
                        <CorporateName Role="Secondary" Type="Corporate"
AuthorityRecord="056535104">
                                <Entry>Couvent des Capucins</Entry>
                                <Subdivision>bibliothèque</Subdivision>
                                <NameAddition>Toulouse, Haute-
Garonne</NameAddition>
                                <Relationship>320</Relationship>
                        </CorporateName>
                </IntellectualResponsibility>
                <PublicationGroup>
                        <Publication>
                                <Publisher>
                                        <Place>En Valladolid</Place>
                                        <Name>en la imprenta de la real
chancilleria que es de la viuda de Joseph de Rueda</Name>
                                </Publisher>
                                <Date>1725</Date>
                        </Publication>
                </PublicationGroup>
                <PublicationDate>
                        <MonographDate Status="Uncertain">
                                <Year>1725</Year>
                                <Year>1725</Year>
                        </MonographDate>
                </PublicationDate>
                <PhysicalDescription>
                        <PhysicalItem>
                                <ItemDescription>
                                        <Material>XX,390 p.</Material>
        <OtherPhysicalDetails>ill.gr.s.b.</OtherPhysicalDetails>
                                        <Dimensions>in-2</Dimensions>
                                </ItemDescription>
                        </PhysicalItem>
                </PhysicalDescription>
        </Description>
        <CodedValues>
                <cdMonographic Illustration="y" FormOfContents="z"
Conference="0" Festschrift="0" Index="1" Literature="c" />
                <cdTextualPhysical Medium="r" />
        </CodedValues>
        <Notes>
                <Note Type="PhysicalDescription">lettres ornées, bandeaux
et culs-de-lampe gravés sur bois</Note>
                <CopyInHandNote Institution="">Reliure de
parchemin</CopyInHandNote>
                <ProvenanceNote Institution="">Cachet de la bibliothèque
du couvent des capucins de Toulouse</ProvenanceNote>
                <Note Type="Reproduction">Texte numérisé d après l
exemplaire Res Cap A 8621 DEL de la Bibliothèque universitaire centrale
```

```
du Mirail (SCD Toulouse2), consultable en ligne http://www.biu-
toulouse.fr/num150/Timbres.pdf</Note>
        </Notes>
        <Subjects>
                <TargetAudience Code="k" />
                <PlaceAccess>
                        <Country>Espagne</Country>
                        <City>Valladolid</City>
                </PlaceAccess>
                <OtherClass System="brp-sys">
                        <Class>531ANC</Class>
                </OtherClass>
        </Subjects>
        <LocalData />
</BiblioRecord>
```

Figure V.2. Exemple du contenu d'un document issu du corpus des notices de livre.

La Figure V.3 présente le schéma XML du document présenté dans la Figure V.2. Un tel schéma peut être considéré, dans nos travaux, comme une vue générique qui n'a subi aucune adaptation.



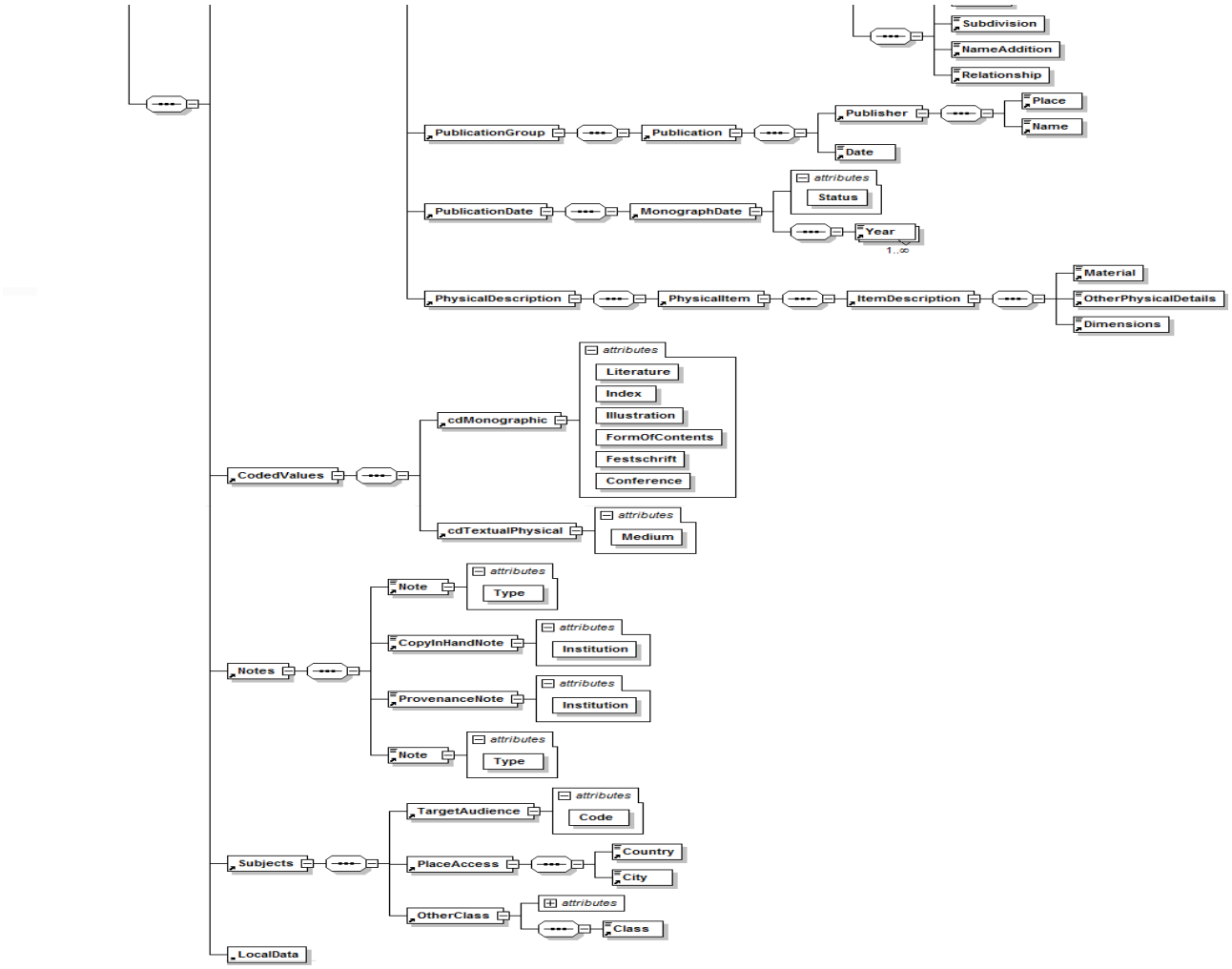


Figure V.3. Schéma XML du document présenté dans la Figure V.2 généré par l'outil XMLSpy.

III.2. Description des expériences

Afin d'évaluer notre démarche de classification (Cf. Chapitre IV – Section II.2.3) et le calcul de distance (Cf. Chapitre IV – Section II.2.2), nous proposons une démarche d'évaluation basée sur la variation des critères de classification et de distance. Une telle démarche permet de montrer l'influence de chacun des paramètres sur les résultats de la classification. Dans cet objectif, nous avons élaboré cinq tests :

- StrAdaptRepCons : dans ce test, nous utilisons les trois pondérations proposées avec conservation de la représentativité ;
- StrAdaptRep : dans ce test, nous utilisons les trois pondérations proposées sans conservation de la représentativité ;
- StrAdaptCons : dans ce test, nous utilisons la pondération structurelle et la pondération d'adaptation avec conservation de la représentativité ;
- AdaptRepCons : dans ce test, nous utilisons les pondérations d'adaptation et de représentativité avec conservation de la représentativité ;
- StrRepCons : dans ce test, nous utilisons la pondération structurelle et la pondération de représentativité avec conservation de la représentativité.

Dans ces cinq tests, nous avons intégré les 78 documents dans le même ordre. Dans chacun des tests, une vue spécifique est créée pour chaque document. Cette vue spécifique sera comparée à un sous-ensemble de vues génériques. Ce sous-ensemble est sélectionné à partir de l'ensemble des vues génériques existantes en fonction d'un seuil minimal de filtrage fixé à 80%. La vue générique la plus proche dans ce sous-ensemble est sélectionnée. Cette vue sera le représentant de la vue spécifique si le degré de similarité entre ces deux vues est supérieur au seuil d'agrégation (seuil minimal de degré de similarité) fixé à 0,8 (80% de similarité) pour tous nos tests. Dans les quatre premiers tests et en cas d'adaptation d'une vue générique, nous exécutons la phase de conservation de représentativité. Nous avons fixé le seuil de dispersion à 0,03 (3%). Nous avons fixé l'ensemble des seuils suite à plusieurs expérimentations. Etant donné le corpus, soit les vues spécifiques à classer sont identiques aux représentants, soit les différences se situent relativement haut dans l'arborescence ce qui induit une distance importante entre les vues. Dans tous les cas, les vues spécifiques ne sont pas radicalement différentes, elles ont une sous-arborescence commune. L'écart type dépasse rarement 3% et ne va jamais au-delà de 4%. Le seuil de dispersion en dessous de 3% est trop strict : les vues spécifiques ne peuvent être rattachées à aucune des classes, sauf si elles sont strictement identiques. Ceci amènera à une multiplication des classes ayant très peu de vues spécifiques rattachées. Ces observations ont été confirmées par les différentes expérimentations (Cf. Tableau V.1). En effet, afin de fixer les différents seuils, nous avons effectué 12 tests. Ces tests résultent de la combinaison de trois valeurs fixées d'une part pour le seuil d'agrégation (70%, 80% et 90%) et d'autre part de quatre valeurs pour le seuil de dispersion (2%, 3%,4% et 5%).

Tableau V.1. Nombre de classes et moyenne de similarités en fonction des seuils.

Seuil de dispersion / **Seuil d'agrégation**	Nb de Classes/Moyenne de similarités	**2%**	**3%**	**4%**	**5%**
70%		5/80,9	6/79,2	7/78,2	7/78,2
80%		15/83,6	16/89,4	18/87,3	18/87,3
90%		31/95,2	32/95,8	34/93,2	34/93,2

Le choix d'un seuil d'agrégation égale 90% génère un surnombre de classes en contrepartie les seuils 70% et 80% donnent des résultats semblables à ce qui est observé dans la réalité. Au vu des résultats, un nombre de classes avoisinant 11 et une moyenne de similarités maximale étant recherchée, une unique combinaison de seuils a été envisagée (80%, 3%).

Une fois les 78 vues spécifiques classées, nous vérifions si elles sont rattachées à la bonne vue générique. A cette fin, nous recalculons la distance (en utilisant les paramètres choisis dans le test en question) entre chaque vue spécifique et l'ensemble des vues génériques. Dans le cas où une vue spécifique est plus proche d'une vue générique d'une autre classe alors nous marquons cette vue spécifique comme une vue à reclasser. Un tel cas s'explique par l'évolution des vues génériques : une vue générique peut subir plusieurs

adaptations suite au rattachement de vues spécifiques « suffisamment » proches, mais non identiques.

Afin d'évaluer les résultats obtenus dans chacun des tests effectués, nous avons calculé pour chaque classe identifiée :

- le nombre de vues spécifiques rattachées « Nombre de vues spécifiques par classe » ;
- le nombre de vues spécifiques qui seront « mieux » représentées dans une autre classe « Nombre de vues spécifiques à reclasser ». Comme nous l'avons expliqué précédemment, une fois la classification terminée, nous avons recalculé les similarités entre chaque vue spécifique et l'ensemble des vues génériques afin de retrouver le nombre de vues spécifiques qui seraient mieux représentées par une autre vue générique ;
- la moyenne des similarités entre les vues spécifiques rattachées « Moyenne des similarités dans une classe » ;
- l'écart type des similarités entre les vues spécifiques rattachées.

III.3. Résultats et Analyses

- Test 1 : StrAdaptRepCons

Le premier test est effectué en utilisant tous les paramètres de classification et de calcul de distance proposés. Le Tableau V.2. récapitule l'ensemble des résultats obtenus.

Tableau V.2. Récapitulatif des résultats des expérimentations avec les paramètres StrAdaptRepCons.

Classe	Nombre de vues spécifiques par classe	Nombre de vues spécifiques à reclasser	Moyenne des similarités dans une classe	Ecart type
1	4	2	0,80539915	0,018365802
2	16	5	0,852316801	0,023249129
3	3	1	0,815614541	0,021383064
4	9	1	0,84641546	0,013938163
5	7	0	0,860182634	0,017845131
6	3	2	0,839617002	0,015906674
7	11	4	0,829318394	0,017241559
8	1	0	1	0
9	1	0	1	0
10	14	9	0,838447043	0,009045331
11	2	0	0,8894687	0,009597836
12	2	2	0,799007237	0,000670373
13	1	0	1	0
14	1	0	1	0
15	2	0	0,932957033	0,011733433
16	1	0	1	0

Les vues spécifiques sont réparties en seize classes. En moyenne, chaque classe comprend 4,8 vues spécifiques. Cependant, il y a des classes qui sont plus peuplées que d'autres. La Figure V.4 montre la répartition des vues spécifiques par classe. Nous remarquons en particulier que la moitié des classes comporte seulement une ou deux vues

spécifiques. Si on se focalise sur l'une de ces classes, en particulier la classe 12, nous constatons que cette classe admet deux vues spécifiques et ces deux vues sont à reclasser ce qui veut dire que cette classe doit disparaitre. Ceci nous amène à penser qu'il faut ajouter une troisième phase dans la démarche de classification. Cette phase doit permettre de redistribuer certaines vues spécifiques et éventuellement de fusionner les classes qui deviennent proches suite à l'adaptation de l'une d'entre elles.

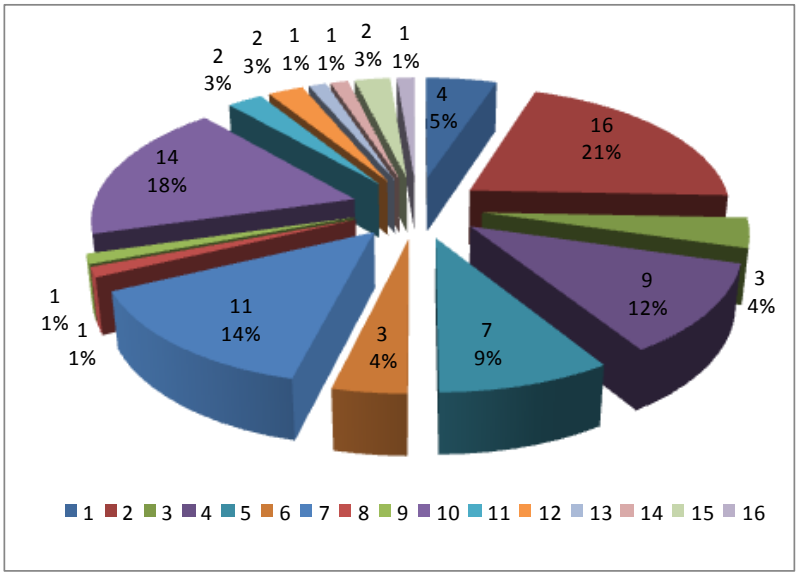


Figure V.4. Répartition des vues spécifiques par classe avec les paramètres StrAdaptRepCons.

La prise en compte de l'ensemble des paramètres (pondération structurelle, pondération d'adaptation, pondération de représentativité et conservation de représentativité) nous a permis d'avoir des classes marquées par une dispersion inférieure à 0,01 pour la moitié des classes. La Figure V.5 permet de visualiser la moyenne des similarités et l'écart type pour chaque classe. Nous remarquons que dans la majorité des classes, la similarité moyenne se situe autour de 80% avec une dispersion inférieure à 0,01.

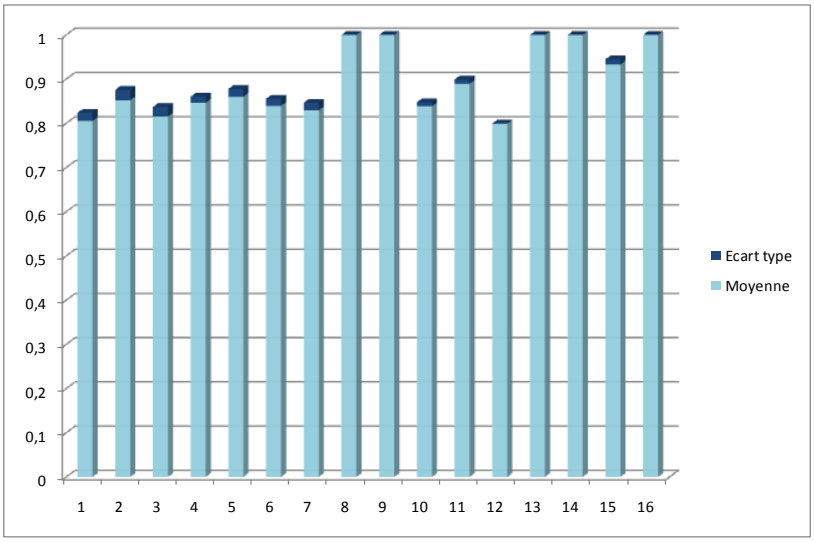


Figure V.5. Moyenne et écart type de chaque classe avec les paramètres StrAdaptRepCons.

- Test 2 : StrAdaptRep

Concernant le deuxième test, nous avons enlevé, par rapport au test 1, la phase de conservation de la représentativité. Ceci va nous permettre de voir l'impact de cette phase sur les résultats de classification. En d'autre terme, nous allons voir si les classes obtenues sont réellement plus dispersées sans cette phase. Le Tableau V.3 présente les résultats de ce test.

Tableau V.3. Récapitulatif des résultats des expérimentations avec les paramètres StrAdaptRep.

Classe	Nombre de vues spécifiques par classe	Nombre de vues spécifiques à reclasser	Moyenne des similarités dans une classe	Ecart type
1	4	4	0,80539915	0,01836579
2	16	4	0,85231686	0,0232491
3	3	0	0,85563987	0,01855552
4	11	0	0,93324861	0,03897024
5	5	5	0,81620489	0,02950682
6	11	7	0,78257715	0,03101675
7	1	0	1	0
8	15	0	0,82808808	0,00950439
9	2	0	0,8894687	0,01175484
10	4	0	0,91879894	0,01733554
11	2	0	0,88128889	0,0115239
12	1	0	1	0
13	1	0	1	0
14	2	0	0,9037717	0,00549251

Nous remarquons que le classifieur a généré quatorze classes. Les 78 vues spécifiques relatives aux 78 documents du corpus utilisé sont réparties sur ces 14 classes comme le présente la Figure V.6. Nous remarquons à ce niveau que 53 vues spécifiques (68% du corpus) sont réparties sur quatre classes (2, 4, 6 et 8). Les autres classes n'admettent pas plus de cinq vues spécifiques. D'autre part, nous remarquons qu'il y a 20

vues spécifiques à reclasser concentrées sur 4 classes (1, 2, 5 et 6). La totalité des vues spécifiques rattachées à la classe 1 doit être reclassée.

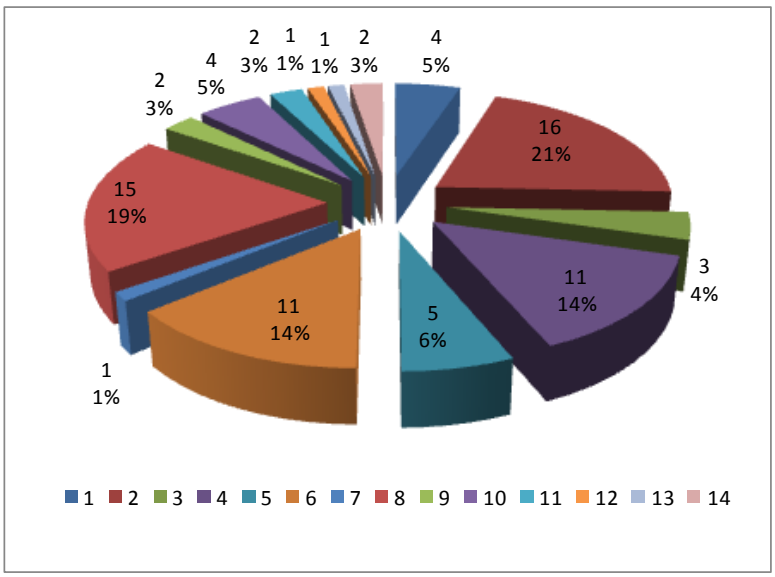


Figure V.6. Répartition des vues spécifiques par classe avec les paramètres StrAdaptRep.

Concernant la moyenne des similarités dans chaque classe, nous remarquons que 6 classes admettent des moyennes de similarités supérieures à 0,9% et la moitié des classes ont des moyennes de similarités comprises entre 0,8% et 0,9%. Ceci dit, l'écart type de ces sept classes est supérieur à 0,015% c'est-à-dire que ces classes sont plus dispersées que celles retrouvés dans le premier test. La Figure V.7 permet de visualiser la moyenne des similarités et l'écart type pour chaque classe.

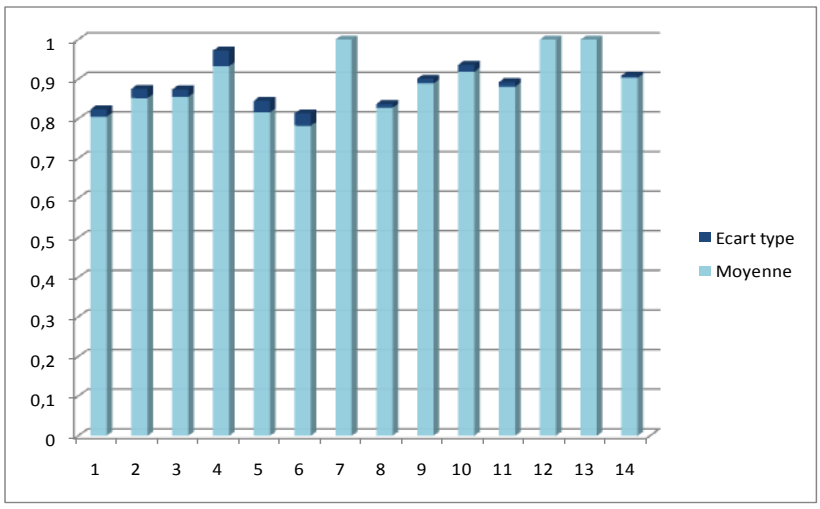


Figure V.7. Moyenne et écart type de chaque classe avec les paramètres StrAdaptRep.

- Test 3 : StrAdaptCons

Dans ce troisième test, nous avons enlevé, par rapport au test 1, un paramètre relatif à la distance : la pondération de représentativité. Un tel test va nous permettre de voir si les

vues spécifiques admettent des sous-arborescences communes. Ceci doit se répercuter sur la classification et notamment sur la dispersion des classes qui doit augmenter. Le Tableau V.4 présente les résultats de ce test.

Tableau V.4. Récapitulatif des résultats des expérimentations avec les paramètres StrAdaptCons.

Classe	Nombre de vues spécifiques par classe	Nombre de vues spécifiques à reclasser	Moyenne des similarités dans une classe	Ecart type
1	5	3	0,84108073	0,02547231
2	12	12	0,80525485	0,02551633
3	4	2	0,82546078	0,026189
4	7	0	0,89331121	0,01876203
5	5	1	0,8947992	0,02578021
6	7	4	0,84469215	0,02095044
7	3	1	0,82078848	0,02844502
8	19	0	0,85850834	0,01380414
9	1	0	1	0
10	4	1	0,82369967	0,02119893
11	1	0	1	0
12	2	0	0,97382483	0,00753775
13	1	0	1	0
14	3	0	0,85684512	0,00550728
15	2	0	0,82138637	0,00453493
16	2	0	0,89896598	0,00071821

Avec ce test, nous obtenons 16 classes dont 6 admettent une ou deux vues spécifiques seulement. La majorité des vues spécifiques (62%) sont reparties sur 14 classes d'une manière plus ou moins équivalente : chacune de ces classes n'admet pas plus de 9% des vues spécifiques (Cf. Figure V.8). D'autre part, nous remarquons qu'il y a 24 vues spécifiques à reclasser réparties sur 7 classes. En particulier, la classe 2 peut être fusionnée avec une autre et par conséquent toutes ses vues spécifiques doivent être reclassées. Les expérimentations ont montré que les vues spécifiques de la classe 2 doivent être reclassées dans classe 6.

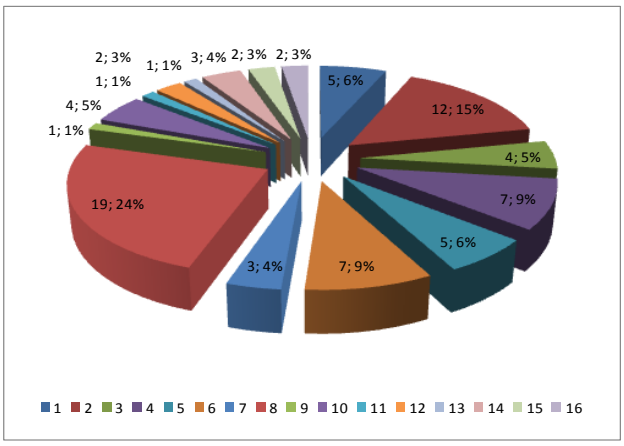


Figure V.8. Répartition des vues spécifiques par classe avec les paramètres StrAdaptCons.

En ce qui concerne l'homogénéité des classes, nous remarquons que la plupart des moyennes de similarité sont comprises entre 0,8% et 0,9% et les écarts types dépassent 0,2 pour 7 classes. La Figure V.9 illustre l'ensemble des moyennes de similarité ainsi que des écarts type obtenus.

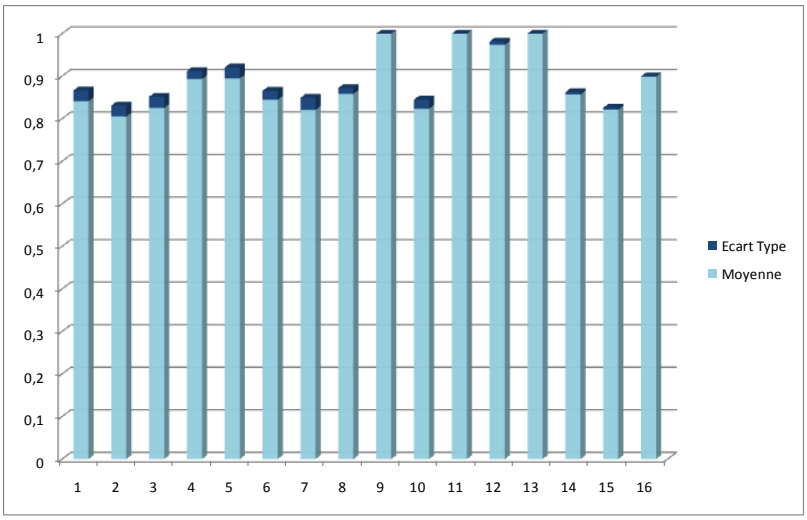


Figure V.9. Moyenne et écart type de chaque classe avec les paramètres StrAdaptCons.

- Test 4 : AdaptRepCons

Dans le quatrième test, nous avons enlevé, par rapport au test 1, un autre paramètre relatif à la distance : la pondération structurelle. Un tel test permet de montrer l'intérêt de prendre en compte l'ordre et la hiérarchie dans la classification. Le Tableau V.5 présente les résultats de ce test.

Tableau V.5. Récapitulatif des résultats des expérimentations avec les paramètres AdaptRepCons.

Classe	Nombre de vues spécifiques par classe	Nombre de vues spécifiques à reclasser	Moyenne des similarités dans une classe	Ecart type
1	7	2	0,84220028	0,01978151
2	20	11	0,83648407	0,01554198
3	6	5	0,80967099	0,01720059
4	2	2	0,73388022	0,0170626
5	1	1	1	0
1	1	1	1	0
7	29	15	0,83527158	0,01952881
8	1	1	1	0
9	8	4	0,86633421	0,01566
10	1	0	1	0
11	2	0	0,87200135	0,01053482

Ce test se distingue par un nombre de classes plus faible (seulement 11 classes). Plus de la moitié des vues spécifiques sont réparties sur deux classes : la classe 2 admet 26% des vues spécifiques et la classe 7 admet 37% des vues spécifiques (Cf. Figure V.10). Nous remarquons également que 6 classes n'admettent que 1 ou 2 vues spécifiques. D'autre part, nous remarquons qu'il y a 42 vues spécifiques à reclasser réparties sur 9 classes. En

particulier, les deux classes les plus peuplées (2 et 7) admettent plus que la moitié de leurs vues à reclasser. La diminution du nombre de classes par rapport aux tests précédents est expliquée par la nature du corpus utilisé. En effet, les documents du corpus sont marqués par des nœuds communs mais qui ne sont pas au même endroit dans la hiérarchie. En levant la pondération structurelle, on lève une contrainte forte qui est double : ordre et hiérarchie. Ceci amènera à l'augmentation des distances et par conséquent à la diminution du nombre de classes.

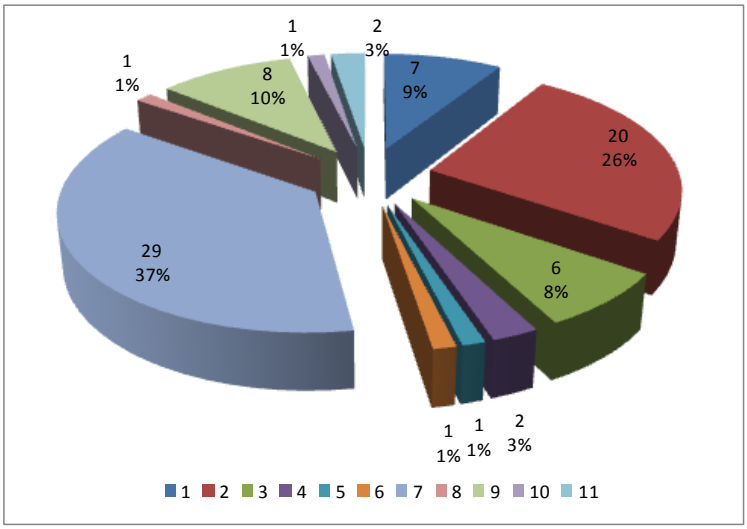


Figure V.10. Répartition des vues spécifiques par classe avec les paramètres AdaptRepCons.

Concernant la moyenne des similarités dans chaque classe, nous remarquons que sept classes admettent des moyennes de similarités inférieures à 0,9%. L'écart type de ces sept classes est supérieur à 0,015%. La Figure V.11 permet de visualiser la moyenne des similarités et l'écart type pour chaque classe.

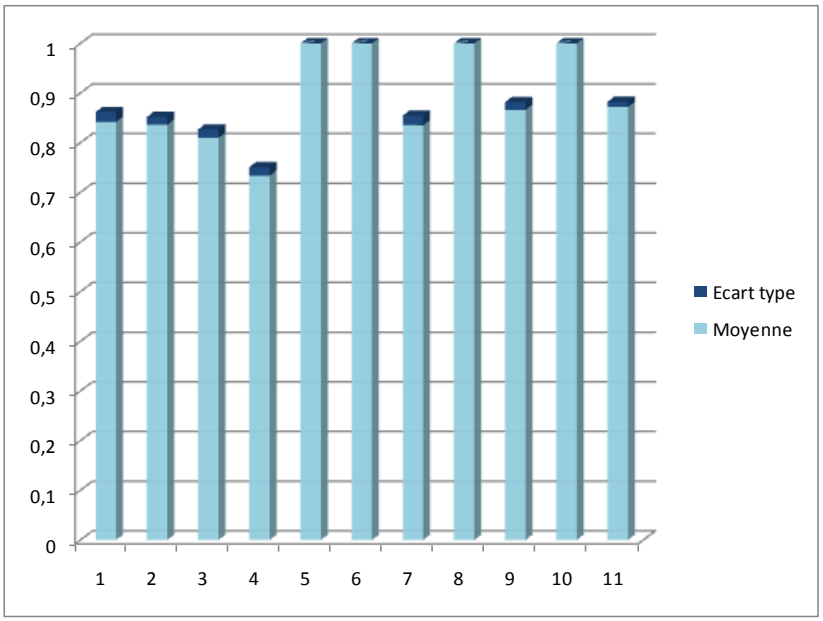


Figure V.11. Moyenne et écart type de chaque classe avec les paramètres AdaptRepCons.

- Test 5 : StrRepCons

Dans le dernier test, nous avons enlevé, par rapport au test 1, le troisième paramètre relatif à la distance : la pondération d'adaptation. Le but d'un tel test est de voir si les modifications aperçues entre les vues ont le même coût c'est-à-dire s'il y a des modifications de relations qui sont proches de la racine. Le Tableau V.6 présente les résultats de ce test.

Tableau V.6. Récapitulatif des résultats des expérimentations avec les paramètres StrRepCons.

Classe	Nombre de vues spécifiques par classe	Nombre de vues spécifiques à reclasser	Moyenne des similarités dans une classe	Ecart type
1	9	5	0,85794523	0,02625081
2	19	14	0,84617602	0,02800279
3	1	0	1	0
4	9	2	0,87826843	0,01641909
5	7	2	0,86862	0,02576597
6	1	1	1	0
7	2	0	0,92726877	0,0022414
8	1	0	1	0
9	19	0	0,90225007	0,02570511
10	6	0	0,86370199	0,0154908
11	2	0	0,97086424	0,00159222
12	1	0	1	0
13	1	0	1	0

Nous remarquons que le classifieur a généré 13 classes. Il y a 7 classes qui admettent 1 ou 2 vues spécifiques seulement. 38 vues spécifiques (49%) sont réparties sur seulement les classes 2 et 9 (Cf. Figure V.12). D'autre part, nous remarquons qu'il y a 24 vues spécifiques à reclasser réparties sur 5 classes. En particulier, les deux classes 1 et 2 admettent plus que la moitié de leurs vues à reclasser.

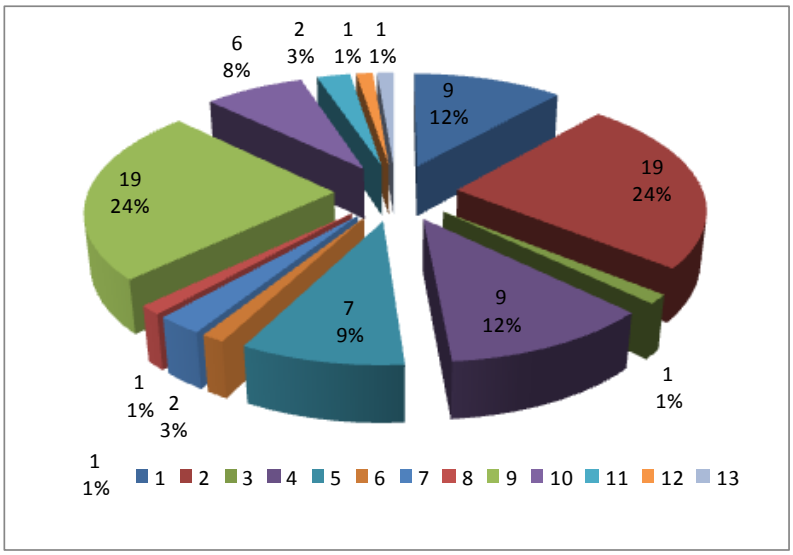


Figure V.12. Répartition des vues spécifiques par classe avec les paramètres StrRepCons.

Les moyennes de similarité et les écarts types sont illustrés dans la Figure V.13. Bien que nous retrouvions des classes marquées par une dispersion nulle ou presque nulle, celles qui admettent 1 ou 2 vues spécifiques, le reste des classes admet un écart type qui dépasse 0,015.

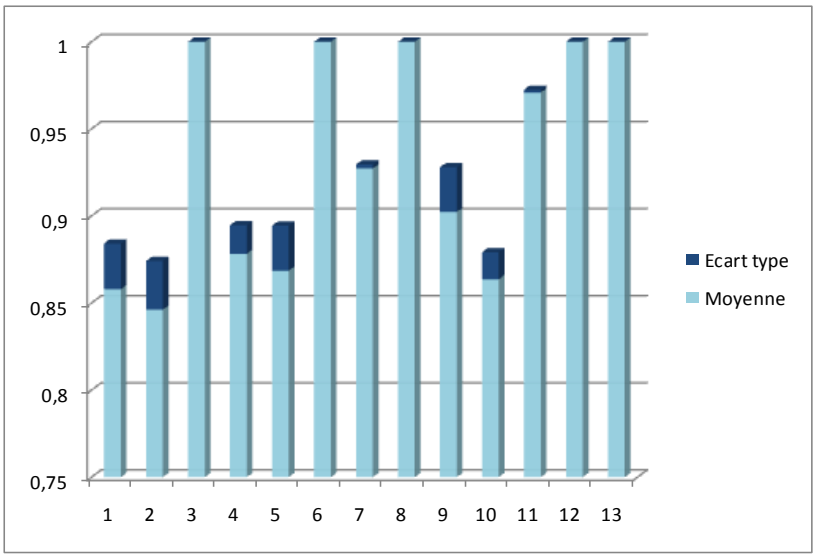


Figure V.13. Moyenne et écart type de chaque classe avec les paramètres StrRepCons.

III.4. Bilan et synthèse

Les résultats de l'ensemble des tests effectués sont synthétisés dans le Tableau V.7. Ce tableau présente pour chaque test le nombre de classes obtenues, la moyenne du nombre de vues spécifiques par classe, la moyenne des similarités moyennes, la moyenne des écarts types et le pourcentage de vues spécifiques à reclasser. Les valeurs présentées sont calculées à partir des tableaux correspondant à chaque test effectué (Cf. Tableau V.2 ; Tableau V.3 ; Tableau V.4 ; Tableau V.5 ; Tableau V.6).

Tableau V.7. Récapitulatif des résultats des cinq tests effectués.

	StrAdaptRepCons	StrAdaptRep	StrAdaptCons	AdaptRepCons	StrRepCons
Nombre de classes	16	14	16	11	13
Moyenne du nombre de vues spécifiques par classe	4,875	5,5714	4,875	3,818	6
Moyenne des similarités moyennes	0,8942	0,8904	0,8849	0,8905	0,9319
Moyennes des écarts types	0,0099	0,0153	0,0140	0,0104	0,0108
Pourcentage des vues spécifiques à reclasser	33,33%	25%	30,7%	53,84%	30,7%

Sur le Tableau V.7, les résultats montrent que la phase de conservation de représentativité agit directement sur la dispersion des classes. Ceci est remarquable sur l'écart type de StrAdaptRep où nous avons enlevé cette phase (écart type égal à 0,0153 pour StrAdaptRep contre 0,0099 StrAdaptRepCons).

En nous focalisant sur le nombre de vues spécifiques à reclasser, nous remarquons que la prise en compte de la structure est essentielle (53% de documents à reclasser sans la pondération structurelle).

L'observation des résultats du test effectué sans pondération d'adaptation montre que la moyenne des similarités moyennes est relativement élevée (0,9319). En comparant cette moyenne avec les moyennes des similarités moyennes des autres tests, nous pouvons dire que les différences entre les vues spécifiques se trouvent notamment sur des éléments proches de la racine. L'observation des structures des documents du corpus confirme ces résultats. En effet, nous remarquons que certaines sous-arborescences constituant des différences entre deux vues spécifiques telles que les sous-arborescences qui décrivent la localisation physique ou celles qui décrivent la localisation de la version électronique (Cf. section III.1) sont rattachées directement à la racine de la structure documentaire. Bien que les résultats de ce test semblent être les plus proches de la réalité (nombre de classes égales à celui retrouvé lors de classification manuelle, une moyenne des similarités moyennes relativement élevées : 0,9319, etc.), les classes obtenues regroupent des vues spécifiques admettant des différences au niveau des éléments proches de la racine. Ceci peut agir sur la sémantique de la classe elle-même si on considère que les nœuds proches de la racine sont plus significatifs. Par exemple, dans ce test, on ne retrouve pas une classe spécifique au livre ancien comme il est observé lors de la classification manuelle.

En comparant le bilan du premier test (la prise en compte des trois pondérations avec conservation de la représentativité : StrAdaptRepCons) avec celles des quatre autres tests (StrAdaptRep, StrAdaptCons, AdaptRepCons et StrRepCons), les résultats ne sont pas si éloignés les unes des autres : un nombre de classes proche, une moyenne des similarités moyennes satisfaisante. Seulement, le pourcentage de vues spécifiques à reclasser est important (33,33%).

En examinant les documents du corpus, nous constatons que les vues spécifiques à reclasser retrouvées dans le premier test (StrAdaptRepCons) pourront s'intégrer dans d'autres classes engendrant la suppression des classes dont elles dépendent. Ceci est cohérent avec les résultats retrouvés lors de classification manuelle où nous avons retrouvé 11 classes contre 16 pour le test StrAdaptRepCons. De tels résultats nous amènent à penser que la classification basée sur une distance structurelle avec conservation de représentativité fournit de bons résultats si toutefois on fusionne certaines classes jugées « proches » en redistribuant les vues spécifiques à reclasser.

Si les résultats retrouvés en utilisant un corpus assez restreint sont encourageants, des expérimentations sur un corpus plus étendu et plus hétérogène devraient être menées afin de mieux mesurer l'impact des différents paramètres sur les résultats de la classification.

IV. Restitution des documents multistructurés : Cas d'une analyse multidimensionnelle

Nous proposons dans cette section une validation de la démarche d'analyse multidimensionnelle. Parmi les cinq types d'analyses distingués dans le chapitre IV, nous présentons le cas d'une analyse par structure générique.

Dans ce qui suit, nous détaillons la nature du corpus utilisé et ensuite les différentes étapes d'analyse multidimensionnelle.

IV.1. Description du corpus

Les documents ayant servi à nos expérimentations ont été extraits de sources diverses : il s'agit des documents XML issus des émissions de RFI (Radio France International) et RTM (Radio Télévision du Maroc) annotées dans le cadre des projets RAIVES (Parlangeau-Vallès et al. 2003). RAIVES (Recherche Automatique d'Informations Verbales Et Sonores) est un projet d'indexation des documents sonores. Ce projet a pour objectif de structurer les documents sonores, en particulier radiophoniques, à partir de l'indexation de leur contenu, de manière à leur donner un sens du point de vue d'un utilisateur du Web et de produire à partir de ces documents des connaissances exploitables. Ce contenu pourrait alors être accessible aux moteurs de recherche et devenir disponible aux internautes au même titre que le contenu textuel de pages HTML.

Afin de valider notre approche d'analyse multidimensionnelle (Djemal et al. 2010a), nous avons utilisé trente documents audio annotés dans le cadre du projet RAIVES. Pour chaque document audio, nous avons retrouvé trois fichiers qui matérialisent trois segmentations différentes du même document. Ces trois fichiers présentent des découpages en Musique/NonMusique, Parole/NonParole. Pour les segments parole trois découpages sont présentés : en Thème, en Locuteur et en Langue. Au niveau de ces fichiers, certains découpages ne sont pas repérés par des balises. De ce fait, nous avons opté pour une réorganisation de ces fichiers à l'aide de moulinettes XSLT. En résultat, nous avons obtenu trois fichiers dont chacun présente une structuration particulière du document audio. Il comprend en moyenne 60 nœuds répartis sur quatre niveaux max. Chaque nœud comprend au maximum huit fils.

L'intégration de ces documents nous a permis d'avoir une structure générique « RaivesCorpus » comprenant trois vues génériques : « Language », « Speaker » et « Topic ». Chaque vue générique représente un découpage particulier d'un même document audio. Si ces trois vues partagent les nœuds génériques « AudioSegment », « Music », « Noise » et « Speech »,

- dans la vue générique « Language », l'élément « Speech » est décomposé en « Language » ensuite en « Dialect » et « Trans » ;
- dans la vue générique « Speaker », l'élément « Speech » est décomposé en « Speaker » ensuite en « Name » et « Trans » ;

- dans la vue générique « Topic », l'élément « Speech » est décomposé en « Topic » ensuite en « Trans ».

IV.2. Démarche

IV.2.1. Choix du type d'analyse approprié

La première phase de la restitution multidimensionnelle consiste en la construction du schéma du magasin. La première étape est la sélection du type d'analyse. Dans notre exemple, nous avons choisi de montrer une analyse par structure générique (Cf. Figure V.14). Nous rappelons que la structure générique englobe plusieurs vues génériques. Pour cela, nous activons les menus : *Exploitation* puis *Analyse* puis *Type* puis *Par structure générique*

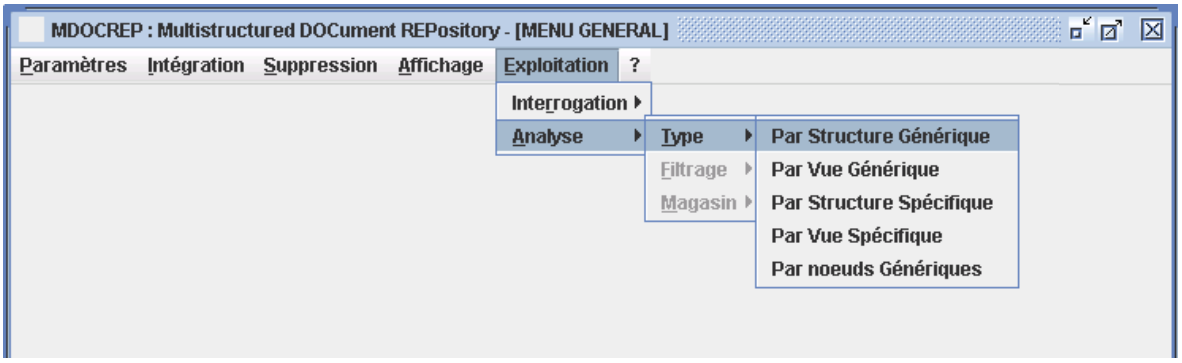


Figure V.14. Activation des menus.

Ainsi, le système affiche la liste de toutes les structures existantes. Parmi ces structures, nous devons choisir la structure générique « RaivesCorpus » qui représente notre corpus.

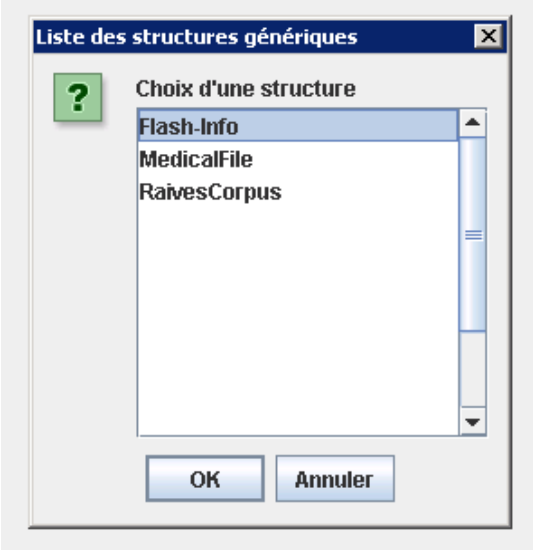


Figure V.15. Choix d'une structure générique.

IV.2.2. Sélections des composants

Une fois que le choix de la structure générique a été effectué, le système affiche d'une manière automatique l'ensemble de ses vues génériques ainsi que leur organisation (Cf. Figure V.16). Nous rappelons que chaque élément est représenté par un ovale et chaque attribut est symbolisé par un rectangle.

La sélection des composants d'analyse consiste en la précision des dimensions et du fait souhaité. L'affectation de ces rôles se fait au travers des menus contextuels (Cf. Figure V.16). Pour cela, nous devons pointer l'élément désiré et avec un clic sur le bouton droit nous fixons notre choix (Fait ou Dimension) ainsi que les attributs, à savoir : l'ordre pour les dimensions ou la fonction d'agrégation pour le fait (Compte, Somme, Maximum, Minimum, Moyenne, Contenu).

Dans notre exemple, nous allons choisir les dimensions et le fait à partir des trois vues génériques proposées : « Language », « Speaker » et « Topic ».

A partir de la première vue générique, nous sélectionnons la dimension « Language » (Cf. Figure V.16).

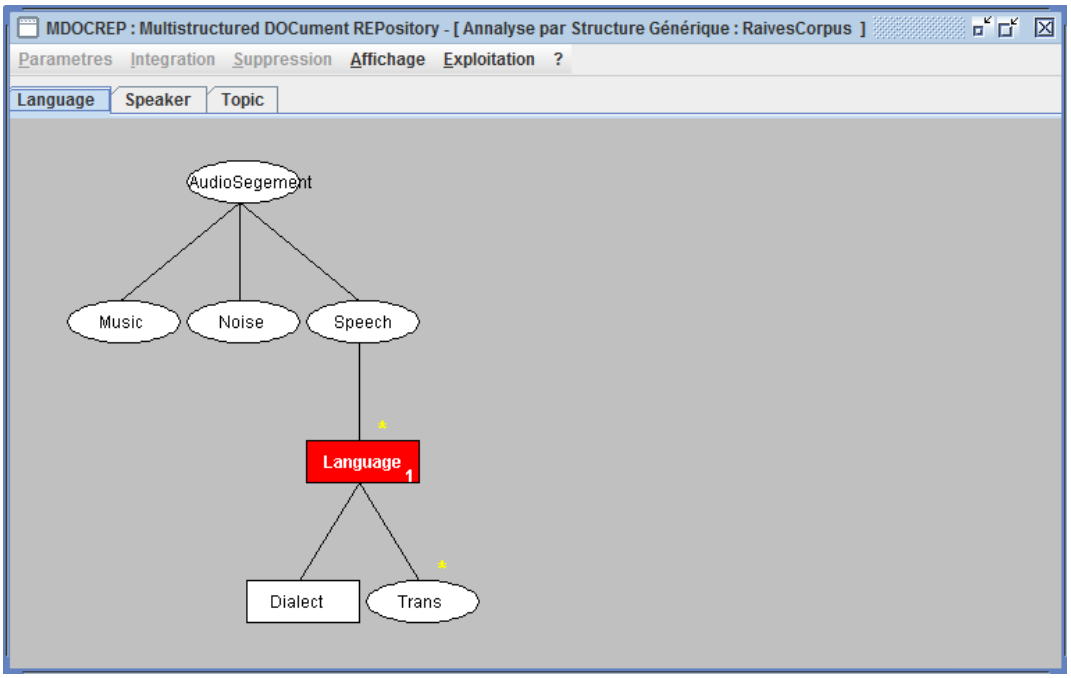


Figure V.16. Composantes d'analyse sélectionnées à partir de la vue générique « Language ».

A partir de la deuxième vue, nous sélectionnons la dimension « Name » (Cf. Figure V.17). Cette dimension est relative au nom du locuteur (« Name »).

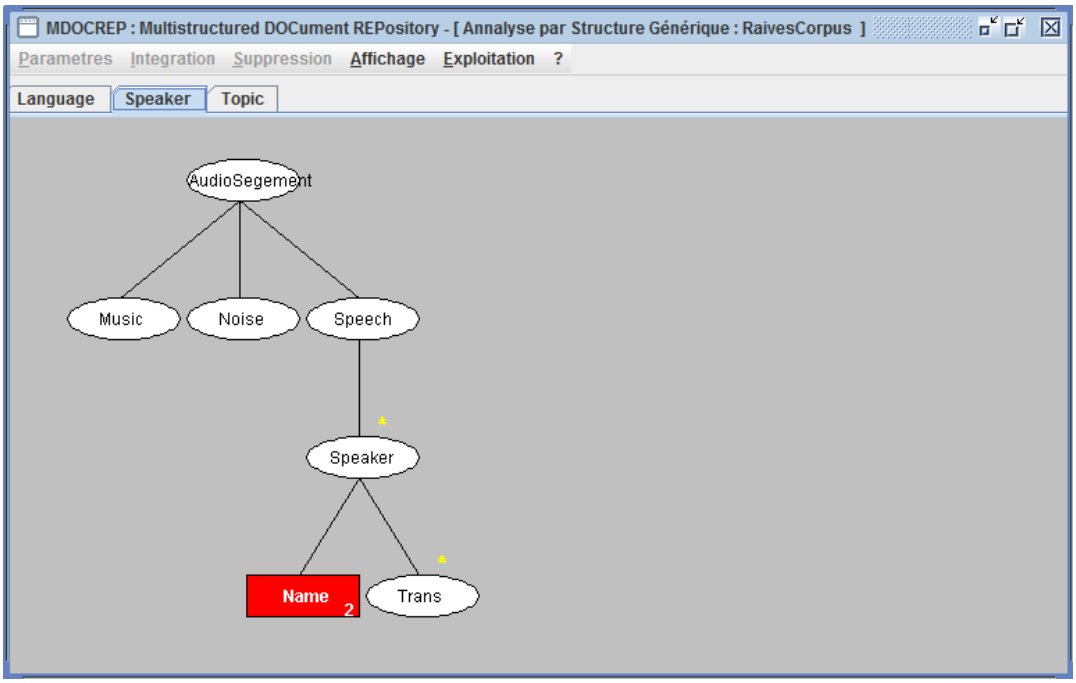


Figure V.17. Composantes d'analyse sélectionnées à partir de la vue générique « Speaker ».

A partir de la troisième vue, nous sélectionnons la dimension « Topic » et le fait « Trans » (Cf. Figure V.18). La dimension choisie est relative au thème abordé dans les segments audio et le fait concerne les différentes transcriptions.

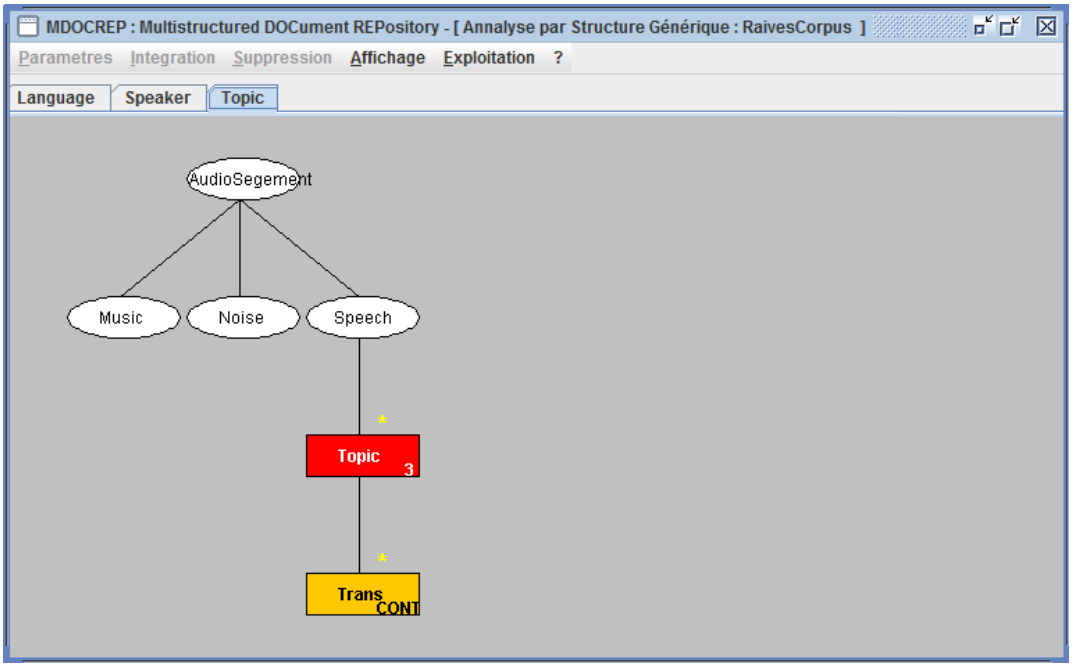


Figure V.18 Composantes d'analyse sélectionnées à partir de la vue générique « Topic ».

IV.2.3. Filtrage

Après la sélection des composantes d'analyses, le système doit générer les magasins correspondant à ces composantes. Ces magasins ne doivent analyser que les langues français « Fr » et espagnol « Esp ». A cette fin, l'application d'un premier filtre sur la dimension « Language » permet de sélectionner les valeurs correspondantes (Cf. Figure V.19). Par la suite, un deuxième filtre sur la dimension « Name » permet de ne prendre en compte que des locuteurs « Julien Coubet » et « Michel Drucker » (Cf. Figure V.20). Enfin, un troisième filtre sur la dimension « Topic » permet de ne sélectionner que les thèmes « Economy » et « Current Affairs » (Cf. Figure V.21).

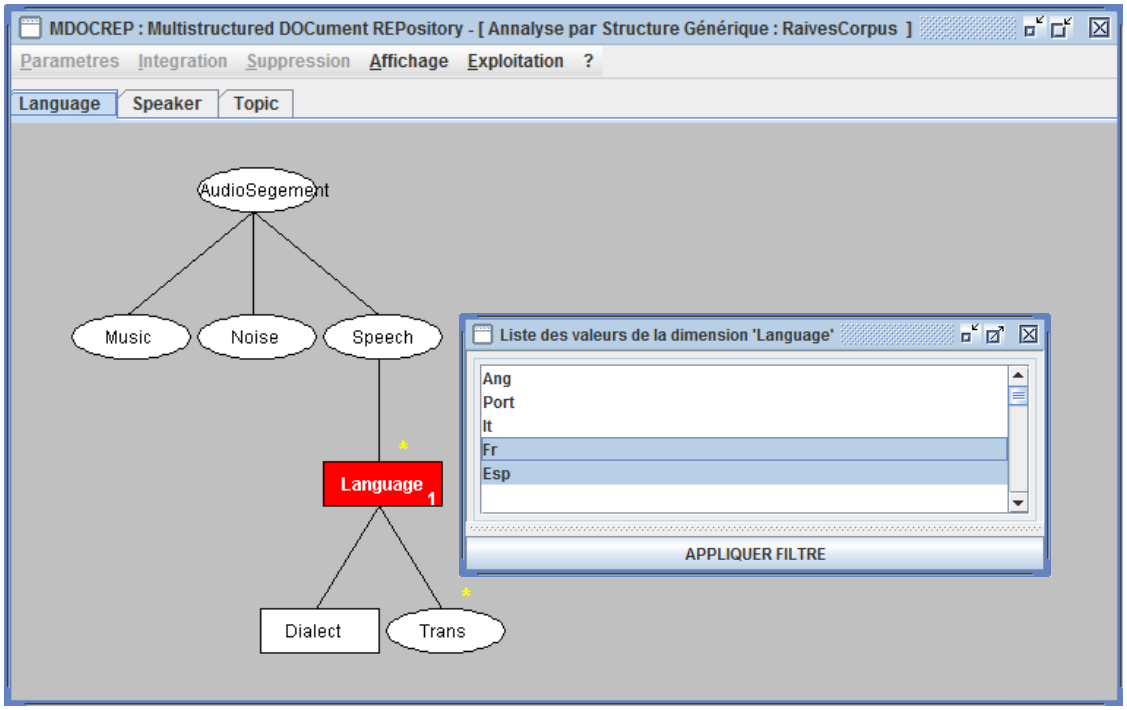


Figure V.19. Filtrage relatif à la dimension « Language ».

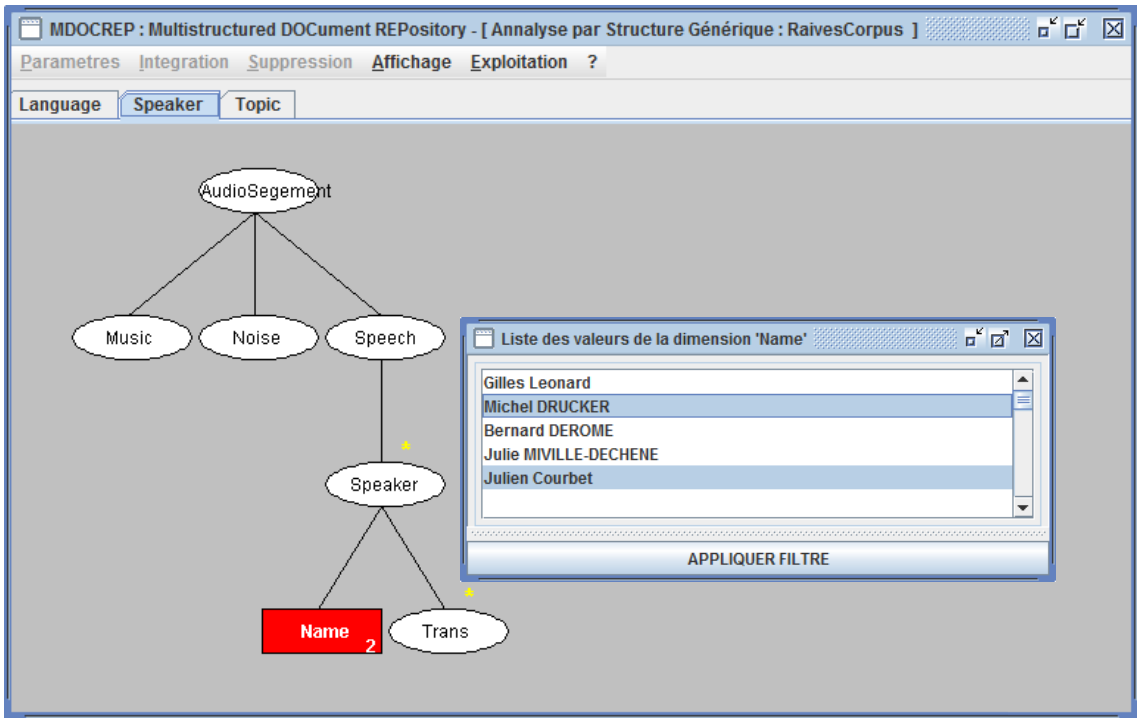


Figure V.20. Filtrage relatif à la dimension « Name ».

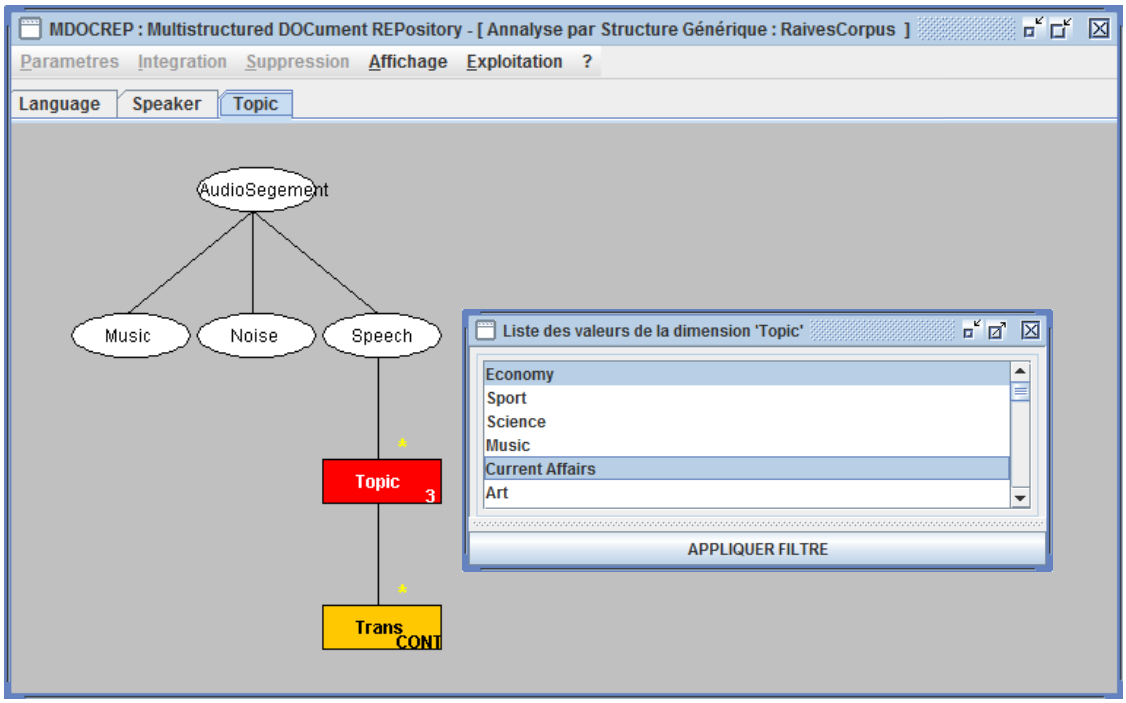


Figure V.21. Filtrage relatif à la dimension « Topic ».

IV.2.4. Résultat

Pour visualiser le résultat de l'analyse, nous activons les menus : *Exploitation* puis *Analyse* puis *Magasin* puis *Visualisation*. Ainsi, le système crée les vues selon la démarche décrite dans le Chapitre IV Section III.2 et affiche le résultat sous forme de table multidimensionnelle.

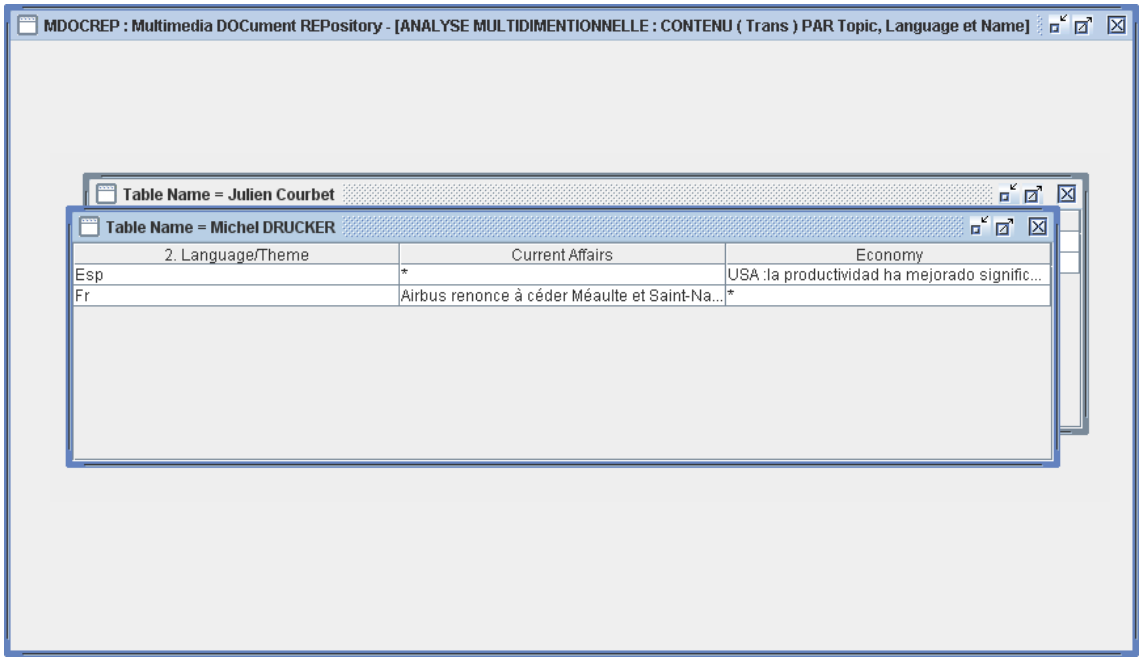


Figure V.22. Résultat de l'analyse sous forme de table multidimensionnelle.

V. Conclusion

Nous avons présenté dans ce chapitre le prototype MDOCREP. Ce prototype nous a permis de valider nos propositions d'intégration et d'exploitation des documents selon le modèle MVDM.

MDOCREP nous a permis d'évaluer et de valider la démarche de classification proposée dans cette thèse. Les résultats obtenus sont satisfaisants (moyenne de similarité des classés élevé, faible écart type…). Cependant, les expériences ont montré qu'il y a toujours des documents à reclasser et certaines classes qui doivent disparaitre. Ceci nous amène à penser qu'il faut ajouter une nouvelle phase à la démarche de classification proposée. Cette phase doit nous permettre d'augmenter la distance inter classes.

MDOCREP nous a également permis de valider la démarche de restitution multidimensionnelle proposée. Nous avons présenté en particulier un exemple d'analyse multidimensionnelle de documents audio. Cette analyse multidimensionnelle se base sur la structure générique des documents audio. Cette structure générique est composée de trois vues génériques qui nous ont servi pour le choix de différents paramètres d'analyse. Ceci nous a permis de montrer comment combiner des paramètres issus de plusieurs vues pour la restitution des documents.

VI. Bibliographie

Djemal, K., Soulé-Dupuy, C., et Vallès-Parlangeau, N. (2010a). "Multistructured documents: from modelling to multidimensional analyses." SciWatch Journal.

Parlangeau-Vallès, N., Farinas, J., Fohr, D., Illina, I., Magrin-Chagnolleau, I., Mella, O., Pinquier, J., Rouas, J. L., et Sénac, C. (2003). "Audio Indexing on the Web: A preliminary study of some audio descriptors." Proceedings of SCI.

Conclusion générale

I. Bilan et synthèse de nos propositions

Dans cette thèse, nous nous sommes intéressés à la multistructuralité des documents et à sa contribution dans l'optimisation de tout processus de recherche de documents. Nous avons présenté une approche de gestion de ces documents. Cette approche est basée sur trois axes : la modélisation, la classification et l'exploitation.

Dans un premier temps, nous avons présenté le modèle MVDM. Ce modèle comprend deux niveaux de description : spécifique et générique.

Le niveau spécifique modélise les propriétés et les caractéristiques de chaque document et de leurs structures. Ce modèle représente les entités d'un document sous forme de nœuds spécifiques. Les relations qui relient ces nœuds spécifiques sont également spécifiées de façon à traduire plusieurs types de liens entre deux entités. MVDM s'articule autour du concept de vue. Une vue telle qu'elle a été conçue traduit une organisation particulière du document selon une vision ou un contexte défini. Au travers de ce concept, le modèle MVDM permet de représenter plusieurs structures de même type (par exemple plusieurs structures logiques), mais aussi plusieurs structures de type différent (telle que par exemple une structure logique et une structure physique). L'originalité du modèle réside également dans la flexibilité de définition des vues. Ces vues peuvent être définies au niveau global du document comme elles peuvent être définies au niveau d'une entité permettant ainsi de décrire des structures multiples d'un document ainsi que des entités qui le composent.

Le niveau générique modélise les caractéristiques des classes de documents multistructurés selon le principe de vues génériques. La classification de documents repose sur ce niveau générique. Elle est traduite au travers de la classification des vues. Une classe de vues est alors considérée comme un ensemble homogène et cohérent d'un point de vue structurel et sémantique.

Dans un deuxième temps, nous avons présenté une démarche d'intégration de documents multistructurés selon le modèle MVDM. Dans un premier temps, un document est dématérialisé. Cette dématérialisation est basée sur l'extraction de la structure et du contenu de ce nouveau document. Ceci nous permet de générer une vue spécifique de ce document. Dans un deuxième temps, cette vue spécifique est rattachée à une vue générique.

Pour rattacher une vue spécifique à la vue générique adéquate, nous avons proposé une démarche de classification non supervisée des vues. Cette classification se base sur l'exploitation des paramètres structurels pour caractériser la ressemblance des documents. La similarité entre deux vues est jugée à partir d'une distance appelée distance structurelle. L'originalité de cette distance réside dans les différents paramètres qu'elle intègre. Cette distance repose sur la combinaison de trois pondérations : *une pondération structurelle* permettant de traduire l'organisation structurelle (hiérarchie et ordre) ; *une pondération d'adaptation* permettant d'évaluer le coût de modification (ajout ou suppression) d'un nœud d'une vue par rapport à l'autre. Plus la sous-arborescence liée au nœud à modifier est

importante plus le coût d'adaptation sera fort ; et *une pondération de représentativité* permettant de tenir compte du nombre de relations spécifiques rattachées à chaque relation générique. L'originalité de notre démarche de classification réside dans la spécification d'un représentant sous forme de graphe pour chaque classe. Un tel représentant permet de couvrir le maximum d'arborescences structurellement proches. L'originalité de notre démarche de classification réside également dans la conservation de la représentativité des classes qui permet de reclasser des vues spécifiques jugées mal classées soit parce que leurs classes d'origine ont subi plusieurs adaptations, soit parce que lors de l'intégration de cette vue spécifique la nouvelle classe n'était pas encore créée.

Enfin, nous avons proposé deux techniques d'exploitation des documents multistructurés : la recherche et la restitution multidimensionnelle. La recherche consiste à retrouver de données factuelles et des éléments répondant à un ou plusieurs critères relatifs à une ou plusieurs vues de documents. L'analyse multidimensionnelle consiste à analyser les informations documentaires de la base selon des axes d'analyse (dimensions) et un sujet (fait) non prédéfinis. L'originalité de ces deux techniques réside dans, d'une part, l'ajout de paramètres d'exploitation (des nœuds) issus des structures multiples définies sur le document et d'autre part, la gestion de chevauchement entre les nœuds définis sur un même contenu. Ceci nous permet de mieux localiser l'information et par conséquent d'avoir des résultats plus pertinents et plus précis.

II. Perspectives de recherche

Dans le travail que nous avons présenté, il reste encore quelques problèmes à résoudre.

Lors de l'intégration d'un document, nous avons supposé que l'utilisateur se charge de décider s'il s'agit d'une nouvelle vue spécifique d'un document existant ou d'une vue spécifique d'un nouveau document. Par conséquent, l'utilisateur décide de l'agrégation d'une vue spécifique dans une structure spécifique.

L'agrégation des vues génériques dans leur structure générique correspondante est également décidée par l'utilisateur. En effet, lorsque le classifieur décide la création d'une nouvelle vue générique, l'utilisateur doit décider s'il faut créer une nouvelle structure générique ou agréger cette vue générique dans une structure générique existante.

Bien que les résultats des expérimentations présentées dans le chapitre V soient encourageants, ces résultats ont montré que, après la classification, il existe des vues spécifiques relatives à des documents qui peuvent être « mieux » classées. En effet, ces vues deviennent plus proches d'autres classes.

Nos perspectives à cours terme visent naturellement à combler ces limites.

Définir une démarche d'agrégation des vues spécifiques dans leur structure spécifique revient à définir une méthodologie de construction de documents multistructurés. Si l'on considère qu'un document multistructuré est construit à partir de n

documents « monostructurés », il est envisageable de comparer le contenu ainsi que les structures de ces n documents afin de construire un document multistructuré.

De la même façon que l'on opère une classification des vues spécifiques, il est intéressant de définir une démarche de classification des structures spécifiques. La classification de ces structures doit prendre en compte les vues spécifiques agrégées dans la structure spécifique à classer et doit assurer l'agrégation des vues génériques représentant les vues spécifiques agrégées dans la structure spécifique.

L'amélioration des résultats de classification passe par la proposition d'une phase supplémentaire à la démarche de classification présentée dans le chapitre IV. Cette phase doit permettre de redistribuer certaines vues spécifiques et éventuellement de fusionner les classes qui deviennent proches. Si la phase de conservation de la représentativité permet de réduire l'inertie intra classes, la nouvelle phase doit permettre d'augmenter la distance entre classes appelée encore inertie inter classes.

L'application des méthodes de classification basées sur l'estimation de la probabilité d'erreur peut être une alternative pour réduire le nombre des vues spécifiques à reclasser. Les méthodes les plus adoptées sont les arbres de décisions, les chaines de Markov cachées, les réseaux de neurones. Ces méthodes telles qu'elles sont décrites dans (Duda et al. 2000) se basent sur la théorie « bayésienne ». Cette théorie revient à calculer des probabilités conditionnelles d'appartenance à une classe sur un ensemble desparamètres discriminants permettant la construction des classes soit les unes par rapport aux autres, soit par rapport à la probabilité d'erreur de classification.

Nos travaux peuvent s'orienter vers d'autres sous-domaines de l'ingénierie documentaire. A moyen et long terme, nous envisageons d'exploiter le modèle MVDM afin d'assurer d'une part, la gestion des versions et d'autre part l'adaptation des documents selon un contexte particulier.

Les documents, une fois écrite sont rarement figés dans le temps. Ils peuvent être marqués par des évolutions de contenu ou de structures constituant ainsi plusieurs versions d'un même document (Djemal et al. 2009b). Ces versions peuvent être considérées comme des vues différentes sur un même document. Si le modèle MVDM permet le partage de nœuds et de contenu entre les vues relatives à ces versions, il faut l'étendre pour prendre en compte des caractéristiques à l'évolution des versions telle que l'évolution du contenu lui-même.

L'apparition des systèmes d'information pervasifs a imposé le développement de nouveaux outils de communication. Ces outils doivent fournir des informations adaptées selon des contextes variables. Dans le cas où ces informations sont traduites par des documents, il faut en fournir une version adaptée à chaque situation contextuelle. Si les vues spécifiques du modèle MVDM peuvent fournir des versions adaptées à des situations contextuelles prédéfinies (adaptation statique), les vues génériques permettent de générer de nouvelles vues spécifiques dédiées à des nouvelles situations contextuelles (adaptation dynamique). Nous envisageons de définir les mécanismes nécessaires afin d'assurer

l'adaptation statique et dynamique des documents et proposer les architectures qui permettront d'implanter de tels mécanismes.

Bibliographie générale

A

Abascal, R., Beigbeder, M., Benel, A., Calabretto, S., Chabbat, B., Champin, P. A., Chatti, N., Jouve, D., Prie, Y., et Rumpler, B. (2003). « Modéliser la structuration multiple des documents ». *H2PTM*, Hermès, Paris, France, 253-258.

Abascal, R., Beigbeder, M., Bénel, A., Calabretto, S., Chabbat, B., Champin, P. A., Chatti, N., Jouve, D., Prié, Y., et Rumpler, B. (2004). « Documents à structures multiples ». *SETIT 2004*.

Adler, S., Berglund, A., Deach, S., Graham, T., Grosso, P., et Gutentag, E. (2001). "W3C Recommendation Extensible Stylesheet Language (XSL) Version 1.0 ." <http://www.w3.org/TR/2001/REC-xsl-20011015/>.

Aguiar, F., et Beigbeder, M. (2004). « Construction et utilisation de contextes autour des nœuds d'un hypertexte pour la recherche d'information. » *Revue Document numérique*, (2004/3), 71-82.

Allen, J. F. (1991). "Time and time again: The many ways to represent time." *International Journal of Intelligent Systems*, 6(4).

Allen, J. F. (1983). "Maintaining knowledge about temporal intervals." Communication ACM, 26(11), 837-843.

Apers, P. M. G., Blanken, H. M., et Houtsma, M. A. (1997). *Multimedia databases in perspective*. Springer-Verlag New York, Inc. Secaucus, NJ, USA.

Arenas, M., et Libkin, L. (2002). "A Normal Form for XML Documents."

Auffret, G., Carrive, J., Chevet, O., Dechilly, T., Ronfard, R., et Bachimont, B. (1999). "Audiovisual-based hypermedia authoring: using structured representations for efficient access to AV documents." *Proceedings of the tenth ACM Conference on Hypertext and hypermedia: returning to our diverse roots: returning to our diverse roots*, ACM New York, NY, USA, 169-178.

B

Bachimont, B. (1998). « Bibliothèques numériques audiovisuelles : Des enjeux scientifiques et techniques. » *Revue Document numérique*, 2(3), 219-242.

Bachimont, B., et Crozat, S. (2004). « Instrumentation numérique des documents: pour une séparation fonds/forme. » *Information-Interaction-Intelligence I3*, 4(1), 95-104.

Barnard, D., Burnard, L., Gaspart, J., Price, L., Sperberg-McQueen, C., et Varile, G. (1995). "Hierarchical encoding of text: Technical problems and SGML solutions." *Computers and the Humanities*, 29(3), 211-231.

Bechhofer, S., Carr, L., Goble, C., Kampa, S., et Miles-Board, T. (2002). "The semantics of semantic annotation." *Lecture notes in computer science*, 1152-1167.

Beigbeder, M. (2004). « Les temps du document et la recherche d'information. » *Document numérique*, (2004/4), 55–64.

Bertino, E., Guerrini, G., Mesiti, M., Rivara, I., et Tavella, C. (2002). "Measuring the Structural Similarity among XML Documents and DTDs." *null*, http://www.disi.unig.

Bird, S., et Liberman, M. (2001). "A formal framework for linguistic annotation." *Speech Communication*, 33(1-2), 23-60.

Blasselle, B. (1998). *Histoire du livre*. Gallimard.

Bray, T., Paoli, J., Sperberg-McQueen, C., et Maler, E. (2000). "W3C Recommendation Extensible Markup Language (XML) 1.0 (Second Edition)." <http://www.w3.org/TR/2000/REC-xml-20001006>.

Breiman, L. (1998). *Classification and Regression Trees*. Chapman & Hall/CRC.

Brickley, D., et Guha, R. V. (2004). "Rdf vocabulary description language 1.0: Rdf schema." *World Wide Web Consortium recommendation* .

Bringay, S., Barry, C., et Charlet, J. (2004). « Les documents et les annotations du dossier patient hospitalier. » *Revue I3 : Information-Interaction-Intelligence*, 4(1), 191-211.

Bruno, E., et Murisasco, E. (2006). "MSXD: A Model and a Schema for Concurrent Structures Defined over the Same Textual Data." *Database and Expert Systems Applications*, 172-181.

Bulterman, D., Jasen, J., Cesar, P., Mullender, S., Hyche, E., DeMeglio, M., Quint, J., et Kawamura, H. (2008). "Synchronized Multimedia Integration Language (SMIL 3.0) W3C Recommendation."

Burnard, L. (1992). "The Text Encoding Initiative: A progress report." *New Directions in English Language Corpora: Methodology, Results, Software Developments*, 97.

C

Cabanac, G., Chevalier, M., Chrisment C. et Julien, C. (2010). "Social validation of collective annotations: Definition and experiment". Dans : Journal of American Society for Information Science and Technology, Wiley, Vol. 61 N. 2, 271-287.

Cabanac, G., Chevalier, M., Chrisment C. et Julien, C. (2009). « Activités documentaires des usagers au sein de l'organisation : amélioration par la pratique d'annotation collective ». Dans : Ingénierie des Systèmes d'Information, Hermès Science Publications, Numéro spécial Prise en compte des utilisateurs dans les SI, Vol. 14, N. 3, 97-117.

Charhad, M. (2005). "Modèles de Documents Vidéo basés sur le Formalisme des Graphes Conceptuels pour l'Indexation et la Recherche par le Contenu Sémantique." Thèse de doctorat, UNIVERSITÉ JOSEPH FOURIER.

Charhad, M., et Quénot, G. (2004). "Semantic video content indexing and retrieval using conceptual graphs." *Information and Communication Technologies: From Theory to Applications, 2004. Proceedings. 2004 International Conference on*, 399-400.

Chatti, N. (2006). "Documents multi-structurés: De la modélisation vers l'exploitation." Thèse de doctorat, L'institut National Des Sciences Appliquées De Lyon.

Chatti, N., et Calabretto, S. (2007). "Adaptation de XML et XQuery pour la représentation et l'interrogation des documents multi-structurés." Saint-Etienne, 109-124.

Chatti, N., Calabretto, S., et Pinon, J. M. (2004). « Vers un environnement de gestion de documents à structures multiples ».

Chatti, N., Calabretto, S., Pinon, J. M., et Kaouk, S. (2007). "MultiX: an XML-based formalism to encode multi-structured documents." *Proceedings of Extreme Markup Languages 2007*, 2007.

Clark, J., et Murata, M. (2001). "RELAX NG Specification. OASIS Committee Specification." *WWW: http://www.relaxng.org/spec-20011203.html* .

Cobena, G., Abiteboul, S., et Marian, A. (2002a). *XyDiff, tools for detecting changes in XML documents*.

Cobena, G., Abiteboul, S., et Marian, A. (2002b). "Detecting Changes in XML Documents." *PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON DATA ENGINEERING*, IEEE Computer Society Press; 1998, 41-52.

Costa, G., Manco, G., Ortale, R., et Tagarelli, A. (2004). "A Tree-Based Approach to Clustering XML Documents by Structure." *LECTURE NOTES IN COMPUTER SCIENCE*, 137-148.

D

Dalamagas, T., Cheng, T., Winkel, K. J., et Sellis, T. (2006). "A methodology for clustering XML documents by structure." *Information Systems*, 31(3), 187-228.

Debarbieux, D. (2005). "Modélisation et requêtes des documents semi-structurés : exploitation de la structure de graphe." Thèse de doctorat, .

Dekhtyar, A., et Iacob, I. (2003). "A Framework for Management of Concurrent XML Markup." *Conceptual Modeling for Novel Application Domains*, <http://www.springerlink.com/content/apx60p65eubpm8la> (Mai. 26, 2009).

Dekhtyar, A., et Iacob, I. E. (2005). "A framework for management of concurrent XML markup." *Data Knowl. Eng.*, 52(2), 185-208.

Dekhtyar, A., Iacob, I. E., et Methuku, S. (2005). "Searching Multi-hierarchical XML Documents: The Case of Fragmentation." *Database and Expert Systems Applications*, 576-585.

Denoyer, L., et Gallinari, P. (2004). "Bayesian network model for semi-structured document classification." *Information Processing and Management*, 40(5), 807-827.

DeRose, S. (2004). "Markup overlap: A review and a horse." *Extreme Markup Languages*, Citeseer.

DeRose, S., Maler, E., et Ron, D. (2001). "W3C XML Pointer Language (XPointer) Version 1.0." <http://www.w3.org/TR/WD-xptr>.

Diligenti, M., Gori, M., Maggini, M., et Scarselli, F. (2001). "Classification of HTML documents by Hidden Tree-Markov Models." *Proceedings of ICDAR*, 849–853.

Djemal, K. (2007a). "A Multi-Views Repository for Multi-Structured Documents." ***9th International Conference on Enterprise Information Systems (ICEIS)*****, 544-548.**

Djemal, K. (2007b). « Vers une exploitation de documents multi-structurés ». ***Congrès de l'INFormatique des Organisations et Systèmes d'Information et de Décision (INFORSID'07)*****, Perros-Guirec, 37-52.**

Djemal, K., Soule-Dupuy, C., et Valles-Parlangeau, N. (2008a). "Modeling and Exploitation of Multistructured Documents." ***Information and Communication Technologies: From Theory to Applications, 2008. ICTTA 2008. 3rd International Conference on*****, 1-6.**

Djemal, K., Soule-Dupuy, C., et Valles-Parlangeau, N. (2008b). "Formal modeling of multistructured documents." ***Second International Conference on Research Challenges in Information Science, 2008. RCIS 2008*****, Marrakech, 227-236.**

Djemal, K., Mbarki, M., et Vallès-Parlangeau, N. (2007). « Une approche multi-vues pour la gestion des documents multistructurés. » ***Document numérique, Entreposage de documents et données semi-structurées*****, 10(2), 37–61.**

Djemal, K., Soulé-Dupuy, C., et Vallès-Parlangeau, N. (2009a). « Analyse multidimensionnelle des documents multistructurés. » ***Colloque Veille Stratégique Scientifique et Technologique (VSST 2009)*****, Nancy.**

Djemal, K., Soulé-Dupuy, C., et Vallès-Parlangeau, N. (2009b). "Management of Document Multistructurality: Case of Document Versions" ***Third International Conference on Research Challenges in Information Science, 2009. RCIS 2009.*** **Fes, 359-366.**

Djemal, K., Soulé-Dupuy, C., et Vallès-Parlangeau, N. (2010a). "Multistructured documents: from modelling to multidimensional analyses." ***SciWatch Journal.***

Djemal, K., Soulé-Dupuy, C., et Vallès-Parlangeau, N. (2010b). « Classification de documents : calcul d'une distance structurelle. », Extraction et la Gestion des Connaissances EGC2010, Hammamet, Tunisie, Du 26/01/2010-29/01/2010, 609-614.

Doucet, A., et Ahonen-Myka, H. (2002). "Naive Clustering of a Large XML Document Collection." *Proceedings of the First Workshop of the Initiative for the Evaluation of XML Retrieval (INEX)*, 81–87.

Duda, R.O., Hart, P.E., et Stork, D.G., (2000). Pattern Classification. Second Edition, Wiley- Interscience Publication.

Dupoirier, G. (1995). *Technologie de la GED: techniques et management des documents électroniques*. Hermès.

Durusau, P., et DeRose, S. J. (2003). "OSIS: A Users' Guide to the Open Scripture Information Standard." *Bible Technologies Group*.

Durusau, P., et O'Donnell, M. B. (2002). "Concurrent markup for XML documents." *Proc. XML Europe*.

Durusau, P., et O'Donnell, M. B. (2004). "Tabling the overlap discussion." *Extreme Markup Languages 2004*.

E, F

Egenhofer, M. J. (1994). “Spatial SQL: A query and presentation language.” *IEEE Transactions on knowledge and data engineering*, 6(1), 86–95.

Egenhofer, M. J., Clementini, E., et Sharma, J. (1994). “Modelling topological spatial relations: Strategies for query processing.” *Computers and Graphics*, 18(6), 815–822.

Egenhofer, M. J., et Franzosa, R. D. (1991). “Point-Set Topological Spatial Relations.” *The International journal of geographical information science and systems*, 5(2), 161-174.

Fernandez, M., Malhotra, A., Marsh, J., Nagy, M., et Walsh, N. (2002). “XQuery 1.0 and XPath 2.0 data model.” *W3C Working Draft*, 15.

Fernandez, M., Malhotra, A., Marsh, J., Nagy, M., et Walsh, N. (2007). “XQuery 1.0 and XPath 2.0 Data Model (XDM).”

Fourel, F. (1998). “Modélisation, indexation et recherche de documents structurés.” Thèse de doctorat, Université Joseph Fourier, Grenoble.

Fourel, F., et Mulhem, P. (1996). “Modelling multimedia structured documents: a retrieval oriented approach.” *Proceedings of the 7th International Workshop on Database and Expert Systems Applications*, IEEE Computer Society, 179-184.

Frank, A. U. (1992). “Qualitative spatial reasoning about distances and directions in geographic space.” *Journal of Visual Languages and Computing*, 3, 343–343.

Frank, A. U. (1996). “Qualitative spatial reasoning: Cardinal directions as an example.” *The International Journal of Geographical Information Science And Systems*, 10(3), 269–290.

Fuhr, N., et Großjohann, K. (2001). “XIRQL: A query language for information retrieval in XML documents.” *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, ACM New York, NY, USA, 172-180.

G, H

Goldfarb, C. F. (1981). “A generalized approach to document markup.” *ACM SIGPLAN Notices*, 16(6), 68-73.

Goldfarb, C. F. (1990). *The SGML handbook*. Oxford University Press, Inc., 664.

Golfarelli, M., Rizzi, S., et Vrdoljak, B. (2001). “Data warehouse design from XML sources.” *Proceedings of the 4th ACM international workshop on Data warehousing and OLAP*, ACM Press New York, NY, USA, 40-47.

Gyssens, M., et Lakshmanan, L. V. S. (1997). “A Foundation for Multi-Dimensional Databases.” *Proceedings of the international conference on very large data bases*, Institute of electrical & electronics engineers (IEEE), 106-115.

Hernandez, N. (2005). “Ontologies pour l'aide à l'exploration d'une collection de documents.” *Ingénierie des Systèmes d'Information*, 10(1), 11–31.

Hilbert, M., Schonefeld, O., et Witt, A. (2005). “Making CONCUR work.” *Extreme Markup Languages*.

Hors, A. L., Byrne, S., Champion, M., Nicol, G., Robie, J., Le Hégaret, P., et Wood, L. (2004). “Document object model (DOM) level 3 core specification.” *W3C Recommendation. http-J/www. xv3. org/TR/DOMrLevel-3-Core*.

Huitfeldt, C. (1993). “MECS-A Multi-Element Code System.” *ACH-ALLC*, 16-19.

Huitfeldt, C., et Sperberg-McQueen, C. M. (2001). “TexMECS: an experimental markup meta-language for complex documents.”

Huitfeldt, C., et Sperberg-McQueen, C. M. (2004). *Markup Languages for Complex Documents–an Interim Project Report*.

I, J

Iacob, I. E., Dekhtyar, A., et Kaneko, K. (2004). “Parsing concurrent XML.” *Proceedings of the 6th annual ACM international workshop on Web information and data management*, ACM, Washington DC, USA, 23-30.

ISO-10179. (1996). “DSSSL - Document Style Semantics and Specification Language. ISO/IEC 10179:1996.”

ISO-10744. (1997). “HyTime. ISO 10744:1997 -- Hypermedia/Time-based Structuring Language (HyTime), 2nd Edition.” <http://xml.coverpages.org/hytime.html> (Sep. 1, 2009).

ISO-5127. (2001). “Information and documentation - Vocabulary.”

ISO-8879. (1986). “SSGM - Information processing, Text and Office Systems, Language, International Organization for Standardization (ISO), Geneva, first edition edition.” 15.

Jagadish, H. V., Lakshmanan, L. V. S., Scannapieco, M., di Roma, U., Sapienza, L., Srivastava, D., et Wiwatwattana, N. (2004). “Colorful xml: One hierarchy isn’t enough.” ACM New York, NY, USA, 251-262.

Jiang, T., Wang, L., et Zhang, K. (1995). “Alignment of trees—an alternative to tree edit.” *Theoretical Computer Science*, 143(1), 137-148.

Joachims, T. (1997). *Text Categorization with Support Vector Machines: Learning with Many Relevant Features*. Springer.

Julien, C. (1988). “Bases d'informations généralisées : contribution à l'étude des mécanismes de consultation d'objets multimedia.” IRIT/CERFIA Université Paul Sabatier .

K, L

Khrouf, K., et Soulé-Dupuy, C. (2004). “A Textual Warehouse Approach: A Web Data Repository.” *Intelligent Agents for Data Mining and Information Retrieval*, Idea Group Publishing, 101-124.

Khrouf, K. (2004). “Entrepôts de documents : De l'alimentation à l'exploitation.” Thèse de doctorat, Université Paul Sabatier.

Kutty, S., Tran, T., Nayak, R., et Li, Y. (2008). “Clustering XML Documents Using Closed Frequent Subtrees: A Structural Similarity Approach.” *Lecture Notes In Computer Science*, 183-194.

Lalanne, D., et Ingold, R. (2004). “Documents statiques et multimodalité: L'alignement temporel pour structurer des archives multimédias de réunions.” *Document numérique*, 8(4), 65–89.

Lassila, O., et Swick, R. (2000). “Resource Description Framework.” *IEEE Intelligent Systems*, 15(6), 67-69.

Le Maitre, J. (2006). “Describing multistructured XML documents by means of delay nodes.” *Proceedings of the 2006 ACM symposium on Document engineering*, ACM New York, NY, USA, 155-164.

Lefèvre, S., et Sèdes, F. (2004). “Indexation de séquences vidéo: Indices liés au temps.” *Document numérique*, 8(4), 41–48.

Lopez-Ornelas, E. (2005). “Segmentation d'images satellitaires à haute résolution et interaction avec l'information géographique. Application à l'extraction de connaissances..” Thèse de doctorat, Université Paul Sabatier.

M, N

MacQueen, J. B. (1966). “Some Methods for Classification and Analysis of Multivariate Observations.” Western Management Science Inst Univ of CALIFORNIA LOS ANGELES, 281-297.

Manjunath, B. S., Salembier, P., et Sikora, T. (2002). *Introduction to MPEG-7: multimedia content description interface*. Wiley.

Marcoux, Y. (1994). “Les formats normalisés de documents électroniques.” *ICO. Intelligence artificielle et sciences cognitives au Québec*, 6(1-2), 56-65.

Marshall, C. C. (1998). “Toward an ecology of hypertext annotation.” *Proceedings of the ninth ACM conference on Hypertext and hypermedia: links, objects, time and space---structure in hypermedia systems: links, objects, time and space---structure in hypermedia systems*, ACM New York, NY, USA, 40-49.

Martinez, J. M. (2002). “Standards-MPEG-7 overview of MPEG-7 description tools, part 2.” *IEEE Multimedia*, 9(3), 83-93.

Mbarki, M. (2008). “Gestion de l’hétérogénéité documentaire : le cas d’un Entrepôt de documents multimédia.” Thèse de doctorat, Université Paul Sabatier.

McKelvie, D., Brew, C., et Thompson, H. S. (1999). “Using SGML as a Basis for Data-Intensive Natural Language Processing.” *COMPUTERS AND THE HUMANITIES*, 31, 367-388.

Mechkour, M. (1995). “EMIR²: An Extended Model for Image Representation and Retrieval.” *DEXA*, 395-404.

Metzger, J. P., et Lallich-Boidin, G. (2004). “Temps et documents numériques.” *Document numérique*, (2004/4), 11–21.

Michard, A. (1998). *XML, langage et applications*. Edition Eyrolles.

Miller, G. A. (1995). “WordNet: a lexical database for English.” *Communications of the ACM*, 38(11), 39-41.

Murata, M. (1999). “Hedge automata: a formal model for XML schemata.” <http://www.xml.gr.jp/relax/hedge_nice.html>.

Nanard, J. (2004). “Formalismes de manipulation du temps par l'auteur dans les documents multimédias.” *Document numérique*, (2004/4), 23–39.

Navarro, G., et Baeza-Yates, R. (1997). “Proximal nodes: a model to query document databases by content and structure.” *ACM Transactions on Information Systems (TOIS)*, 15(4), 400-435.

NISO. (2004). “Understanding Metadata.” *National Information Standards Organisation (NISO)*, <http://www.niso.org/publications/press/UnderstandingMetadata.pdf>.

P, R

Papadias, D., et Theodoridis , Y. (1997). “Spatial relations, minimum bounding rectangles, and spatial data structures.” *International Journal of Geographical Information Science*, 11(2), 111–138.

Parlangeau-Vallès, N., Farinas, J., Fohr, D., Illina, I., Magrin-Chagnolleau, I., Mella, O., Pinquier, J., Rouas, J. L., et Sénac, C. (2003). “Audio Indexing on the Web: A preliminary study of some audio descriptors.” *Proceedings of SCI*.

Pédauque, R. T. (2003). “Document : forme, signe et médium, les re-formulations du numérique.” Document de travail STIC-CNRS, RTP 33, Documents et contenu : creation, indexation, navigation.

Pemberton, S., Austin, D., Axelsson, J., Çelik, T., Dominiak, D., et Elenbaas, H. (2000). “XHTML™ 1.0 The Extensible HyperText Markup Language (Second Edition).” <http://www.w3.org/TR/xhtml1/>.

Piwowarski, B., Denoyer, L., et Gallinari, P. (2002). “Un modèle pour la recherche d’information sur des documents structurés.” *Actes des 6 èmes Journées internationales d’Analyse statistique de Données Textuelles (JADT’02)*, 605–616.

Pokorny, J. (2001). “Modelling stars using XML.” *Proceedings of the 4th ACM international workshop on Data warehousing and OLAP*, ACM Press New York, NY, USA, 24-31.

Poullet, L. (1997). “Formaliser la sémantique des documents: Un modèle unificateur.” INFORSID, Toulouse-France, 339-352.

Poullet, L., Pinon, J. M., et Calabretto, S. (1997). “Semantic structuring of documents.” *Proceedings of the Third Basque International Workshop on Information Technology, BIWIT*, 118–124.

Prié, Y., et Garlatti, S. (2004). “Méta-données et annotations dans le Web sémantique.” *Web Sémantique Revue Hors-Série I*, 3, 1-24.

Raggett, D., Le Hors, A., et Jacobs, I. (1999). “World Wide Web Consortium - HTML 4.01 Specification.” <http://www.w3.org/TR/1999/REC-html401-19991224/>.

Ribière, M., Dieng-Kuntz, R., et Sophia-Antipolis, I. (2002). “AViewpoint Model for Cooperative Building of an Ontology.” U. Priss, D.

Roisin, C. (1999). “Documents structurés multimédia.” Habilitation à diriger des recherches, Institut National Polytechnique de Grenoble.

Romany, L., et Bonhomme, P. (2000). “Parallel Alignment of Structured Documents.” *Parallel Text Processing*, Kluwer Academic Publishers, Dordrecht, Boston, London, 201-218.

Roxin, I., et Mercier, D. (2004). *Multimédia: les fondamentaux. Introduction à la représentation numérique* . Vuibert.

RTF, U. (1999). “OMG Unified Modeling Language Specification, Version 1.3, UML RTF proposed final revision.” *OMG, June*, 4(83), 486.

S, T

Salazar, F., et Valero, F. (1995). *Analyse automatique de documents vidéo*. Université Paul Sabatier, Toulouse.

Saleem, K. (2008). “schema matching and integration in large scale snario.” Université Montpellier II - Sciences et Techniques du Languedoc.

Schonefeld, O. (2008). “A Simple API for XCONCUR Processing concurrent markup using an event-centric API.”

Schütze, H., Hull, D. A., et Pedersen, J. O. (1995). “A comparison of classifiers and document representations for the routing problem.” *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, ACM New York, NY, USA, 229-237.

Shasha, D., et Zhang, K. (1997). “Approximate Tree Pattern Matching.” *Pattern Matching Algorithms*, Oxford University Press, 341–371.

Soulé-Dupuy, C. (2001). “Bases d'informations textuelles : des modèles aux applications.” Habilitation à diriger des recherches, Université Paul Sabatier.

Sperberg-McQueen, C. M., et Burnard, L. (2007). *TEI P5: Guidelines for Electronic Text Encoding and Interchange*. Oxford, Providence, Charlottesville, and Bergen.

Sperberg-McQueen, C. M., et Huitfeldt, C. (2000). “GODDAG: A Data Structure for Overlapping Hierarchies.” *LECTURE NOTES IN COMPUTER SCIENCE*, 139-160.

Tannier, X. (2006). “Extraction et recherche d'information en langage naturel dans les documents semi-structurés.” Thèse de doctorat, Ecole Nationale Supérieure des Mines de Saint-Etienne.

Tekli, J., Chbeir, R., et Yetongnon, K. (2007). “Structural Similarity Evaluation Between XML Documents and DTDs.” *Web Information Systems Engineering – WISE 2007*, 196-211.

Tennison, J., et Piez, W. (2002). “The Layered Markup and Annotation Language (LMNL).” *Extreme Markup*, Montreal.

Tennison, J., Piez, W., et Nicol , G. T. (2002). “LMNL: the Layered Markup and Annotation Language.” *LMNL*, <http://lmnl.net/index.html>.

Termier, A., Rousset, M. C., et Sebag, M. (2002). “TreeFinder: a First Step towards XML Data Mining.” *Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'02)*, IEEE Computer Society Washington, DC, USA, 450.

Tseng, F. S. C., et Chou, A. Y. H. (2006). “The concept of document warehousing for multi-dimensional modeling of textual-based business intelligence.” *Decision Support Systems*, 42(2), 727-744.

Tummarello, G., Morbidoni, C., et Pierazzo, E. (2005). “Toward textual encoding based on RDF.” *Proceedings ELPUB'2005*.

V, W

Vazirgiannis, M., Theodoridis, Y., et Sellis, T. (1998). “Spatio-temporal composition and indexing for large multimedia applications.” *Multimedia Syst.*, 6(4), 284-298.

Vercoustre, A. M., Fegas, M., Lechevallier, Y., Despeyroux, T., et Rocquencourt, I. (2006). “Classification de documents XML à partir d'une représentation linéaire des arbres de ces documents.” Paris, France, 433–444.

Vilain, M., Kautz, H., et Beek, P. (1986). “Constraint propagation algorithms for temporal reasoning.” *READINGS IN QUALITATIVE REASONING ABOUT PHYSICAL SYSTEMS*, 377-382.

Vu, T. H., Denoyer, L., et Gallinari, P. (2003). “Un modèle statistique pour la classification de documents structurés.” *Journees francophones d’Extraction et de Gestion des Connaissances (EGC 2003), Lyon, France, Jan*.

Wagner, R. A., et Fischer, M. J. (1974). “The String-to-String Correction Problem.” *J. ACM*, 21(1), 168-173.

Wang, Y., DeWitt, D. J., et Cai, J. Y. (2003). “X-Diff: an effective change detection algorithm for XML documents.” *Data Engineering, 2003. Proceedings. 19th International Conference on*, 519-530.

Witt, A. (2004). “Multiple Hierarchies: New Aspects of an Old Solution.” *Proceedings of Extreme Markup Languages*.

Witt, A., Goecke, D., Sasaki, F., et Lungen, H. (2005). “Unification of XML Documents with Concurrent Markup.” *Literary and Linguistic Computing*, 20(1), 103-116.

Wiwatwattana, N., Jagadish, H. V., Lakshmanan, L. V. S., et Srivastava, D. (2006). “Making Designer Schemas with Colors.” *Proceedings of the 22nd International Conference on Data Engineering*, IEEE Computer Society Washington, DC, USA, 15.

Y, Z

Yi, J., et Sundaresan, N. (2000). “A classifier for semi-structured documents.” *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM New York, NY, USA, 340-344.

Zacklad, M., Lewkowicz, M., Boujut, J. F., Darses, F., et Détienne, F. (2003). “Formes et gestion des annotations numériques collectives en ingénierie collaborative.” *Actes de la conférence Ingénierie des Connaissances IC, Laval*, 207–224.

Zhang, K., Wang, J. T. L., et Shasha, D. (1996). “On the Editing Distance Between Undirected Acyclic Graphs.” *International Journal of Foundations of Computer Science*, 7(1), 43-58.

Annexe

Sommaire de l'Annexe

I. Algorithme « GénérationVueGénérique »

Algorithme : GénérationVueGénérique

```
Donnée :            Elément nc, List listNS, List listRS;
Résultat :          List listNG, List listRG ;
Variable Globale :  List listNG, List listRG ;
Début
    traiterFils (listNS[0], listNS, listRS) ;
    vérifierUnicité (listNG, listRG) ;
Fin
```

II. Algorithme « traiterFils »

Algorithme : traiterFils

```
Donnée :    Elément nc, List listNS, List listRS;
Variable :  List listEF, list listAF, NoeudSpe nc, NoeudGen ng, RelationGen rg,
            Boolean trouveEle, Boolean trouveAtt ;
Début
    listEF ←chercherElémentFils (nc, listNS, listRS) ;
    listAF ←chercherAttributFils (nc, listNS, listRS) ;
    Pour i de 0 à listEF.taille() Faire
        trouveEle←faux ;
        trouveEle← chercherNoeud(listEF[i], listNG) ;
        Si (trouveEle) Alors
            rg ← RechercherRelation (nc, listEF[i], listRG) ;
            rg.cardinalité= "+" ;
        Sinon
            ng ← CréerNoeud (nc) ;
            listNG.ajouter(ng) ;
            rg ← CréerRelation (nc, listEF[i]) ;
            listRG.ajouter(rg) ;
        FinSi
        traiterFils(nc, listNS, listRS) ;
    FinPour
    Pour i de 0 à listAF.taille() Faire
        trouveAtt←faux ;
        trouveAtt← chercherAtt(listAF[i], listNG) ;
        Si (trouveAtt) Alors
            rg ← RechercherRelation (nc, listAF[i], listRG) ;
            rg.cardinalité= "+" ;
        Sinon
            ng ← CréerNoeud (nc) ;
            listNG.ajouter(ng) ;
            rg ← CréerRelation (nc, listAF[i]) ;
            listRG.ajouter(rg) ;
        FinSi
    FinPour
Fin
```

III. Algorithme « vérifierUnicité »

Algorithme : vérifierUnicité

```
Donnée   :   List listNS, List listRS;
Variable :   Chaîne nomPereI, Chaîne nomPereJ ;
Début
    Pour i de 0 à listNG.taille() Faire
        Pour j de 0 à listNG.taille() Faire
            Si (listNG(i).nomNG=listNG(j).nomNG) Alors
                nomPereI=listNG(i).NomPereNS() ;
                listNG(i).nomNG= Concaténer( listNG(i).nomNG, nomPereI) ;
                nomPereJ=listNG(j).NomPereNS() ;
                listNG(j).nomNG= Concaténer( listNG(j).nomNG, nomPereJ) ;
            Finsi
        FinPour
    FinPour
Fin
```

IV. Algorithme « PondérationStructurelle »

Algorithme : PondérationStructurelle

```
Donnée : List noeudGén listNG, List relationGén listRG ;
Résultat : Rèel M[long][long] ;
Variable : List nœudGén listNCourant, List nœudGén listNMarqué, List nœudGén listFils,
           Entier long, Boulean Marqué ;
Début
    long← listNG.longueur ;
    Pour i de 1 à long Faire
        M[i][0] ← listNG(i).nomNG ;
        M[0][i] ← listNG(i).nomNG ;
    FinPour
    listNCourant.ajouter(listNS(0)) ;
    listNMarqué.ajouter(listNS(0)) ;
    Tant que (¬ listNCourant.estVide)
        listFils←chercherFils(listNCourant(0), listRG) ;
        Tant que (¬ listFils.estVide)
            Pour i de 1 à long Faire
                Si (M[i][listNCourant(0)] < > 0) Alors
                    T←T+ M[i][listNCourant(0)] ;
                    n++ ;
                FinSi
            FinPour
            β ← listNcourant(0).Ordre(); α ← listNcourant(0).Niveau() ;
            M[pos(listNcourant(0))][pos(listFils(0))] ← T/n + β/10^α ;
            Marqué=faux ;
            Pour i de 1 à listNMarqué.longueur() Faire
                Si (listNMarqué (i)=listFils(0)) Alors
                    Marqué ←vrai ;
                FinSi
```

```
                    FinPour
                    Si (Marqué=vrai) || (détectionBoucle(listFils(0),listRG)=vrai) Alors
                        listNCourant.ajouter(listFils(0) ;
                    FinSi
                    listNMarqué.ajouter(listFils(0)) ;
                    listFils.supprimer(0) ;
                Fin Tant que
                listNCourant.supprimer(0) ;
        Fin Tant que
Fin
```

V. Algorithme « PondérationAdaptation »

Algorithme : PondérationAdaptation

```
Donnée : List noeudGén listNG, List relationGén listRG ;
Résultat : Rèel M[long][long] Matrice représentative ;
Variable Globale : List noeudGén listNG', List relationGén listRG' ;
Variable : List relationGén listRG'', Entier nbrFeuille, Entier long ;
Début
    listNG' ← listNG
    listRG'' ← supprimerCycles(listRG) ;
    listRG' ← dupliquerRelations(listRG'') ;
    nbrFeuille← nbrFeuilles(listNG);
    long← listNG.longueur ;
    M[long][long] ← calculerCoûts (listNG(0), nbrFeuille) ;
Fin
```

VI. Algorithme « CalculerCoûts »

Algorithme : CalculerCoûts

```
Donnée : NœudSpe nc, Entier nbrFeuille ;
Résultat : Réel M[long][long];
Variable Globale : List noeudGen listFils, Entier S, Entier i ;
Début
        listFils←chercherFils(nc,ListNG', listRG') ;
        Tant que ¬ listFils.estVide() Faire
        Si listFils(0).estFeuille Alors
                M[pos(nc)][pos(listFils(0))] ← M[pos(nc)][pos(listFils(0))]+1/nbrFeuille ;
                Sinon
                    M[long][long] ← CalculerCoûts(nc, nbrFeuille) ;
            Pour i de 0 à M.longeur () Faire
                            S←S+M[pos(listFils(0)][i] ;
                 FinPour
                M[pos(nc)][pos(listFils(0))] ← M[pos(nc)][pos(pos(listFils(0))]+(S/nbrPere) ;
                Fin Si
                listFils(0).supprimer ;
        Fin Tant que
Fin
```

Liste des figures

Chapitre I

Chapitre II

Chapitre III

Chapitre IV

Chapitre V

Liste des tableaux

Chapitre I

Chapitre II

Chapitre V