Méthodologie pour la création de documents multistructurés

Pierre-Edouard Portier – Sylvie Calabretto

Université de Lyon, CNRS
INSA-Lyon, LIRIS, UMR5205, F-69621, France
{pierre-edouard.portier, sylvie.calabretto}@insa-lyon.fr

RÉSUMÉ. *Dans cet article, nous rappelons la problématique des documents multistructurés puis nous proposons un tour d'horizon des solutions existantes. Nous remarquons alors que ces dernières ne considèrent pas le problème de la construction de tels documents. Nous utilisons notre expérience acquise à travailler avec des chercheurs philosophes qui construisent une édition numérique des travaux de Jean-Toussaint Desanti, pour proposer une méthodologie de construction de documents multistructurés. Cette dernière repose sur le système MultiX² qui permet de représenter de tels documents.*

ABSTRACT. *We present the multi-structured documents problem and offer an overview of existing solutions. We then notice that they do not consider the problem of constructing such documents. So we use our experience with philosophers who are building a digital edition of the work of Jean-Toussaint Desanti, so as to propose a methodology for construction of multi-structured documents. The latter is based on the MultiX² system in order to represent such documents.*

MOTS-CLÉS : *bibliothèques numériques, documents multistructurés, XML*

KEYWORDS: *digital libraries, overlapping hierarchies, XML*

1. Introduction

Ce travail introduit une nouvelle problématique : la construction de documents multistructurés. La multiplicité des usages d'un même document a entraîné une multiplication des types de structures documentaires (physique, logique, sémantique, …). La définition simultanée de plusieurs structures pour un même document a rapidement introduit la problématique dite des documents multistructurés (Bruno et al., 2007). Cette dernière s'inscrit dans un contexte historique particulier où les formalismes les plus utilisés pour la représentation de documents (SGML puis XML) imposent une structuration arborescente. Elle a ainsi toujours pris la dénomination technique de structures hiérarchiques concurrentes (overlapping hierarchies) (Iacob et al., 2005).

En étudiant le problème de la construction des documents multistructurés, nous nous rapprochons des pratiques concrètes des utilisateurs qui construisent quotidiennement des documents. Notre travail repose sur l'expérience acquise à côtoyer des chercheurs qui construisent une édition numérique des archives manuscrites du philosophe Jean-Toussaint Desanti (1914-2002) qui est entre autres connu pour son travail de recherches épistémologiques sur le développement de la théorie des fonctions de variables réelles (travail qui lui demanda d'acquérir une très grande proximité pratique avec les mathématiques). Rappelons que l'édition numérique recouvre l'ensemble du processus non seulement éditorial mais scientifique et critique qui aboutira à la publication d'une ressource sous forme électronique. Dans le cas de l'édition numérique de manuscrits, la première étape de ce processus consiste à construire un facsimile numérique du manuscrit auquel s'ajoutera le travail de transcription et d'analyse critique des chercheurs. En échangeant avec les responsables d'autres projets similaires nous avons déterminé que la problématique de construction de documents multistructurés était au coeur de leurs travaux. Mais, ce qui intéresse les chercheurs en sciences humaines n'est pas le problème technique de la représentation de quasi-hiérarchies au moyen de structures arborescentes, mais bien les moyens de faire coexister plusieurs structures descriptives qui sont autant de registres d'interaction différents pour un même document. C'est-à-dire que chaque structure est considérée comme un point de vue distinct depuis lequel l'utilisateur peut interagir (consulter, éditer, interroger, …) avec le document. Notre travail consiste donc à *favoriser* la création d'une multiplicité de structures lors de la construction du document.

Illustrons notre propos avec la figure 1 qui représente une double page extraite d'un cahier de J.T. Desanti. Sur cette image, la zone appelée ZI représente une unité textuelle qu'un interprétant désire isoler et qui recouvre les deux pages. Nous sommes donc face à deux hiérarchies concurrentes (c'est-à-dire qui ne peuvent pas être représentées au moyen d'un seul arbre) : les pages et les zones d'intérêt. Nous voyons aussi des équations (E1 et E2) qui, même si elles ne soulèvent pas le problème technique des hiérarchies concurrentes, pourraient appartenir à une troisième structure, celle des expressions mathématiques. Ainsi, il s'agit non

seulement de proposer une solution au problème des hiérarchies concurrentes mais encore de concevoir une méthodologie de création de documents multistructurés qui facilite les choix de modélisation de l'interprétant.

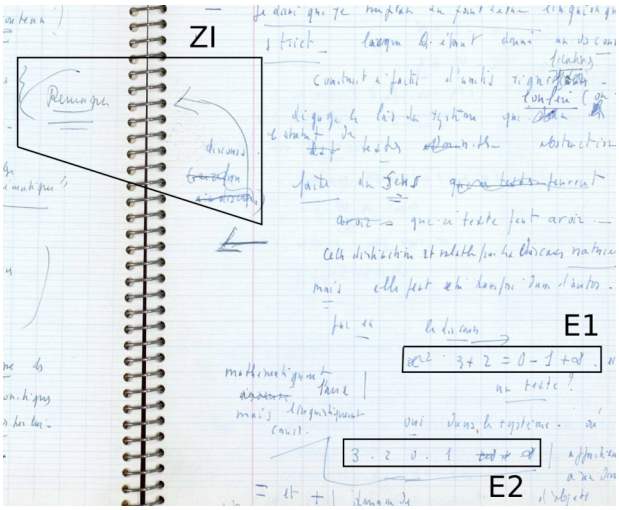

Figure 1. *Deux pages d'un cahier*

Dans la section 2 nous proposons un état de l'art des solutions de modélisation de documents multistructurés. Dans la section 3 nous présentons en détail un modèle particulier : MultiX^2. Enfin, dans la section 4 nous utilisons ce dernier modèle pour proposer une méthodologie de construction de documents multistructurés.

2. Les solutions de modélisation de documents multistructurés

2.1. *Démarche*

Nous divisons en quatre classes l'ensemble des réponses à la problématique des documents multistructurés. D'abord une solution historique, CONCUR, ensuite des solutions *ad-hoc* proposées par la TEI (Text Encoding Initiative), puis les modèles qui ne sont pas compatibles avec le langage XML et enfin ceux qui sont compatibles avec XML. Chaque solution est analysée aussi objectivement que possible selon six axes :

- L'expressivité du modèle évalue si un modèle est explicitement défini et s'il répond de manière satisfaisante à la problématique des documents multistructurés telle que définie jusqu'à présent, c'est-à-dire de manière statique (et non dynamique

où le document est *en construction*). Nous considérons cet axe comme un prérequis pour toute solution qui prétend répondre à la nouvelle problématique ici introduite.

- La généricité du modèle évalue, lorsqu'un modèle existe, s'il est facile de le modifier pour répondre à des problèmes qui dépassent le contexte initial des seules représentation et interrogation de documents multistructurés. Il s'agit pour nous de savoir si le modèle évalué peut servir de base à l'établissement d'une solution au problème de la construction de documents multistructurés.

- La qualité de l'implémentation évalue le soin mis à développer une implémentation efficace du modèle proposé.

- La compatibilité avec les outils XML mesure s'il est possible d'intégrer la solution dans le contexte plus large des nombreux outils existants pour traiter des documents XML (en particulier les outils de typage : les schémas XML, …). Une solution qui ne s'inscrirait pas dans la galaxie XML, quand même répondrait-elle à l'ensemble des autres critères, trouverait difficilement à s'appliquer aux nombreux cas concrets de construction de documents multistructurés, ces derniers apparaissant souvent au sein de communautés qui utilisent déjà massivement XML (voir par exemple l'importance des recommandations de la TEI dans les humanités).

- L'existence de mécanismes de requêtes pour interroger les documents multistructurés.

- La gestion du changement dans les données ou les structures évalue si le modèle est robuste au changement.

2.2. *Une solution historique*

CONCUR (Goldfarb, 1990) est une option SGML qui permet la gestion de plusieurs DTD pour le même contenu. Dans un tel document SGML, toutes les structures cohabitent dans le même fichier. Dans ce fichier, une première structure est encodée de manière standard. Puis, pour chacune des structures supplémentaires, un préfixe est ajouté à chaque balise ouvrante pour déterminer dans quelle DTD elle se trouve spécifiée.

Cependant, avec ce mécanisme, il n'est pas possible d'établir des relations entre les différentes structures. De plus, comme décrit dans (Hilbert et al., 2005), si deux éléments qui appartiennent à des DTD différentes décrivent la même région, l'ordre des balises est indistinct. C'est pourquoi, cette solution a rarement été implémentée et même Charles Goldfarb (Hilbert et al., 2005), le principal développeur du standard SGML, préconisait d'éviter son utilisation.

L'option CONCUR répond à la plupart des problèmes soulevés par les documents multistructurés bien qu'elle ne permette pas d'établir des relations entre structures. Cependant, en tant qu'option intégrée au standard SGML, elle ne répond pas à notre critère de généricité. De plus, il n'y a pas de mécanisme de requête.

Notons que, puisque le document est unique, des changements dans les structures ou les données sont possibles.

2.3. *Des solutions ad-hoc*

La TEI (Text Encoding Initiative) (Burnard et al., 2007) est un consortium qui développe et maintient un standard pour la représentation des textes électroniques. Ses recommandations sont exprimées sous la forme modulaire et extensible d'un schéma XML documenté. Au fil des recommandations, pour chaque instance du problème de multistructuralité, une solution locale est proposée. De plus, dans la dernière version des recommandations, un chapitre a été ajouté qui fait la synthèse de ces solutions *ad-hoc*. Elles sont au nombre de quatre (DeRose, 2004) (Bruno et al., 2007).

Il est possible de dupliquer le contenu autant de fois qu'il y a de structures arborescentes concurrentes … Solution pauvre qui empêche le document d'évoluer.

On peut utiliser des éléments vides (« fin de ligne » <lb/>, « fin de page » <pb/>, …). Il est alors impossible de profiter des outils standard de validation des documents XML, ou de construire un schéma pour spécifier la structure qui utilise des éléments vides.

Pour deux structures concurrentes qui ne forment pas un arbre, on peut découper l'une d'entre elles en éléments suffisamment fins pour qu'ils ne provoquent pas de chevauchements avec l'autre structure, et par le jeu d'attributs bien choisis, se laisser la possibilité de reconstruire la structure initiale. Comme les précédentes, cette solution demande des traitements particuliers pour reconstruire les différentes structures et ne permet pas l'utilisation des outils de typage XML.

Enfin, on peut isoler le contenu dans une structure de base et construire les structures documentaires par références à cette dernière. Cette solution est sans doute la plus générique, elle sera reprise dans le contexte de modèles bien définis par plusieurs des solutions présentées dans les sections suivantes.

2.4. *Modèles non compatibles XML*

2.4.1. *TexMECS*

Plutôt que de s'accommoder des langages de balisage existants, certains travaux ont défini de nouvelles syntaxes. MECS (Multi-Element Code System) (Huitfeldt, 1998) fut le premier langage autorisant des chevauchements de structures. TexMECS (Huitfeldt et al., 2003) est basé sur ce langage mais il est beaucoup plus expressif. Il permet de définir des structures complexes pour lesquelles un élément peut avoir plusieurs parents. Le modèle MECS est suffisamment expressif pour

répondre aux problématiques actuellement identifiées des documents multistructurés, mais trop complexe pour satisfaire le critère de généricité. De plus, il n'est pas accompagné de mécanismes de requête.

2.4.2. *LMNL*

LMNL (Layered Markup and aNnotation Language) (Tennison et al., 2002) définit une syntaxe spécifique basée sur la notion d'intervalle qui permet d'encoder de multiples structures avec des éléments qui se chevauchent. C'est une solution seulement syntaxique qui ne propose pas de mécanismes de requêtes. Comme pour la solution précédente, puisque les structures sont encodées dans un document unique, des changements dans les données ou les structures sont envisageables. Ces deux solutions étant relativement complexes et incompatibles avec XML, elles sont restées à un stade expérimental.

2.4.3. *Graphes d'annotations*

Les graphes d'annotations (Maeda et al., 2002) ont été développés à l'origine pour représenter les phénomènes du domaine de la linguistique (phonétique, prosodie, morphologie, syntaxe, …). Si l'on considère ces sous domaines comme différentes structures, les graphes d'annotations sont une solution à la problématique des documents multistructurés. Comme le montre la figure 2, un même fragment de texte peut être annoté par des arcs appartenant à différentes structures. La granularité des noeuds correspond aux mots, ou même aux caractères si nécessaire. Les labels associés aux arcs indiquent le type de la structure (par exemple S2:M pour les mots, S1:L pour les lignes). Puisque les structures partagent le même graphe, leurs mises à jour ne posent pas de problèmes particuliers. De plus, un prototype de langage de requêtes existe (Bird et al., 2000) pour interroger les graphes d'annotations. Le modèle sous-jacent étant un graphe, il est suffisamment expressif pour répondre aux problématiques des documents multistructurés, mais puisqu'il est fortement orienté linguistique, il manque de généricité.

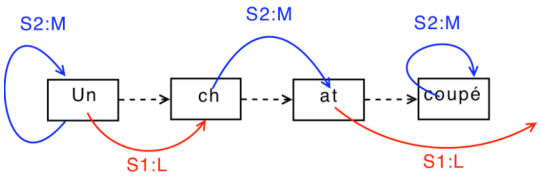

Figure 2. *Graphe d'annotations*

2.4.4. *Utilisation de graphes RDF*

Il est possible d'utiliser le formalisme de représentation de graphes RDF (Resource Description Framework) pour représenter des documents multistructurés

(Tummarello et al., 2005). Cette méthode est très similaire à la précédente, mais possède l'avantage de reposer sur un modèle de graphes générique. Les outils d'interrogation standard pour les graphes RDF (par exemple SPARQL) sont utilisables, mais les requêtes complexes peuvent être difficiles à formuler. Même si RDF est sérialisable en XML, l'utilisation des outils XML standards n'est pas satisfaisante puisque ces derniers travaillent sur des structures arborescentes. Enfin les structures et les données partagent le même graphe, la gestion du changement ne pose donc a priori pas de problèmes.

2.5. *Modèles compatibles XML*

2.5.1. *MuLaX*

MuLaX (Hilbert et al., 2005) est une adaptation de l'option SGML CONCUR au formalisme XML. C'est un format documentaire qui permet d'unifier une série de documents XML qui partagent le même contenu en un seul document. Pour distinguer les différents niveaux d'annotations, les balises sont préfixées par un identifiant de structure. Cette solution répond au problème des documents multistructurés, mais manque de généricité puisqu'un éditeur spécifique est nécessaire pour interpréter les documents produits. Comme pour l'option CONCUR, les changements de structures ou de données sont possibles puisque le document est unique. Aucun opérateur de requête n'est défini, mais les auteurs expliquent qu'il est possible de construire des expressions de chemins similaires à celles du langage XPath.

2.5.2. *GODDAG*

(Sperberg-McQueen et al., 2000) présente une solution basée sur les GODDAG (General Ordered Descendant Directed Acyclic Graph) pour représenter des documents multistructurés. Le modèle de graphe répond au problème de chevauchement de balises. Pour interroger les documents, les auteurs ont proposé une extension au langage XPath. Cette extension permet de prendre en compte de nouveaux types de relations entre éléments de différentes structures. Ce modèle ne peut pas gérer les relations génériques entre structures. Il est possible d'importer (ou d'exporter) depuis (ou dans) le formalisme XML. Avec ce modèle, il est possible de gérer le changement dans les structures.

2.5.3. *MCT*

Le modèle MCT (Multi-Colored Trees) (Jagadish et al., 2004) est une extension du modèle de données XML qui permet de représenter plusieurs arbres partageant les mêmes données. Il s'appuie sur la technique de coloration d'arbres. Une couleur unique est associée à chaque arbre. Un noeud peut avoir plusieurs couleurs : celle de l'arbre auquel il appartient et d'autres qui correspondent à d'autres arbres. Ainsi, un

noeud avec plusieurs couleurs possède (au moins logiquement) plusieurs parents. Le modèle sous-jacent n'est ni suffisamment expressif, ni générique puisqu'il impose un isomorphisme des segments de données (les segments de données doivent être du même type : un mot, un caractère, …). Une extension de XQuery a été développée pour pouvoir interroger les documents construits au moyen de ce modèle.

2.5.4. *MSXD*

MSXD (Multi-Structured XML Documents) (Bruno et al., 2006) est un modèle de représentation de documents multistructurés accompagné d'une extension XQuery pour l'interrogation, il permet de plus aux utilisateurs d'annoter les structures. Une proposition de schéma pour documents textuels multistructurés est introduite. La nécessité de définir un grand nombre de relations entre structures rend cependant l'utilisation du modèle délicate. De plus, il n'est pas possible de prendre en compte les changements dans les structures ou les données.

2.5.5. *Delay Nodes*

Jacques LeMaître (LeMaître, 2006) propose d'ajouter un nouveau type de noeud (delaynode) au modèle de données XDM sur lequel sont basés XPath et XQuery. Ces noeuds sont une représentation virtuelle sous forme de requête XQuery de certains des enfants de leur noeud parent. L'avantage des « delay nodes » est de pouvoir représenter des arbres XML avec des noeuds qui ont plus d'un parent. De plus, aucune extension de XPath ou XQuery ne sont nécessaires à la navigation dans ces nouvelles structures. Avec cette approche, il n'est pas possible de remonter aux parents d'un « delay node » mais seulement de naviguer dans le sens des descendants.

2.5.6. *MonetDB/XQuery*

MonetDB/Xquery est un SGBD XML. Il possède une extension (Alink et al., 2006) pour gérer les documents multistructurés grâce à la technique du « stand-off markup », il s'agit donc d'isoler le contenu dans une structure de base et de construire les structures documentaires par références à cette dernière. Des opérateurs d'interrogation optimisés sont implémentés mais aucun modèle n'est explicitement défini, seule une description informelle est proposée (telle celle de la TEI).

2.5.7. *MSDM, MultiX*

MSDM (Chatti et al., 2007) est un modèle pour la représentation de documents multistructurés proposé par N.Chatti et dont une instance, MultiX, s'exprime dans le formalisme XML. Cette proposition s'inscrit dans la catégorie des solutions à base de « stand-off markup ».

Dans ce modèle, un document est un graphe D composé :

- d'un ensemble SB de noeuds aussi appelé structure de base
- d'une famille $(SD_j)_{j\in J}$ d'arbres aussi appelés structures documentaires

De plus, $\forall j \in J$, il existe une relation R_j qui associe à chaque noeud de SD_j un sous-ensemble de SB ; pour chaque feuille de SD_j ce sous-ensemble ne doit pas être vide. La figure 3 illustre chacun des éléments du modèle.

Enfin, remarquons que seuls les noeuds de la structure de base possèdent un contenu associé et dépendent donc du type des données à structurer (textes, vidéos, …). Dans le cas qui nous intéresse des données textuelles, à chaque noeud de la structure de base est associée une chaîne de caractères que nous appelons aussi *fragment*.

Cette solution répond au problème des documents multistructurés telle que formulée jusqu'à présent. Se basant sur un modèle simple de « stand-off markup », elle est suffisamment générique pour suivre les évolutions futures de la problématique. Elle propose des fonctions XQuery qui permettent d'interroger les documents multistructurés. Mais elle n'est pas bien adaptée à la gestion du changement dans les données ou les structures, puisqu'il faudrait reconstruire la structure de base à partir des structures documentaires à chaque modification.

2.6. *En résumé*

Aucun des modèles présentés ne satisfait pleinement aux critères que nous avons établis. Cependant, MSDM fait partie des bons modèles et puisque nous le connaissons bien et qu'il est de plus cité dans les recommandations de la TEI, nous avons choisi d'en développer une nouvelle instance, MultiX^2, que nous présentons dans la section suivante et sur laquelle repose notre méthodologie de construction de documents multistructurés.

3. MultiX^2, un modèle pour la représentation des documents multistructurés

MultiX^2 est une instance de MSDM qui, contrairement à MultiX, privilégie les recommandations du W3C aux mécanismes spécifiques. Ainsi, les liens entre structures documentaires et structure de base sont assurés au moyen du standard XInclude.

Nous illustrons le modèle sur un exemple extrait des archives Jean-Toussaint Desanti (voir la figure 1). Il s'agit d'une double page de cahier pour laquelle une zone de texte, chevauche les deux pages. Remarquons que nous sommes aussi en présence d'équations mathématiques mélangées à un texte philosophique.

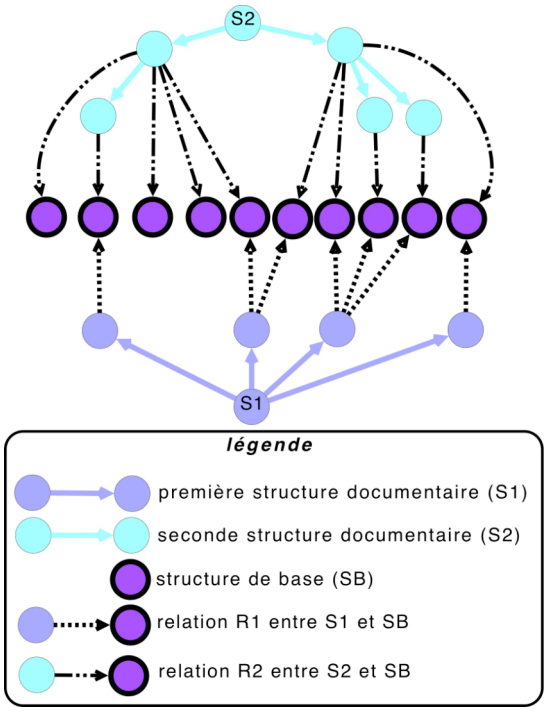

Figure 3. *Illustration du modèle MultiX*

Pour l'exemple, nous distinguons deux structures. D'abord une structure physique des pages :

```
<s1>
  <page>Début de la première page ...
Remarque,</page>
  <page>ce discours, ...
par ex le discours 3 + 2 = 0 - 1 est-il un texte ? ...</page>
</s1>
```

Ensuite une structure logique des zones d'intérêt :

```
<s2>
  <p> Début de la première page ...</p>
  <p>Remarque, ce discours, ...</p>
  <p>par ex le discours
    <eq>3 + 2 = 0 - 1</eq> est-il un texte ? ...</p>
</s2>
```

Nous construisons la structure de base en identifiant les fragments partagés par les structures documentaires :

```
<seg xml:id="F1"> Début de la première page ...</seg>
<seg xml:id="F2">Remarque, </seg>
```

```
<seg xml:id="F3">ce discours, ...</seg>
<seg xml:id="F4">par ex le discours </seg>
<seg xml:id="F5">3 + 2 = 0 - 1</seg>
<seg xml:id="F6"> est-il un texte ? ...</seg>
```

Nous remplaçons le contenu des structures documentaires par des références à la structure de base grâce au standard XInclude. D'abord la structure physique :

```
<s1>
  <page><xi:include href="base.xml" xpointer="element(F1/1)"/>
    <xi:include href="base.xml" xpointer="element(F2/1)"/></page>
  <page><xi:include href="base.xml" xpointer="element(F3/1)"/>
    <xi:include href="base.xml" xpointer="element(F4/1)"/>
    <xi:include href="base.xml" xpointer="element(F5/1)"/>
    <xi:include href="base.xml" xpointer="element(F6/1)"/></page>
</s1>
```

Ensuite la structure logique :

```
<s2>
  <p> <xi:include href="base.xml" xpointer="element(F1/1)"/></p>
  <p><xi:include href="base.xml" xpointer="element(F2/1)"/>
       <xi:include href="base.xml" xpointer="element(F3/1)"/></p>
  <p><xi:include href="base.xml" xpointer="element(F4/1)"/>
       <eq><xi:include href="base.xml" xpointer="element(F5/1)"/></eq>
       <xi:include href="base.xml" xpointer="element(F6/1)"/></p>
</s2>
```

Nous utilisons les outils XML standards pour valider les structures documentaires, et construire des requêtes grâce aux fonctions XQuery développées pour MultiX. Ci-après une requête qui trouve les paragraphes qui chevauchent deux pages :

```
let $physique := doc("physique.xml")
let $logique := doc("logique.xml")
for $page in $physique//page,
    $para in $logique//p
where multix:share-fragments($page,$para) and
        not(multix:include-content-of($page,$para))
return $para
```

Et le résultat retourné sera :

```
<p>Remarque, ce discours, ...</p>
```

La fonction *share-fragments(a,b)* vérifie si les éléments a et b ont au moins un fragment en commun, *include-content(a,b)* vérifie si tous les fragments qui composent l'élément b composent aussi l'élément a.

4. Une méthodologie de construction des documents multistructurés

Dans cette section, nous utilisons le modèle MultiX pour forger un point de vue propre à présenter le problème de la construction de documents multistructurés. En effet, les structures documentaires sont rarement données, mais nécessitent d'être construites. Par exemple, de nombreux projets d'éditions numériques partent des images de manuscrits puis les transcrivent et les annotent. Lors de ces opérations, il faudra gérer des cas de multistructures. C'est pourquoi nous proposons une méthodologie qui cartographie le domaine de la construction de documents multistructurés selon trois catégories de méthodes : l'utilisation de conditions formelles qui déclenchent la création automatique de nouvelles structures, la présentation du résultats des restructurations automatiques à l'interprétant afin qu'il puisse éventuellement les réviser, l'incitation aux interactions entre interprétants afin de favoriser les mises en relations des structures documentaires.

4.1. Les moments de restructuration

Nous analysons les conditions sous lesquelles il faut construire une nouvelle structure documentaire. Pour plus de clarté et puisque nous connaissons bien le domaine, nous nous plaçons dans le contexte de l'édition critique numérique de manuscrits. Admettons que pour transcrire un manuscrit des chercheurs disposent des éléments de la TEI. Si nous reprenons l'exemple précédent, des pages, des paragraphes et des équations ont été balisés jusqu'à ce qu'un paragraphe chevauche deux pages (voir les arcs en pointillés de la figure 4). Il faut alors distinguer deux structures. La création d'une nouvelle structure est une opération purement formelle (voir figure 5), qui consiste à transformer un graphe en deux arbres, mais qui peut aussi être l'occasion pour l'interprétant de faire des choix de modélisation. Par exemple, demander que la division en équations rejoigne une structure « mathématique » et profiter de l'occasion pour nommer les nouvelles structures (voir figure 6).

4.2. *Le partage des responsabilités*

Puisque la création d'un nouveau document s'inscrit souvent dans un environnement multi-utilisateurs (c'est toujours le cas pour les projets d'éditions numériques de manuscrits), il faut définir des rôles et des responsabilités. Nous supposons un responsable par structure. Le responsable devra intervenir lors d'une modification de la structure de base. Nous distinguons quatre types de modifications (voir figure 7).

Dans la première situation (S1), un fragment en lien avec la structure du responsable est modifié, c'est-à-dire que du texte y est ajouté ou supprimé (f11 devient f12). Alerté, le responsable n'a pas de décision nécessaire à prendre. Dans la

seconde situation (S2), un fragment de la structure de base en lien avec un élément de la structure du responsable est scindé (f21 est changé en f22 et f23). Le responsable doit être alerté car sa structure n'est plus cohérente. Il pourra choisir de regrouper les éléments scindés, de n'en conserver que quelques uns, etc. Dans la troisième situation (S3), des éléments d'autres structures coupent leurs liens avec un fragment de la structure de base relié à un des éléments de la structure du responsable. Le responsable doit être alerté à titre indicatif, en interprétant la situation, en interrogeant les autres responsables, … il pourra se construire une connaissance nouvelle. Dans la quatrième situation (S4), des fragments sans rapport avec la structure du responsable sont créés (par exemple f41). Ce dernier doit être tenu informé pour éventuellement mettre à jour sa structure.

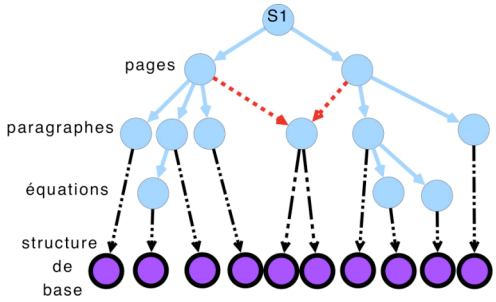

Figure 4. *Une restructuration est nécessaire (un paragraphe chevauche deux pages)*

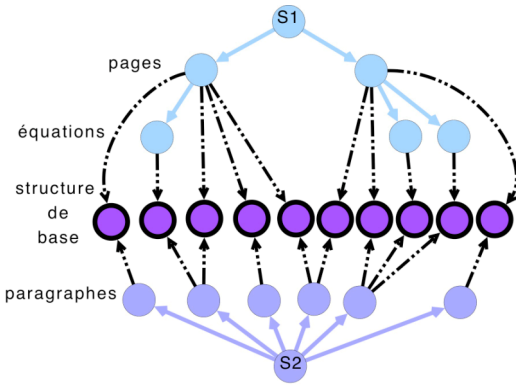

Figure 5. *Restructuration automatique (deux structures sont distinguées)*

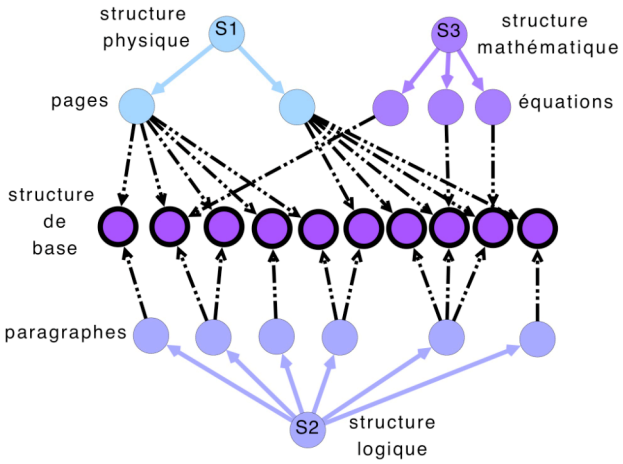

Figure 6. *Intervention de l'interprétant dans la restructuration (trois structures sont distinguées et nommées)*

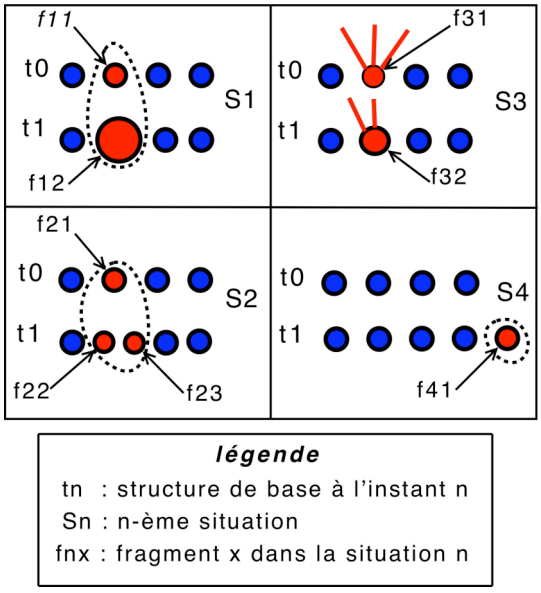

Figure 7. *Types de modifications de la structure de base*

5. Conclusions

Nous avons énoncé un nouveau problème, celui de la *construction* des documents multistructurés. Ce dernier nous a permis de reprendre l'ancienne problématique de la représentation des documents multistructurés en nous éloignant de sa formulation jusqu'aujourd'hui seulement technique pour nous rapprocher de l'usage concret des personnes qui construisent des documents. Nous avons, entre autres, montré que même si l'imposition de structures documentaires arborescentes fut longtemps considérée comme le noeud du problème, nous pouvions en faire le noyau d'une solution en l'utilisant comme déclencheur formel pour détecter les moments qui nécessitent une restructuration et l'intervention d'un interprétant ! De plus, nous développons un prototype logiciel qui implémente les différentes opérations algébriques décrites dans l'article et permet de soutenir le processus de construction de documents multistructurés. Notons que cette implémentation repose sur une correspondance entre les schémas XML et les types du langage de programmation fonctionnel Haskell (Jones, 2002), ce qui assure une bijection entre un document XML et les valeurs typées qui existent à l'exécution du programme. Travailler dans l'espace des valeurs du langage facilite la gestion dynamique des structures. De plus les opérations fournies par ce prototype sont accessibles au moyen d'une interface Web qui utilise le protocole HTTP en suivant le « design pattern » dit REST (Fielding, 2000), ceci nous permet d'inscrire ce programme dans une plateforme que nous développons et qui est destinée à assister les chercheurs du domaine des Humanités dans l'édition numérique de textes.

6. Bibliographie

Alink W., Bhoedjang R. A. F., De Vries A. P., Boncz P. A., « Efficient XQuery Support for Stand-Off Annotation », *XIME-P*, Chicago, June 2006.

Bird S., Buneman P., Chiew Tan W., « Towards a Query Language for Annotation Graphs », *Proceedings of the Second International Conference on Language Resources and Evaluation*, 2000, p. 807-814.

Bruno E., Murisasco E., « Multistructured XML textual documents », *GESTS International Transactions on Computer Science and Engineering*, vol. 34, n° 1, 2006, p. 200-211.

Bruno E., Calabretto S., Murisasco E., « Documents textuels multi structurés : un état de l'art. », *Revue i3*, vol. 7, n° 1, 2007.

Burnard L., Bauman S., TEI P5 : Guidelines for Electronic Text Encoding and Interchange, 2007.

Chatti N., Kaouk S., Calabretto S., Pinon J.-M., « MultiX : an XML-based formalism to encode multi-structured documents », *Proceedings of Extreme Markup Languages'2007*, Montréal (Canada), août 2007.

De Rose S. J., « Markup Overlap : A Review and a Horse », *Extreme Markup Languages*, 2004.

Fielding R. T., Architectural styles and the design of network-based software architectures, PhD thesis, 2000.

Goldfarb C. F., The SGML handbook, Oxford University Press, New York, NY, USA, 1990.

Hilbert M., Witt A., Québec M., Schonefeld O., « Making CONCUR work », *Extreme Markup Languages*, 2005.

Huitfeldt C., « MECS – a Multi-Element Code System », *Working Papers from Wittgenstein Archives at the University of Bergen*, vol. 3, 1998.

Huitfeldt C., Sperberg-McQueen M., « TexMECS : An experimental markup meta-language for complex documents », *Working paper of the project Markup Languages for Complex Documents (MLCD)*, Univeristy of Bergen, Available on the Web at http://decentius.aksis.uib.no/mlcd/2003/Papers/texmecs.html, 2003.

Iacob I. E., Dekhtyar A., « Processing XML documents with overlapping hierarchies », *JCDL'05 : Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries*, New York, NY, USA, 2005, p. 409-409.

Jagadish H. V., Lakshmanan L. V. S., Scannapieco M., Srivastava D., Wiwatwattana N., « Colorful XML : one hierarchy isn't enough », *SIGMOD'04 : Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, New York, NY, USA, 2004, ACM, p. 251-262.

Jones S. P., Haskell 98 Language and Libraries : The Revised Report, http://haskell.org/, September 2002.

Le Maître J., « Describing multi-structured documents by means of delay nodes », *DocEng'06 : Proceedings of the 2006 ACM symposium on Document engineering*, New York, NY, USA, 2006, ACM, p. 155-164.

Maeda K., Bird S., Ma X., LEE H., « Creating Annotation Tools with the Annotation Graph Toolkit », *Proceedings of the Third International Conference on Language Resources and Evaluation*, April 2002.

Sperberg-McQueen C. M., Huitfeldt C., « GODDAG : A Data Structure for Overlapping Hierarchies », *DDEP/PODDP*, 2000, p. 139-160.

Tennison J., Piez W., « The Layered Markup and Annotation Language (LMNL) », *Extreme Markup Languages*, 2002.

Tummarello G., Morbidoni C. Pierazzo E., « Toward Textual Encoding Based on RDF », *ELPUB*, 2005, p. 57-63.