Structuration de l'information

La structuration de l'information représente toutes les techniques de production, de stockage et d'accès au contenu. Elle fait une certaine différence entre les bases de données et les corpus documentaire dans le sens où les techniques vont différer selon si l'on souhaite retrouver le contenu de chaque champ d'une base de données ou si nous souhaitons une recherche d'information en texte intégral. Structurer un document permet d'inscrire le contenu dans une forme afin de le retrouver grâce à des index spécifiques par exemple, ou de le publier sous des formes diverses. Une structuration logique permet de définir des éléments hiérarchisés et d'assigner des attributs pour chaque éléments.

Les objectifs de la structuration de l'information sont de permettre un meilleur traitement, ici automatique, des données. Pour rappel, une donnée est une information qui n'a pas été transformée pour être traitée dans le but de leur donner un sens, d'être communiquée. Une structuration des données peut être différente en fonction du traitement informatique que l'on souhaite lui approprier, et cela faciliterait le travail d'application information et diminuer le nombre d'erreurs de traitement.

Dans notre cas, nous parlerons essentiellement de structuration de données numériques puisque les données à traiter sont celles du corpus d'articles de presse issus de La Voix du Nord et récupérés sous format HTML via le serveur Europresse.

Dans le cadre du stage, la structuration de l'information va être utilisée afin de produire une base de connaissance dans le but de valoriser le patrimoine minier. Les données étant diffuses dans les contenus de la presse, la première tâche a donc été de collecter les informations, pour ensuite les structurer avant de les analyser. Il y a donc des enjeux de préparation du corpus et d'identification des éléments pertinents. Plus précisément, et comme expliqué de manière plus abstraite dans la méthodologie, il a fallut transformer et structurer de façon automatique le corpus d'articles de presse afin de permettre au logiciel Tropes un meilleur traitement des documents pour l'analyse de leur contenu mais aussi dans le but de réutiliser ces documents dans le projet sans pour autant être gêner par le bruit des métadonnées non pertinentes à quelconques analyse.

Dans cette partie, nous essaierons de savoir comment s'intègre la structuration de l'information dans une analyse de contenu ? Comment sont redistribués les rôles du balisage et des langages informatiques dans la structuration de l'information puis dans une analyse de corpus d'articles de presse ?

Afin de répondre à cette problématique, nous allons tout d'abord présenter quelques définitions des notions de base dans la structuration des informations afin de mieux cadrer le sujet et nous focaliser sur les notions que nous rencontrerons au fur et à mesure des explications. Nous démontrerons ensuite le rôle du balisage dans la structuration de l'information mais aussi son rôle dans le cadre de l'analyse de corpus de presse dans le projet ANR MémoMines. Et nous terminerons par aborder les langages informatiques qui nous ont été utile pour le traitement numérique des articles de presse, notamment les langages XSLT, HTML et XML.

3.1 Définitions

Pour commencer, nous allons définir très rapidement les notions d'information, notamment en tant que signal, de systèmes d'information, de données et de document.

L'information est « *une connaissance inscrite (enregistrée) sous forme écrite (imprimée ou numérisée), orale ou audiovisuelle sur un support spatio-temporel. L'information comporte un élément de sens. C'est une signification transmise à un être conscient par le moyen d'un message inscrit sur un support : imprimé, signal électrique, onde sonore, etc.* » (Le Coadic, 2004). Selon Shannon, l'information est un flux physique circulant entre un émetteur et un récepteur lors d'un processus de communication, comme

l'indique le Modèle de Shannon et Weaver[3]. Dans ce modèle, la source d'information énonce un message que l'émetteur va encoder et transformer en signal qui va être acheminé par le canal, puis décodé par le récepteur, qui reconstitue un message à partir de ce signal et le transmet au destinataire.

L'information est une sorte de séries de codes, comme par exemple le fait d'afficher un titre sous une grande police,de surligner des éléments importants dans le contenu, ou de créer une certaine hiérarchie des titres, qui permettent de définir la structure de l'information ou du document.

Pour parler de la notion de système d'information, nous allons surtout nous intéresser à son enjeu informatique. Un système d'information est « *un ensemble organisé de ressources qui permet de collecter, stocker, traiter et distribuer de l'information...* » (Wkipédia). Dans notre cas, les systèmes d'information vont être toutes les solutions informatiques qui vont permettre de collecter, stocker, traiter et communiquer les informations. Ces systèmes vont faire appel à des langages informatiques en fonction de leur utilité, par exemple, pour permettre une meilleure gestion des informations, nous nous orienterons vers du langage Java ou SQL. Dans le stage, l'enjeu est de baliser l'information pour un traitement automatique, nous utilisons donc les langages de balisage comme HTML ou encore XML.

Un document est l'ensemble formé par une information et son support. Il est fabriqué dans le but d'expliquer, de décrire et peut être utilisé comme une preuve. Sa forme numérique est celle qui nous intéresse le plus ici. Le document numérique est celui qui est utilisé dans ce stage. Sous cette forme, il permet une séparation entre les métadonnées, soit toutes les informations relatives au document, que ce soit sa nature, son auteur, sa date de création, etc. et le contenu, soit les informations destinées à être communiquée par ce document, comme par exemple des informations textuelles, des images, des tableaux.

3.2 Balisage et langages de structuration de l'information

Avant d'expliquer les termes de balisage et de langages de structuration de l'information, il est important de préciser que notre travail s'appuie sur la modélisation de l'information. La modélisation est « *une technique d'ingénierie visant à comprendre un système, déjà existant ou à créer. Elle permet de « visualiser » […] un système tel qu'il est, ou tel que nous voudrions qu'il soit ; d'en préciser la structure ou le comportement suivant des points de vue qui éclairent la réalité de différentes façons, et ceci indépendamment d'un langage de programmation* » (Dalbin, 2003). La modélisation permet donc de structurer les idées et simplifier la réalité dont la représentation est abstraite. En d'autres termes, le but est de construire un système pour le documenter. En prenant l'exemple de notre mission, il a fallut modéliser et donc de structurer le corpus d'articles de presse en balisant totalement les fichiers afin de repérer très rapidement les titres des articles, le nom de leur source, leur numéro, leur date de publication et évidemment leur contenu. Il est plus aisé pour l'humain de se concentrer sur une zone du document à la fois plutôt que sur un ensemble de données non structurées limitant la perception ne serait-ce que d'un unique élément de l'information. Pour faire une comparaison avec une situation du quotidien, l'humain a du mal à retrouver l'objet qu'il cherche dans une maison qui n'est pas rangé alors que dans une maison bien rangé, il visualise déjà la pièce dans lequel l'objet peut se trouver éliminant ainsi toutes les autres pièces de la maison.

Dans un langage informatique, une balise permet de repérer une position dans un processus de structuration de l'information dans un document. Elle marque l'emplacement de cette information par rapport au flux d'information que propose un document. Si plusieurs information doivent être marquées, il faudra utiliser plusieurs types de balise. Dans ce cas, la balise n'est plus seulement un marquage mais devient un élément d'information aussi important que le contenu du document.

Dans ce même contexte, le balisage permet de définir une zone dans le document. Cela permet de repérer rapidement la partie du document qui nous intéresse étant donné qu'elle possède une caractéristique

3 Cf Modèle de Shannon et Weaver en Annexe

particulière qui la différencie du reste du contenu. Sans parler de langage de balisage, le moyen le plus abordable de baliser une information dans un document est l'application de style dans un texte, que ce soit la mise en gras, en italique, entre parenthèses ou entre guillemets. Plus techniquement, le balisage est le fait de mettre entre deux balises indiquant le début et la fin de la zone à marquer, une information. On parle ici de balises ouvrante et de balises fermantes. Dans certains langages, une balise ouvrante doit absolument correspondre à une balise fermante.

Dans le cadre du stage, nous avons eu l'occasion de pratiquer le balisage par des langages de structuration de l'information, notamment grâce aux langages informatiques sur lesquels nous reviendront plus tard. Nous avons retrouver cet aspect de la structuration de l'information notamment au moment de la transformation du corpus. Au vu des problèmes rencontrer lors du nettoyage et de l'analyse sémantique de Tropes, il a fallut trouver une solution afin de mener ces tâches le plus rapidement possible et de manière automatique.

Comme expliqué plus généralement dans la partie méthodologie de ce mémoire, nous avons dû procéder à un nettoyage du corpus afin de permettre un meilleur traitement par le logiciel Tropes, et limiter donc les erreurs, mais aussi pour que ce corpus puisse être réutiliser à l'avenir sans que les futurs réutilisateurs soient gêné par le bruit que pouvait comporter la version originale téléchargée en HTML sur le serveur Europresse. Le corpus de la version originale en HTML a donc été créer en trois autres nouvelles versions : la première en texte brut, pour une utilisation à court terme dans le logiciel Tropes, comportant uniquement le titre des articles et leur contenu ; la seconde en HTML, pour une utilisation à long terme dans d'autres analyses, incluant en plus des titres des articles et leur contenu, leur date de publication, leur source et leur numéro ; pour finir la troisième version en XML, pour la même utilisant et comportant les mêmes éléments que la version en HTML. Afin de procéder à ces transformation, il a fallut apprendre à utiliser le langage XSLT. Le langage XSLT a été conçu pour transformer des documents XML en document d'autres formats. Il permet de créer des règles de transformation[4] sur un document donné en entrée pour générer en sortie un nouveau document dans le format que l'on souhaite. Un document XSLT est en fait un document XML que l'on peut aussi appeler comme feuille de style XSLT. Une feuille de style XSLT possède une structure de base comprenant un prologue et un élément racine. Les autres éléments qui seront ajoutés sur cette feuille devront commencer par xsl :, comme par exemple, l'élément xsl:output, qui permet de préciser les caractéristiques de sortie du document à créer, ou l'élément xsl :template, qui définit le nom des règles à appliquer et sur quelle partie du document d'entrée la règle doit être appliquée, que l'on peut voir ci-dessous :

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
3      xmlns:xs="http://www.w3.org/2001/XMLSchema"
4      exclude-result-prefixes="xs"
5      version="2.0">
6      <xsl:output method="html" encoding="UTF-8"/>
7      <xsl:template match="body" name="regle1">
```

Figure 17: Exemple de structure de base d'une feuille de style XSLT

Aussi, afin de générer un document, il est important de connaître le langage utilisé pour tel ou tel format et de respecter les styles attribués à ce dernier. Par exemple, si le document que l'on souhaite générer sortira sous format HTML, il faudra structurer le document comme un document HTML :

4 *cf schéma d'une transformation par XSLT du cours Cours3-XSLT.pptx*

```
<xsl:template match="body" name="regle1">
    <html>
        <head>
            <title>Resultats de la transformation</title>
        </head>
        <body>
            <header>
            </header>
            <section>
                <xsl:for-each select="article">
                    <article>
                        <xsl:for-each select="header/div/p">
                            <xsl:if test="@class='titreArticleVisu rdp__articletitle'">
                                <h1><xsl:value-of select="."/></h1>
                            </xsl:if>
                        </xsl:for-each>
                        <xsl:for-each select="header/div/span">
                            <xsl:if test="@class='DocPublicationName'">
                                <h2><xsl:value-of select="."/></h2>
                            </xsl:if>
                            <xsl:if test="@class='DocHeader'">
                                <h3><xsl:value-of select="."/></h3>
                            </xsl:if>
                        </xsl:for-each>
                        <xsl:for-each select="section/div/div/p">
                            <p><xsl:value-of select="."/></p>
                        </xsl:for-each>
                        <xsl:for-each select="footer/div/div">
                            <xsl:if test="@class='publiC-lblNodoc'">
                                <p><xsl:value-of select="."/></p>
                            </xsl:if>
                        </xsl:for-each>
                    </article>
                </xsl:for-each>
            </section>
        </body>
    </html>
</xsl:template>
</xsl:stylesheet>
```

Figure 18: Exemple d'une feuille de style XSLT structurée pour une sortie HTML

Sur cette image, on distingue bien la structure attribuée au format HTML avec ses balises : <html>, <head>, </head>, <body>, <section>, … On aperçoit aussi au sein de cette structure que des éléments de langage XSLT apparaissent. La balise <section> va concerner tous les articles se trouvant dans le document HTML du corpus orignal. Pour chaque article, nous mettrons dans la balise <article> du document de sortie, les titres (‘*titreArticleVisu rdp_articletitle*’), le nom du document de publication (‘*DocPublicationName*’), etc.

On retrouve ces mêmes éléments dans la feuille de style XSLT pour la transformation du corpus HTML original en XML. La structure attribuée au format XML est respectée et les éléments du langage XSLT sont incorporés dans cette structure sans pour autant la déformer :

xsl:stylesheet

```
        <resultats>
            <article>
                <titre>
                    <xsl:for-each select="header/div/p">
                        <xsl:if test="@class='titreArticleVisu rdp__articletitle'">
                            <xsl:value-of select="."/>
                        </xsl:if>
                    </xsl:for-each>
                </titre>
                <source>
                    <xsl:for-each select="header/div/span">
                        <nom_lieu>
                            <xsl:if test="@class='DocPublicationName'">
                                <xsl:value-of select="."/>
                            </xsl:if>
                        </nom_lieu>
                        <date>
                            <xsl:if test="@class='DocHeader'">
                                <xsl:value-of select="."/>
                            </xsl:if>
                        </date>
                    </xsl:for-each>
                </source>
                <contenu>
                    <xsl:for-each select="section/div/div/p">
                        <xsl:value-of select="."/>
                    </xsl:for-each>
                </contenu>
                <numero>
                    <xsl:for-each select="footer/div/div">
                        <xsl:if test="@class='publiC-lblNodoc'">
                            <xsl:value-of select="."/>
                        </xsl:if>
                    </xsl:for-each>
                </numero>
            </article>
        </resultats>
    </xsl:template>
</xsl:stylesheet>
```

Figure 19: Exemple d'une feuille de style XSLT structurée pour une sortie XML