Académie de Montpellier
Université Montpellier II
Sciences et Techniques du Languedoc

ÉTUDE BIBLIOGRAPHIQUE DE

STAGE RECHERCHE MASTER M2

effectuée au Laboratoire d'Informatique de Robotique
et de Micro-électronique de Montpellier

Spécialité : **AIGLE**

Personnalisation de page web : application à l'amélioration de l'accessibilité du web

par **franck PETITDEMANGE**

Date : **17 mars 2014**

Sous la direction de **Marianne HUCHARD, Michel MEYNARD, Yoann BONAVERO**

Table des matières

1 Introduction et motivations

Le World Wide Web (www) est un réseau de ressources. Leur publication repose sur un langage universellement compréhensible et accepté par tous les ordinateurs : HTML. Historiquement conçu pour faciliter l'échange d'articles dans la communauté scientifique, la démocratisation du web a fait radicalement évoluer le contenu d'une page web, sans pour autant que le langage ne suive cette évolution. Ainsi, les auteurs de pages web ont détourné les pratiques de conception d'une page de manière anarchique. Ce manque d'homogénéité complique la compréhension du contenu publié sur le web par une machine. Ceci faisant perdre la propriété universelle de la toile voulue par son créateur Tim Berners Lee :

> « La puissance du web réside dans son universalité. L'accès à tous, quel que soit son handicap est un aspect essentiel »

Cette question d'universalité sous-tend la motivation de ce stage et les problématiques qui en découlent.

Nous nous intéressons à la personnalisation des pages web. L'objectif est de fournir des méthodes et des outils afin d'adapter une page suivant les souhaits d'un lecteur. Nous nous focalisons plus particulièrement sur une application permettant l'amélioration de l'accessibilité des pages web pour les personnes en situation de handicap visuel. Certains défauts de vision impliquent une restriction du champ visuel et des difficultés à voir certains contrastes. La consultation d'une page devient difficile puisqu'elle se limite à un fragment. De plus des informations peuvent devenir illisibles du fait d'une mauvaise combinaison de couleurs. L'outil doit acquérir des informations sur le profil du lecteur et, par exemple isoler et mettre en avant la barre de navigation de la page avec un contraste adapté au profil du lecteur. Cela nécessite de reconnaître la barre de navigation comme un élément manipulable en tant que tel.

L'adaptation d'une page web implique notre première problématique qui est la restructuration d'une page web. Nous souhaitons expérimenter une approche de restructuration basée sur les méta-modèles et les transformations de modèles. L'idée étant d'extraire la structure d'une page et d'en construire une représentation plus abstraite. Cela doit nous permettre de nous affranchir de la diversité de conception de ces dernières et des langages utilisés (versions d'HTML, CSS, etc.). A partir de cette représentation nous voulons appliquer des transformations, puis générer une nouvelle page conforme aux besoins du lecteur. L'une des problématiques du stage portera également sur l'acquisition des souhaits d'un lecteur au travers d'exemples de ces transformations, exemples eux-mêmes décrits dans cette représentation abstraite. Dans la section 2, nous précisons l'exploitation que nous souhaitons faire des modèles. Puis, en section 3, nous étudions différents méta-modèles de page en vue de la création d'un nouveau. La section 4 présente plus en détail la problématique d'extraction de structures dans une page web et différentes solutions. Lors du stage, l'un des objectifs consistera à adapter ou à concevoir une technique d'extraction de telles structures pour alimenter la représentation abstraite à partir de pages concrètes.

2 Éléments sur l'Ingénierie Dirigée par les Modèles

Dans le cadre de ce sujet, nous nous plaçons dans un contexte de restructuration (d'adaptation) d'une page web. Nous souhaitons définir cette restructuration à un niveau d'abstraction plus élevé, et pour cela il nous faudra construire un modèle (plus abstrait, avec plus de sémantique) d'une page web à partir de cette dernière. Un modèle de page se construit conformément à un méta-modèle. Nous donnons dans cette section les définitions nécessaires à la compréhension de l'approche traitée du point de vue de l'Ingénierie Dirigée par les Modèles.

Definition. « Un modèle est une description d'une partie d'un système écrit dans un langage formel »[10]

C'est une représentation simplifiée d'une partie d'un système, une abstraction du système étudié suivant un point de vue. Par exemple, une carte routière présente un plan suivant le point de vue du réseau routier, il existe plusieurs types de carte suivant ce que l'on veut étudier (chemin pédestre, chemin routier etc).

« Pour un observateur A, M est un modèle de l'objet O, si M aide A à répondre aux questions qu'il pose sur O » (Minsky)

La notion de modèle fait référence à un langage formel définissant les éléments conceptuels du modèle : le méta-modèle. La syntaxe et la sémantique du modèle sont conformes à un méta modèle.

Definition. « Un méta-modèle est un modèle qui définit un langage formel pour exprimer un modèle »[10]

Littéralement, c'est un modèle de modèle.

L'avantage d'utiliser un méta-modèle est de manipuler une représentation indépendante de la diversité de conception des éléments d'une page. En effet, les éléments ou concepts récurrents peuvent être structurés ou mis en forme de manière différente d'une page web à l'autre. [16] présente une méthodologie similaire à l'approche que nous voulons mettre en œuvre. Les auteurs souhaitent appliquer des traitements homologues sur des systèmes orientés objets implémentés dans différentes plateformes (C++, Java, ADA, etc). Pour cela, ils ont réalisé un méta-modèle de système objet dans le but d'effectuer des traitements indépendamment d'une représentation sous-jacente (du code source écrit dans un langage spécifique).

Nous voulons exploiter ce modèle (et son méta-modèle) de deux façons différentes. D'une part ils doivent nous servir de support pour appliquer un processus de transformation sur une page. D'autre part, ils doivent nous servir de support à l'extraction de règles de transformation à partir d'exemples. Ce dernier point est traité dans [12], nous comptons l'aborder techniquement par la suite dans le stage.

Definition. « Une transformation est la génération automatique d'un modèle cible depuis un modèle source selon une transformation définie »[10]

Une transformation est donc un programme, parfois donné sous forme d'un ensemble de règles qui décrivent comment un modèle du langage source peut être transformé dans un modèle du langage cible. Plus précisément, elle est la description de la manière dont une ou plusieurs constructions d'un langage source peuvent être transformées dans un langage cible. Plusieurs approches pour la transformation de modèle existent [9].

Suivant le langage du modèle source et du modèle cible, intervient une distinction (*cf.* figure 1). On parle de transformation endogène (rééchelonnement) lorsque les modèles sont exprimés dans le même langage. On parle de transformation exogène (traduction) dans le cas contraire.

Il s'agit de transformation horizontale quand les modèles source et cible possèdent le même niveau d'abstraction. Lorsque ce n'est pas le cas, il s'agit de transformation verticale (*cf.* figure 2).

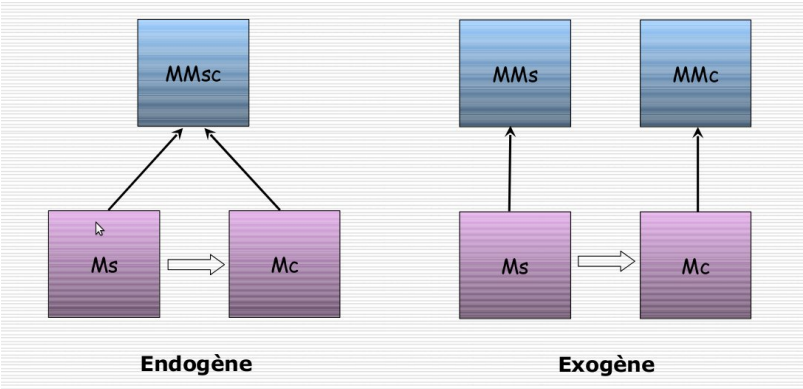

FIGURE 1 – Transformation endogène et exogène

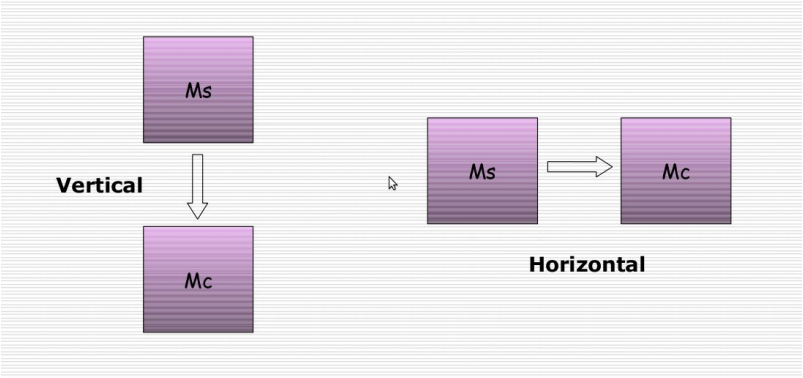

FIGURE 2 – Transformation horizontale et verticale

On distingue plusieurs transformations pour notre approche (*cf.* figure 3). Une première est une transformation verticale et exogène (*cf.* figure 3 transformation 1). Cette étape correspond à la construction d'un modèle source de page depuis une page web. Ce processus est l'objet de la section 4. Le modèle construit est conforme à un possible méta-modèle intermédiaire étudié dans la section 3 et que nous approfondirons lors du stage. Puis une transformation horizontale et endogène (*cf.* figure 3 transformation 2). Cette étape sera étudiée plus tard dans le stage. Elle intègre dans ses transformations les souhaits de personnalisation d'une page web par un utilisateur. L'acquisition de ces transformations fera l'objet d'un protocole d'acquisition des souhaits. Et pour finir, on retrouve une transformation verticale et exogène (*cf.* figure 3 transformation 3) qui va nous permettre de

construire la page web consultée par l'utilisateur suivant ses souhaits.

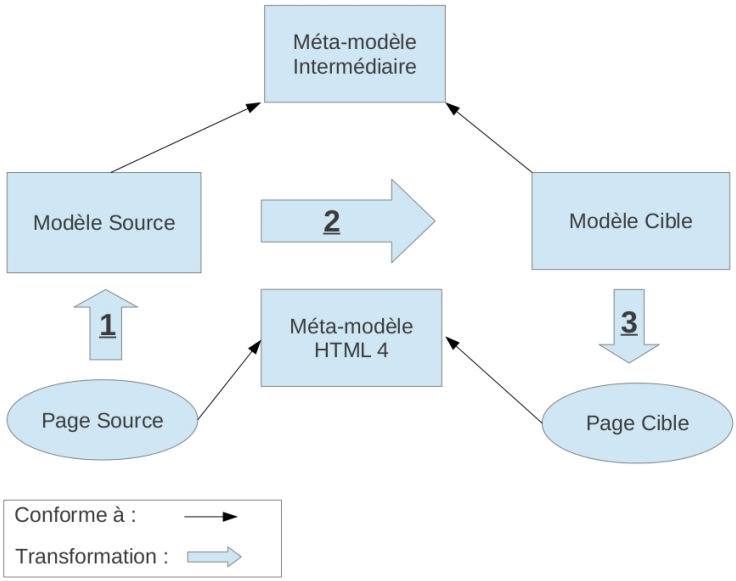

FIGURE 3 – Processus de transformation

3 Méta-modèle de page web

3.1 Introduction

Comme expliqué dans la section 2, les méta-modèles sont le support à l'expression de transformations. Ainsi, dans notre cas, les concepts décrits par le méta-modèle doivent comprendre les éléments, ou concepts que l'on souhaite transformer dans une page. Par exemple, si l'on souhaite modifier la couleur du texte de la barre de navigation, on doit pouvoir exprimer dans la transformation le concept de barre de navigation et le concept de couleur.

On s'intéresse à la conception d'un méta-modèle proche de la représentation qu'un lecteur peut avoir d'une page. Le but étant d'exprimer des transformations conformes aux souhaits du lecteur. On propose d'étudier et de comparer le langage standard de publication de page sur le web, HTML, dans les normes 4 et 5 mais aussi une taxonomie pour la description d'interface graphique ARIA.

3.2 HTML 4

HTML 4 [5] est un langage permettant la publication de contenu sur le web. C'est le langage standard actuel des pages web. Il permet de structurer le contenu et de lui associer une mise en forme. Le contenu peut être du texte, des images, ou plus généralement du multimédia. Ce contenu est organisé de manière hiérarchique en le découpant en section et sous-section.

Contenu Le contenu principal décrit dans les pages HTML 4 est un contenu textuel. Il peut également contenir du multimédia comme des images, des vidéos et applets (des programmes qui sont automatiquement chargés puis lancés sur la machine de l'utilisateur). L'inclusion de contenu multimédia se fait par l'élément générique : <OBJECT>. Il possède une collection d'attributs prédéfinis qui décrivent l'objet inclus dans la page. Le principal étant *type* décrivant le type de contenu des données (*e.g.* figure 4). La valeur de ces attributs n'est pas prédéfinie. Elle est interprétée librement par la machine qui charge la page web.

```
<object data="data/test.mpg" type="video/mpeg">
        Ceci est une vidéo
</object>
```

FIGURE 4 – Exemple contenu multimédia

Structuration générique HTML 4 propose un mécanisme générique pour la composition de contenu formant la structure des pages web. Ce mécanisme gravite autour des éléments de type <DIV> et de leurs attributs respectifs : *id* et *class*.

DIV Signifiant division, la balise DIV est utilisée comme conteneur générique, il peut contenir n'importe quel élément. Il est exploité pour :
- regrouper les éléments pour leur appliquer un style (une mise en forme particulière).
- signaler une section ou une sous-section.

id et class Chaque élément peut se voir attribuer un identifiant ou une classe d'appartenance. *id* assigne un nom à un élément. Ce nom est unique dans le document. *class* au contraire, assigne un ou plusieurs noms de classe à un élément. Un nom de classe peut être partagé par plusieurs instances d'éléments. Les identifiants et les classes sont des suites de caractères quelconques décidées arbitrairement par l'auteur du document.

Les éléments DIV utilisés conjointement avec les attributs *id* et *class* sont au cœur du mécanisme générique de structuration d'un document. DIV permet de diviser le contenu d'un document en sections et sous-sections (*e.g.* figure 6) pour décrire sa structure. Les balises <DIV> ayant une sémantique neutre, c'est l'auteur du contenu qui attribue (de manière arbitraire) un nom de *class* ou un *id*. L'*id* ou la *class* est associé à une mise en forme définie a priori. La mise en forme est définie au travers d'un langage : CSS[6] que l'on appelle feuille de style. CSS permet d'appliquer un ensemble de règles de style ou un agencement des éléments dans l'espace de la page. Par exemple, l'auteur peut déclarer une classe "aside" et définir que les éléments appartenant à la classe "aside" doivent être placés sur le côté droit de la page avec un fond blanc. Ce mécanisme est illustré par la figure 5. L'auteur associe à chaque <DIV> une *class* ou un *id* auquel s'applique une mise en page et une mise en forme définies par l'auteur dans une feuille de style CSS.

3.3 HTML 5

HTML 5 [8] étend HTML 4 en apportant de nouveaux éléments lexicaux. Ces nouveaux éléments apportent une sémantique standard et explicite à la structure d'une page ainsi qu'aux contenus.

```
<body>
        <div id="header" ></div>
        <div id="navigation_bar"></div>
        <div class="aside"></div>
        <div class="section">
                <div class="article"></div>
                <div class="article"></div>
        </div>
        <div class="aside"></div>
        <div id="footer"></div>
</body>
```

FIGURE 5 – Architecture page web HTML 4

Contenu HTML 5 fournit de nouveaux éléments comme <VIDEO>, <AUDIO> avec un ensemble d'attributs propres à chaque balise (a contrario de l'élément <OBJECT> de HTML 4). Les attributs spécifiques permettent de renseigner l'état d'un élément. Par exemple, la balise <AUDIO> possède un attribut spécifique *muted* indiquant si le son de l'élément audio est coupé ou non.

Structuration Les nouveaux éléments de HTML 5 spécifient donc une sémantique standard :

- SECTION : représente une section générique dans un document, c'est-à-dire un regroupement de contenu par thématique.
- ARTICLE : représente un contenu autonome dans une page, facilite l'inclusion de plusieurs sous-documents.
- NAV : représente une section de liens vers d'autres pages ou des fragments de cette page
- ASIDE : représente une section de la page dont le contenu est indirectement lié à ce qui l'entoure et qui pourrait être séparé de cet environnement
- HEADER : représente un groupe d'introduction ou une aide à la navigation. Il peut contenir

```
<body>
<div class="section" id="elephants-foret" >
        <h1>Les éléphants des forêts</h1>
        <p>Dans cette partie, nous abordons le sujet
moins connu des éléphants des forêts.</p>
        <div class="sous-section" id="habitat-foret" >
        <h2>L'habitat</h2>
        <p>Les éléphants des forêts ne vivent pas
dans les arbres mais au milieu d'eux.</p>
        </div>
</div>
</body>
```

FIGURE 6 – Exemple découpage en sections et sous-sections

des éléments de titre, mais aussi d'autres éléments tels qu'un logo, un formulaire de recherche, etc.
- FOOTER : représente le pied de page, ou de la section, ou de la racine de sectionnement la plus proche

La figure 7 montre un découpage explicite de la structure avec HTML 5 en opposition au découpage implicite de HTML 4 montré dans la figure 5.

3.4 ARIA

ARIA (Acessible Rich Internet Application) [7] est la spécification d'une ontologie décrivant une interface graphique. Elle fournit des informations sur la structuration d'un document et plus généralement elle décrit les éléments qui composent une interface au moyen d'un ensemble de rôles, d'états et de propriétés.

Rôle Les rôles permettent d'identifier la fonction de chaque élément d'une interface. Ils sont regroupés en trois catégories :
- Widget Roles : référence un ensemble de widget prédéfinis (alertdialog, button, slider, scrollbar, menu, etc.)
- Document Structure Roles : décrit les structures qui organisent un document (article, definition, entête, ect.)
- Landmark Roles : décrit les régions principales d'une interface graphique (main, navigation, search, etc.)

États et propriétés ARIA prend en compte l'aspect dynamique et interactif des éléments d'une interface. Elle permet d'associer des états et des propriétés aux éléments d'une interface. Un état est une configuration unique d'un objet. Par exemple, on peut définir l'état d'un bouton par l'état *aria-checked* qui peut prendre trois propriétés suivant l'interaction avec l'utilisateur : *true* - *false* - *mixed* . Dans le cas d'une checkbox, *true* indique si la checkbox est cochée, *false* si elle ne l'est pas et *mixed* dans le cas d'un ensemble de checkbox indique que certaines sont cochées.

Aria prévoit même un système d'annotation pour les objets ayant des comportements asynchrones. Par exemple, on peut indiquer par une annotation qu'un élément se met à jour de manière autonome.

```
<body>
        <header></header>
        <nav></nav>
        <section>
                <article></article>
                <article></article>
        </section>
        <aside></aside>
        <footer></footer>
</body>
```

FIGURE 7 – Exemple d'attribution de rôle

3.5 Discussion

L'étude ci-dessus présente de manière sommaire les concepts des langages HTML 4, 5 et de la norme ARIA. Au regard de notre problématique, nous pouvons voir quelques perspectives à la conception d'un méta-modèle.

HTML 4 fournit une sémantique pour décrire les éléments textuels d'une page et ses hyperliens. Cependant le mécanisme de structuration fourni est trop générique et ambigu. En effet, il n'est pas possible de délimiter explicitement les éléments d'une page par thématique. Le mécanisme d'imbrication hiérarchise le contenu alors qu'un élément n'est pas forcement sous-section de l'élément qui l'imbrique. Une page web contient des éléments qui ne sont pas en rapport avec le contenu d'un document, mais plutôt avec le site web (*e.g.* menu de navigation, un logo, etc.).

HTML 5 semble combler ces limites en apportant de nouveaux éléments lexicaux amenant une sémantique plus riche et surtout standard sur la structure d'une page et la nature du contenu. Elle permet donc de modéliser les limites de chaque partie d'une page suivant sa thématique (*e.g.* <SECTION>). Elle permet de décrire un contenu non-linéaire (*e.g.* <ASIDE>) et introduit également

des balises pour la description d'éléments propres à une page web comme des menus de navigation, bannière, logo, etc.

La norme ARIA intègre plus d'éléments pour décrire une page web et notamment une description du comportement des éléments d'une page web particulièrement utiles avec le développement des éléments graphiques appelés widget.

Un méta-modèle basé sur la syntaxe de HTML 5 semble être un compromis intéressant entre HTML 4 et ARIA. HTML 4 est peu adapté à l'expression de transformations reflétant l'intention d'un lecteur et ARIA est trop riche au regard de nos besoins immédiats. Les spécifications de HTML 4 sont disponibles sous forme de DTD [1]. Une DTD est un document décrivant la grammaire d'un langage. Il est simple d'en construire un méta-modèle. HTML 5 n'étant pas encore au rang de standard, aucune DTD n'est encore disponible, seule une définition exhaustive des éléments lexicaux et leurs fonctions est fournie par le W3C. Il n'y pas de grammaire clairement spécifiée. La tâche de réalisation d'un méta-modèle semble complexe pour HTML 5. Aria fournit, quant à lui, un modèle UML [2]. UML est un langage de modélisation graphique à base de pictogrammes [3]. Ce modèle UML est pour nous un méta-modèle directement exploitable. Ainsi nous pourrons utiliser le modèle fourni par ARIA pour la construction d'un méta-modèle. Ce dernier serait restreint aux éléments de description d'un document (*Document Structure Roles*) et (*Landmark Roles*). Suivant l'évolution de nos besoins de transformation nous pourrions facilement intégrer d'autres aspects, notamment les widget et comportements asynchrones des éléments d'une page. Ce méta-modèle intégrera les éléments de HTML 4, en particulier ceux décrivant le contenu textuel non modélisé avec ARIA (*e.g.* paragraphe).

4 Extraction de structure

4.1 Introduction

L'extraction correspond au processus de transformation 1 de la figure 3. On veut extraire les éléments de la structure d'une page qui correspondent aux différentes briques conformes aux éléments d'un méta-modèle de page défini a priori. La problématique est due au langage de conception des pages web qui manque de sémantique, rendant un processus d'extraction automatique difficile.

En effet, la conception des pages web s'articule autour d'un langage pour décrire la structure du document (HTML) et d'un langage pour décrire la mise en forme du document (CSS). Les pages sont constituées d'éléments hétérogènes : une page est constituée d'un ou plusieurs contenus principaux, d'un menu de navigation, de publicité, etc... Chacun de ces éléments représente une sous-structure de la page. Lorsque l'on regarde une page web depuis un navigateur, on constate que ces éléments sont structurés de façon sémantique, ils sont organisés selon leur sens. La difficulté dans la tâche d'extraction de la structure d'une page est due au manque d'expressivité du langage HTML. La norme actuelle de HTML (HTML 4) ne fournit pas de moyen de délimiter les éléments du document en fonction de leur sémantique. Par exemple, on ne peut pas délimiter de manière explicite la structure d'un menu dans une page avec ce langage. Le constat est que l'information de la structure d'une page apparaît principalement dans la mise en page. La structure d'une page est explicitée à travers l'utilisation de polices, de couleurs ou plus généralement d'éléments visuels pour caractériser les contenus qui ont la même signification.

1. Document Type Definition
2. Unified Modeling Language
3. wikipédia

Une approche étudiée dans la tâche d'extraction de la structure est un processus de segmentation. Ce dernier permet de découper une page en régions, on peut ainsi délimiter le contenu d'une page de façon explicite. Nous avons étudié également des approches de reconnaissance de structure syntaxique et fonctionnelle.

4.2 Segmentation de contenu

Comme présenté dans la section 3.2, HTML 4 fournit une sémantique neutre avec la balise <DIV>. On ne sait pas si elle délimite une section, sous-section ou si elle est utilisée pour appliquer un style. Le sens n'en est connu que par le concepteur de la page. La compréhension de la structure d'une page est alors implicite. Les auteurs de l'article « Extracting content structure for web pages based on visual representation » [2] proposent une approche pour rendre explicite la structure de la page.

Approche segmentation visuelle L'approche proposée par les auteurs [2] présente un algorithme de partitionnement basé sur les éléments de mise en forme des pages web. Le partitionnement extrait une structure qui regroupe les éléments d'une page sémantiquement proches en bloc (*e.g.* figure 8). Le postulat est que les éléments d'une page possédant des caractéristiques de mise en forme proches, tels que la police, la couleur, la taille, sont sémantiquement proches.

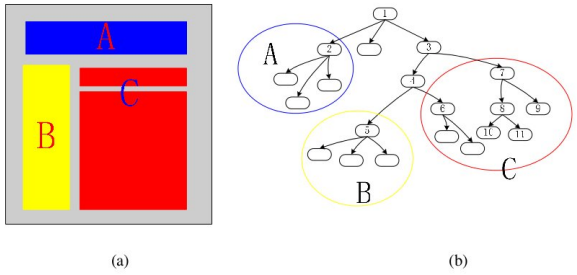

FIGURE 8 – Exemple de partitionnement, (a) page (b) DOM de la page [2]

L'algorithme exploite le DOM[4] de la page web. Le DOM est une API[5] pour les documents HTML (ou plus généralement XML). Il fournit une représentation arborescente d'un document et les moyens d'accéder à son contenu et sa mise en forme.

Le processus de segmentation, figure 9, se décompose en trois phases : un processus d'extraction de blocs, un processus de détection de séparateur et un processus de reconstruction.

Le processus d'extraction détecte les éléments du niveau courant du DOM susceptibles de former un contenu cohérent. Cette détection repose sur des séparateurs explicites : on sait que certains éléments délimitent le contour d'un contenu (par exemple les balises <DIV>). Mais elle repose également sur une fonction de distance visuelle comparant les nœuds parents et frères du nœud courant : une balise <DIV> a de grandes chances de délimiter un contenu sémantiquement différent du nœud parent si la couleur de fond est différente de celle de ce dernier. Pour chaque nœud,

4. Document Object Model
5. Application Programming Interface

l'algorithme vérifie s'il forme un bloc ou non. Si oui, il associe un degré de cohérence au bloc. Ce degré de cohérence est un indicateur de l'importance sémantique du bloc. Si non, il est appliqué le même processus aux enfants du nœud. Quand tous les nœuds du bloc courant sont extraits, ils sont mis dans un pool.

Des séparateurs entre les blocs sont ensuite détectés. L'algorithme détecte ici des séparateurs implicites, c'est-à-dire n'apparaissant pas dans la structure HTML. Les séparateurs implicites sont les espaces entre les blocs d'un pool. Un poids est attribué à chaque séparateur suivant son importance (par exemple, plus l'espacement entre deux blocs est grand, plus le poids sera élevé). Ce poids est un indicateur de différence sémantique entre les blocs adjacents. Plus le poids du séparateur est élevé entre deux blocs, plus leur contenu sera sémantiquement éloigné.

Une construction hiérarchique des blocs est créée. Cette construction hiérarchique repose sur le degré de cohérence attribué à chaque bloc.

Pour chaque nouveau bloc de la structure hiérarchique construite, l'algorithme teste le degré de cohérence attribué par rapport à un seuil de cohérence défini. Ce seuil est défini suivant la granularité de la structure que l'on veut en sortie de l'algorithme. Si le degré de cohérence n'est pas supérieur au seuil de cohérence, le bloc est de nouveau proportionné. La structure finale est construite après que tous les blocs soient traités.

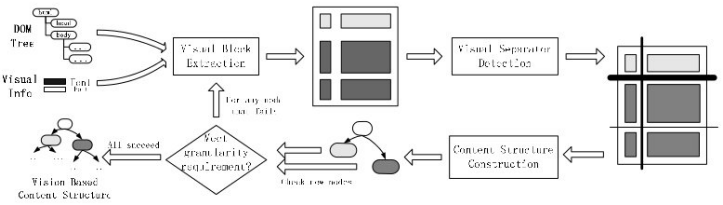

FIGURE 9 – Algorithme de segmentation [2]

En conclusion, l'algorithme regroupe les éléments d'une page en structure ayant une sémantique forte. Cette méthode nous permet de délimiter les différentes régions d'une page. Cependant l'inconvénient est que l'algorithme ne nous permet pas de connaître le rôle associé à chaque sous-structure isolée dans la page.

4.3 Compréhension de contenu

La segmentation du contenu n'est pas suffisante dans le processus de transformation. Pour construire notre modèle de page, nous avons besoin de faire correspondre les éléments d'une page web aux éléments du méta-modèle intermédiaire qui contient des concepts plus abstraits ou ayant un sens plus fort. Comme soulevé dans la section 3.2, la sémantique des éléments de structuration est neutre. On ne connaît pas le rôle des conteneurs génériques <DIV>. On ne peut pas faire correspondre les éléments d'une page directement avec les éléments d'un méta-modèle contenant des concepts du type de ceux que l'on trouve en HTML 5.

On propose d'étudier plusieurs approches pour reconnaître les sous-structures composant une page. Une première approche est une analyse syntaxique. On s'intéresse aux processus exploités en bio-informatique pour isoler des structures dans des séquences génomiques et déterminer quelles

sont leurs fonctions. On s'intéresse également à une analyse fonctionnelle des structures en analysant leurs propriétés.

4.3.1 Analyse syntaxique

Similarité de séquence L'un des postulats de base en bio-informatique est que deux séquences génomiques similaires correspondent à deux protéines présentant la même fonction [1]. En d'autres termes, trouver des séquences ayant des similitudes (syntaxiques) est un signe de proximité fonctionnelle. Appliqué à une page web, cela signifie que par exemple la structure syntaxique d'un menu d'une page web A est similaire à la structure syntaxique d'un menu d'une page web B. Il semble cohérent d'appliquer ce postulat à une page web. En effet, la publication de contenu sur le web se standardise par l'intermédiaire des scripts [17]. Ceux-ci génèrent des pages de manière automatique et standardisée (Wordpress, joomla, etc). En construisant une séquence représentative d'un élément de notre méta-modèle, il devrait être possible de déterminer la fonction d'un fragment de page web. Pour déterminer la similarité de deux séquences, les bio-informaticiens utilisent le concept d'alignement [14].

Definition. L'alignement est la mise en correspondance de deux séquences. Soit deux séquences X1 de taille n et Y1 de taille m dont la valeur est définie dans le même alphabet fini Λ. Un alignement est une correspondance entre les lettres de la première séquence et celles de la deuxième, sans en changer l'ordre, et en autorisant éventuellement des « trous ».

```
G A A T C _ T G A C
C A _ _ C G T _ A _
```

FIGURE 10 – Alignement possible des séquences X1=GAATCTGAC,Y1=CACGTA

La mise en correspondance repose sur trois types d'opérations élémentaires : la substitution, l'insertion, suppression. Plusieurs combinaisons d'alignements existent. Lorsque l'on souhaite comparer la similitude de deux séquences, la meilleure solution est celle qui minimise le nombre d'opérations d'insertions et de suppressions.

Il existe deux types d'alignements : local et global.

L'alignement global L'alignement global est conçu pour comparer des séquences sur toute leur longueur. Une méthode optimale pour trouver un alignement global maximal de chaîne de caractères est l'algorithme de Needleman-Wunsch [11].

L'alignement local L'alignement local est conçu pour rechercher dans la séquence A des régions semblables à la séquence B (ou à des parties de la séquence B). Une méthode pour trouver un alignement local maximal de chaîne de caractères est l'algorithme de Smith & Waterman [13].

Comme on le voit dans la figure 12 l'alignement global tente d'aligner les séquences sur toute leur longueur, tandis que l'alignement local se focalise sur les zones de forte homologie. L'alignement global est adapté à la comparaison de deux fragments de taille approximativement égales. Alors que l'alignement local permet de faire correspondre un fragment dans une plus grande séquence.

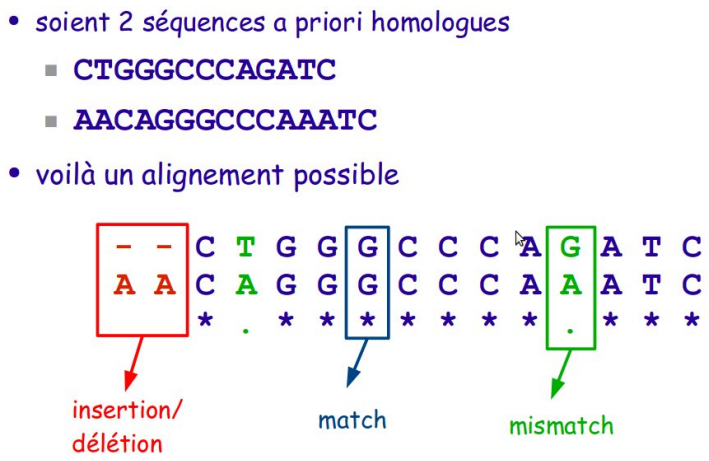

FIGURE 11 – opérations d'alignements de séquence

```
Global FTFTALILLAVAV
       F--TAL-LLA-AV

Local  FTFTALILL-AVAV
       --FTAL-LLAAV--
```

FIGURE 12 – Comparaison de séquences par alignement global et local

Similarité d'arbre Sur le web, une page web est accessible et manipulable sous la forme d'une structure arborescente que l'on appelle DOM (*cf.* section 4.2). Une autre approche dans l'étude de similarité de structure est la comparaison d'arbre. Cette approche consiste à trouver la plus petite ou la moins coûteuse séquence d'opérations d'édition (substitution, suppression et insertion) qui permet la transformation d'un arbre vers un autre.

Notons Λ un nœud vide. Une opération d'édition est écrite $b \to c$, où b et c sont soit un nœud, soit Λ.

- $b \to c$ est une opération de substitution si $b \neq \Lambda$ et $c \neq \Lambda$
- une opération de suppression si $b \neq \Lambda \doteq c$
- une opération d'insertion si $b = \Lambda \neq c$

Pour exprimer une séquence d'opérations élémentaires qui transforment l'arbre, on utilise le concept de mapping, introduit par [15]. Un mapping établit une correspondance un-à-un entre les nœuds de deux arbres ordonnés et qui préserve l'ordre des nœuds.

Definition. Un Mapping M de l'arbre T1 vers l'arbre T2 est un ensemble de paires ordonnées d'entiers (i, j), $1 \leqslant i \leqslant n1$, $1 \leq j \leq n2$, satisfaisant les conditions suivantes, pour tous $(i1, j1), (i2, j2) \in M$:

- i1=i2 si et seulement si j1=j2 (one-to-one condition)
- t1[i1] est à gauche de t1[i2], si est seulement si t2[j1] est à gauche de t2[j2] (préservation de l'ordre des nœuds frères)
- t1[i1] est un ancêtre de t1[i2] si et seulement si t2[j1] est un ancêtre de t2[j2] (préservation de l'ordre des ancêtres)

Definition. Soit M un mapping entre les arbres T1 et T2 décrivant des opérations de modification. S est l'ensemble de paires $(i, j) \in M$, D l'ensemble des nœuds T1[i] n'ayant pas de paire $(i, j) \in M$, et I l'ensemble des nœuds T2[j] n'ayant pas de paire $(i, j) \in M$. Le coût du mapping est donné par $|S|p + |I|q + |D|r$, où p est le coût des substitutions non identiques, q est le coût des insertions (1), r est le coût d'une suppression (1), le coût des substitutions identiques est 0.

Plusieurs classes de mapping existent, la différence se situe sur les restrictions des opérations autorisées dans le mapping. Les quatre principales sont : la distance d'alignement, la distance de sous arbre-isolé, la distance ascendante, la distance descendante [18]. [17] utilise un mapping de distance descendante pour isoler des sous-structures communes entre plusieurs pages afin d'en identifier des patterns (motif représentatif d'une structure). Cette approche semble intéressante pour partionner une page d'après notre méta-modèle intermédiaire (qui est lui-même un arbre).

4.3.2 Analyse fonctionnelle

L'article "Function-based object model towards website adaptation" [4] propose une analyse des fonctions d'un élément d'une page. Cette approche repose sur une segmentation du contenu d'une page a priori pour identifier chaque élément. Cette segmentation est similaire à celle de la section 4.2. Les éléments isolés par cette segmentation sont vus comme des objets possédant des propriétés (type d'agencement, nombre d'hyperliens, *etc.*). Les auteurs proposent un modèle : Function-based object. Ce modèle est une classification de chaque élément d'une page. Cette classification se fait sur la valeur des propriétés de chaque objet. Les principales propriétés spécifiées sont basées sur la présentation (type de média, agencement, *etc.*), les hyperliens (le nombre d'hyperliens, les pages vers lesquelles ils pointent), l'interaction avec l'utilisateur (uniquement affichage, soumission de contenu ?). L'extraction des propriétés fonctionnelles est faite principalement par analyse de la nature des balises et des attributs associés aux balises. Suivant la valeur des propriétés, les objets sont rangés dans différentes catégories : objet de type informatif, de navigation, d'interaction, de décoration, etc. L'analyse de la catégorie d'un objet se fait au moyen d'un arbre de décision (*e.g.* figure 13).

4.4 Discussion

L'alignement de séquence locale peut sembler une bonne approche. On peut comparer directement une séquence du méta-modèle à la page pour trouver la zone qui s'en approche le plus. Du point de vue de la complexité, pour deux séquences de taille m et n la complexité de l'algorithme de Smith & Waterman est $\mathcal{O}(nm)$. A l'échelle d'une page web le temps de calcul est acceptable. Aucune solution basée sur ce genre d'approche appliquée aux données du web n'a été trouvée. On peut cependant prendre en compte qu'une page web est fournie par les navigateurs web sous la

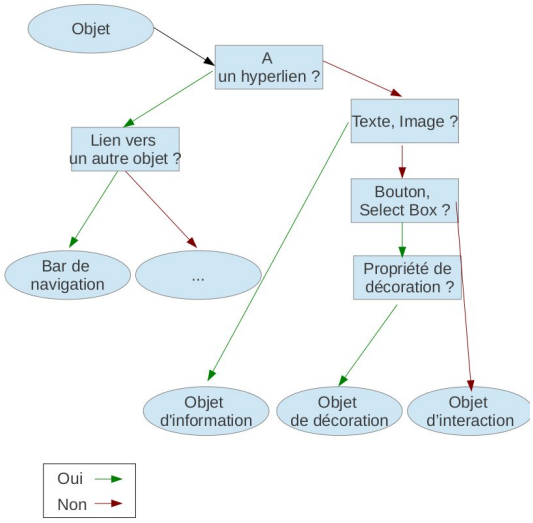

FIGURE 13 – Exemple d'arbre de décision basique

forme d'une structure arborescente. Cette solution implique donc un pré-traitement (aplatissement d'arbre). Cette solution paraît efficace pour trouver un élément de notre méta-modèle, mais trop complexe si l'on souhaite aligner tous les éléments de notre méta-modèle dans une page. La similarité d'arbre paraît plus naturelle comme approche. D'un point de vue complexité, soit *T1* arbre de la page et *T2* l'arbre du méta-modèle, et d un entier une distance définie, un mapping basé sur un distance descendante aurait un coût théorique de $\mathcal{O}(d^2 + |T1| + |T2|)$. La taille des données en entrée rend cette solution acceptable.

Les deux approches présentent un inconvénient majeur, elles sont trop dépendantes de la représentation sous-jacente des données. Les données publiées par HTML sont des formats de données semi-structurées [19]. Il n'y a pas de syntaxe formelle pour décrire un type de donnée. Un menu peut être écrit avec différentes syntaxes (*e.g.* figure 14). Cela complexifie une solution basée sur une analyse syntaxique. Une solution mise en oeuvre par [3] est une translation des éléments d'une page vers une représentation plus abstraite. Suivant leur classe d'appartenance les balises sont remplacées par un type plus générique (*e.g.* figure 15), par exemple les éléments <UL>, <OL>, <DL> sont différents types de liste, ils seraient remplacés par un type plus générique <LISTS>. Les solutions basées sur une analyse syntaxique requièrent la conception d'un pattern. La conception de ce genre de pattern est une opération coûteuse [17, 3].

L'approche par analyse fonctionnelle (section 4.3.2) semble limiter les inconvénients de l'analyse syntaxique. D'après les résultats de l'expérimentation faite dans l'article, l'approche fonctionne bien. Les éléments de la page sont bien identifiés conformément au modèle défini. Cependant le modèle est rustique et ne couvre pas tous nos besoins. Les auteurs de l'approche décrivent leur méthodologie pour la conception des différents objets du modèle. Nous devrions pouvoir l'étendre facilement à nos besoins.

```
<div class='menu'>
        <ol><a href='#'><span>Home</span></a></ol>
        <ol><a href='#'><span>About</span></a></ol>
        <ol><a href='#'><span>Services</span></a></ol>
        <ol><a href='#'><span>Contact</span></a></ol>
</div>
```

```
<div class='menu'>
<ul>
        <li><a href='#'>Home</a></li>
        <li><a href='#'>About</a></li>
        <li><a href='#'>Services</a></li>
        <li><a href='#'>Contact</a></li>
</ul>
</div>
```

FIGURE 14 – Exemple de différentes conceptions de menu

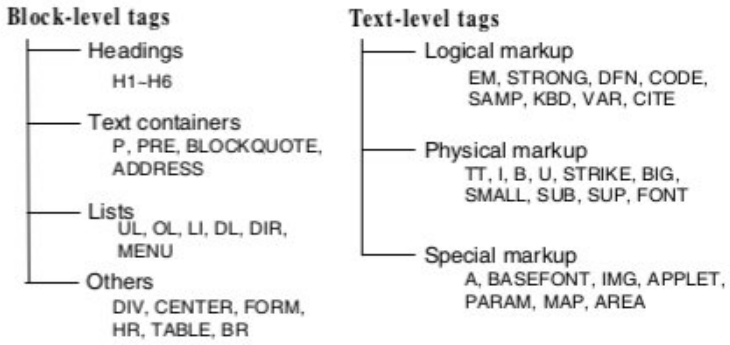

FIGURE 15 – Classification de balises [3]

5 Conclusion et futurs travaux

Cette bibliographique nous a permis de cerner les avantages d'un processus dirigé par les modèles dans l'environnement des pages web. Au travers de l'étude d'un possible support à la réalisation d'un méta-modèle, nous avons synthétisé les principaux concepts des langages de publication existants sur le web. Cette partie était nécessaire à la compréhension de ces langages et de leurs futures évolutions. Elle a également été l'amorce pour cerner la difficulté du traitement de la reconnaissance de types de structures et de leurs rôles dans une page de manière automatique. Ceci nous a permis d'étudier des solutions existantes et d'explorer d'autres pistes possibles à l'extraction de structures d'intérêt. Cependant d'autres approches peuvent encore être étudiées, notamment des approches stochastiques avec une application des Réseaux Bayésiens pour reconnaître le rôle d'une structure par une classification définie a priori. Des techniques de reconnaissance de formes issues du domaine du traitement des images semblent très intéressantes. En effet, ces techniques cherchent à regrouper des éléments (pixels) en structures de haut niveau (des objets). Cette problématique ressemble

fortement aux problèmes liés à l'extraction de structure dans l'environnement actuel des pages web mais ne sont pas applicables directement car les données en entrée ne sont pas les mêmes. En traitement d'image, les données en entrée sont des matrices, ici nous manipulons des arbres.

La suite des travaux va se diriger vers la conception d'un méta-modèle intermédiaire basé sur des éléments du modèle de ARIA. Puis nous définirons un processus d'extraction de structures et de leurs rôles dans une page web conformes au méta-modèle intermédiaire. Ce processus s' appuiera sur l'algorithme vu en section 4.2 et sur une technique d'analyse fonctionnelle vue en section 4.3.2. Nous nous emploierons par la suite à l'étude et à la réalisation d'un protocole d'acquisition des souhaits de personnalisation d'une page web. Ce protocole présente un grand intérêt pour nous puisque c'est grâce à cette partie que nous souhaitons rendre le web plus accessible aux personnes en situation de handicap visuel, en leur mettant à disposition un outil capable d'adapter une page web de façon personnalisée.

Références

[1] Kaizhong Zhang Bruc A.Shapiro. Comparing multiple RNA secondary structures using tree comparisons. *CABIOS*.

[2] Deng Cai, Shipeng Yu, Ji-Rong Wen, and Wei-Ying Ma. Extracting content structure for web pages based on visual representation. In *Web Technologies and Applications*, pages 406–417. Springer, 2003.

[3] Chia-Hui Chang and Shao-Chen Lui. IEPAD : information extraction based on pattern discovery. In *Proceedings of the 10th international conference on World Wide Web*, pages 681–688. ACM, 2001.

[4] Jinlin Chen, Baoyao Zhou, Jin Shi, Hongjiang Zhang, and Qiu Fengwu. Function-based object model towards website adaptation. In *Proceedings of the 10th international conference on World Wide Web*, pages 587–596. ACM, 2001.

[5] World Wide Web Consortium et al. HTML 4.01 specification. http ://www.w3.org/TR/REC-html40/, 1999.

[6] World Wide Web Consortium et al. Cascading Style Sheets. http ://www.w3.org/Style/CSS/, 2010.

[7] World Wide Web Consortium et al. Accessible Rich Internet Applications 1.0. http ://www.w3.org/WAI/intro/aria, 2014.

[8] World Wide Web Consortium et al. HTML 5 Specification. http ://www.w3.org/TR/html5/, 2014.

[9] Krzysztof Czarnecki and Simon Helsen. Classification of model transformation approaches. In *Proceedings of the 2nd OOPSLA Workshop on Generative Techniques in the Context of the Model Driven Architecture*, volume 45, pages 1–17, 2003.

[10] Anneke G Kleppe, Jos B Warmer, and Wim Bast. MDA explained : the model driven architecture : practice and promise. 2003.

[11] Saul B Needleman and Christian D Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3) :443–453, 1970.

[12] Hajer Saada. *Exploiting Model Transformation Examples for Easy Model Transformation Handling*. PhD thesis, Université Montpellier 2, 2113.

[13] Temple F Smith and Michael S Waterman. Identification of common molecular subsequences. *Journal of molecular biology*, 147(1) :195–197, 1981.

[14] Eric Sobel and Hugo M Martinez. A multiple sequence alignment program. *Nucleic acids research*, 14(1) :363–374, 1986.

[15] Kuo-Chung Tai. The tree-to-tree correction problem. *Journal of the ACM (JACM)*, 26(3) :422–433, 1979.

[16] Sander Tichelaar, Stéphane Ducasse, and Serge Demeyer. Famix and xmi. In *Reverse Engineering, 2000. Proceedings. Seventh Working Conference on*, pages 296–298. IEEE, 2000.

[17] Karane Vieira, Altigran S da Silva, Nick Pinto, Edleno S de Moura, Joao Cavalcanti, and Juliana Freire. A fast and robust method for web page template detection and removal. In *Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 258–267. ACM, 2006.

[18] Jason TL Wang and Kaizhong Zhang. Finding similar consensus between trees : an algorithm and a distance hierarchy. *Pattern Recognition*, 34(1) :127–137, 2001.

[19] Guillaume Wisniewski, Francis Maes, and Ludovic Denoyer. Modèle probabiliste pour l'extraction de structures dans les documents web. *Document numérique*, 10(1) :89–107, 2007.