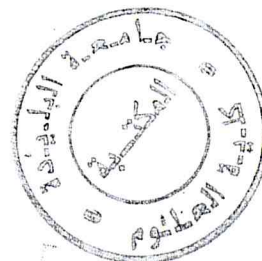


1110 2004 - 3 2 - 1
République Algérienne Démocratique et Populaire.
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique.

Université Saad Dahlab, Blida
USDB.

Faculté des sciences.
Département informatique.



**Mémoire pour l'obtention
d'un diplôme d'ingénieur d'état en informatique.**
Option : Système d'information

Sujet :

**La segmentation et l'extraction du contenu
sémantique d'une page web**

Présenté par : Mellak Sarah

Promoteur : Mme S.Khouas Oukid
Encadreur : Mr H.Hocini

Organisme d'accueil : Centre de Développement des Technologies Avancées C.D.T.A

Soutenue le: date soutenance, devant le jury composé de :

Président

Examineur

Examineur

- Novembre 2005-



Dédicaces

Je dédie ce modeste mémoire à :

La mémoire de mon feu grand père.

Ma très chère grande mère.

Mon très cher père Ali.

Ma très chère mère Hadjira.

Mon fiancé Kamel.

Mes chères soeurs Asma, Atika ; Khadija, Mariem et Soumia.

Mes oncles maternels Mohamed Amin, Mustapha, Abd El Hamid, et ma tente maternels Fatma Zohra.

Mes oncles paternels Mohamed, Mustapha, Sid Ahmed, Kamel, et ma tente paternels Yamna.

Mes cousins et cousines.

A ma deuxième famille la famille Sidi Moussa, en particulier mon père Abd El Wahab, et mes sœurs Souhila, Anissa, et Bahia.

A amies d'enfances Amina, Karima, et mes amies d'étude Fella , Hayette,

Samia, Fatiha, Nassima.

Remerciements

Nous tenons à adresser au Directeur Général du Centre de Développement de Technologies avancées (CDTA) nos sincères remerciements pour nous avoir accueilli dans son laboratoire et permis de réaliser ce modeste travail.

Nous avons eu un réel plaisir à travailler avec notre promotrice Mme S.Khouas Oukid, notre encadreur Mr H.Hocini et notre co-encadreur Mr A.R. Boumaaraf qui ont su rester à notre écoute dans les moments où nous avons besoin d'eux.

Nous tenons à remercier tous les professeurs de l'USDB qui nous ont accompagnés dans nos études supérieures, ainsi qu'à tous nos enseignants et enseignantes d'étude primaire, moyenne et lycéenne, et à tout ceux qui nous aiment.

Nous remercions tous ceux qui ont contribué à la réalisation de ce travail de prêt ou de loin.

Résumé

Le développement explosif de l'information sur le web rend critique et indispensable le développement de techniques et de modèles permettant de distinguer l'information importante et utile de celle inutile. Aussi, il est prouvé que cette classification de l'information non structurée du web facilite considérablement la recherche et l'exploration des données sur le web.

Ce travail entre dans le cadre du web mining et représente une contribution primaire à l'élaboration d'un système d'exploration de données sur le web. Dans cette phase nous nous intéressons à la recherche d'un modèle permettant la segmentation et l'extraction du contenu sémantique d'une page web.

Les principaux travaux exposés dans ce mémoire concernent une étude de l'Etat de l'art sur le web mining et les techniques de représentation de l'information sur le net. Plusieurs méthodes de segmentation ont été prises en considération et la méthode VIPS (Vision-based Page Segmentation) a été choisie et implémentée vu ses performances basées sur l'aspect visuel.

Mots clés: web mining, data mining, segmentation, classification

SOMMAIRE

Dédicaces	I
Remerciements	II
Résumé	III
Sommaire	IV
Liste des figures	VII
Liste des tableaux	IX
INTRODUCTION GENERALE	01

Chapitre 1: *Web Mining*

I. INTRODUCTION	05
II. METHODES DEFINITION	05
III. LES FACTUR DE WEB	06
IV. LES TACHES DU WEB MINING	08
IV.1. Mining les données des moteurs de recherche	08
IV.2. Analyse la structure des liens hypertexte	09
IV.3. L'extraction de la structure et le contenu sémantiques d'une page	10
web	
IV.5. Construire un Web multicouche et multidimensionnel	11
IV.6. Déterminer des informations pertinentes et appropriées	11
IV.8. Personnalisation d'information	12
V. LA RELATION DU WEB MINING AVEC D'AUTRE DOMAINES	12
V.1. Le web mining et IR	12
V.2. Le web mining et information extraction (IE)	12
V.3. Le Web mining et machine learning appliqué sur le Web	13
V.4. Web mining et l'agent paradigm	13
VI. LE PROCESSUS DU WEB MINING	14
VII. LES CATEGORIES DU WEB MINING	14
VII.1. Le web usage mining	14
VII.2. Le web structure mining	15
VII.3. Le web content mining	16
VII.4. La relation entre les différentes catégories	17
VIII. CONCLUSION	17

Chapitre 2: *Modèle Objet de Document (DOM)*

I. INTRODUCTION	19
------------------------	-----------

II. LES FAMILLES DE DOM	19
III. Le W3C DOM	21
III.1. Définition	21
III.2. Le modèle objet de document	22
III.3. Ce qui n'est pas le modèle d'objet de document	24
III.4. L'origine du modèle objet de document	25
III.5. Les niveaux de DOM	25
III.5.1. DOM niveau 0	25
III.5.2. DOM niveau 1 (DOM1)	26
III.5.3. DOM Niveau 2 (DOM2)	26
III.5.4. DOM niveau 3 (DOM3)	27
IV. LA MANIPULATION DU DOM AVEC JAVASCRIPT	28
IV.1. Les types de noeud les plus utilisés	30
IV.2. Les propriétés générales	31
IV.3. Les méthodes générales	32
V. CONCLUSIONS	33

Chapitre 3: Vision-based Page Segmentation (VIPS)

I. INTRODUCTION	35
II. ETAT DE L'ART	36
III. LE PRINCIPE DU VISION-BASED CONTENT STRUCTURE	37
IV. L'ALGORITHME VIPS	39
IV.1. Les propriétés visuelles	41
IV.2. L'extraction des Blocs Visuels	42
IV.3. Détection des Séparateurs Visuels	46
IV.4. La construction de la structure du contenu	48
VI. CONCLUSION	49

Chapitre 4: Démarche de Développement de l'outil VIPS

I. INTRODUCTION	51
II. DEMARCHE DE DEVELOPEMENT	51
III. ANALYSE ET SPECIFICATION DES BESOINS	52
III.1. Les cas d'utilisation (Use cases)	52
III.1.1. Les acteurs	52
III.1.2. Les cas d'utilisation principaux	52
III.1.3. Diagramme de cas d'utilisation général	56
III.2. Diagramme de séquence	57
III.3. Diagramme d'activité	62
IV. CONCEPTION	65
IV.1. Conception globale	65
IV.1.1 Architecture de notre système	65
IV.2. Conception détaillée	66
IV.2.1. Le module « chargement de page »	66
IV.2.1.1. Chargement du document HTML	67

	IV.2.1.2. La réinitialisation du processus	67
	IV.2.2. Le module « détermination du DOM »	67
nœuds	IV.2.2.1 L'accès et l'extraction des attributs visuels des	68
68	IV.2.2.2. Création de la représentation associée au nœud	
	IV.2.3 Le module « détermination du VIPS »	68
	IV.2.3.1. Nettoyage de données	68
	IV.2.3.2. L'extraction des blocs visuels	68
	IV.2.3.3. La détection des séparateurs	69
	IV.2.3.4. la construction de l'arbre sémantique	69
	IV.2.4. Interface utilisateur	69
	IV.2.5. Diagramme de classes	69
	V. IMPLEMENTATION	70
	V.1. Contexte matériel et logiciel	70
	V.2. Implémentation des modules	71
	V.2.1. Implémentation du module « chargement de page »	71
	V.2.1.1. Chargement du document HTML	71
	V.2.1.2. La réinitialisation du processus	71
	V.2.2. Implémentation du module « détermination du DOM »	71
nœuds	V.2.2.1 L'accès et l'extraction des attributs visuels des	71
nœud	V.2.2.2. Création de la représentation associée au	73
	V.2.3. Implémentation du module «détermination du VIPS»	74
	V.2.3.1. Nettoyage de données	74
	V.2.3.2. L'extraction des blocs visuels	74
	V.2.3.3. La détection des séparateurs	78
	V.2.3.4. la construction de l'arbre sémantique	79
	V.3. Interface utilisateur	79
	V.3.1. La vue commande	80
	V.3.2. La vue document	80
	V.3.3. La vue attribut	81
	V.3.4 La vue structurelle	81
	VI. TEST EST VALIDATION est et Validation	82
	VII. CONCLUSION	83
	CONCLUSION GENERALE	84
	Annexe A	85
	Références bibliographiques	92

Liste des Figures

- Figure 1.1** : Le Web mining, les sous catégories, et les données manipulés.
- Figure 1.2** : Les pages hub et autoritaire.
- Figure 2.1** : Le modèle objet de base implémenté dans les explorateurs.
- Figure 2.2** : Représentation graphique du DOM du tableau d'exemple.
- Figure 2.3** : La hiérarchie d'objet de document d'IE4.
- Figure 3.1** : La page Web et la structure de contenu à base de la vision d'une page. (d) et (e) les spécifications la structure de contenu à base de la vision.
- Figure 3.2** : L'algorithme VIPS.
- Figure 3.3** : (a) Page layout, (b) DOM tree de la page.
- Figure 3.4** : (a) Layout du bloc C (b) DOM tree du bloc C.
- Figure 3.5** : Vision-based content structure.
- Figure 3.6** : L'algorithme de visual block extraction.
- Figure 3.7** : Visual block extraction d'une sous page.
- Figure 3.8** : Processus de détection de séparateur d'une page simple.
- Figure 4.1** : Le processus de développement en cascade.
- Figure 4.2** : Cas d'utilisation principale
- Figure 4.3** : Diagramme de cas d'utilisation «chargement de page »
- Figure 4.4** : Diagramme de cas d'utilisation «Détermination du DOM »
- Figure 4.5** : Diagramme de cas d'utilisation « détermination du VIPS »
- Figure 4.6** : Diagramme de cas d'utilisation « Nettoyage de données »
- Figure 4.7** : Diagramme de cas d'utilisation « Extraction de blocs visuels »
- Figure 4.8** : Diagramme de cas d'utilisation «Détection des séparateurs »
- Figure 4.9** : Diagramme de cas d'utilisation « construction d'arborescence »
- Figure 4.10** : diagramme de cas d'utilisation général.
- Figure 4.11** : Diagramme de séquence pour le chargement de page.
- Figure 4.12** : Diagramme de séquence modification de la page charger.
- Figure 4.13** : Diagramme de séquence détermination du DOM.
- Figure 4.14** : Diagramme de séquence Parcours de l'arbre est visualisation des nœuds et les attributs visuels associés
- Figure 4.15** : Diagramme de séquence nettoyage de données
- Figure 4.16** : Diagramme de séquence extraction des blocs visuels
- Figure 4.17** : Diagramme de séquence détection des séparateurs.
- Figure 4.18** : Diagramme de séquence construction de l'arborescence

Figure 4.19: Diagramme de séquence parcours de l'arbre sémantique est visualisation des segments et les attributs visuels associés

Figure 4.20: Diagramme d'activité du système.

Figure 4.21 : Diagramme de collaboration des modules de système.

Figure 4.22 : Diagramme de composants de notre système.

Figure 4.23 : Les éléments du module chargement de page.

Figure 4.24 Les éléments du module détermination du DOM

Figure 4.25 : 22 Les éléments du module détermination du VIPS

Figure 4.26 Diagramme de classe correspondant à notre outil.

Figure 4.27 Vue globale de notre outil.

Figure 4.28 Vue commande de notre outil.

Figure 4.29 Vue document de notre outil.

Figure 4.30 Vue attribut de notre outil.

Figure 4.31 Vue structurelle de notre outil.

Figure 4.32 Chargement d'un nouveau document.

Figure 4.33 Visualisation des nœuds et de leurs attributs la partie DOM.

Figure 4.34 Construction de l'arbre sémantique.

LISTE DES TABLEAUX

Tableau 2.1 : Les familles de modèle objet.

Tableau 2.2: Les collections et leurs descriptions.

Tableau 2.3: Les types de noeud les plus utilisés

Tableau 2.4: Les propriétés générales.

Tableau 3.1: Les propriétés visuelles.

Tableau 3.2: Les règles heuristiques pour la phase de block extraction.

Tableau 3.3: Les différentes règles pour les différents nœuds.

INTRODUCTION GENERALE

Suite aux différents types de média qui sont apparus au cours de plusieurs années, l'avènement de l'Internet a fait un grand bouleversement dans le monde de l'information du point de vue:

- Accessibilité (à partir de n'importe quel terminal connecté à Internet).
- Diversité: on trouve plusieurs types d'informations (textes, image, sons, vidéos, etc.).
- Hétérogénéité aussi bien dans la forme (la structure et la présentation des documents) que dans le contenu des documents.

Aujourd'hui le Web joue un rôle de plus en plus important pour la distribution et la diffusion de l'information. Sachant cette importance, chaque organisation (entreprises, associations, universités, etc.) est accessible via son site web. Cela provoque d'un côté la croissance du nombre d'utilisateurs d'Internet (les internautes) dans le monde (119 millions en 1998, 333 millions en 2000, et à plus de 500 millions en 2001 [1]). D'un autre côté, Le Web est devenue un gigantesque espace d'information hétérogène, distribué à l'échelle planétaire, qui connaît une croissance exponentielle (320 millions de documents disponibles en 1998 [1], à 800 millions de documents en 1999 [1], et à plus de 2 milliards en 2000 [1]). Ces données sont largement sous-estimés en raison de la taille considérable du "Web invisible (par exemple les différents base de données qui alimente les différents sites web), qui est 500 fois plus importante que la taille du "Web de surface" [1].

Le progrès explosif du World Wide Web par des milliards de pages Web, créés avec HTML et XML, ou produits dynamiquement; nous a mis face à un grand défi, non seulement pour satisfaire les besoins du client, mais aussi pour les concepteurs du web. Quant un internaute se connecte pour rechercher une information, qui doit être précise, atteinte d'une manière rapide, pertinente et fiable. La même chose pour les concepteurs du Web: mieux connaître ses visiteurs mieux les servir (adaptation du contenu, de la forme et l'architecture). Mais avec les facteurs suivant est ce qu'on peut satisfaire les besoins des deux acteurs (les internautes et les concepteurs) [1]:

- ✓ La surinformation

D'après les statistiques le nombre de site web a été estimé a 40 millions en 2002, par conséquent il y a eu une explosion des systèmes d'informations

- ✓ les limites des moteurs de recherche actuels

- ✓ Information retrieval.
- ✓ Information extraction.
- ✓ l'adaptation automatique de page.

Parmi ces avantages on peut citer

- ✓ distinguer les différents sujets dans une page web qui présente plusieurs sujets à la fois (page multi-sujets).
- ✓ Distinguer l'information pertinente (qui est lié au sujet de la page) de l'information non pertinente (la publicité, les bars de menu, les copyrights, etc...).
- ✓ Adapter le contenu de la page en fonction des segments les plus visés par les internautes.

Dans ce travail nous allons donc nous intéresser à la segmentation et à l'extraction du contenu sémantique à partir des propriétés visuelles. Le mémoire sera présenté de la façon suivante :

- ✓ Le chapitre 1
Introduit le Web Mining, ses tâches, ses relations avec d'autres domaines, et ses catégories
- ✓ Le chapitre 2
Introduit les notions relatives au modèle objet de document (DOM) et la manipulation de celle-ci avec JavaScript.
- ✓ Le chapitre 3
A pour objet la présentation de la segmentation et l'extraction du contenu sémantique de la page web à partir des propriétés visuelles.
- ✓ Le chapitre 4
Dans ce chapitre nous présentons la démarche de développement de l'outil VIPS que nous avons réalisé.
- ✓ Dans la conclusion, nous résumerons l'apport essentiel de ce travail, et nous proposerons quelques perspectives qui pourront améliorer et enrichir notre prototype.

Les moteurs de recherche ne sont pas adaptés aux caractéristiques des documents Web. Ces moteurs de recherche permettent de retrouver des pages suivant différents critères, qui portent principalement sur le contenu textuel des documents. Ces systèmes traitent d'importants volumes de documents avec plusieurs centaines de millions de pages indexées. Ils sont néanmoins très rapides et sont capables de résoudre plusieurs milliers de requêtes par seconde. Malgré les moyens mis en oeuvre, les réponses fournies par ces systèmes s'avèrent généralement peu satisfaisantes, car elles sont trop nombreuses, bruitées, et peu précises. Ces moteurs privilégient la puissance (nombre de documents indexés, nombre de requêtes par jour) souvent au détriment de la qualité des résultats, car ils traitent une page HTML comme une seule unité.

✓ La multiplicité des accès à l'Internet

Aujourd'hui avec le développement technologique et les besoins des utilisateurs on peut accéder à l'Internet à partir de différents terminaux (ordinateur, téléphone portable, etc.) ce qui implique des caractéristiques différentes (débit, capacité de la mémoire, taille de l'écran, etc.), donc il faut adapter l'information en fonction du support à la place de la dupliquer, c.à.d qu'à partir d'un site unique, on génère la page en fonction du support

✓ L'apparition de nouvelles activités sur le Web comme le e-commerce (electronic commerce) et le e-learning (electronic learning) nécessite une haute personnalisation des sites web et le suivi et la gestion des informations émis par les utilisateurs.

Le web est une immense collection de pages qui a les caractéristiques suivantes :

- ✓ Distribution géographique
- ✓ Dynamacité
- ✓ Evolutivité
- ✓ Ouverture

Fournit une source riche et sans précédent pour appliquer les techniques du Data Mining (Web Mining), afin d'extraire des connaissances à partir des données. Le Web Mining a plusieurs tâches pour alléger la complexité du Web, parmi celles-ci, c'est "la segmentation et l'extraction de la structure sémantique du contenu". Quand un être humain regarde une page web, il essaie d'identifier les différentes parties (segments) de la page. La segmentation et l'extraction du contenu sémantique simule la perception d'un être humain, cette simulation se fait à partir de la structure logique² du document HTML et les propriétés visuelles (la taille, la couleur du font, taille de police, etc...). La segmentation et l'extraction du contenu sémantique à partir de propriétés visuelles peuvent apporter des avantages à plusieurs domaines comme

Chapitre 1

Web mining

I. INTRODUCTION

Le Web est un milieu où on peut accéder à une grande variété d'information stockée dans les différentes régions du monde, il est aussi une immense et dynamique collection de pages qui inclut des hyperliens innombrables et des volumes énormes d'information. Par conséquent il fournit une source d'extraction de données riches et sans précédent. L'expansion rapide du Web cause la croissance exponentielle de cette information, menant à plusieurs problèmes : une plus grande difficulté de trouver l'information appropriée, d'extraire la connaissance utile et de se renseigner sur les consommateurs ou les différents utilisateurs. Le Web Mining est un secteur de recherche approprié pour résoudre ces problèmes. Le Web Mining comme technologie peut jouer un rôle de plus en plus important en relevant les défis de développer le Web intelligent.

II. DEFINITION [1,2]

"Le Web Mining est l'utilisation des techniques de data mining afin d'automatiser la recherche et l'extraction de connaissances à partir des documents et services Web"

Le Web Mining peut être divisé suivant les données manipulées en trois champs:

- *Web Content Mining* : Etude du contenu d'un document web.
- *Web Structure Mining* : Etude de la structure de liens hypertextes des pages ou des site web.
- *Web Usage Mining* : Etude des habitudes de navigation

Le Web mining peut manipuler plusieurs types de données (Web Data).ces données sont classifiées en quatre catégories:

- *Content data*

Elles sont présentées à l'utilisateur final d'une manière convenable et structurée. Elles peuvent être simple comme : texte, images, ou des données structurées, telles que l'information rapporté à partir des bases de données.

- *Structure data*

Représentent la manière dont le contenu est organisé. Elles peuvent être des entités de données utilisées dans une page Web, tel que des étiquettes de HTML ou de XML, ou des entités de données employées pour rassembler un site Web, comme des hyperliens reliant une page à une autre.

- *Usage data* :

représentent la façon d'utilisation d'une page ou un site Web, telle que l'adresse IP d'un visiteur, l'heure et la date de l'accès, le chemin complet consulté, et d'autres attributs qui peuvent être inclus dans un fichier log¹.

¹Fichiers log : se sont des fichiers qui stockent tout les accès et les interactions avec le serveur.

- *User profile data* :

Fournissent des informations au sujet des utilisateurs d'un site Web. Un profil d'utilisateur contient des informations démographiques (telle que nom, âge, pays, état civil, éducation, intérêts etc.), aussi bien que des informations sur les intérêts et les préférences. Ces informations sont récupérées à partir des fiches ou des questionnaires, ou peuvent être inférées en analysant des fichiers log.

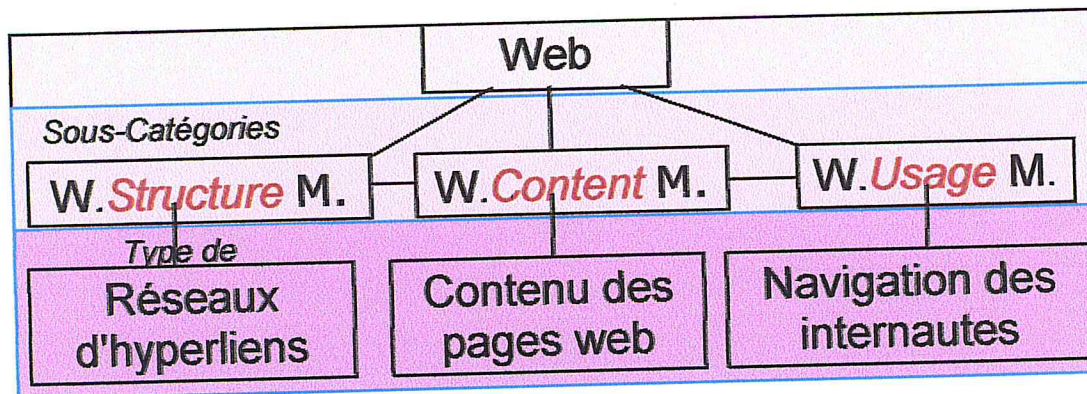


Figure 1.1 : Le Web mining, les sous catégories, et les données manipulés

III. LES FACTEURS DU WEB

Le Web offre un défi sans précédent au Web Mining, cela est dû aux caractéristiques suivantes

- La quantité de l'information sur le Web est énorme, et facilement accessible : Actuellement l'Internet fait interconnecter plus de 20 million de machines et met à la disposition des utilisateurs plus de 580 million de pages Web.
- La quantité d'information sur le Web est très large et diverse : On peut trouver des informations sur presque n'importe quoi.
- Information / données : presque tous les types existent sur le Web, par exemple, les tables, les textes, les images, et même pour les données multimédia (sons, vidéos), etc.
- Une grande partie des informations sur le Web est semi-structuré : ce type d'information tire son origine à partir de la structure du code HTML, cette structure satisfait les besoins des concepteurs des pages Web pour présenter l'information d'une manière simple afin de faciliter le visionnement humain et la lecture rapide.
- Une grande partie de l'information sur le Web est liée : Il y a des hyperliens non seulement lient des pages dans le même site Web, mais aussi interconnectent plusieurs sites Web. les liens servent comme outil d'organisation des informations et également d'indications de Hub / Authority des pages et les emplacements liés.

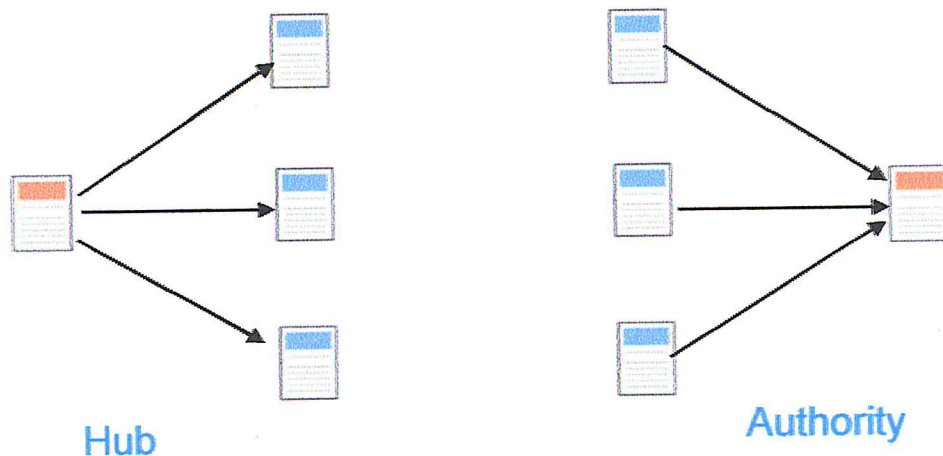


Figure 1.2 : Les pages hub et autoritaire.

- Une grande partie de l'information sur le Web est redondante : on remarque que la même information ou ses variantes peut apparaître à plusieurs emplacements (différents pages).
- Le Web est un environnement bruyant : Une page Web contient typiquement un mélange de beaucoup de genres d'information, par exemple : le contenu principal, annonces, panneaux de navigation, notices de copyright, etc. Un utilisateur donné se concentre généralement sur seulement une partie minuscule du Web, écartant le reste des données (données inintéressantes ou données non pertinentes) qui servent à perturber seulement les résultats de recherche désirés.
- Le Web se compose du "surface Web" et du "deep Web".
 - ✓ Surface Web : c'est la partie visible aux utilisateurs, c à d les pages qui peuvent être affichées sur l'écran d'un internaute en utilisant un browser.
 - ✓ Deep Web : c'est la partie du Web non visible, cette partie est représenté par les bases de données qui sont accessible seulement via une interface. En juillet 2000, les analystes ont estimé que le nombre des bases de données sur le Web est au moins 100.000. Ces bases de données fournissent les informations de haute qualité et bien maintenues, mais ne sont pas efficacement accessibles. Puisque les crawlers courants ne peuvent pas questionner ces bases de données. Donc elles restent invisibles aux moteurs de recherche traditionnels. Pour accéder efficacement au deep Web, nous devons intégrer toutes ces bases de données.
- Le Web est également un fournisseur de services : Beaucoup de sites et de pages Web permettent à des personnes d'effectuer des opérations avec des paramètres d'entrée (par exemple : envoyer des

données via un formulaire, chat, forums). En plus le e-commerce et le e-learning sont deux services qui exigent une haute personnalisation, l'identification et le suivi des utilisateurs.

- Le Web est dynamique : il constitue un émetteur d'informations fortement dynamique. certaines informations reçoivent des mises à jour régulières comme : Les news, les informations proviennent à partir des marchés des actions ou à partir des centres commerciaux. Par conséquent des mises à jour sont opérées au niveau des hyperliens et aussi au niveau des bases de données. Suivre et surveiller ces changements sont les questions les plus importantes.
- Le Web est une société virtuelle : Il est non seulement une collection de données et d'information, fournisseur de services, mais également il permet des interactions entre les personnes, les organismes et les systèmes automatiques, c.-à-d., les communautés.
- Le Web est un environnement complexe : La complexité de page Web dépasse de loin la complexité de n'importe quelle collection traditionnelle de documents textes. Bien que le Web fonctionne comme une énorme bibliothèque numérique, les pages dont il contient manquent de structure uniforme. D'ailleurs, le nombre énorme de documents dans cette bibliothèque numérique n'a pas été classé, ce qui fait l'opération de rechercher est extrêmement difficile.

IV. LES TACHES DU WEB MINING

Les tâches suivantes représentent les principaux problèmes qui doivent être résolus. Nous devons employer le Web mining d'une manière efficace pour l'amélioration des performances du Web et développer un Web plus intelligent.

IV.1. Mining les données des moteurs de recherche

Dans le processus de recherche, les moteurs de recherches se basent sur de grand index qu'ils possèdent. Ceux-ci permettent l'indexation des pages Web, la construction et la sauvegarde des indices de mots clés énormes pour aider à localiser des ensembles de pages Web qui contiennent des mots-clés spécifiques. En employant un ensemble de mots-clés et d'expressions bien précises, un utilisateur expérimenté peut rapidement localiser les documents appropriés.

Cependant, les moteurs de recherches actuelles souffrent de plusieurs insuffisances. D'abord, les résultats de recherche peuvent facilement contenir des centaines de milliers de documents. Ceci peut mener à des moteurs de recherches renvoyant un nombre énorme d'entrées de document, dont beaucoup sont seulement marginalement appropriées au sujet ou contiennent seulement des informations de mauvaise qualité. En second lieu, beaucoup de documents fortement appropriés peuvent ne pas contenir des mots clés qui définissent explicitement le sujet, phénomène connu sous le nom de " *problème de polysemy* " .

Le Web mining devrait être intégré avec le service des moteurs de recherche Web pour augmenter la qualité des résultats de recherche. Pour faire ainsi, nous pouvons commencer par agrandir l'ensemble des mots-clés par un ensemble de synonymes. Cette solution peut mener à une recherche parallèle (à partir du mot-clé et ses synonymes) pour obtenir un plus grand ensemble de documents pertinents. Aussi l'analyse des liens hypertextes et la dynamique du Web fournissent ainsi la base pour découvrir les documents de haute qualité.

IV.2. Analyse la structure des liens hypertexte

L'identification des pages Web "autoritaires" pour un sujet donné augmentera la qualité d'un résultat de recherche. Le secret de l'autorité se cache dans les liens hypertextes de page Web. Ces hyperliens contiennent une énorme quantité d'informations qui peut aider automatiquement à impliquer la notion de l'autorité. Quand l'auteur d'une page Web crée un hyperlien se dirigeant à une autre page Web, cette action peut être considérée comme action de validité de cette page.

La convergence collective vers une page donnée par différents auteurs sur le Web peut indiquer l'importance de cette page et en même temps c'est une preuve de l'autorité de cette page. Ainsi les liens hypertextes fournissent une source d'extraction riche. La structure des liens hypertextes a les propriétés suivantes :

- Les auteurs de page Web créent quelques liens pour des buts, tels que la navigation ou pour servir d'annonces payées. En générale, bien que la plupart des hyperliens visent des pages pertinentes, l'opinion collective domine.
- Une autorité appartenant à un intérêt commercial ou concurrentiel aura rarement un lien vers des pages Web des concurrents.
- Les pages autoritaires fournissent rarement des descriptions d'éclairage.

Les propriétés de structures de lien Web ont mené des chercheurs à considérer une autre catégorie importante de page Web, les hubs. Un hub est une page Web qui fournit des collections de liens aux pages autoritaires.

Ces pages peuvent être des listes de liens visant sur différents homepage, ou d'autre page Web. Un hub confère implicitement l'autorité de la page cible. Généralement, un bon hub cible généralement de bonnes autoritaires, et, réciproquement. Un rapport si mutuel entre les hubs et les autoritaires aide à extraire des pages Web de haute qualité. Les algorithmes les plus célèbres sont PageRank et HITS et il y a quelques moteurs de recherche disponibles dans le Web, tels que Google, qui sont construits autour de telles méthodes. En analysant des liens hypertextes et l'information textuelle, ces systèmes peuvent produire des résultats de recherches de meilleures qualités.

de la dynamique du Web. L'accès à cette information exige des techniques d'extraction sophistiquée.

L'analyse du Web log peut aider à établir des services adaptés aux besoins des utilisateurs. Puisque les données du Web log fournissent des informations sur la popularité des pages, les méthodes d'accès, la nature du trafic, la réaction des utilisateurs vers les différents types de conceptions, celles-ci peuvent être utilisées pour améliorer la structure et le contenu des pages ou des sites Web.

IV.5. Construire un Web multicouche et multidimensionnel

Nous pouvons construire et employer le Web multidimensionnel dans trois étapes principales:

- D'abord, nous analysons systématiquement un ensemble de pages Web, cette analyse couvre le contenu, la structure, les liens hypertextes, et les modèles d'utilisation.
- Grouper un ensemble de pages Web similaires dans un cluster, qui est appelé une page sémantique.
- Traiter une page individuellement, si elle forme un cluster indépendant, comme une page sémantique.

L'analyse produit alors un descripteur pour chaque page sémantique, qui contient des propriétés qui seront critiques pour la construction du Web directory multicouche et multidimensionnel. La première couche est le Web lui-même car on ne peut pas créer un warehouse qui contient tout les documents Web, la deuxième couche est la base des descripteurs. Le Web directory peut faciliter les opérations d'interrogation, d'extractions, et d'analyse.

IV.6. Déterminer des informations pertinentes et appropriées

Souvent les utilisateurs recherchent une information bien spécifique en utilisant les moteurs de recherche. Quand un utilisateur veut lancer le processus de recherche, la première chose à faire c'est la détermination du mot clé qui sert comme une entrée. Le résultat est une liste de pages rangés en fonction de leurs similitudes au mot clé. Aujourd'hui les outils de recherches ont certains problèmes comme:

- la basse précision: cela est due à la qualité des résultats de recherche qui contiennent beaucoup d'informations non pertinentes.
- l'incapacité d'indexer toutes les informations disponibles sur le Web.

IV.7. Créer de nouvelles connaissances hors de l'information disponible sur le Web

On a plusieurs types de données sur le Web. Le problème est comment tirer une bonne connaissance à partir de ces données pour avoir la bonne décision, avec la prise en compte de toute la caractéristique du Web et la nature de données. Cela peut conduire à mieux servir les utilisateurs.

IV.3. L'extraction de la structure et le contenu sémantiques d'une page Web

L'extraction des structures de page Web et du contenu sémantique peut être difficile, vu les limitations courantes dans le domaine d'analyse du langage naturel. Cependant, il y a des méthodes qui peuvent identifier une grande partie de telles structures. Alors un système d'extraction de structure de page peut analyser une page Web pour voir comment ajuster les segments d'un contenu dans une des structures. Examiner le comportement des utilisateurs peut augmenter et améliorer la qualité des structures et du contenu extraits de la page Web.

L'analyse détaillée des mécanismes d'extraction de page Web indique que les différents genres de pages ont différentes structures sémantiques. Par exemple, le homepage d'un département, le homepage d'un professeur, et une page d'annonce de travail peuvent tout avoir des structures différentes.

Pour identifier une structure pertinente et intéressante, un expert indique manuellement cette structure pour une classe donnée de page Web. Ensuite il y a le développement des techniques pour induire automatiquement une telle structure d'un ensemble d'exemples pré marqués de page Web. Les réalisateurs peuvent employer les méthodes d'extraction de structure et de contenu de page Web pour l'extraction automatique à partir des classes de page Web, des structures sémantiques possibles, et toute autre information sémantique. L'identification de classe de page aide à extraire les structures et le contenu sémantiques. L'extraction de telles structures aide à confirmer à quelle classe les pages extraites appartiennent. La structure sémantique et l'identification du contenu d'une page Web augmenteront considérablement l'analyse détaillée du contenu.

IV.4. Le mining du Web dynamique

Le Web subit souvent des changements dans le contenu et même dans la structure ce qui le rend dynamique. Pour garder l'historique de certaines informations avec la grande masse de celles-ci au niveau du Web, cette opération peut être impossible. Mais cela peut être faisable à partir des fichiers logs. Avec cette technique, on peut découvrir des modèles d'accès aux pages Web. Les régularités d'analyse et d'exploration des fichiers log peuvent :

- Augmenter la qualité et la livraison des services d'information d'Internet pour l'utilisateur final.
- Améliorer les performances des systèmes de serveurs web.
- Identifier les clients potentiels pour le commerce électronique, etc.

Un serveur web enregistre habituellement une entrée de Web log pour chaque accès de page Web. Cette entrée inclut les informations les plus importantes tel que l'Url visité, l'adresse IP et sans oublier de garder une empreinte de temps. Les serveurs rassemblent un nombre énorme de Web log d'accès Web. Les sites Web populaires peuvent enregistrer des centaines de méga-octets chaque jour de Web log. Les bases de données de Web log fournissent des informations riches au sujet

IV.8. Personnalisation d'information

Ce problème est souvent associé au type et à la présentation d'information. Quand un concepteur présente une page Web, il essaie de l'organiser de façon quelle soit agréé par au moins la majorité des visiteurs de cette page. La personnalisation est très important surtout pour les sites de e-commerce et de e-learning, ou d'une façon générale pour les sites Web qui ont des points d'interactions avec leurs visiteurs (par exemple via des formulaires). Mais cela n'est faisable que si et seulement si on a des renseignements sur les différents types d'utilisateurs, leurs préférences, les emplacements visités, etc.

V. LA RELATION DU WEB MINING AVEC D'AUTRE DOMAINES [3]

Le web mining rentre en conflit avec plusieurs domaines comme information retrieval (IR), information extraction (IE), machine learning, et agent paradigm. Mais le web mining est différent de tous ces domaines:

V.1. Le web mining et IR

Certains ont réclamé que la découverte de ressource ou de document (IR) sur le Web est une instance du Web (content) mining et d'autres associent le Web mining à IR intelligent. Actuellement IR fait la récupération automatique de tous les documents pertinents. IR a comme buts primaires d'indexer le texte et rechercher les documents utiles dans une collection. Maintenant IR inclut la modélisation, la classification et la catégorisation de documents, les interfaces utilisateur, le filtrage et la visualisation des données, etc. les tâches qu'elles peuvent être considérées comme des instances du Web mining sont la classification et la catégorisation des pages Web, celles-ci peuvent être employées pour l'indexation. Donc on peut dire que le Web mining est une phase du processus du (Web) IR. Cependant il faut noter que les tâches d'indexation n'emploient pas forcément les techniques du Data mining.

V.2. Le web mining et information extraction (IE)

IE a pour but de transformer une collection de documents, habituellement avec l'aide des systèmes de IR, en information facilement assimilée et analysée. IE vise à extraire des faits pertinents à partir des documents, tandis qu'il est intéressé par la structure ou la représentation d'un document. IR considère le texte dans le document juste comme un sac de mots désordonnés. En général IE travaille à un niveau plus fin sur les documents que IR.

Construire les systèmes d'IE manuellement n'est pas une tâche simple pour un milieu si dynamique et divers tel que le contenu Web. En raison de cette nature, la plupart des systèmes d'IE se concentrent sur des sites Web spécifiques. D'autres emploient les technique de machine learning ou de Data mining (Web mining) pour avoir des modèles ou des règles d'extraction pour les documents Web. Donc, le Web mining fait partie du processus d'IE. D'un autre point de vue, les résultats du processus d'IE peuvent être sous forme de base de données structurée, une

compression, un sommaire du texte, ou des documents originaux. Donc on peut considérer que IE est un genre d'étape de prétraitement dans le processus du Web mining. Celle-ci se trouve après le processus d'IR et avant que les techniques du Web mining sont exécutées. Dans une vue semblable, IE peut également être employé pour améliorer le procédé d'indexation. Réciproquement, on peut également dire que IE est une instance du texte ou du Web mining, puisque le sommaire est une nouvelle forme d'information qui n'existait pas avant. Cependant, le Web mining est employé pour améliorer l'IE à partir du Web (le Web mining fait partie du processus d'IE)

V.3. Le Web mining et machine learning appliqué sur le Web

Le Web mining n'est pas semblable aux techniques d'apprentissage appliquées sur le Web. D'une part, Il y a des applications de machine learning qui ne sont pas une instance du Web mining, un exemple de ces applications est la recherche du plus court chemin à prendre à partir d'un lien ou d'une adresse Url. D'autre part, il y a des méthodes utilisées pour le Web mining qui n'ont pas de relations avec machine learning comme la détermination des pages autoritaires et des hubs. Mais les deux domaines sont étroitement liés, machine learning soutient et aide le Web mining car elle peut être appliqué au cours du processus du Web mining. Les recherches récentes montrent que l'application des techniques de machine learning peut améliorer le procédé de classification des textes comparé aux techniques IR traditionnelles. En bref, le Web mining intersecte avec l'application de machine learning sur le Web.

V.4. Web mining et l'agent paradigm

Le Web mining a un rapport étroit avec les softwares agents ou les agents intelligents. En effet certains de ces agents accomplissent des tâches au cours du processus de Web mining. Il y a trois sous catégories de software agents: users interface agents, agents distribués, et mobile agents. Les sous catégories qui sont intéressantes pour le Web mining sont les users interface agents et les agents distribués. Les users interface agents essaient de maximiser le rendement de l'interaction des utilisateurs avec l'interface. La technologie de distributed agents est concernée par la résolution des problèmes par un groupe d'agents et les relevant agents. Il y a deux approches fréquemment utilisées pour développer les agents intelligents qui aident les utilisateurs à trouver et rechercher l'information appropriée, l'approche content-based et l'approche collaborative. Dans l'approche content-based, le système cherche l'information en se basant sur l'analyse du contenu des documents avec l'utilisation des performances des utilisateurs. Dans l'approche collaborative, le système essaie de trouver les utilisateurs avec des intérêts similaires. Le système fait ceci en analysant les profils d'utilisateur et les sessions ou les transactions. Principalement cette approche utilise usage data. Dans ce contexte on peut classer l'approche content-based autant que Web content mining et l'approche collaborative autant que Web usage mining, mais l'approche collaborative peut être aussi combiné ou utilisé avec le Web content mining.

VI. LE PROCESSUS DU WEB MINING [1,3]

Le processus du Web mining peut être décomposé en quatre étapes qui seront présentés comme suit:

- Retrouver les ressources: c'est l'étape de recherche et d'assemblage des données Web.
- La sélection et prétraitement de l'information: cette étape concerne le nettoyage de l'information et la construction de la base de test.
- La généralisation: la découverte automatique des différents modèles généraux.
- L'analyse : validation et/ou interprétation des modèles extraits.

L'étape de Retrouver les ressources veut dire le processus de recherche des données. Ces données peuvent être un contenu textuel comme: les messages électroniques, les newsgroup, le contenu textuel des documents HTML (obtenus en enlevant les balises HTML), ou le code source des fichiers HTML. On peut également inclure des données choisies manuellement ou des données extraites à partir des bases de données, etc. L'étape de sélection et prétraitement est un genre de processus de transformation des données originales définies à l'étape précédente. Cette étape a comme résultat des données bien présentées sous la forme désirées. Dans l'étape 3 la machine learning et le Web mining sont typiquement employés pour la généralisation. Nous devrions également noter que l'être humain joue un rôle important dans le procédé de knowledge discovery puisque le Web est un milieu interactif. C'est particulièrement important pour la validation et/ou l'interprétation dans l'étape 4. Les étapes 1, 2, 4 dans cet ordre sont les plus utilisées.

VII. LES CATEGORIES DU WEB MINING [1, 3,4]

Le web mining se divise en trois catégories en se basant sur les parties ou les données manipulées : Web content mining, web usage mining, et web structure mining.

VII.1. Le web usage mining

Le rôle du web usage mining est de comprendre et mettre un sens aux données générées par les sessions ou par le comportement des utilisateurs. Contrairement aux web content mining et web structure mining qui utilisent les données primaires, le web usage mining utilise des données secondaires dérivées à partir des interactions de l'utilisateur avec le web. Le Web usage mining utilise des données comme les fichiers logs (web server access logs, proxy server logs, browser logs), les sessions ou les transactions des utilisateurs, les cookies, les clics de souris, les défilements, ou toutes autres résultats conclues à partir des interactions. En bref ce sont les usage data et les profile data. Le processus de web usage mining peut être classé en deux approches. La première organise les usage data du serveur web dans des tables relationnelles avant d'exécuter les techniques de Web mining. La seconde utilise directement les fichiers log avec l'utilisation des

techniques de prétraitement. Les issues de la qualité des données sont très important ici. Il y a plusieurs problèmes surtout en ce qui concerne l'identification des sessions d'utilisateurs, une seule adresse IP et plusieurs utilisateurs ou vice versa, ce problème peut être résolu par l'utilisation des cookies ou l'introduction des identifiants pour les sessions, etc. un autre problème est que Les serveurs clients et les proxy sauvent des copies locales des pages qui ont été consultées par l'utilisateur. L'utilisation des boutons "arrière" et "avant" sur un navigateur peut accéder à la copie locale au lieu de l'apporter à partir du serveur. Ce problème peut être réglé en utilisant un petit délai pour les pages ce qui force le navigateur de demander une nouvelle version, mais cela peut trop occuper le serveur. Parfois le web usage mining utilise quelques domaines comme : web content, site topology, et concept hiérarchies.

Les applications du Web usage mining peuvent être classifiées en deux principales catégories, apprendre (learning) les profils des utilisateurs dont le contexte d'une interface adaptée et apprendre (learning) les méthodes de navigation. Les utilisateurs web ont besoin d'autres techniques pour prendre en compte leurs besoins et leurs préférences, cela peut être faisable avec la combinaison du Web content mining. D'autre part, les concepteurs doivent être intéressés par ce genre de techniques pour mieux adapter (contenu, design, structure, etc.) leurs sites web aux besoins d'utilisateurs. L'implication des techniques de web usage mining dans le e-commerce a apporté plusieurs bénéfices:

- Concevoir les stratégies de vente de produits.
- Evaluer les campagnes promotionnelles.
- Viser les bons groupes d'utilisateur basés sur leurs modèles d'accès.
- Prévoir le comportement d'utilisateur à partir des règles précédemment instruites et le profil d'utilisateurs.
- Présenter l'information dynamique aux utilisateurs basés sur leurs intérêts et profils.

Le e-commerce n'est pas le seul domaine qui a tiré des bénéfices du web usage mining. Celui-ci peut aider à :

- Améliorer la présence et la structure du web par: la détermination de la meilleure manière de structurer le site Web ;
- Identifier des "liens faibles" pour l'élimination ou le perfectionnement ;
- Rendre l'information la plus visitée la plus facile à y accéder.

VII.2. Le web structure mining

Le Web structure mining a comme fonction de découvrir le modèle de la structure des hyperliens du web. Le modèle est basé sur la topologie des hyperliens avec ou sans description. Cette catégorie est également appelé "analyse des liens". La nature des hyperliens donne une grande surface d'extraction d'informations utiles.

Avec l'analyse des liens on peut découvrir des types spécifiques de pages se basant sur les liens entrants et sortants comme les hubs et les autoritaires. L'idée de base est ce que les hubs et les autoritaires sont des pages interconnectées par des liens unidirectionnel ou bidirectionnel. Les hubs sont des pages qui contiennent des liens vers les pages autoritaires. Les pages autoritaires sont les pages qui contiennent les informations sur le sujet voulu par l'utilisateur. Une bonne hub page est une page qui pointe sur des bonnes pages autoritaires. Une bonne page autoritaire est pointée par des bonne hubs pages. Le web usage mining est aussi utile pour générer des information comme la qualité des pages par rapport au sujet recherché, la structure intéressante du Web, la classification des pages par rapport aux différents sujets, la similarité ou la relation entre les différent page Web, et découvrir les page miroirs. Trouver une page pertinente signifie trouver d'autre pages pertinentes qui sont liées avec cette page.

Certains algorithmes qui sont proposés pour modeler la topologie ou la structure du web comme HITS, Page Rank et l'amélioration du HITS par l'ajout des informations sur le contenu à la structure des hyperliens. Ces modèles sont principalement appliqués pour calculer la pertinence de chaque page web (par exemple cette technique est utilisée par Google). Ces modèles peuvent découvrir les micro communautés (Les communautés web peuvent être décrites comme collection de pages Web tels que chaque noeud membre a plus d'hyperliens au sein de la communauté qu'en dehors de la communauté).

VII.3. Le web content mining

Web content mining: représente la recherche de l'information utile à partir contenu, données; ou documents Web. Le Web fournit plusieurs sources de données. Le contenu du Web est composé de plusieurs types de données: textuel, image, audio, vidéo, métadata de même que les hyperliens. Les recherche récentes dans le data mining avec la manipulation de multi types de données s'appelle multimédia data mining, cette branche de data mining peut être considérée comme une instance du Web content mining. Cette ligne de recherche reçoit moins d'attention que les recherche qui se basent sur les données textuel ou les hyperliens. Le contenu Web se composent de données non-structurés comme le texte libre, semi-structurés comme les documents HTML, et les données structurés comme les données qui résident dans les bases de données qui génèrent des pages Web. Les recherches qui appliquent les techniques de data mining sur le texte non structuré s'appelle le text mining, donc on peut considérer le text mining comme une instance du Web content mining.

On peut voir les recherches faites dans le contexte du Web content mining de deux point de vue différents : IR view et DB (base de données) view. Le but de web content mining à partir du IR view est surtout d'aider et de filtrer l'information en utilisant le profile d'utilisateur. Alors celui du web content mining à partir du DB view est principalement d'essayer de modeler les données sur le web et les intégrer pour que des queries plus sophistiqué puissent être exécutés.

VII.4. La relation entre les différentes catégories

La distinction entre les différentes catégories du web mining n'est pas bien définie. Le web content mining peut utiliser le texte, les hyperliens, et parfois les profils des utilisateurs qui sont soit inférés ou introduits par les utilisateurs. La même chose pour le web structure mining qui doit utiliser les informations induites à partir des hyperliens. D'ailleurs les hyperliens peuvent être induits à chaque session d'utilisateur, c.à.d ils sont inférés à partir des fichiers logs. Dans la pratique, les trois catégories du Web mining peuvent être employées en isolation ou être combinées dans la même application, particulièrement dans le web content mining et le Web structure mining puisque les hyperliens sont utilisées par les deux.

VIII. CONCLUSION

Le web a été adopté comme milieu critique de communication et d'information à la majorité de la population. Les données web se développent à un taux significatif ce qui permet l'émergence de certaines techniques.

Le Web mining est un secteur de recherche très important, il est employé pour extraire de nouvelles connaissances à partir des différents types de données. Cependant, le web mining doit surmonter beaucoup plus de défis de recherches avant qu'il puisse réaliser ses objectifs. On peut simplement regarder le web mining comme prolongation de *knowledge discovery* qui est appliqué sur les données Web. On a vu que le Web mining se divise en trois sous catégories, le web usage mining, le web structure mining, et le web content mining.

Le web mining peut accomplir plusieurs tâches parmi elles, l'extraction de la structure et le contenu sémantique d'une page Web. Cela peut se faire à partir des documents HTML qui peuvent être représentés sous une forme hiérarchique suivant le modèle objet de document (DOM). Le chapitre suivant sera consacré à la présentation générale du (DOM).

Chapitre2

Document Object Model (DOM)

I. INTRODUCTION

Le plus grand challenge des scripteurs web coté client concerne le modèle objet. Un modèle objet est un mécanisme pour manipuler un document comme un objet. Chaque objet (parent) peut être composé d'autres objets (fils) et il est doté d'un ensemble d'attributs et un ensemble d'événements. Les modèles objets actuels, permettent d'accéder à presque tous les éléments, et à tous les attributs sur un élément. Il n'est pas exigé que les auteurs apprennent de nouvelles balises HTML, et il n'implique aucune nouvelle technologie. La récupération et la manipulation d'une valeur introduite dans un champ de texte d'un formulaire à partir d'un script, ou l'ajout des événements de souris aux hyperliens sont des formes limitées du modèle objet.

Netscape a mis en place le premier modèle objet dans Navigator 2. Avant les explorateurs de la version 4, le modèle objet original s'était étoffé de certains dispositifs intéressants. Le processus de normalisation s'est mis en place à partir du consortium World Wide Web (W3C) qui a établi des recommandations de modèle objet de document (DOM). Le nouveau (DOM) propose une grande partie du modèle objet original ainsi que de nouvelles manières pour s'adresser à des objets dans un document.

II. LES FAMILLES DE DOM [5]

Un aperçu de l'évolution des explorateurs scriptables à partir de NN2 et IE3 jusqu'à IE5.5 et NN6 indique six familles distinctes de modèle objet de document. L'étude de l'évolution de modèle objet est extrêmement intéressante pour les nouveaux venus au langage de script. Le tableau 2.1 énumère les familles de modèle objet (dans l'ordre chronologique) et les versions de l'explorateur qui les supportent.

Tableau 2.1 : Les familles de modèle objet.

Model	Browser Support
Model objet de base	NN2, NN3, E3/J1, IE3/J2, NN4, IE4, IE5, NN6, IE5.5
Model objet de base ainsi que les images	NN3, IE3.01 (Mac seulement), NN4, IE4, IE5, NN6, IE5.5
NN4 extensions	NN4
IE4 extensions	IE4, IE5, IE5.5 (certains dispositifs dans toutes les versions exigent Win32 OS)
IE5 extension	IE5, IE5.5 (certains dispositifs dans toutes les versions exigent Win32 OS)
W3C DOM	IE5 (partiel), IE5.5 (partiel), NN6 (complet)

Le premier Explorateur scriptable, NN2 a implémenté un très petit modèle objet qui est présenté dans la figure 2.1. Seuls les éléments interactifs (champ de texte, formulaire, liens, etc.) sont traités en tant qu'objets avec des propriétés, des méthodes et des manipulateurs d'évènement. IE3 a implémenté le modèle objet de base à partir de NN2, plus l'objet window à partir de NN3.

L'apparition de la deuxième famille a été avec le NN3, qui a sorti un modèle objet construit sur la version originale. Certains objets ont intégré de nouvelles propriétés (comme l'objet window), méthodes et des manipulateurs d'évènements. Les scripts ont pu également communiquer avec les applets Java en tant qu'objet. Mais le plus important a été l'objet image et le tableau d'objets image exposé dans l'objet document.

Des extensions ont été faites avec l'avènement de NN4, celles-ci concernent des ajouts au niveau des objets existants ce qui permet d'augmenter la puissance des scripteurs (comme la manipulation de l'objet screen). Mais le plus important est l'amélioration du modèle d'évènement par l'ajout de nombreux évènements (comme onMouseDown et onMouseUp), et la capture de l'évènement qui traverse toute l'arborescence pour atteindre l'objet ciblé. Il est plus commode que cet évènement soit capturé à un seul endroit plutôt que d'assigner le même manipulateur d'évènement à plusieurs objets. A chaque fois qu'un évènement se produit un objet "event" est créé et il contient toutes les informations concernant cet évènement. L'ajout le plus radical a été un nouvel objet qui reflète un élément entièrement nouveau dans HTML, l'élément LAYER.

La quatrième famille est apparue avec des extensions qui ont été faites au niveau de IE4. La première amélioration est que chaque élément HTML est devenu un objet scriptable, et l'invention du tableau document.all (également appelé collection). Cet objet contient des références de chaque élément dans le document. En remarquant la figure1 tous les éléments HTML sont emboîtés sous l'objet "document." La deuxième amélioration est le support de CSS (Cascading Style Sheets) de niveau 1, ce qui a permis au script de manipuler la propriété style des éléments HTML. Dans cette version IE4 a implémenté un modèle d'évènement différent du modèle de NN4. Un évènement d'IE commence de l'élément ciblé et il remonte vers le haut à travers la hiérarchie d'éléments, pour atteindre l'objet window. Cet évènement peut être modifié au long de la hiérarchie mais ne peut pas être supprimé. Comme dans le NN4. IE4 a également un objet event.

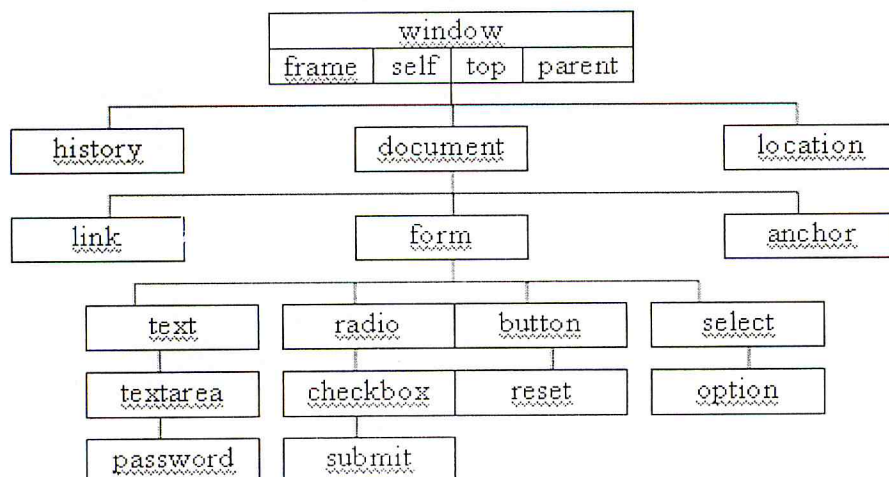


Figure2.1 : Le modèle objet de base implémenté dans les explorateurs.

La cinquième famille est apparue avec la sortie de IE5, la gamme des objets par rapport à IE4 est restée plus au moins la même, mais le nombre de propriétés, de méthodes, et de manipulateurs d'évènement a augmenté considérablement.

La sixième famille est le W3C DOM. Le W3C DOM est une évolution robuste du modèle objet parce qu'il fournit un modèle structuré et une interface logique pour permettre aux auteurs d'accéder et mettre à jour les éléments et les attributs.

On remarque qu'il y a des améliorations qui ont été faite d'une famille à une autre. Mais ce n'est pas suffisant car il y a toujours des manques aux niveaux de ces modèles objet. Ces manques concerne les objets, les attributs, et même les types d'évènement supportés par un objet (Par exemple: l'objet LAYER qui n'est pas supporté par les modèles objet de IE et même pour W3C DOM).

III. Le W3C DOM [5,6]

Les modèles objets contradictoires entre les différents explorateurs ont rendu la vie difficile aux développeurs. Les scripteurs ont demandé une norme qui sert comme dénominateur commun pour tout le HTML et une norme CSS pour le contenu et les styles. Le W3C a réclamé la création d'une norme de modèle objet de document, le W3C DOM.

Le groupe de travail du W3C DOM proposait de créer un modèle objet de document qui peut être appliqué aux documents HTML et XML. Un document XML peut avoir des balises contenant n'importe quel nom (comme le définit le Document Type Définition "DTD"), il n'a aucune structure intrinsèque ou vocabulaire fixe d'éléments comme pour un document HTML. En conséquence les spécifications du DOM ont dû s'adapter à la structure connue de HTML, comme à la structure inconnue de XML.

Pour faire ce travail efficacement, le groupe de travail a divisé les spécifications du DOM en deux sections. La première est appelée le noyau DOM, qui définit les caractéristiques pour la structure du document de base, que les documents HTML et XML partagent. Ceci inclut les notions d'un document contenant des éléments qui ont des noms et des attributs d'étiquette ; un élément est capable de contenir aucun élément ou d'autres éléments. La seconde s'adresse aux éléments et aux caractéristiques qui s'appliquent seulement à HTML. La partie HTML "hérite" de tous les dispositifs du noyau DOM.

III.1. Définition [5, 6]

Le DOM est une abréviation du modèle objet de document (Document Object Model), il indique la façon dont les objets dans une page Web (texte, images, titres, liens, etc...) sont représentés. Le *DOM* définit quels sont les attributs et les évènements associés à chaque objet, comment les objets, les attributs et les évènements peuvent être manipulés. Le HTML dynamique (*DHTML*) se fonde sur le *DOM* pour changer dynamiquement l'aspect des pages Web après qu'elles sont téléchargées au navigateur d'un utilisateur.

Avec le DOM, les programmeurs peuvent établir des documents, diriger leur structure, et ajouter, modifier, ou supprimer des éléments dans un document HTML ou XML, à quelques exceptions - en particulier, les interfaces de DOM pour les sous-ensembles internes et externes de XML.

En tant que spécification W3C, l'objectif important du W3C DOM est de fournir une interface de programmation standard pouvant être utilisée dans une grande diversité d'environnements et d'applications. Le DOM est conçu pour une utilisation dans n'importe quel langage de programmation. De manière à pouvoir offrir une spécification des interfaces du DOM précise et indépendante des langages de programmation, ils ont choisi de définir leur spécification dans le langage de définition d'interface (IDL), de l'Object Management Group (OMG). En supplément de l'OMG IDL, le W3C a proposé des language binding pour Java et ECMAScript (un langage de script standard de l'industrie basé sur JavaScript et JScript).

Employer le W3C DOM a beaucoup d'avantages pour manipuler l'arborescence du document HTML ou XML. Avec le W3C DOM, on peut [4]:

- Déplacer une partie de l'arbre d'un document à un autre sans détruire et recréer le contenu.
- Créer les éléments et les attacher à n'importe quel point dans l'arbre de document.
- Organiser et manipuler les nouvelles ou les branches d'arbre existantes dans un fragment de document avant d'insérer les objets de nouveau dans l'arbre.

III.2. Le modèle objet de document (DOM) [5]

Le DOM est un API de programmation pour les documents XML ou HTML. Il est basé sur une structure d'objet qui ressemble étroitement à la structure des documents

En prenant cet exemple de code pris d'un document HTML, qui représente un tableau :

```
<TABLE>
<TBODY>
<TR>
<TD>Shady Grove</TD>
<TD>Aeolian</TD>
</TR>
<TR>
<TD>Over the River, Charlie</TD>
<TD>Dorian</TD>
</TR>
</TBODY>
</TABLE>
```


La représentation graphique du DOM associée au code précédent est présentée dans la figure 2.2.

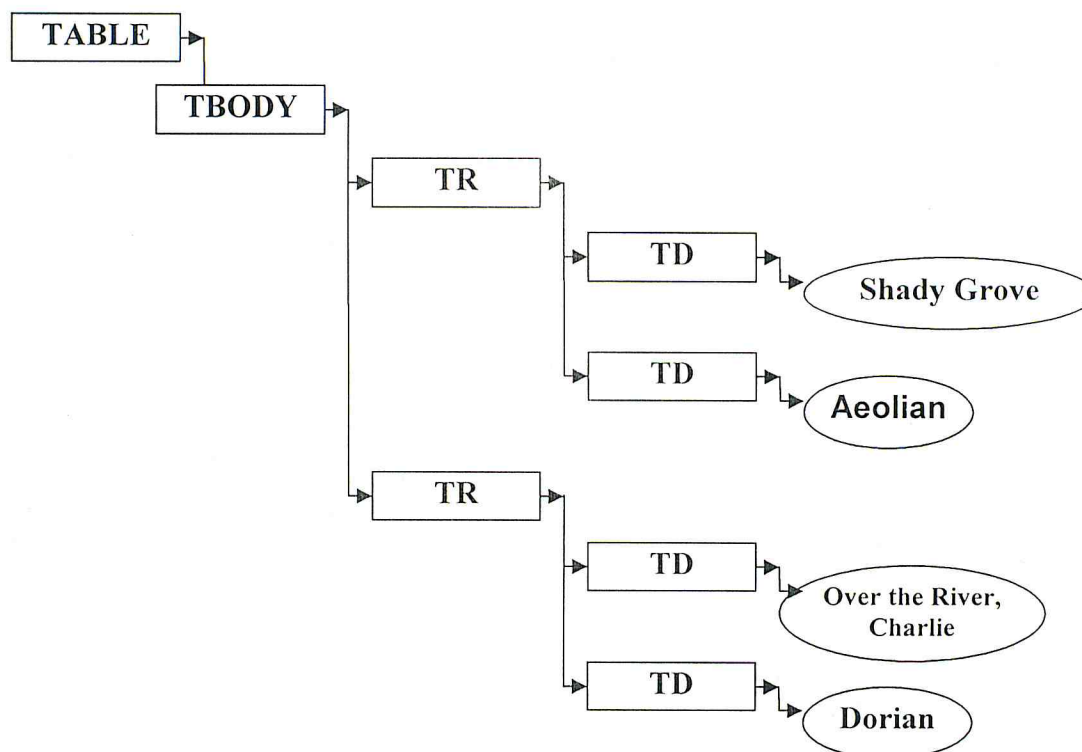


Figure 2.2: Représentation graphique du DOM du tableau d'exemple.

Avec le DOM, les documents ont une structure logique arborescente, qui peut contenir plus d'un arbre. Chaque document contient des nœuds (feuilles ou nœuds intermédiaire) qui sont liés directement ou indirectement au nœud racine du document. Cependant, le DOM n'indique pas que les documents doivent être implémentés comme un arbre ou une plantation, ni l'implémentation des rapports entre les objets. Le DOM structure models est un isomorphisme structurel : si deux réalisations quelconques du modèle objet de document sont employées pour créer une représentation du même document, elles créeront le même modèle de structure, selon XML Information Set.

Les documents sont modélisés en utilisant des objets. Le modèle englobe non seulement la structure du document mais également son comportement, et les objets qui le composent. En d'autres termes, les nœuds, dans la figure 2.2, ne représentent pas une structure de données, mais ils représentent des objets qui ont des fonctions et une identité. Comme modèle objet, le DOM identifie.

- Des interfaces et des objets pour représenter et manipuler un document.
- la sémantique de ces interfaces et de ces objets - y compris leur comportement et leurs attributs.
- les relations et les collaborations entre ces interfaces et ces objets.

La structure des documents HTML a été traditionnellement représentée par un modèle de données abstrait, pas avec un modèle objet. Le modèle de données abstrait est concentré sur les données. Dans les langages de programmation orienté objet, les données elles-mêmes sont encapsulés dans les objets. Des fonctions se sont associées à ces objets pour déterminer comment ils peuvent être manipulés.

III.3. Ce qui n'est pas le modèle d'objet de document [5]

Cette section est conçue pour donner une compréhension plus précise du DOM en le distinguant d'autres systèmes qui peuvent sembler être comme lui.

- Le DOM n'est pas une spécification binaire. Les programmes de DOM écrits dans la même langage binding auront leur code source compatible d'une plateforme à l'autre, mais le DOM ne définit aucune forme d'interopérabilité binaire.
- Le modèle d'objet de document n'est pas une manière de persister les objets XML ou HTML. Au lieu d'indiquer comment les objets peuvent être représentés dans XML ou HTML, le DOM indique comment des documents XML et HTML sont représentés comme des objets, de sorte qu'ils puissent être employés dans des programmes orientés objets.
- Le modèle d'objet de document n'est pas un ensemble de structures de données; c'est un modèle d'objet qui spécifie des interfaces. Bien que les diagrammes montrent des rapports de parent/enfant, ce sont que des rapports logiques définis par les interfaces de programmation, mais pas une représentation de quelconques structures de données internes particulières.
- Le modèle d'objet de document ne définit pas quelle est l'information la plus pertinente du document ou comment l'information est structurée. Pour XML, ceci est indiqué par le jeu d'informations XML du W3C (Infoset). Le DOM est simplement un API à cet ensemble d'informations.
- Le Modèle Objet de Document, en dépit de son nom, n'est pas en compétition avec le Modèle Objet de Composant (COM). COM, tout comme CORBA, est une manière indépendante d'un langage de spécifier des interfaces et des objets ; le DOM est un jeu d'interfaces et d'objets conçu pour la gestion de documents HTML et XML. On peut implémenter le DOM en utilisant des systèmes indépendants du langage comme COM ou CORBA ; on peut également l'implémenter en utilisant des langages binding spécifiques, comme les bindings avec Java et ECMAScript spécifiées dans ce document.

III.4. L'origine du modèle objet de document (DOM) [5,6]

Le DOM a été mis en place pour permettre à des scripts de Javascript et à des programmes de Java d'être portatifs par les browsers."Le HTML dynamique

(DHTML)" est l'ancêtre immédiat du DOM. Cependant, quand le groupe de travail de DOM a été constitué à W3C, il a été également joint par des fournisseurs dans d'autres domaines, y compris des rédacteurs de HTML, de XML, des vendeurs issus d'autres domaines et une base documentaire. Plusieurs de ces fournisseurs avaient travaillé avec le SGML avant que XML ait été développé. En conséquence, le DOM a été influencé par des plantations de SGML et la norme de HyTime standard. Certains de ces vendeurs avait déjà développé leur propre modèle objet pour les documents, de manière à proposer un API aux éditeurs SGML/XML, ou encore une base documentaire, c'est pourquoi ces modèles objet ont également influencé le DOM.

III.5. Les niveaux de DOM:

Le DOM représente des spécifications continuellement en évolution. Les sorties des spécifications coïncident rarement avec les sorties des explorateurs. Par conséquent, il est très courant pour n'importe quelle sortie donnée de l'explorateur d'inclure une partie de la version du W3C la plus récente.

Le *DOM* est séparé en trois parties :Noyau, HTML, et XML. Le noyau *DOM* fournit un ensemble de bas niveau d'objets qui peuvent représenter n'importe quel document structuré. Cette interface est capable de représenter n'importe quel document HTML ou XML. L'interface de noyau est une conception compacte et minimale pour manipuler le contenu du document. Dépendant d'utilisation du *DOM*, l'interface du noyau *DOM* peut ne pas être commode ou appropriée pour tous les utilisateurs. Les caractéristiques de HTML et de XML fournissent des interfaces additionnelles de plus haut niveau qui sont employées avec les spécifications de noyau pour fournir une vue plus commode du document. Ces spécifications se composent d'objets et de méthodes qui fournissent des accès plus facile et plus direct dans les types spécifiques de documents.

III.5.1. DOM niveau 0

Le terme " DOM niveau 0" se rapporte à un mélange (pas formellement indiqué) des fonctionnalités de document HTML offertes par la version 3.0 de Netscape Navigator et la version 3.0 de Microsoft Internet Explorer. Dans certains cas, des attributs ou des méthodes ont été inclus pour des raisons de compatibilité en arrière avec le " DOM niveau 0".

III.5.2. DOM niveau 1 (DOM1)

Les spécifications du niveau 1 de DOM sont séparées en deux parties :Noyau et HTML. La partie du noyau DOM1 fournit un ensemble de bas niveau d'interfaces fondamentales qui peuvent représenter n'importe quel document structuré, aussi bien que définir des interfaces prolongées pour représenter un document XML. La partie du niveau 1 de HTML fournit des interfaces additionnelles qui sont employées avec les interfaces fondamentales définies dans la partie du noyau pour fournir une vue plus commode d'un document HTML. Les interfaces présentées dans *DOM1* incluent, entre autres, les interfaces: Document, Node, Attr, Élément, et Text. Toutes

noyau *DOM2*. Il présente la propriété de *contentDocument*, une manière utile d'accéder au contenu du document dans une frame.

III.5.4. DOM niveau 3 (DOM3)

Ce niveau s'adresse au loading et saving, comme les modèles de contenus (tels que les *DTD* et les schémas) avec le soutènement de validation de document. En plus, il s'adresse également aux views, au formatage de document, des événements et des groupes d'événement.

Le *DOM3* est actuellement composé de 5 spécifications différentes: Le noyau *DOM3*, *DOM 3 Load and Save*, *DOM3 Validation*, *DOM 3 Events* et *DOM3 XPath*.

Le noyau *DOM3* prolonge les fonctionnalités du noyau de *DOM1* et de *DOM2*. Les nouvelles méthodes et propriétés incluent *adoptNode()*, *strictErrorChecking*, et *textContent*, pour citer seulement quelques uns.

La *DOM 3 Load and Save* permet à des programmes et à des scripts de charger dynamiquement le contenu d'un document *XML* dans un document de *DOM* et d'arranger un document de *DOM* dans un document *XML*.

Le *DOM3 Validation* permet à des programmes et à des scripts de mettre à jour dynamiquement le contenu et la structure des documents tout en s'assurant que le document reste ou devient valide.

Le *DOM 3 Events* est la prolongation de *DOM 2 Events*. Cette Spécification se concentre principalement sur des événements de clavier et comment les manipuler.

Le *DOM3 XPath* fournit des fonctionnalités simples pour accéder à un arbre de *DOM* en utilisant *XPath* 1. C'est également un avantage clair pour les applications d'utilisateur qui emploient le *DOM* pour pouvoir employer des expressions de *XPath* pour localiser automatiquement les noeuds. Ces spécifications ont été créées pour arranger entre la représentation *DOM* de *W3C* et le modèle *XPath* pour permettre à des fonctions de *XPath* d'être fournies et des résultats retourné dans le cadre de *DOM APIs* d'une manière standard. Réciproquement, les données, qui ne sont pas présent en compte par les spécifications de *XPath* mais elles sont présentes grâce aux résultats venant de la hiérarchie de *DOM*.

IV. LA MANIPULATION DU DOM AVEC JAVASCRIPT [6]

Le *DOM* peut être manipulé par plusieurs langages de programmation. *W3C* a prévu des langages-bindings pour des langages de programmation comme java, javascript, etc.

Dans Javascript on ne trouve pas de termes tels que les classes, l'héritage et les instances. Mais on ne peut pas ignorer le concept de la hiérarchie, parce qu'une partie du code se fonde sur la capacité d'écrire des références aux objets qui dépendent de leurs positions à l'intérieur de la hiérarchie. Le rôle de cette hiérarchie

est de permettre aux scripts de référencier les objets à l'intérieur du document affiché dans un explorateur.

La plupart des objets qu'un explorateur crée sont établis quand un document HTML est chargé dans l'explorateur. Le même type de code HTML utilisé pour créer des liens, des éléments, crée ces objets en mémoire et ils sont présents si et seulement si ils sont appelés par des scripts. Les objets sont créés en fonction de l'ordre de chargements. Dans un environnement multiframe, un script dans un cadre ne peut pas communiquer avec les objets d'un autre cadre tant que les deux cadres ne sont pas complètement chargés.

Chaque objet possède des propriétés qui peuvent être visible tel que l'état d'une case à cocher, et peuvent contenir des informations qui ne sont pas évidentes, comme l'action et la méthode d'un formulaire soumis.

La méthode d'un objet est une instruction qu'un script peut donner, certaines méthodes renvoient des valeurs. Les méthodes les plus utilisées sont les manipulateurs d'évènement, elles sont déclenchées à chaque fois qu'une action est appliquée sur cet objet. Les manipulateurs d'évènements peuvent être invoqués comme des méthodes ou comme des attributs.

Le DOM propose une hiérarchie d'objets, afin de faciliter leur manipulation. La figure 2.3 monte une hiérarchie d'objet qui est implémentée dans les explorateurs scriptables. On remarque que l'objet window est le plus élevé dans la hiérarchie.

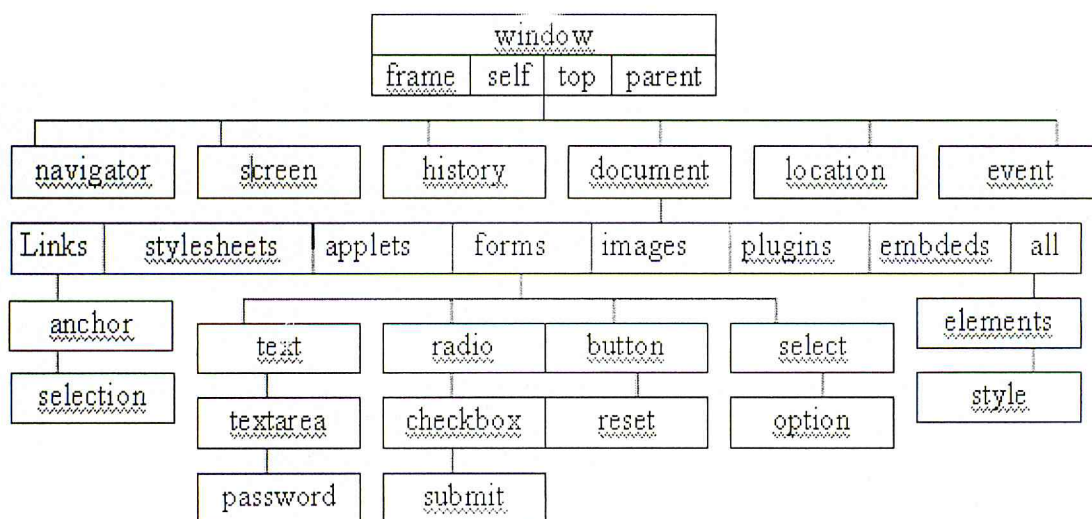


Figure 2.3 : La hiérarchie d'objet de document d'IE4.

▪ L'objet window

L'objet window est l'objet global et ses propriétés sont des variables globales pour JavaScript. Les variables globales window, self, top, et parent se réfèrent à cet objet.

- **L'objet frames**

L'objet frames est une collection d'objets window qui permettent à accéder à tous les objets window d'un document HTML multicaadre.

- **L'objet document**

L'objet document représente le document HTML, il est contenu dans l'objet window et il contient tous les éléments HTML.

- **L'objet event**

L'objet event est créé à chaque fois qu'un utilisateur interagit avec la page web. Il contient toutes les informations concernant l'évènement produit (la nature, le positionnement, etc.)

- **L'objet history**

L'objet history représente une référence de l'historique de navigation de l'utilisateur dans le document.

- **L'objet location**

L'objet location représente l'URL du nouveau document chargé.

- **L'objet screen**

L'objet screen contient des informations sur l'écran où le document est affiché.

- **L'objet navigator**

L'objet navigator contient des informations sur le navigateur où le document est affiché (par exemple: la version du navigateur).

Le tableau 2.2 suivant présente quelque collection d'objets qui sont contenu dans l'objet document:

Tableau 2.2: Les collections et leurs descriptions.

collection	description
all	Référence à tous les objets du document
anchors	Référence à tous les objets <A> du document
applets	Référence à tous les objets <APPLET> du document
children	Référence à tous les objets fils de l'objet document
forms	Référence à tous les objets <FORM> du document
images	Référence à tous les objets du document
links	Référence à tous les objets <A> du document
href	Référence à tous les objets <AREA> du document
plugins	Sert comme un alias pour la collection embeds
Scripts	Référence à tous les objets <SCRIPT> du document
stylesheets	Référence à tous les objets stylesheets du document

IV.1. Les types de noeud les plus utilisés [6,7]

Les types de nœud les plus utilisés ou les plus manipulés sont présentés dans le tableau 2.3. Chaque type de nœud est accompagné par la valeur de la propriété `nodeType`.

Tableau2.3: Les types de noeud les plus utilisés

Noeud	nodeType
Element	1
Text	3
Document	9
Comment	8
Attr	2

- **Element:** Les éléments sont les blocs de base des documents XML ou HTML. Les éléments ont des enfants qui sont des nœuds de Type Element, noeuds de type Text, ou une combinaison des deux. Les noeuds de type Element sont également le seul type de noeud qui peut avoir des attributs.
- **Text:** Un noeud de type Text est exactement identique à un texte. Il peut se composer de plusieurs informations ou juste un espace blanc.
- **Document:** Le noeud de type Document représente le parent global pour tous les autres noeuds dans un document.
- **Comment:** Le nœud de type Comment est simplement un commentaire Les commentaires incluent des informations sur les données, et sont habituellement ignorés par l'application.
- **Attr:** Les noeuds de type Attr contiennent des informations sur un noeud de type Element, mais ne sont pas considérés réellement comme des enfants de Element.

IV.2. Les propriétés générales

Chaque objet du DOM peut contenir des attributs, qui sont soit modifiable (comme la propriété `width`) ou non (comme la propriété `tagName`). Le tableau 2.4 suivant énumère quelques attributs communs à tous les éléments HTML.

Tableau 2.4: Les propriétés générales.

propriété	signification
<code>all[]</code>	Donne accès à tous les nœuds de type Element de l'élément courant.
<code>canHaveChildren</code>	Retourne vraie si un élément peut avoir des fils.
<code>canHaveHTML</code>	Booléen qui indique si un élément peut contenir un code HTML

	entre les balises de l'élément courant.
childNodes[]	Retourne la liste des éléments fils
children	Retourne la liste des éléments fils de type Element.
className	Retourne nom de la classe de l'élément.
currentStyle	Retourne le style courant de l'élément.
Disabled	Active ou désactive un élément.
firstChild	Retourne le premier fils
id	Retourne la valeur d'identifiant.
innerHTML	Retourne le code HTML entre la balise de début et de fin de l'élément courant.
innerText	Retourne tout le texte emboîté entre la balise de début et de fin de l'élément courant.
isDisabled	Retourne vraie si l'élément est désactivé.
isMultiLine	Retourne vraie si un nœud de type Text présenté sur plusieurs lignes
language	Retourne le nom du langage de script qui est appliqué sur l'élément courant.
lastChild	Retourne le dernier fils
length	Retourne la longueur de élément comme les chaînes de caractères ou les tableaux.
nextSibling	Retourne le nœud qui se trouve après le nœud courant. S'il n'existe pas elle retourne null.
nodeName	Retourne le nom de balise.
nodeType	Retourne le type de nœud comme Element, Text, etc.
nodeValue	Retourne la valeur du nœud.
offsetHeight	Retourne la longueur de l'élément par rapport à l'élément offsetParent.
offsetLeft	Retourne le nombre de pixels entre la bordure gauche de l'élément par rapport à l'élément offsetParent.
offsetParent	Retourne une référence à l'objet parent de l'élément courant.
offsetTop	Retourne le nombre de pixels entre la bordure haute de l'élément par rapport à l'élément offsetParent.
offsetWidth	Retourne la largeur de l'élément par rapport à l'élément offsetParent.
outerHTML	Retourne le code HTML entre la balise de début et de fin de l'élément courant.
outerText	Retourne tout le texte emboîté entre la balise de début et de fin de l'élément courant.
parentNode	Retourne une référence à l'objet parent de l'élément courant.
previousSibling	Retourne le nœud qui se trouve avant le nœud courant. S'il n'existe pas elle retourne null.
runtimeStyle	Retourne des informations concernant le style
sourceIndex	Retourne l'index de l'élément dans le document
style	Retourne des informations concernant le style
tagName	Retourne le nom de la balise du nœud.

IV.3. Les méthodes générales [7]

La fonctionnalité du *DOM* se divise en quatre groupes principaux :

- Les méthodes pour parcourir l'arbre de document : Les méthodes de parcours permettent à un script de localiser des noeuds particuliers dans un arbre de document. La racine de l'arbre de document est consultée par la propriété `document`. Parmi les plus importantes on peut citer :
 - **getElementById(id:String)**
Retourne une référence d'un objet à partir de son identifiant "id"
 - **getElementsByTagName(tag:String)**
Retourne un tableau de références des objets à partir des noms de balises "tag".
- Les méthodes pour changer la structure de l'arbre de document: La structure d'un document peut être changée en insérant de nouveaux éléments dans le document, et en remplaçant les éléments existants. on peut citer parmi elles
 - **createElement**
Crée un nœud de type `Element`.
 - **createTextNode**
Crée un nœud de type `Text` (par ex: le contenu d'un élément `<H2>`).
 - **appendChild**
Ajouter un nouveau fils à la fin de la liste des enfants.
 - **insertBefore**
Insérer un nœud avant un autre nœud spécifique.
 - **replaceChild**
Remplacer un nœud par un autre nœud spécifique.
- les méthodes pour rechercher et changer les valeurs qui sont associées aux noeuds de l'arbre: Le *HTML DOM* fournit des méthodes pour changer les attributs et le contenu d'un élément. Des attributs de HTML sont exposés comme propriétés d'objet d'élément. Les noms de propriété sont indépendants des attributs dans le document source et suivent quelques règles d'appellation.
 - **getAttribute(name:String)**
Retourne la valeur d'attribut sous une forme de chaîne de caractères.
 - **setAttribute(name:String, value:String)**
Affecter à des attributs bien spécifiques des valeurs.
 - **removeAttribute(name:String)**
Supprime un attribut bien spécifique.

- Les méthodes pour gérer les différents types d'évènements qui sont associés à chaque nœud de type Element.

En outre, le HTML *DOM* fournit des méthodes spécifiques aux types particuliers de nœud, par exemple la méthode `close ()` qui est propre à l'objet `window`.

V. CONCLUSIONS

Dans ce chapitre on a présenté les différentes familles de modèle objet, la famille W3C DOM qui est venue pour résoudre le problème de compatibilité entre les modèles objets des différents explorateurs. Le DOM à plusieurs niveaux tel que chaque niveau supérieur étend les niveaux les plus bas et apporte des améliorations pour permettre aux programmeurs de mieux manipuler les documents. Enfin la partie manipulation de DOM avec Javascript qui donne une idée sur la façon dans laquelle on peut manipuler le DOM et quelles sont les principales type d'objet, d'attributs et des méthodes qui nous permettent de faire.

Le DOM peut représenter le document HTML sous une forme d'arbre ou les nœuds sont les balises HTML. Le DOM n'indique pas d'informations sur la relation entre les nœuds, donc il faut une autre approche. Dans le chapitre suivant nous allons introduire l'algorithme Vision based Page Segmentation qui utilise le DOM et les propriétés visuelles extraite à partir des nœuds du DOM pour déterminer la structure sémantique des documents HTML.

Chapitre3

Vision-based Page Segmentation

I. INTRODUCTION

Beaucoup d'applications Web peuvent utiliser les structures sémantiques du contenu des pages Web. Par exemple, dans l'accès aux informations sur le Web et pour surmonter les limitations de la recherche à base de mot clé, quelques chercheurs ont essayé d'employer des techniques de base de données et de construire des wrappers¹ pour structurer les données du Web [8,9,10,11]. Dans la construction de wrappers, il est nécessaire de diviser les documents Web en différents gros morceaux d'information [9,12]. Les travaux précédents emploient des méthodes ad hoc pour traiter différents types de pages Web. Si il est possible d'obtenir une structure sémantique du contenu d'une page Web, les wrappers peuvent être plus facilement construits et l'information peut être plus facilement extraite. D'ailleurs, l'analyse des liens [13,14] a tiré beaucoup d'attention ces dernières années. Traditionnellement différents liens dans une page sont traités de la même manière. Le principe de base des algorithmes d'analyse des liens est que s'il y a un lien entre deux pages, alors il y a un certain rapport entre ces deux pages en entier. Mais dans la plupart des cas, un lien de la page A vers la page B indique seulement qu'il pourrait y avoir un certain rapport entre certaine partie de la page A et certaine partie de la page B. mais, l'existence du grand nombre de liens bruyants posera le problème du sens du sujet dans l'algorithme de HITS [14,15]. Des travaux récents dans la distillation de sujets [16,17] renforcent cette observation. Cependant, ces travaux sont basés sur *DOM* (modèle d'objet de document), comme on a vu dans le chapitre précédent c'est la représentation d'une page Web sous forme d'une arborescence, cette dernière n'a aucune puissance suffisante de segmenter sémantiquement une page Web. En outre, la lecture rapide efficace de grands pages Web sur de petits dispositifs tenus dans la main rend nécessaire également la segmentation sémantique des pages Web [18].

En résumé on peut dire que plusieurs applications peuvent tirer des bénéfices de cette approche comme :

- Information retrieval ;
- Information extraction ;
- l'adaptation automatique de pages.

Beaucoup d'essai de travaux récents [12, 19, 16, 20] ont été faite pour extraire l'information structurale à partir du HTML DOM tree. Cependant, en raison de la flexibilité de la syntaxe de HTML, beaucoup de pages Web n'obéissent pas les caractéristiques de HTML de W3C (World Wide Word Consortium), qui pourraient causer des erreurs au niveau de la structure arborescente du DOM. D'ailleurs, le DOM tree est initialement introduite pour la présentation dans les navigateurs plutôt que la description de la structure sémantique d'une page Web. Par exemple, deux noeuds dans le DOM tree ont le même parent, ça n'implique pas forcément que ces deux noeuds sont reliés sémantiquement.

¹ Wrappers : Agent d'extraction d'information.

Dans le sens de la perception humaine, c'est toujours le cas qu'une personne perçoit une page Web en tant qu'un ensemble de différents objets sémantiques plutôt qu'un seul objet simple. Quelques efforts de recherches prouvent que les utilisateurs prévoient toujours que certaines parties fonctionnelles d'une page Web (par exemple liens de navigation, barre de publicité) apparaissent à certaines positions de cette page. En fait, quand une page Web est présentée à l'utilisateur, les propriétés spatiales et visuelles peuvent aider l'utilisateur à diviser la page Web en plusieurs parties sémantiques. Par conséquent, il est possible de segmenter automatiquement les pages Web en employant les propriétés spatiales et visuelles. Dans tout ce chapitre, nous employons *le bloc* pour dénoter la partie sémantique d'une page Web.

Nous présentons l'algorithme VIPS (Vision-based Page Segmentation) pour extraire la structure sémantique d'une page Web. Une telle structure sémantique est une structure hiérarchique dans laquelle chaque noeud correspondra à un bloc. Une valeur (*degré de cohérence*) sera assignée à chaque noeud pour indiquer la cohérence du contenu dans le bloc basé sur la perception visuelle. L'algorithme VIPS extrait d'abord tous les blocs appropriés à partir du HTML DOM tree, puis il essaye de trouver les séparateurs entre ces blocs extraits. Les séparateurs dénotent les lignes horizontales ou verticales dans une page Web qu'ils ne croisent pas visuellement avec les blocs. Enfin, basé sur ces séparateurs, la structure sémantique correspondante à la page Web est construite. L'algorithme VIPS utilise une approche de haut en bas, qui est très efficace.

II. ETAT DE L'ART

Les applications mentionnées dans l'introduction indiquent le besoin des techniques pour extraire la structure du contenu d'une page Web. Beaucoup de recherches sont basées sur les types de balise HTML. Les balises utiles incluent <P> (paragraphe), <TABLE> (tableau), (liste), <H1>~<H6> (titre), etc. Lin [22] considère seulement la balise <TABLE> et sa descendance comme bloc et emploie une approche basé sur l'entropie pour découvrir les blocs les plus pertinents. Kaasinen [18] et Buyukkokten [23] ont divisé la page Web par la prise en compte de certaines balises faciles telles que <P>, <TABLE> et . Wong [24] définit des types de balises pour la segmentation de page et donne des labels à chaque partie de la page Web dont le but est de faire une classification.

Quelques autres algorithmes se servent également des informations du contenu ou des liens. Embley [12] et Buttler [25] emploient quelques règles heuristiques pour découvrir les frontières des segments dans une page Web, qui aident l'extraction de données à partir du page Web. Récemment, la Bar-Yossef [26] propose le problème de détection de template et présente un algorithme facile seulement basé sur les informations extraite à partir des liens. En outre, Rahman [27] donne quelques commentaires sur la décomposition des pages Web mais il ne montre pas des détails sur la technique.

Chen [19] a proposé Function-based Object Model (FOM) pour la compréhension et l'adaptation du contenu d'une page Web. Chaque élément

indivisible dans le DOM tree (les feuilles) s'appelle un objet de base et peut être groupé dans un objet composé. Une fonction peut être défini à chaque objet pour aider à établir une structure hiérarchique de la page. Cependant, les règles et les fonctions pour faire le groupement sont difficiles à définir d'une manière exacte et rendent ainsi le processus de construction entier très inflexible.

Miloš Kovacevic [28] a proposé une méthode qui se nomme « Recognition of Common Areas in a Web Page Using Visual Information ». Cette méthode s'appuie sur un modèle, tel que chaque page peut être vue comme combinaison de quatre parties (menus, header (en-tête), footer (bas de page) et le centre de la page.). L'identification de ces parties se base sur des règles heuristiques. Mais l'inconvénient de cette méthode est que elle suit un modèle ou un design de page web bien précis.

La plupart des méthodes ci-dessus ne tiennent pas compte de la structure visuelle de la page Web. Dans la suite de ce chapitre, nous présentons l'algorithme VIPS (Vision-based Page Segmentation) qui contrairement à toutes ces méthodes fait la segmentation sémantique des page Web à base des propriétés visuelles.

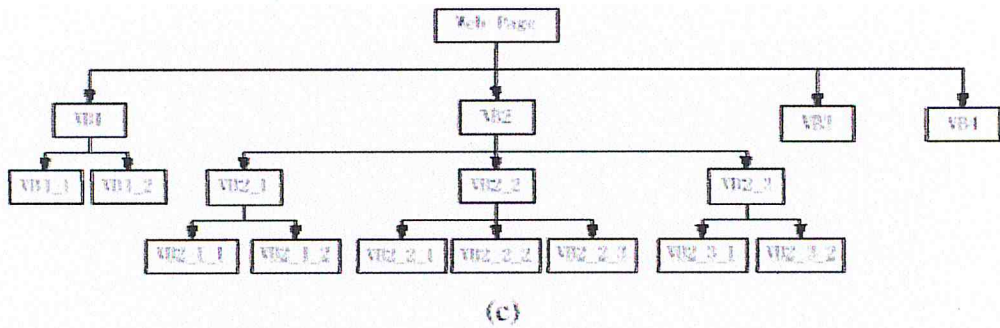
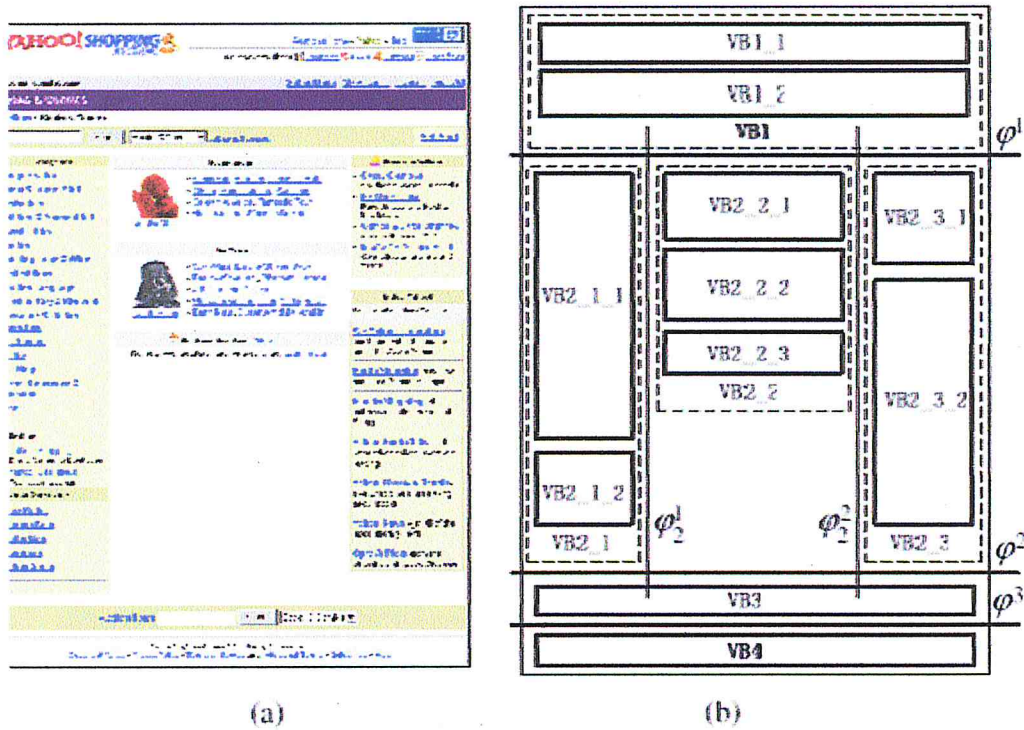
III. LE PRINCIPE DU VISION-BASED CONTENT STRUCTURE

Un objet de base peut être défini comme une feuille dans le DOM tree qui ne peut pas être décomposé. Dans la structure de contenu à base de la vision (vision-based content structure), un nœud est appelé bloc. Un bloc est un objet de base ou un ensemble d'objets de base. Un bloc ne correspond pas nécessairement à un nœud dans le DOM tree.

Similaire à la description de la représentation du document Web donné par Tang [29], Une page web peut être représenté comme un triplet $\Omega = (O, \Phi, \delta)$:

- $O = \{\Omega^1, \Omega^2, \dots, \Omega^N\}$, c'est un ensemble finie de blocs, ou chaque bloc O^i peut être un objet de base ou un ensemble d'objets de base tel que il peut être décomposé de la même manière .
- $\Phi = \{\Phi^1, \Phi^2, \dots, \Phi^N\}$ un ensemble finie de séparateurs (séparateurs horizontal et séparateurs vertical), chaque séparateur a un poids qui indique sa visibilité. Les séparateurs de la même Φ ont le même poids.
- δ qui représente la relations entre deux blocs de O , elle peut être écrite de la façon suivante: $\delta = O \times O \rightarrow \Phi \cup \{NULL\}$. Par exemple, on considère Ω^i, Ω^j deux objets de O , $\delta(\Omega^i, \Omega^j) \neq NULL$ cela indique que Ω^i et Ω^j sont exactement séparés par le séparateur $\delta(\Omega^i, \Omega^j)$ ou d'une autre façon ces deux bloc sont adjacents.

La figure 1 montre que dans le premier niveau la page Web est composée en quatre blocs visuels VB1, ..., VB4, donc il y a trois séparateurs Φ^1, Φ^2, Φ^3 comme le montre la figure 3.1 (d). Ensuite on peut de la même manière construire la structure de contenu à base de la vision pour chaque VB_i (sub page). Par exemple VB2 est divisé en trois blocs qui sont séparé par deux séparateurs.



$$O = (VB1, VB2, VB3, VB4)$$

$$VB2 = (VB2_1, VB2_2, VB2_3)$$

$$\Phi = \{\varphi^1, \varphi^2, \varphi^3\}$$

$$\Phi^2 = \{\varphi_1^1, \varphi_2^1\}$$

$$\delta \begin{pmatrix} (VB1, VB2) \\ (VB2, VB3) \\ (VB3, VB4) \\ else \end{pmatrix} = \begin{pmatrix} \varphi^1 \\ \varphi^2 \\ \varphi^3 \\ NULL \end{pmatrix}$$

$$\delta^2 \begin{pmatrix} (VB2_1, VB2_2) \\ (VB2_2, VB2_3) \\ else \end{pmatrix} = \begin{pmatrix} \varphi_1^1 \\ \varphi_2^1 \\ NULL \end{pmatrix}$$

(d)

(e)

Figure 3.1 : La page Web et la structure de contenu à base de la vision d'une page. (d) et (e) les spécifications la structure de contenu à base de la vision.

On attribue pour chaque bloc visuel, un degré de cohérence (DoC). DoC a les propriétés suivantes:

- Plus la valeur de DoC est grande, plus le contenu du bloc est plus cohérent.
- Dans l'arbre, le DoC de l'enfant n'est pas plus petit que celui de son parent.

Dans cet algorithme le DoC est un entier qui varie de 1 à 10.

On définit le degré de cohérence permet (PDoC) dont l'avantage est que dans le processus de segmentation, un segment ne peut pas être divisé si la valeur du DoC est supérieure à PDoC.

IV. L'ALGORITHME VIPS

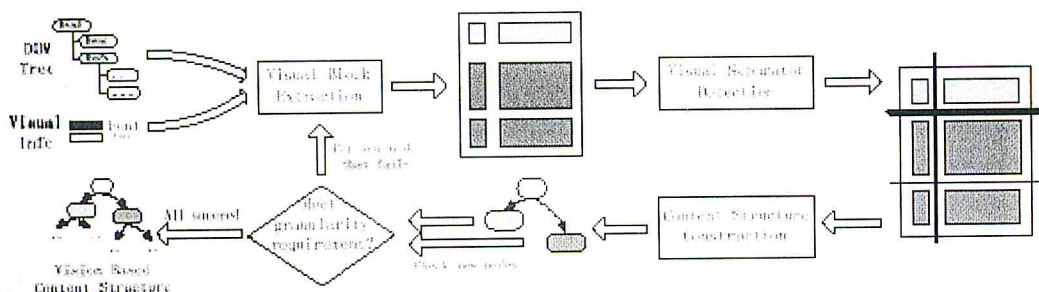


Figure 3.2 : L'algorithme VIPS.

Fondamentalement, la structure de contenu à base de la vision d'une page est obtenue en combinant la structure de DOM et les propriétés visuelles (c à d que les entrées du VIPS sont le DOM tree et les attributs visuelles qui sont expliquées plus loin dans ce chapitre). Le processus de segmentation est illustré dans la figure 3.2. Il y a trois étapes: l'extraction des blocs visuels, détection des séparateurs, et la construction de la structure du contenu. L'algorithme est de haut en bas. La page Web est premièrement segmentée en plusieurs grands blocs et la structure hiérarchique de ce niveau est enregistrée. Pour chaque grand bloc, le même processus de segmentation est effectué récursivement jusqu'à ce que nous obtenions suffisamment des petits blocs qui respectent la contrainte suivante:

$$\text{DoC} \geq \text{PDoC}$$

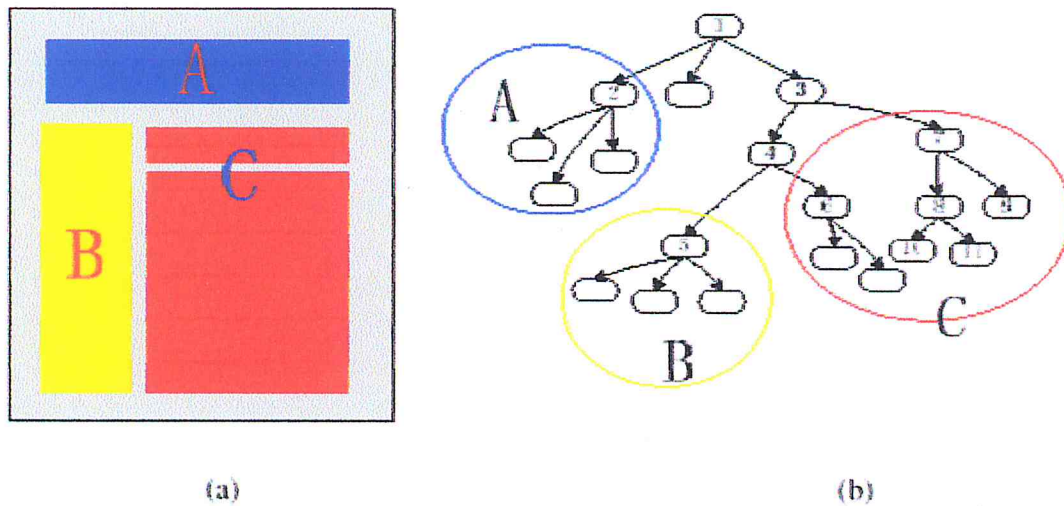


Figure 3.3 : (a)Page layout, (b) DOM tree de la page.

Pour chaque itération, le DOM tree avec les informations visuelles qui correspond au bloc courant (page pour la première itération) est obtenu à partir d'un navigateur web, comme il est montré dans la figure 3.3. Le processus d'extraction des blocs commence par l'extraction des blocs à partir du DOM tree en ce basent sur les propriétés visuelles. Chaque nœud de DOM (nœud 1, 2, 3, 4, 5, 6, 7 figure 3.3b) est vérifié pour juger s'il forme un bloc simple ou non. Si le nœud n'est pas simple (noeud 1, 3, 4 dans figure 3.3b), ces enfants seront traités de la même manière. Une valeur DoC est attribuée à chaque bloc extrait (nœud 2, 5, 6, 7 dans figure 3.3b) basé sur la propriété visuelle du bloc. Quand tous les blocs de la page ou du sous-page sont extraits, ils sont mis dans un pool.

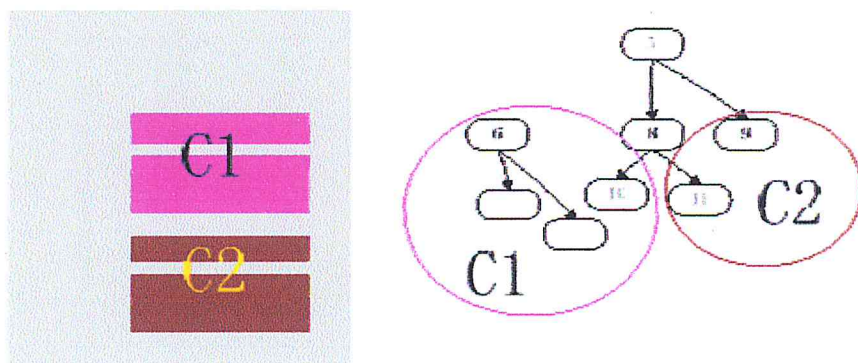


Figure 3.4: (a) Layout du bloc C (b) DOM tree du bloc C.

Des séparateurs entre ces blocs sont identifiés et le poids d'un séparateur est affecté suivant les propriétés de ses blocs voisins. La hiérarchie de layout est construite, basé sur ces séparateurs. Après avoir construit la hiérarchie de layout de cette étape, chaque nœud de feuille de la structure du contenu est vérifié pour voir s'il répond à la condition de granularité. Si non, ce nœud de feuille sera traité comme une sous-page et sera encore segmenté de la même manière. Par exemple, le bloc C dans la figure 3 ne répond pas au condition de granularité donc il sera traité

comme une sous-page et il sera segmenté en deux parts, C1 et C2, comme il est représenté dans la figure 3.4(a) et 3.4 (b).

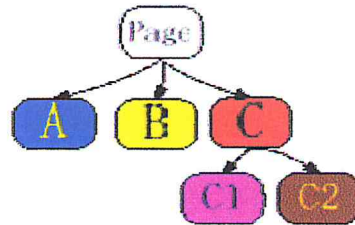


Figure 3.5: Vision-based content structure.

IV.1. Les propriétés visuelles

Les pages Web sont conçus pour les êtres humains! Alors les concepteurs ou les designers des pages Web essaient d'organiser leurs pages de telle façon que chaque information est affectée à un emplacement bien précis, avec un arrière plan bien spécifique. Dans le cas d'un texte on peut changer son apparition avec les propriétés de police (font).

Nous récupérons toutes les propriétés qui affectent l'apparition visuelle d'un nœud sur la page web. Elles sont énumérées dans le tableau suivant:

Tableau 3.1: Les propriétés visuelles.

Attributs	Signification
nodeName	Retourne le nom de la balise (par exemple < HTML>, <P>, etc.)
nodeValue	Retourne la chaîne de caractère dans le cas d'un nœud de type text
innerText	Retourne la chaîne de caractère dans le cas d'un nœud de type Element
length	Retourne la longueur d'une chaîne de caractère
left	Retourne l'espace entre la bordure gauche du nœud et la bordure gauche de la page
top	Retourne l'espace entre la bordure haute de nœud et la bordure haute de la page
width	Retourne la largeur du nœud
height	Retourne la longueur du nœud
fontSize	Retourne la taille de police
fontWeight	Retourne le poids de police
fontColor	Retourne la couleur du texte
fontStyle	Retourne le style de police
backgroundColor	Retourne la couleur d'arrière plan
backgroundImage	Retourne le chemin d'image qui se trouve en arrière plan
padding	Retourne l'espace entre le nœud et sa bordure
borderWidth	Retourne l'épaisseur de la bordure
borderColor	Retourne la couleur de la bordure

IV.2. L'extraction des Blocs Visuels [30]

Cette étape, sera consacré à trouver tous les blocs visuels appropriés contenus dans la sous-page courante. En général, chaque nœud dans le DOM tree peut représenter un bloc visuel. Cependant, certains nœuds " énormes " tels que <TABLE > et < P > sont employés seulement dans le but d'organisation, et ne peuvent pas représenter un bloc visuel simple. Alors le nœud devrait être encore divisé et remplacé par ses enfants. Puisque la grammaire de HTML est flexible, beaucoup de pages web n'obéissent pas entièrement aux spécifications de HTML de W3C, ainsi l'arbre de DOM ne peut pas toujours refléter le rapport réel entre les nœuds.

Pour chaque bloc visuel extrait, sa valeur de DoC est affectée selon son intra différence visuelle. Ce processus est réitéré jusqu' à la détection de tous les blocs visuels du sous-page courante.

<pre> Algorithm DivideDomtree(pRoot, nLevel) { IF (Dividable(pRoot, nLevel) == TRUE){ FOR EACH child OF pRoot { DivideDomtree(child, nLevel); } } ELSE { Put the sub-tree (pRoot) into the pool as a block; } } </pre>	<pre> Algorithm Dividable(pRoot, nLevel) { IF (pRoot is the Top Block){ RETURN TRUE; } ELSE { Special routines for TABLE, TR, TBODY, TD, P, UL, FORM; Heuristic rules for general tags; } } </pre>
(a)	(b)

Figure 3.6:L'algorithm de visual block extraction.

Un nœud quelconque de DOM tree peut être divisé suivant les considérations suivantes:

- Les propriétés du nœud : Par exemple, la balise HTML qui correspond à ce nœud, la couleur du fond, la taille et la forme du bloc correspond à ce DOM nœud.
- Les propriétés de ces enfants : Par exemple, les balises HTML des nœuds enfants, couleur de fond et la taille des enfants. Le nombre de différents genres d'enfants est également une considération.

Nous basons sur les spécifications de WWW HTML 4.01, les nœuds de DOM tree seront classifiés en deux catégories, *Inline node* et *Line-break Node* :

- Inline node

Le noeud de DOM tree avec les balises HTML, qui affectent l'aspect du texte et peuvent être appliquées à une chaîne de caractères (c à d balises de formatage du texte) sans présenter une ligne de coupure. Par exemple: , <BIG>, , , <I>, , <U>, etc.

- Line-break Node

Le noeud avec des balises autres que les balises qui sont orientées pour le texte (c à d : ces balises ne sont pas des balises de formatage du texte).

Nous basons sur l'apparition du nœud sur le browser et les propriétés d'enfants du nœud, nous donnons quelques définitions :

- *Valid node*

Un nœud qui peut être vu par le browser, dont la largeur et la hauteur sont différentes de zéro et aussi il contient un contenu significatif (ce problème aura lieu surtout dans les pages qui sont générées dynamiquement).

- *Text node*

Le noeud du DOM tree qui correspond au texte libre, qui n'est pas entouré avec des balises HTML.

- *Virtual text node(définition récursive)*

- *Inline node* avec seulement *Text node* comme enfants est un *Virtual text node*.
- *Inline node* avec seulement *Text node* ou *Virtual text node* comme enfants est un *Virtual text node*.

Quelques considérations importantes qui sont employées pour produire des règles heuristiques qui sont employées dans la phase de segmentation dans l'algorithme VIPS sont :

- Tag cue

1. Des balises telles que <HR> sont souvent employées pour séparer les différents sujets. Par conséquent il est préférable de diviser un nœud du DOM tree s'il contient ces balises.

2. Si un *Inline node* a un enfant qui est un *line-break node*, il est préférable de diviser un nœud du DOM tree.

- Color cue

Il est préférable de diviser un noeud de DOM tree si sa couleur du fond est différente de la couleur d'un de ses enfants. En même temps, le noeud enfant avec des couleurs de fond différentes ne sera pas divisé à ce rond.

- Text cue

Si la plupart des enfants d'un noeud de DOM sont des *text nodes* ou des *virtual text nodes*, il est préférable de ne pas le diviser.

- Size cue

Un seuil relatif de taille (la taille de nœud comparée à la taille de la page ou de la sub-page) est défini pour les différentes balises (le seuil change suivant les nœuds de DOM ayant différentes balises HTML). Si la taille relative du nœud est inférieure au seuil, il est préférable de ne pas diviser ce nœud.

Nous basons sur ces propriétés, des règles heuristiques seront produites pour juger si un nœud sera divisé ou non. Si un nœud n'est divisé, alors un bloc est extrait et une valeur de DoC sera affectée à ce dernier. Les règles heuristiques sont énumérées dans le tableau 3.2 par leur priorité.

Tableau 3.2: Les règles heuristiques pour la phase d'extraction de bloc

Règle1	si un nœud n'est pas un text node et il n'a aucun enfant valide, alors ce nœud ne peut pas être divisé alors il sera supprimé.
Règle2	si un nœud a seulement un enfant valide et l'enfant n'est pas un text node, ce nœud sera remplacé par son enfant.
Règle3	si un nœud est la racine de sous arbre DOM (correspondant au bloc), et il y a un seul sous arbre DOM correspondant à ce bloc, ce nœud sera divisé.
Règle4	Si tous les enfants d'un nœud sont des text nodes ou des virtual text nodes, ne pas diviser le nœud. <ul style="list-style-type: none"> • Si tout les propriétés de la police (taille, poids, famille, etc.) de tous les nœuds enfants sont les mêmes, le DoC du bloc extrait est égale à 10. • Autrement, le DoC de ce bloc extrait est égale à 9.
Règle5	Si un des nœuds enfants d'un nœud du DOM tree est un line break node, ce nœud sera divisé.
Règle6	si un des nœuds enfants d'un nœud du DOM tree contient la balise <HR>, divisez ce nœud.
Règle7	Si la couleur du fond de ce nœud est différente d'un de ses enfants, divisez ce nœud et en même temps, le nœud enfant avec les couleurs de fond différentes ne sera pas divisé dans ce rond. <ul style="list-style-type: none"> • Affectez une valeur de DoC (de 6 à 8) pour le nœud enfant basé sur les balises HTML et la taille du nœud enfant.
Règle8	si un nœud a au moins un text node ou un virtual text node comme enfant, et la taille relative du nœud est inférieure au seuil, alors le nœud ne peut pas être divisé. <ul style="list-style-type: none"> • Affectez une valeur de DoC (de 5 à 8) basée sur les balises HTML du nœud.
Règle9	Si l'enfant du nœud avec la taille maximale est plus petite qu'un seuil (taille relative), ne divisez pas ce nœud. <ul style="list-style-type: none"> • Donnez une valeur de DoC basé sur la balise HTML et la taille de ce nœud.
Règle10	Si les enfants précédents du même parent n'ont pas été divisés, ne divisez pas ce nœud.
Règle11	Divisez ce nœud.
Règle12	Ne divisez pas ce nœud. <ul style="list-style-type: none"> • Donnez la valeur de DoC basée sur la balise HTML et la taille de ce

nœud.

Les règles énumérées précédemment ne sont pas appliquées à tout les types de balise mais chaque règle est destiné à un certaines type de nœuds seulement.

Le tableau 3.3. indique les règles appliquées pour chaque type de nœuds, par exemple la règle1 est appliquée pour tout les type de nœuds par contre à la règle à la règle10 qui est appliquée sur le <TD> seulement.

Tableau 3.3: Les différentes règles pour les différents nœuds.

	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12
Inline text node	/	/	/	/	/	/		/	/		/	
<TABLE>	/	/	/				/		/			/
<TR>	/	/	/				/		/			/
<TD>	/	/	/	/				/	/	/		/
<P>	/	/	/	/	/	/		/	/		/	
Autre balise	/	/	/	/		/		/	/		/	

Les règles sont appliquées par leurs ordres dans le tableau 3.2 (de 1 à 12) sur les différents nœuds du DOM tree. Dès qu'une règle est applicable sur un nœud (la première règle qui convient) la règle est exécutée et ensuite on passe à un autre nœud.

La figure 3.7 (b) est une table, qui est une partie d'une page Web. Sa structure arborescente de DOM est montrée au côté gauche de la figure7. Dans le processus d'extraction de bloc, quand le nœud < TABLE > est rencontré, il a seulement un seul enfant valide < TR >, donc selon la règle2 le nœud <TABLE> sera remplacé par le nœud < TR >. Le nœud < TR > a cinq enfants < TD > et seulement trois d'entre eux sont valides. Le premier <TD> a une couleur du fond différente de la couleur du fond de son parent. Selon la règle 7, le nœud < TR > est divisé et le nœud <TD > ne peut pas être divisé plus loin dans ce rond, il est mis dans le pool et il sera considéré comme bloc. Le deuxième et le quatrième enfant du nœud de < TR > ne sont pas valides et ils sont supprimés. Selon la règle10, le troisième et le cinquième enfant du < TR > ne seront pas divisés dans ce rond. En conclusion, on obtient trois blocs, VB2_1, VB2_2, et VB2_3.

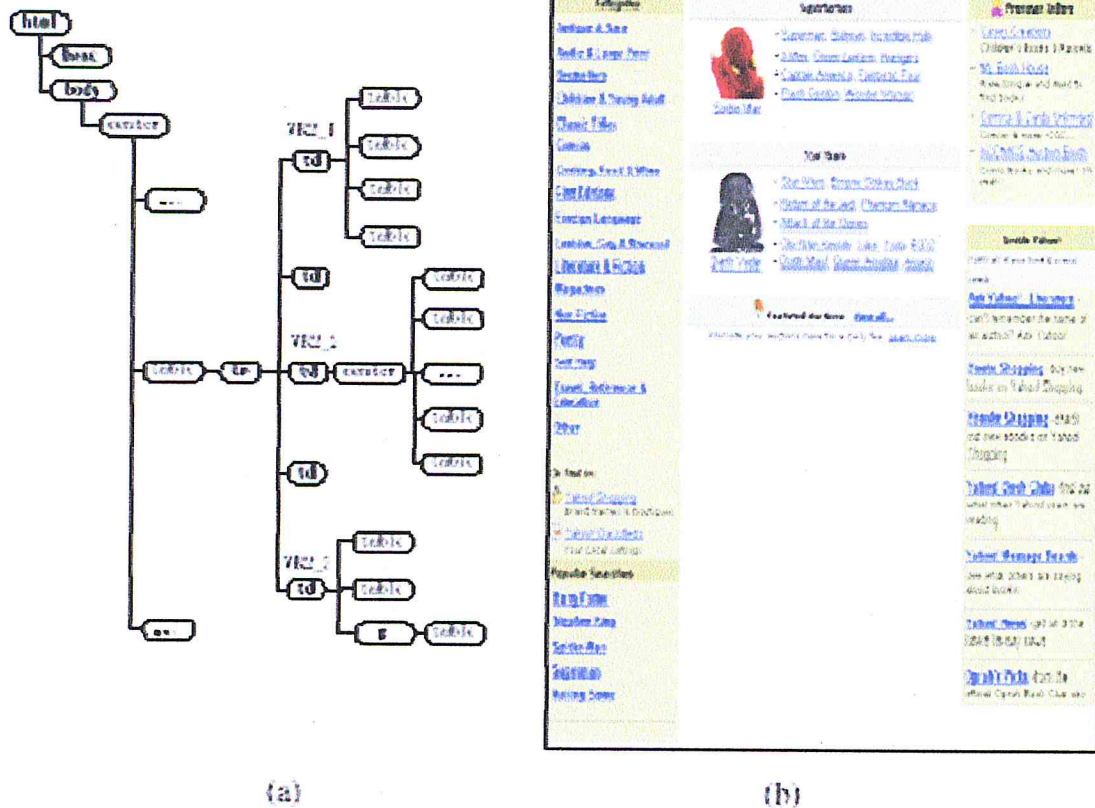


Figure 3.7: Visual block extraction d'une sous page.

IV.3. Détection des Séparateurs Visuels[30]

Les séparateurs sont des lignes horizontales ou verticales dans une page web qui ne croisent pas visuellement avec les blocs dans le pool. D'une perspective visuelle, les séparateurs sont des bons indicateurs de la différence sémantique entre les blocs dans une page Web. Un séparateur visuel est représenté par un 2-tuple: (P_s, P_e) , où la P_s est le Pixel de début de séparateur et P_e est le Pixel de fin de séparateur. La largeur du séparateur est calculée par la différence entre ces deux valeurs.

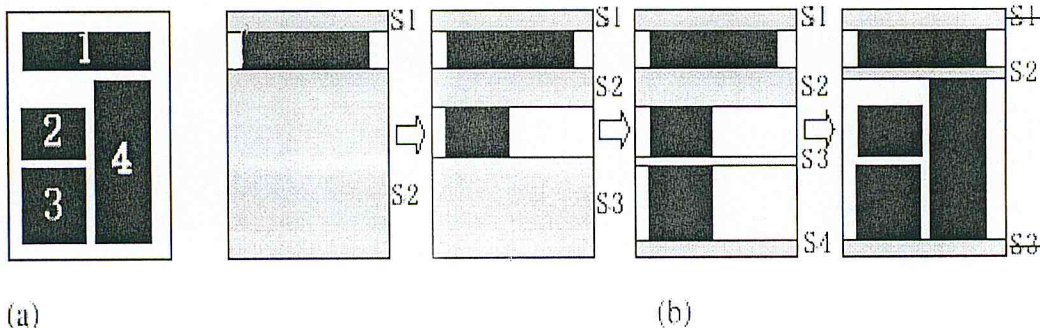
- Détection De Séparateur

L'algorithme de Detection des Separateurs Visuels est décrit comme suit:

1. Initialiser la liste de séparateur : La liste commence par un seul séparateur (P_{be}, P_{ee}) dont le Pixel de début et le Pixel de fin sont les frontières du pool.
2. Pour chaque bloc dans le pool, la relation du bloc avec chaque séparateur est évaluée comme suit :

- a) Si un bloc est contenu dans le séparateur, alors divisez le séparateur;
 - b) Si un bloc croise avec le séparateur, alors il faut mettre à jour les paramètres du séparateur;
 - c) Si un bloc couvre le séparateur, alors supprimez le séparateur.
3. Supprimez les quatre séparateurs qui se trouvent à la frontière du pool.

Prendre la figure 3.8 (a) comme exemple dans laquelle les blocs noirs représentent les blocs visuels dans la page Web. Pour simplifier les choses seulement que le processus de détection des séparateurs horizontaux sont montrés. D'abord et d'après l'étape 1 la liste des séparateurs est initialisée par un seul séparateur qui dont les paramètres de début et de fin sont les frontières du pool. Quand le premier bloc est mis dans le pool, il divise le séparateur en S1 et S2 selon l'étape 2(a). C'est pareil pour le deuxième et troisième bloc. Quand le quatrième bloc est mis dans le pool, il croise avec le séparateur S2 et couvre le séparateur S3, les paramètres de S2 sont mis à jour selon l'étape 2(b) et S3 est supprimé selon l'étape 2(b). À la fin de ce processus et d'après l'étape 3, les deux séparateurs S1 et S3 qui se trouvent aux frontières du pool sont enlevés.



(a) (b)
Figure 3.8 : Processus de détection de séparateur d'une page simple.

- Affectation des poids pour des séparateurs

Après la détection des séparateurs visuels, chaque séparateur a une valeur d'importance qui sera exprimé par son poids. Le poids d'un séparateur est affecté suivant la différence visuelle entre ses blocs voisins, par exemple la distance, la différence de couleur, etc. Les règles suivantes sont employées pour donner un poids ou une valeur d'importance à chaque séparateur:

1. Plus la distance entre les blocs des différents côtés du séparateur est grande, plus le poids est élevé.
2. Si un séparateur visuel est recouvert avec certaines balises HTML (par exemple, la balise HTML<HR>), son poids est plus élevé.

3. si les couleurs de fond des blocs des deux côtés du séparateur sont différentes, le poids sera élevé.
4. pour les séparateurs horizontaux, si la différence des propriétés de police telles que la taille et le poids des deux côtés du séparateur est grande, le poids sera élevé. D'ailleurs, le poids sera augmenté si la taille de police du bloc au-dessus du séparateur est plus petite que la taille de police du bloc au-dessous du séparateur.
5. pour les séparateurs horizontaux, quand les structures des blocs des deux côtés du séparateur sont très semblable (par exemple tous les deux sont des texte), le poids du séparateur sera diminué.

Prendre le troisième noeud $\langle TD \rangle$ sur la figure 3.7 comme exemple. La sub-page correspond à ce noeud est montrée sur la figure 9 (b) et la structure arborescente du DOM est montrée sur la figure 3.9 (a). Comme on peut voir beaucoup de noeuds dans le DOM tree sont non valides. Ils sont ignorés dans le processus d'extraction des blocs visuels. Après que l'étape d'extraction de bloc est terminée, six blocs sont mis dans le pool et cinq séparateurs horizontaux sont détectés. Alors les poids de ces séparateurs sont affectés sur la base des cinq règles ci-dessus. Dans cet exemple, le séparateur entre le bloc 2 et le bloc 3 aura un poids plus élevé que le séparateur entre le bloc 1 et le bloc 2 en raison de la différence des poids de police et la règle 4. Pour la même raison, le séparateur entre le bloc 4 et le bloc 5 aura également un poids élevé. Les séparateurs et les poids finals sont montrés sur la figure 9(c), dans lequel une ligne plus épaisse signifie un poids plus élevé.

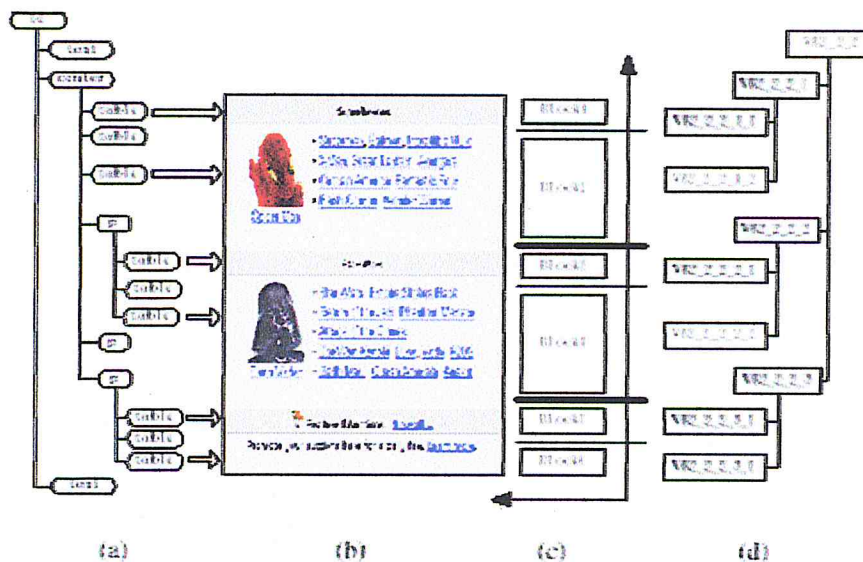


Figure 3.9 : Les séparateurs et leurs poids.

IV.4. La construction de la structure du contenu [30]

Quand les séparateurs sont détectés et leurs poids sont affectés, la structure du contenu peut être construite. Le processus de construction commence à partir

des séparateurs avec le plus petit poids et les blocs autour de ceux-ci sont fusionnés pour former de nouveaux blocs. Ce processus de fusionnement sera répété jusqu'à ce que les séparateurs avec les poids maximaux soient rencontrés.

Après, chaque nœud de feuille est vérifié s'il répond à la condition de granulite. Pour chaque nœud qui échoue, il sera considéré comme une sous-page et il passe à l'étape de l'extraction des blocs visuels encore pour construire la sous structure de contenu avec ce nœud. Si tous les nœuds répondent à la condition, le processus itératif est alors arrêté et la structure de contenu à base de la vision de la page entière est obtenu. La condition pour DoC est $DoC > PDoC$, si PDoC est prédéfini.

Prenant la figure 3.9 comme exemple. Dans la première itération, les premiers, troisième et cinquièmes séparateurs sont choisis car ils ont le plus petit poids, les blocs 1 et 2 sont fusionnés pour former le nouveau bloc $VB2_2_2_1$. Le fusionnement semblable est appliqué sur les blocs 3 et 4 (dont le résultat est le bloc $VB2_2_2_2$) et les blocs 5 et 6 (dont le résultat est le bloc $VB2_2_2_3$). Les nouveaux blocs $VB2_2_2_1$, $VB2_2_2_2$ et $VB2_2_2_3$ sont les enfants de $VB2_2_2$ et peuvent également être regardés comme une partition de $VB2_2_2$. Chaque nœud de feuille, tel que $VB2_2_2_1_1$, $VB2_2_2_1_2$ et $VB2_2_2_2_1$, sera vérifié pour voir s'il répond à la condition de granulite. Après plusieurs itérations, la vision-based content structure finale de la page est construite.

En résumé, l'algorithme VIPS tire profit des propriétés visuelles pour obtenir la structure de contenu à base de la vision d'une page Web et établit ainsi avec succès le lien entre la structure du DOM tree et la structure sémantique. La page est divisée à base des séparateurs visuels et elle est structurée sous une forme d'une hiérarchie. Cette hiérarchie sémantique est conformée à la perception humaine dans certaine mesure. Le VIPS est également très efficace par rapport à la structure du DOM car deux fils appartient au même parent n'implique pas qu'ils sémantiquement liés. Puisque le VIPS commence de traiter la page Web de la racine de la structure du DOM pour l'extraction visuelle des blocs et il n'analyse pas chaque nœud de base de DOM tree, donc on conclue que l'algorithme est totalement de haut en bas.

V. CONCLUSION

Dans ce chapitre on a donné une vue globale sur le VIPS dont le but est d'extraire la structure du contenu d'une page web sur la base de la représentation visuel. La structure du contenu d'une page web produite est très utile pour plusieurs domaines telles que l'adaptation du web, information retrieval et information extraction. La structure du contenu d'une page web peut efficacement représenter la structure sémantique de la page Web. Un algorithme automatique de haut en bas, pour détecter la structure du contenu d'une page web. Il simule comment un utilisateur comprend la structure de layout d'une page Web basé sur sa représentation visuelle. Comparé au résultat obtenu par le DOM tree, les résultats obtenus par l'algorithme VIPS sont plus logiques.



Chapitre4

*Démarche de
développement de*

I. INTRODUCTION

Dans le chapitre précédent, nous avons parlé sur le principe du vision-based content structure et plus précisément sur *VIPS* (VIsion-based Page Segmentation), qui a pour but d'extraire la structure sémantique d'une page Web.

Dans ce présent chapitre, nous présentons les étapes les plus importantes dans la conception et la réalisation de notre système, de l'analyse des besoins jusqu'à l'implémentation.

II. DEMARCHE DE DEVELOPEMENT

Pour développer un logiciel, il est nécessaire de passer par plusieurs étapes. Pour cela, des méthodes de développement ont été définies, ces méthodes permettent de mieux organiser, d'avoir une meilleure compréhension, de réduire la complexité des applications et permettent une plus grande facilité dans l'interprétation des concepts logiciels.

Pour la modélisation de notre travail, nous utilisons la notation UML qui représente un langage de modélisation et non pas une méthode objet. Donc on peut dire que UML ne décrit pas une démarche de développement de logiciel. Le processus de développement que nous utilisons est le modèle "en cascade" [32].

Le modèle "en cascade" présente un cycle de vie d'un logiciel par une suite de phases (analyse, conception, implémentation, test et validation) (figure 4.1) qui s'enchaînent dans un déroulement linéaire depuis l'analyse des besoins jusqu'à la maintenance.

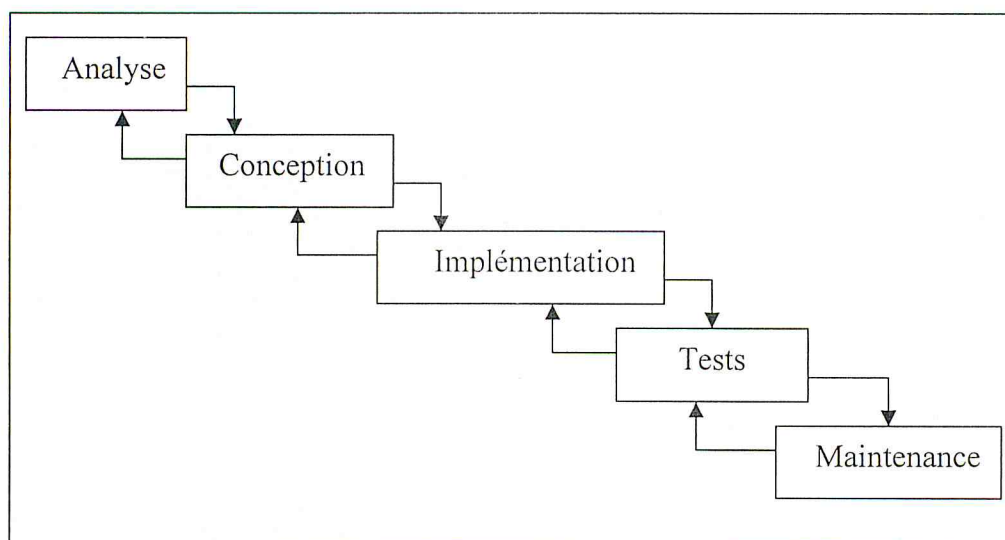


Figure 4.1: Le processus de développement en cascade.

III. ANALYSE ET SPECIFICATION DES BESOINS

L'analyse et spécification des besoins est une étape très importante au début du processus de développement. Son but est d'éviter de développer en logiciel non adéquat [32].

La finalité de cette étape est la description générale des fonctionnalités du système. Par la réponse à ces questions:

- Quelles sont les fonctions du système?
- Quels sont les utilisateurs du système?
- Qu'attendent t'ils du système?

Cette étape étudie le comportement du système exprimé sous la forme des cas d'utilisation, le contexte du système, les acteurs et les scénarios.

III.1. Les cas d'utilisation (Use cases)

La spécification des cas d'utilisation détermine le "le quoi faire", c'est-à-dire les besoins de l'utilisateur. L'expérience montre que la technique des cas d'utilisation se prête bien à la détermination des besoins d'utilisateurs.

Un cas d'utilisation est un ensemble d'actions réalisées par le système en réponse à une action d'un acteur.

L'étude des cas d'utilisation débute par la détermination des acteurs (catégories des utilisateurs) du système.

III.1.1. Les acteurs

Un acteur est une entité externe qui agit sur le système (opérateur, autre système...) et qui peut consulter ou modifier l'état du système.

Dans notre cas il y a un seul utilisateur principal (l'utilisateur de système) qui peut être un informaticien ou non.

III.1.2. Les cas d'utilisation principaux

Notre système est divisé en trois cas principaux, qui sont les suivants (voir figure 4.2)

- 1) Chargement de page.
- 2) Détermination du DOM.
- 3) Détermination du VIPS.

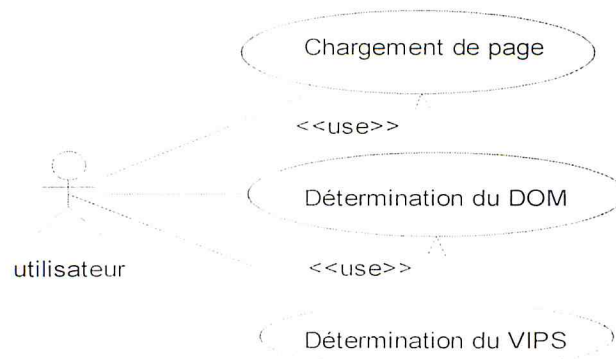


Figure 4.2 Cas d'utilisation principale

Nous expliquerons, par la suite chaque cas d'utilisation en donnant les diagrammes des sous cas d'utilisation qui lui correspondent, afin de pouvoir comprendre le diagramme général (récapitulatif de tous les diagrammes particuliers).

1) Chargement de page

L'utilisateur se charge du chargement d'un document HTML, en spécifiant son emplacement.

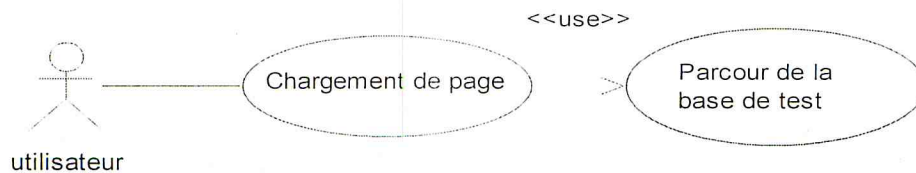


Figure 4.3 : Diagramme de cas d'utilisation «chargement de page »

2) Détermination du DOM

Pour chaque document HTML charger correspond un modèle objet de document. Chaque nœud dans le DOM à des propriétés visuelles qui les correspondent.

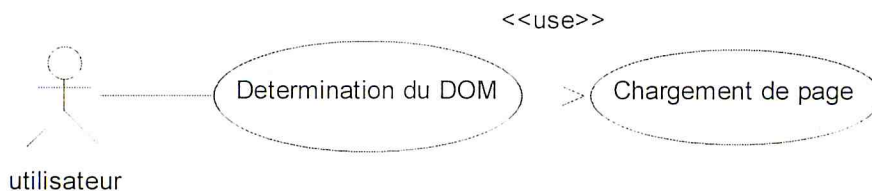


Figure 4.4 : Diagramme de cas d'utilisation «Détermination du DOM »

3) Détermination du VIPS

La détermination du VIPS consiste à extraire la structure sémantique d'une page Web. Elle incluse :

- Le nettoyage.
- La Segmentation.
- La détection des séparateurs.
- La construction de l'arborescence.

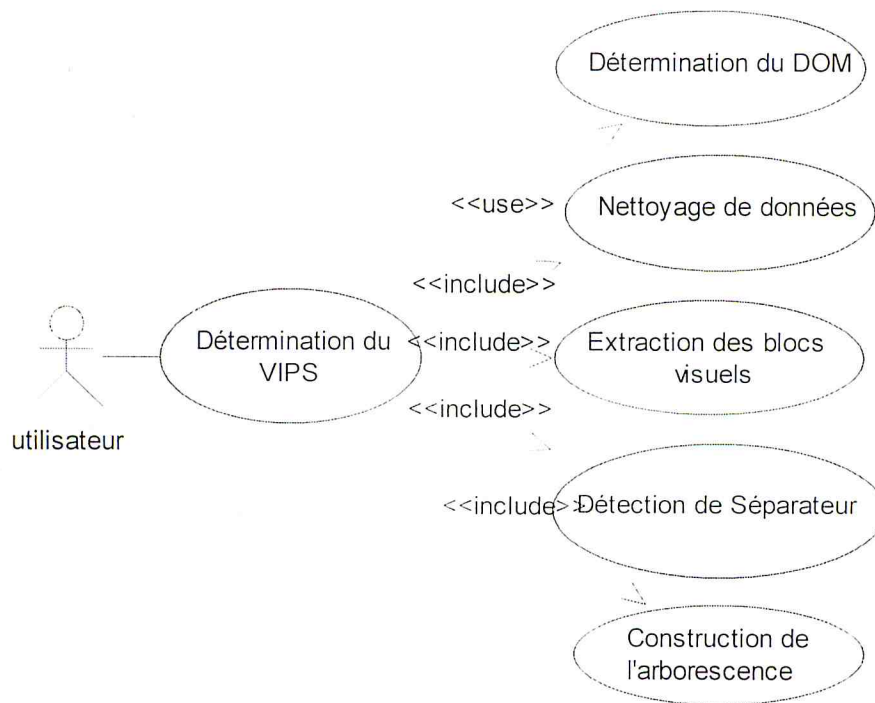


Figure 4.5 Diagramme de cas d'utilisation « détermination du VIPS »

✓ Le nettoyage

Le nettoyage est une étape très importante dans la détermination du VIPS, elle consiste à supprimer les fils non valide de l'arborescence du DOM.

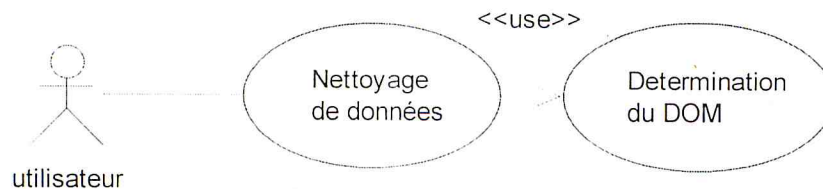


Figure4.6 : Diagramme de cas d'utilisation « Nettoyage de données »

✓ Extraction des blocs visuels

L'extraction des blocs visuels consiste à déterminer tout les segments ou les blocs de page ou du sub-page.

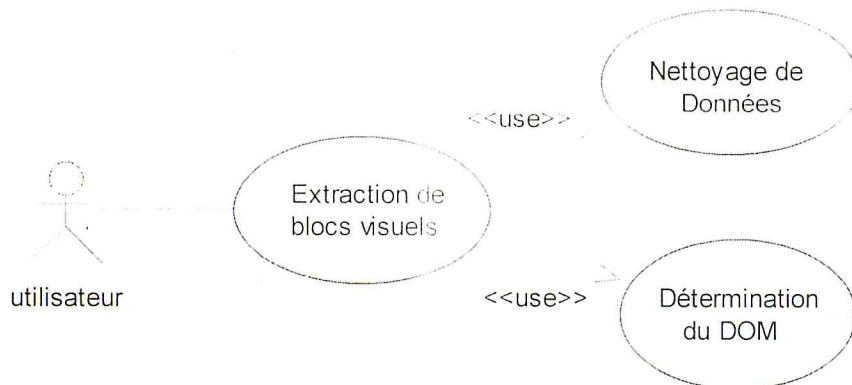


Figure4.7 : Diagramme de cas d'utilisation « Extraction de blocs visuels »

✓ Détection des séparateurs

La détection des séparateurs consiste à trouver tous les séparateurs horizontaux et verticaux qui séparent les segments de page ou de sub-page.

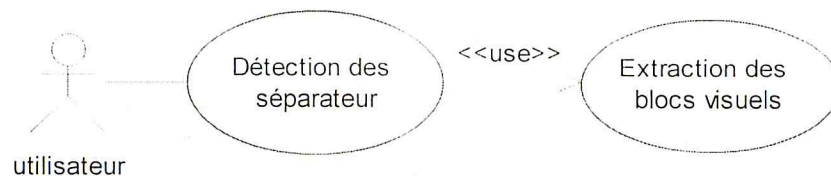


Figure 4.8 : Diagramme de cas d'utilisation «Détection des séparateurs »

✓ Construction d'arborescence

L'ensemble des blocs visuels et l'ensemble des séparateurs sont utilisés pour construire l'arbre sémantique qui correspond au document HTML.

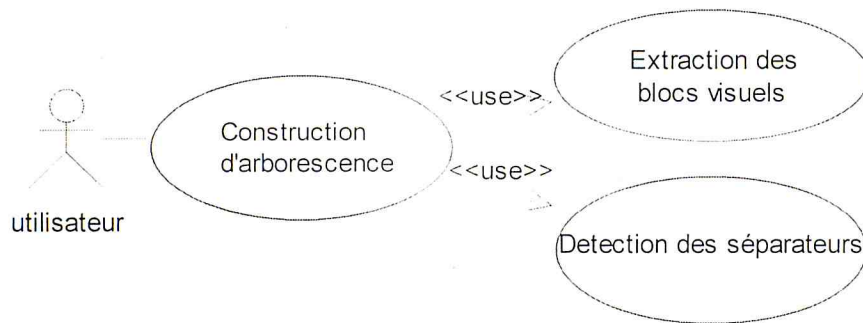


Figure 4.9 Diagramme de cas d'utilisation « construction d'arborescence »

III.1.3. Diagramme de cas d'utilisation général

Les différentes fonctionnalités offertes par notre outil forme ainsi un ensemble de cas d'utilisation, exprimés dans les sections précédentes. Afin de les formaliser, UML offre à travers les diagrammes de cas d'utilisations toutes les fonctionnalités que le futur système offrira à ses utilisateurs. La figure suivante illustre ce diagramme. (Figure 4.10)

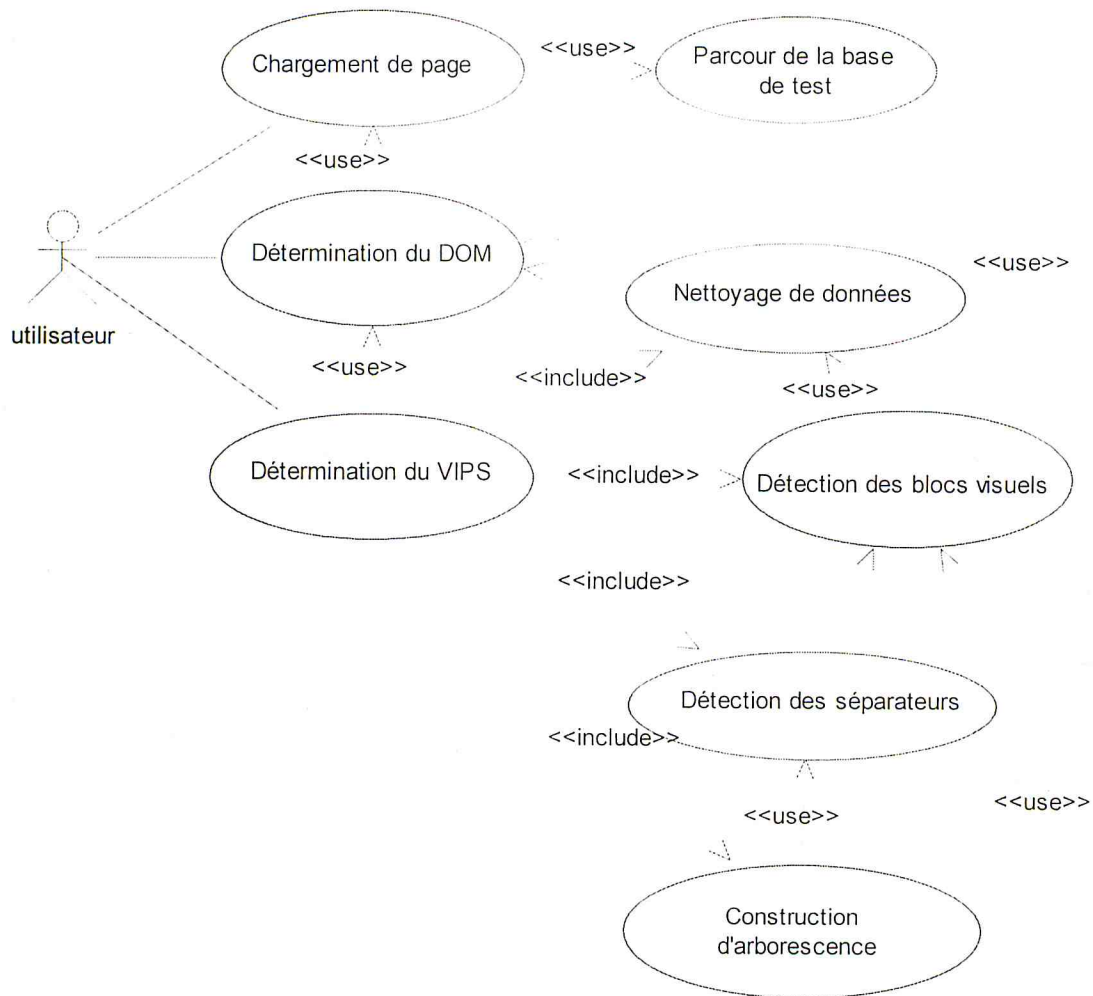


Figure 4.10 diagramme de cas d'utilisation général.

III.2. Diagramme de séquence

L'avantage des cas d'utilisation est d'être graphiquement très simple et donc facile à appréhender. Malheureusement, cette simplicité ne va pas sans certaine pauvreté sémantique [32]. Cependant les diagrammes de séquence nous permettent de bien schématiser les scénarios des cas d'utilisation et montrent les interactions entre plusieurs objets selon un point de vue temporel.

1) Chargement d'une page

Scénario

- ✓ L'utilisateur demande le chargement d'une page à partir de la base de test.
- ✓ Le système ouvre une fenêtre de parcours.
- ✓ L'utilisateur sélectionne une page web

- ✓ Le système charge la page.
- ✓ Après le chargement le système rend la main à l'utilisateur.

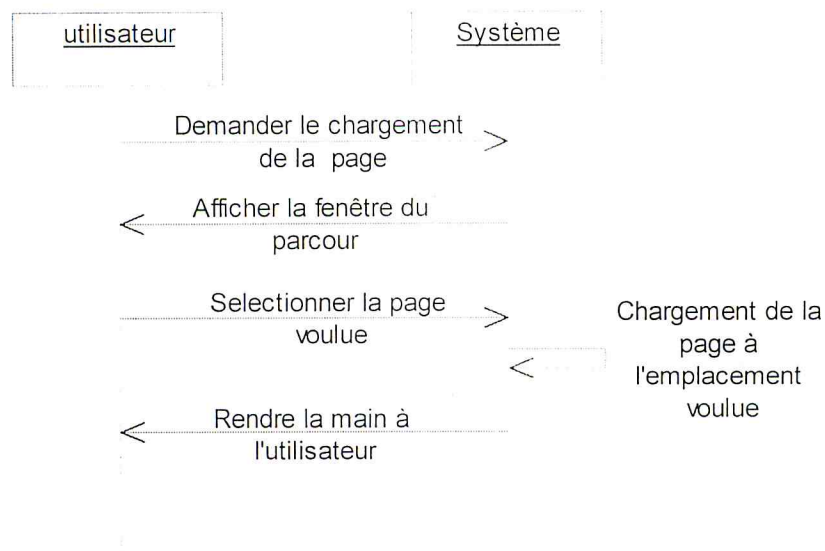


Figure 4.11 : Diagramme de séquence pour le chargement de page.

2) Modification de la page charger

Scénario

- ✓ L'utilisateur demande le chargement d'une page à partir de la base de test.
- ✓ Le système ouvre une fenêtre de parcour.
- ✓ L'utilisateur sélectionne une nouvelle page web
- ✓ Le système réinitialise l'espace de travail et charge la nouvelle page.
- ✓ Après le chargement le système rend la main à l'utilisateur.

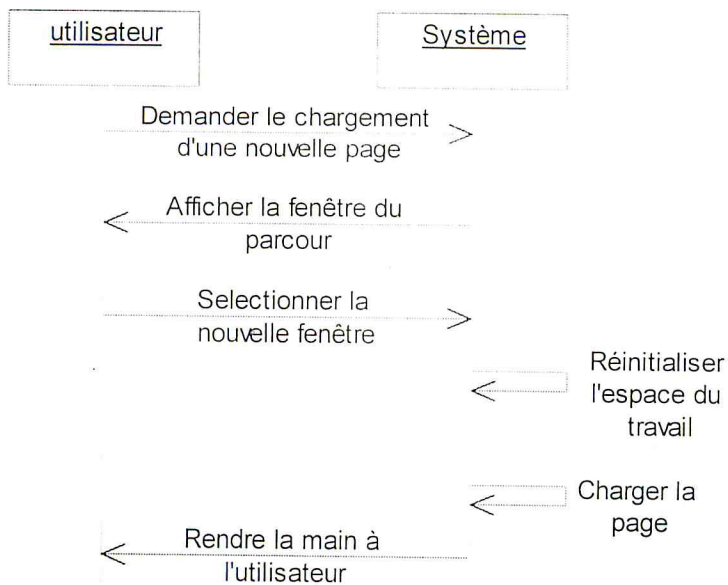


Figure 4.12 : Diagramme de séquence modification de la page charger.

3) Détermination du DOM

Scénario

- ✓ L'utilisateur lance l'opération d'extraction des blocs visuels.
- ✓ Le système extrait la structure du DOM.
- ✓ Le système affiche l'arborescence DOM associée au document.
- ✓ Le système rend la main à l'utilisateur.

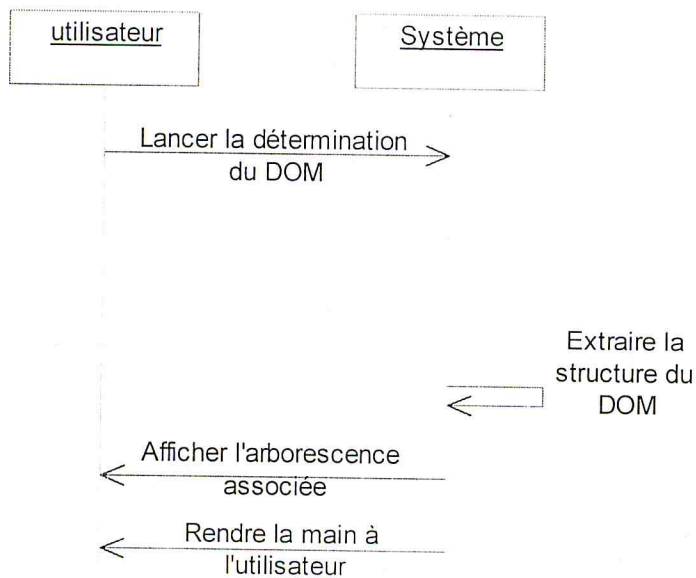


Figure 4.13 : Diagramme de séquence détermination du DOM.

4) Parcours de l'arbre est visualisation des nœuds et les attributs visuels associés

Scénario

- ✓ L'utilisateur sélectionne un nœud.
- ✓ Le système génère les attributs associés.
- ✓ Le système affiche l'ensemble des attributs dans la zone associée.
- ✓ Le système rend la main à l'utilisateur.

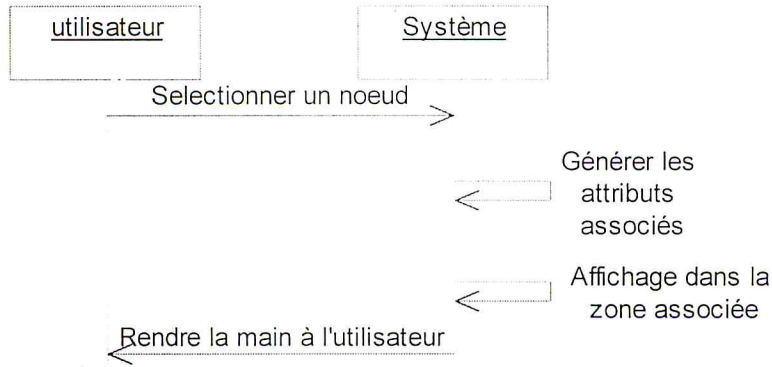


Figure 4.14 : Diagramme de séquence Parcours de l'arbre est visualisation des nœuds et les attributs visuels associés

5) Nettoyage de données

Scénario

- ✓ L'utilisateur lance l'opération de nettoyage de données.
- ✓ Le système supprime les fils non valides.
- ✓ Le système rend la main à l'utilisateur.

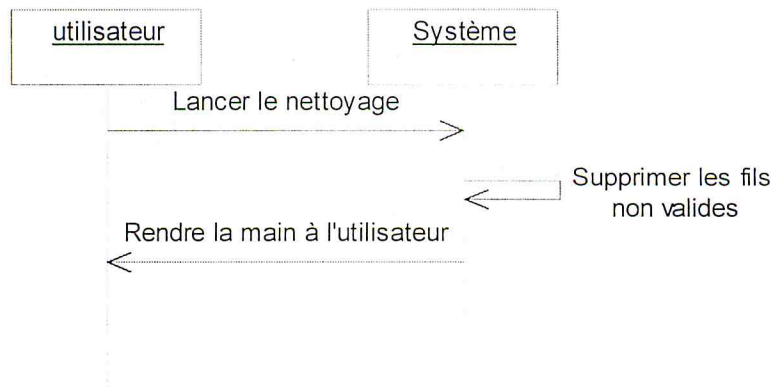


Figure 4.15 : Diagramme de séquence nettoyage de données

6) Extraction des blocs visuels

Scénario

- ✓ L'utilisateur lance l'opération d'extraction des blocs visuels.
- ✓ Pour l'extraction le système applique des règles heuristiques.
- ✓ Le système affiche les segments extraits.
- ✓ Le système rend la main à l'utilisateur.

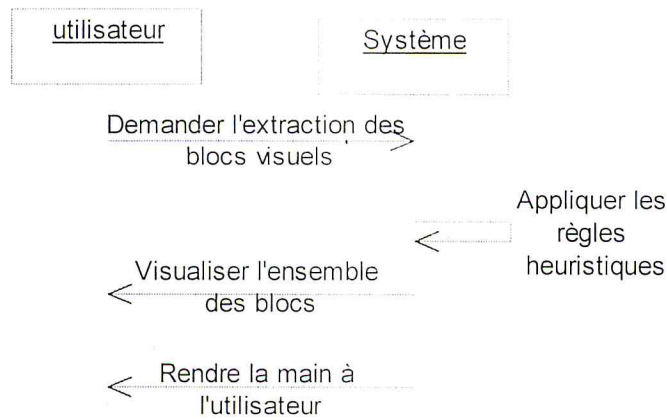


Figure 4.16 Diagramme de séquence extraction des blocs visuels

7) Détection des séparateurs

Scénario

- ✓ L'utilisateur lance l'opération de détection des séparateurs.
- ✓ Le système détecte les séparateurs.
- ✓ Le système identifie les blocs au bord des séparateurs.
- ✓ Le système détermine les poids des séparateurs.
- ✓ L'utilisateur visualise les blocs au bord des séparateurs
- ✓ Le système rend la main à l'utilisateur.

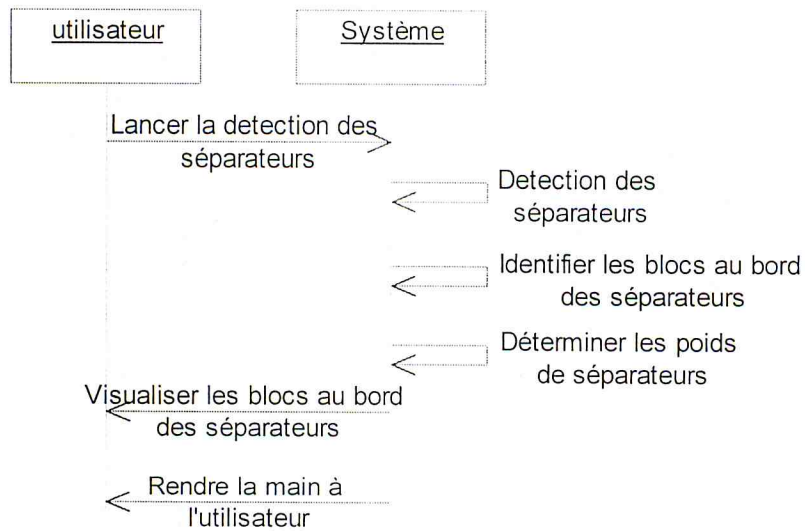


Figure 4.17 : Diagramme de séquence détection des séparateurs.

8) Construction de l'arborescence

Scénario

- ✓ L'utilisateur lance l'opération de construction d'arborescence.
- ✓ Le système visualise l'arbre sémantique.
- ✓ Le système rend la main à l'utilisateur.

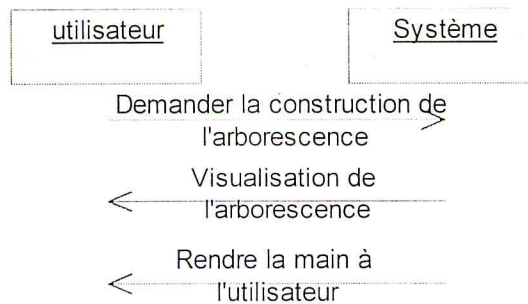


Figure 4.18: Diagramme de séquence construction de l'arborescence

9) Parcours de l'arbre sémantique est visualisation des segments et les attributs visuels associés

Scénario

- ✓ L'utilisateur sélectionne un segment.
- ✓ Le système génère les attributs associés.
- ✓ Le système affiche l'ensemble des attributs dans la zone associée.
- ✓ Le système rend la main à l'utilisateur.

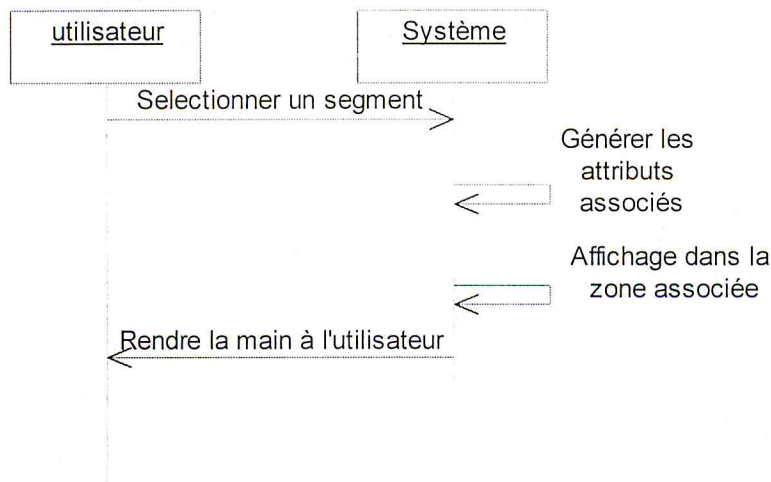


Figure 4.19: Diagramme de séquence parcours de l'arbre sémantique est visualisation des segments et les attributs visuels associés

III.3. Diagramme d'activité

UML permet de représenter graphiquement le comportement d'une méthode ou le déroulement d'un cas d'utilisation à l'aide de diagrammes d'activités. Une

activité représente une exécution d'un mécanisme, un déroulement d'étapes séquentielles.

Il est possible de décrire les acteurs responsables de chaque activité par l'utilisation des "couloirs d'activités" qui permettent de répartir graphiquement les différentes activités entre les acteurs opérationnels [32]. Chaque activité est placée dans le couloir correspondant à l'acteur qui assume celle-ci. Nous utilisons ce formalisme pour présenter le diagramme d'activité générale du système.

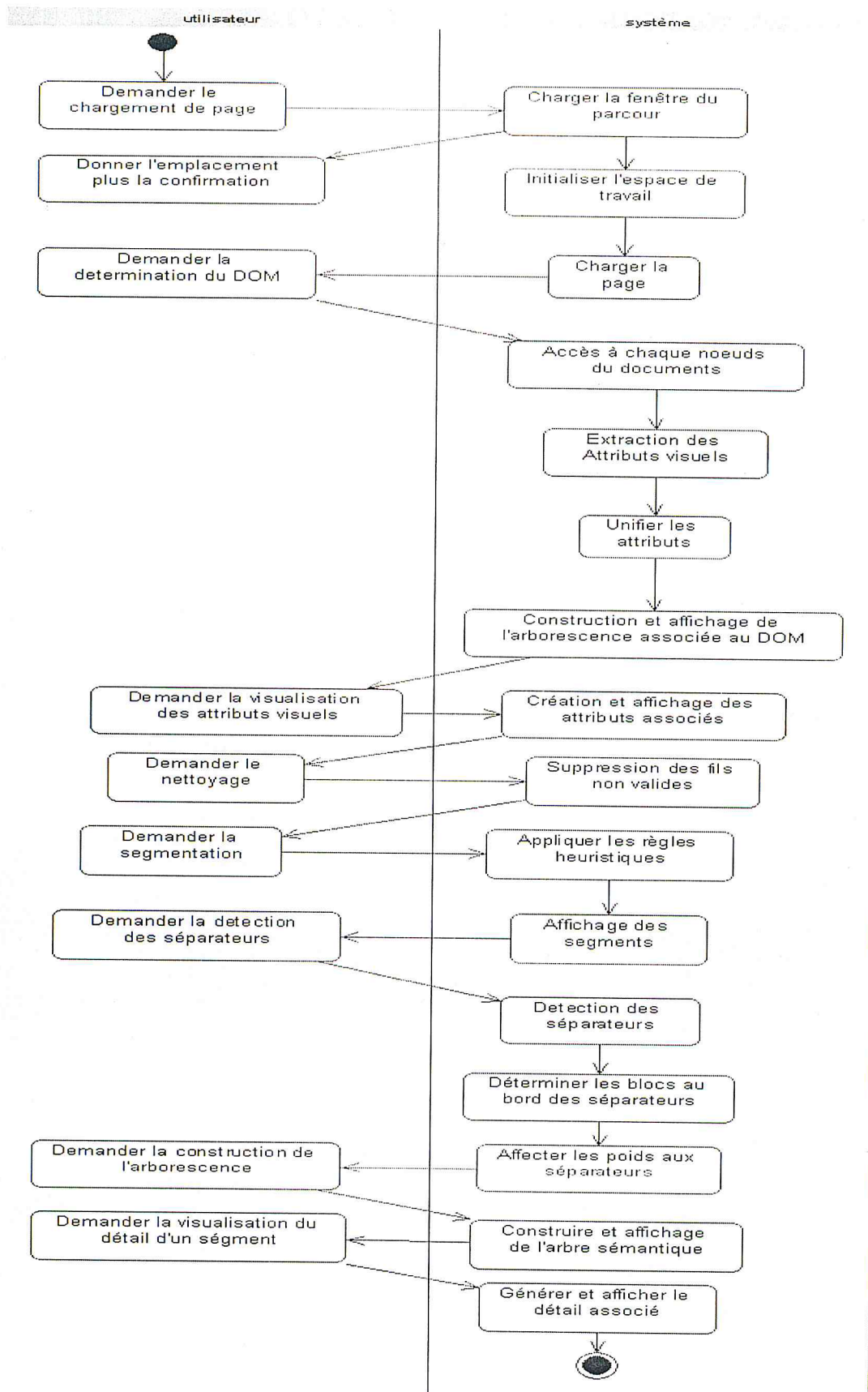


Figure 4.20: Diagramme d'activité du système.

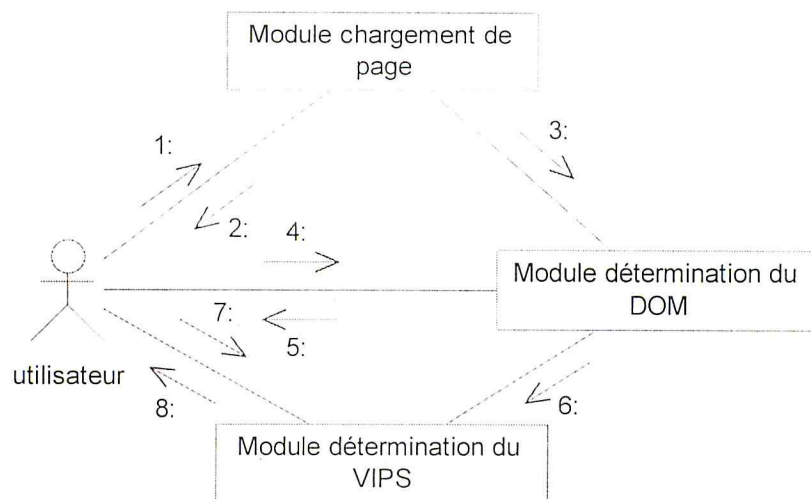
IV. CONCEPTION

La phase d'analyse est suivie de la phase de conception. Cette phase généralement s'intéresse au "comment" dans la réalisation du logiciel. La conception commence par une conception dite "globale" qui décrit l'architecture du système, et elle sera suivie par une conception détaillée.

IV.1. Conception globale

La conception globale a pour but de décomposer le logiciel en module. Ce dernier représente une unité pour la manipulation des applications.

Voici le diagramme de collaboration qui montre les différents modules qui composent notre système, et leurs interactions.



- 1) Demander le chargement de page par l'utilisateur.
- 2) Le chargement de page par le système.
- 3) Donner l'accès à la page charger.
- 4) Déclencher la détermination du DOM.
- 5) Structure DOM extraite.
- 6) Donner l'accès aux nœuds du DOM et leurs attributs visuels.
- 7) Demander la détection du VIPS.
- 8) Structure VIPS extraite.

Figure 4.21 : Diagramme des collaboration des modules de système.

IV.1.1 Architecture de notre système

Dans notre cas, nous proposons une architecture de trois modules :

- Module chargement de page.
- Module détermination du DOM.

- Module détermination du VIPS.

Les composants de notre système peuvent être décrits à l'aide du diagramme de composants (figure 4.22).

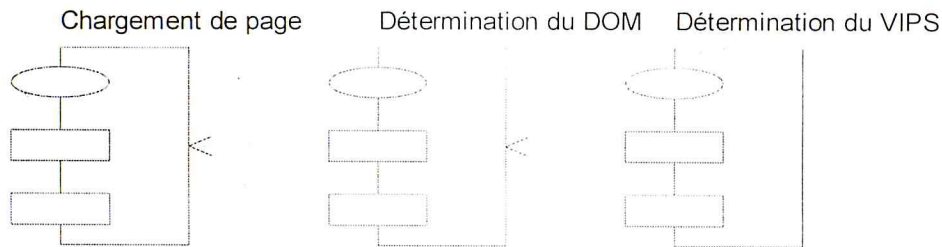


Figure 4.22 : Diagramme de composants de notre système.

- Le module « chargement de page »
Il se charge du chargement de la page sélectionnée par l'utilisateur à partir de la base de test.
- Le module « détermination du DOM »
Ce module se charge de la génération de l'arborescence associée au document HTML et la création de la représentation associée. Celle-ci doit être accessible par l'utilisateur pour qu'il puisse visualiser les nœuds et leurs attributs.
- Le module « détermination du VIPS »
Le module se charge de la génération de l'arbre sémantique associée au document HTML et la création de la représentation associée. Celle-ci doit être accessible par l'utilisateur pour qu'il puisse visualiser les segments et leurs attributs.

IV.2. Conception détaillée

La conception détaillée fournit pour chaque module, une description détaillée de la manière dont les fonctions du composant sont réalisées (algorithmes et représentation des données)

IV.2.1. Le module « chargement de page »

Le module chargement de page contient deux fonctions qui sont : le chargement du document HTML (depuis l'emplacement spécifié par l'utilisateur), et la réinitialisation du processus en cas du chargement d'un nouveau document.

Ces éléments sont présentés dans la figure suivante (figure 4.23) sous le format d'un paquetage :

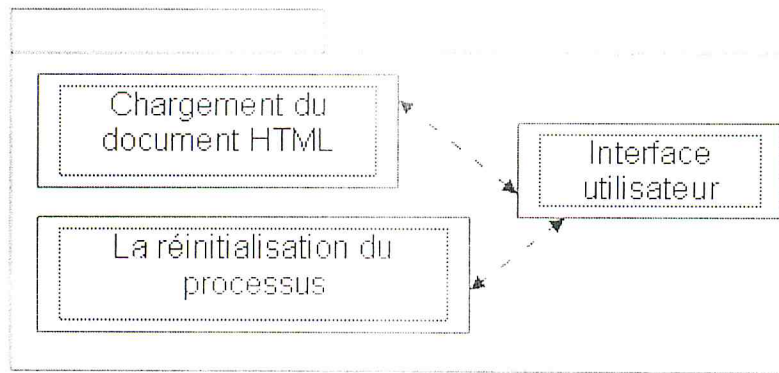


Figure 4.23 : Les éléments du module de chargement de page.

IV.2.1.1. Chargement du document HTML

Le chargement d'un document HTML se fait à la demande de l'utilisateur dans l'emplacement spécifié dans l'interface utilisateur.

IV.2.1.2. La réinitialisation du processus

Quant un utilisateur demande un chargement d'un nouveau document HTML, l'interface utilisateur sera réinitialisée (vue document, vue structurelle, vue attribut).

IV.2.2. Le module « détermination du DOM »

Le module de chargement de page contient deux fonctions qui sont : l'accès et l'extraction des attributs visuels associés à chaque nœud, et la création de la représentation associée dans l'interface utilisateur.

Ces éléments sont présentés dans la figure suivante (figure 4.24) sous le format d'un paquetage :

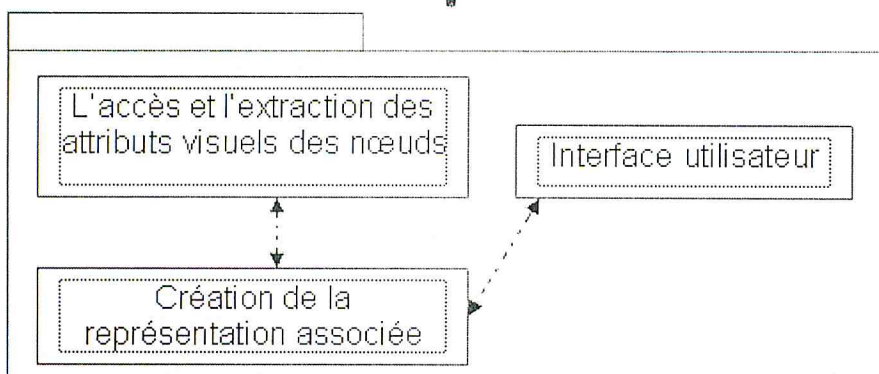


Figure 4.24 Les éléments du module de détermination du DOM

IV.2.2.1 L'accès et l'extraction des attributs visuels des nœuds

Quand un utilisateur veut déterminer le DOM du document HTML, il faut que ce-ci soit parcouru. Donc il doit y avoir un accès à chaque nœuds au document et au même temps une extraction des attributs visuels.

IV.2.2.2 création de la représentation associée au nœud

La structure DOM doit être visualisée par l'utilisateur via une interface (vue document, vue structurelle, vue attribut).

IV.2.3 Le module « détermination du VIPS »

Le module chargement de page contient quatre fonctions qui sont : le nettoyage, l'extraction des blocs visuels (segmentation), la détection des séparateurs (horizontaux ou verticaux), la construction de l'arbre sémantique.

Ces éléments sont présentés dans la figure suivante (figure 4.25) sous le format d'un paquetage :

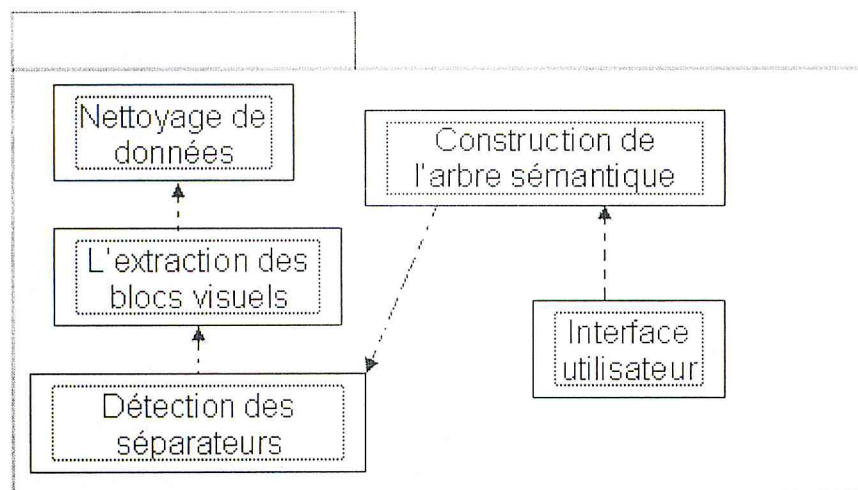


Figure 4.25 : 22 Les éléments du module détermination du VIPS

IV.2.3.1. Nettoyage de données

Quand un utilisateur veut déterminer la structure correspondante au VIPS la première chose à faire c'est le nettoyage de données, c'est-à-dire supprimer les fils non valide (qui non pas d'apparition sur le navigateur)

IV.2.3.2. L'extraction des blocs visuels

L'extraction des blocs visuels consiste à appliquée les règles heuristiques à une page ou une sub-page. Ces règles sont appliquées suivant leurs priorités sur un nœud du DOM et ensuite sur ces fils (si ce dernier est divisé) jusqu'à tous les segments sont dans le pool.

IV.2.3.3. La détection des séparateurs

Un séparateur est un bon indicateur de différence sémantique entre deux segments. La détection des séparateurs consiste à

- trouver l'ensemble des séparateurs (horizontaux, ou verticaux).
- détecter les blocs qui sont au bord de chaque séparateur car la différence entre les deux segments revient à déterminer la différence entre les blocs qui se trouve au bord de séparateur.
- calculer le poids des séparateurs.

IV.2.3.4. la construction de l'arbre sémantique

La construction de l'arbre sémantique revient à établir les relations entre les segments. Ces relations sont basées sur les poids de séparateurs, car la construction consiste à établir des relations entre les segments qui se trouvent au bord des séparateurs qui ont de faible poids et ce processus est répété jusqu'à l'arrivé au séparateurs qui ont le poids le plus élevé.

IV.2.4. Interface utilisateur

Nous remarquons des sections précédentes que tous les modules ont une relation avec l'interface utilisateur. Notre outil a une interface contenant

- Une zone décrivant l'arborescence (DOM ou VIPS) qui sert d'une part à donner un aperçu de tous les éléments composants le document, et d'autre part, de décrire l'organisation hiérarchique de celui-ci. Cette zone est appelée « **vue structurelle** ».
- Une zone où seront affichés les différents attributs d'un objet sélectionné dans la vue hiérarchique, cette zone est appelée « **vue attribut** ».
- Une zone qui sert à visualiser le document charger par l'utilisateur à partir de la base de test et de visualiser l'objet sélectionné dans la vue hiérarchique. Cette zone est appelée « **vue document** »

IV.2.5. Diagramme de classes

Pour exprimer de manière générale la structure statique de notre système, nous utilisons le diagramme de classe qui représente cet aspect statique en terme de classe et de relations entre ces classes [32].

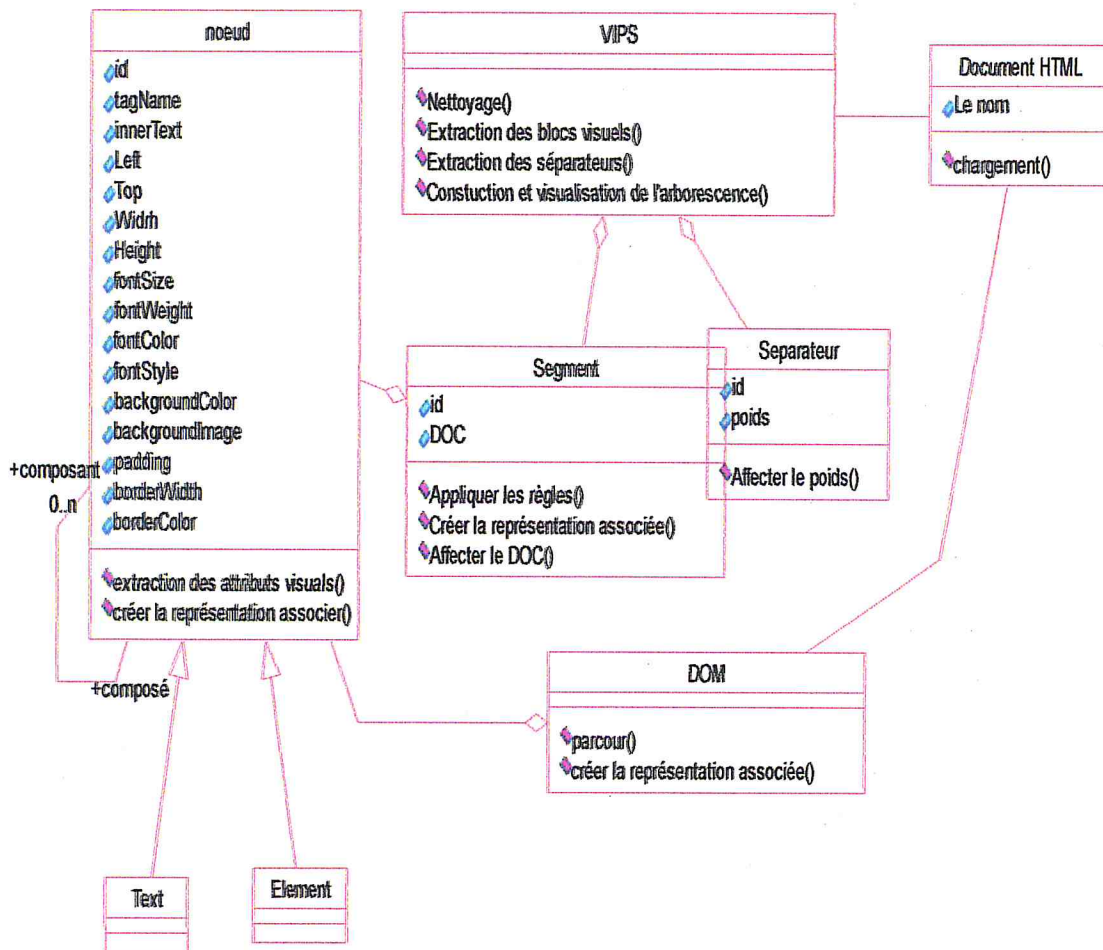


Figure 4.26 Diagramme de classe correspondant à notre outil.

V. IMPLEMENTATION

Dans cette phase nous exprimons la partie implémentation de notre projet, dont il s'agit d'implémenter la solution évoquée dans la partie conception. Autrement dit, l'implémentation est une phase aux cours de la quelle les algorithmes (dans notre cas les diagrammes de séquence) définis dans la partie conception sont traduits dans un langage de programmation.

V.1. Contexte matériel et logiciel

L'outil que nous avons conçu est implémenté sous le système d'exploitation Windows. Cette implémentation est effectuée sur un PC (Personal Computer) en utilisant le langage JavaScript via le navigateur Internet Explorer 5.5. Nous avons choisis ce langage car il est adapté à la programmation web.

V.2. Implémentation des modules

Dans cette partie, nous allons détailler les différents modules réalisés dans notre architecture

V.2.1. Implémentation du module « chargement de page »

V.2.1.1. Chargement du document HTML

Il est réalisé par la fonction suivante :

```
chargerPage ()
Début
  Activer le module de détermination du DOM et faire apparaître son
  interface ;
  Désactiver le module de détermination du VIPS et faire cacher son
  interface ;
  Charger la page à partir de l'emplacement introduit par l'utilisateur dans la
  zone de texte dédié ;
Fin
```

V.2.1.2. La réinitialisation du processus

Cette fonction est appelée lors d'un chargement d'un nouveau document

```
clearPage ()
Début
  chargerPage ()
  Si (un nouveau document est charger)
    Création d'un élément body vide c à d qui n'a pas de fils pour la vue
    structurelle et la vue attributs ;
    Remplacer les éléments body de la vue structurelle et la vue attributs
    par les nouveaux éléments body ;
  Fin si
Fin
```

V.2.2. Implémentation du module « détermination du DOM »

La variable globale dans ce module est « ele » qui est un tableau de deux dimensions, dont les lignes sont les objets du document et les colonnes sont les attributs visuels.

V.2.2.1 L'accès et l'extraction des attributs visuels des nœuds

Le rôle de cette fonction est d'unifier les unités de mesure pour tous les attributs visuels. Le paramètre s1 représente un attribut visuel.

Unifier les paramètres (s1)

Début

 Sj (s1 ne respecte pas les unités de mesure utilisées)

 Convertir le s1 ;

 Fsi

Fin

Cette fonction fait la création du détail de chaque objet du document. Elle a comme paramètres : obj qui représente un nœud dans la document, tableAttr qui est le tableau ou les paramètres sont placés, id qui est l'identifiant du tableau, text qui est un booléen indiquant si obj est de type Text ou non

createTrDétail(obj, tableAttr, id, text){

Début

 Si(text est vrai)

 Div= creation d'un element (<DIV>)

 Insérer Div comme parent du nœud obj ;

 Extraire les attributs visuels du Div qui sont les même pour le nœud obj ;

 Remplacer l'element Div par sont fils obj ;

 Détruire Div ;

 Sinon(c'est-à-dire que obj est de type Element)

 Extraire les attributs visuels du nœud obj ;

 Fsi

 Pour (tout les attributs visuels de l'element obj)

 Faire

 s=ele[obj][attributi)+"\n" ;

 Finfaire

 Tr= creation d'un element (<TR>);

 Td= creation d'un element (<TD>);

 Text= creation d'un Text (s);

 Affecter un identifiants id au nœud Tr ;

 Ajouter le nœud text comme fils au nœud Td ;

 Ajouter le nœud Td comme fils au nœud Tr ;

 Ajouter le nœuds Td au tableau tableAttr ;

 Cacher tr ;

Fin

Cette fonction fait le parcours du document afin d'extraire la structure et les attributs visuels.

Parcour()

Début

 Activer le module VIPS ;

 Pour (tout les element du Document charger)

 Faire

 Créer un tableau vide de deux dimensions qui a comme ligne l'identifiant de l'élément et chaque colonne représente un attribut visuel ;

 createTableDétail (identifiant de l'élément) ;

 Unifier les paramètres (ele[identifiant de l'élément][attribut]) ;

 Finfaire

Fin

V.2.2.2 création de la représentation associée au nœud

Cette fonction a pour but de créer le tableau qui contient les attributs et leurs valeurs dans la vue attributs, elle a comme paramètre id l'identifiant de tableau.

createTableDétail(id)

Début

```

table = création d'un Element (<TABLE>);
tbody = création d'un Element (<TBODY>);
tr1 = création d'un Element (<TR>);
tr2 = création d'un Element (<TR>);
td1 = création d'un Element (<TD>);
td2 = création d'un Element (<TABLE>);
td3 = création d'un Element (<TABLE>);
text1 = création d'un Text ("valeur");
text2 = création d'un Text ("attributs");
Donner un identifiant (id) au tableau (table);
Affecter des valeurs aux attributs visuels comme width et height des elements;
Ajouter le nœud text1 comme fils au nœud td1;
Ajouter le nœud text2 comme fils au nœud td2;
Ajouter le nœud td1 et td2 comme fils au nœud tr1;
Ajouter le nœud td3 comme fils au nœud tr2;
Ajouter les nœud tr1 et tr2 comme fils au nœud tbody;
Ajouter le nœud tbody comme fils au nœud table;

```

Fin

Cette fonction a pour but de créer la représentation des nœuds dans la vue structurelle, elle a comme paramètres id l'identifiant du nœud, Name qui représente l'attribut nodeName, li qui représente un élément de liste, et tr qui est la représentation la représentation graphique des attributs dans la vue attribut.

créerReprésentationNoeudVueStructurelle(id,Name,li,tr){

Début

```

Affecter un identifiant id à li;
Text = création d'un Text (Name);
Img = création d'un Element ('img');
Ajouter le nœud img et text comme fils au nœud li;
Si (l'élément qui a l'identifiant id n'a pas de fils)
    Affecter le chemin à l'attribut src du img;
    img.onmousedown
        Début
            Cacher les attributs de l'élément sélectionnée précédemment par
            l'utilisateur;
            Afficher les attributs de l'élément sélectionnée par l'utilisateur (tr);
            Encadrer l'élément associée dans la vue document;
        Fin
    Sinon

```

```

| Affecter le chemin à l'attribut src du img ;
| img.onmousedown
| Début
|   | Si (l'arborescence n'est pas affichée)
|   |   | Afficher l'arborescence ;
|   |   | Sinon
|   |   |   | Cacher l'arborescence ;
|   |   |   | Fsi
|   |   | Cacher les attributs de l'element sélectionnée précédemment par
|   |   | l'utilisateur ;
|   |   | Afficher les attributs de l'element sélectionnée par l'utilisateur ;
|   |   | Encadrer l'element associée dans la vue document ;
|   |   | Fin
|   | Fsi
| Fin
Fsi
Fin

```

V.2.3. Implémentation du module «détermination du VIPS»

Les variables globales sont les tableaux attente qui contient les nœuds du DOM non traités par VIPS, le tableau pool qui contient les blocs visuels, et le tableau R qui contient l'état de la règle par rapport au nœud.

V.2.3.1. Nettoyage de données

Cette fonction a pour objet de supprimer (logiquement) les fils non valides, elle correspond à la règle une dans l'étape de segmentation.

VIPSNettoyage()

```

| Début
|   | Activer la détection des blocs visuels ;
|   | Pour (tout les elements du document)
|   |   | Faire
|   |   |   | Si (l'element n'est pas valide)
|   |   |   |   | Supprimer logiquement le nœud ;
|   |   |   |   | Fsi
|   |   |   | Finfaire
|   |   | Fin
|   | Fsi
| Fin

```

V.2.3.2. L'extraction des blocs visuels

Les fonctions Regle2 jusqu'à Regle12 sont les règles heuristiques de l'étape de segmentation, elles ont le paramètre obj qui représente le nœud sur lequel la règle est appliquée

Regle2(obj)

Début

Si (obj à un seul enfant valide et cet enfant n'est pas de type Text)

 | Insérer le fils dans le tableau attente ;

 | Supprimer obj du tableau attente ;

 | R[2]=vraie ;

Finsi

Sinon

 | R[2]= faux ;

Finsion

Fin

Regle3(obj)

Début

Si (obj est la racine de la page ou du sous page et il y a une seul racine)

 | Insérer tous les fils dans le tableau attente ;

 | Supprimer obj du tableau attente ;

 | R [3]=vraie ;

Finsi

Sinon

 | R[3]= faux ;

Finsion

Fin

Regle4(obj)

Début

Si (tous les fils enfants du obj sont des text node ou virtual text node)

 | Mettre obj dans le pool ;

 | Supprimer obj du tableau attente ;

 | R [4]=vraie ;

Finsi

Sinon

 | R[4]= faux ;

Finsion

Fin

Regle5(obj)

Début

Si (un des enfants de obj est un line break node)

 | Mettre tous les fils dans le tableau attente ;

 | Supprimer obj du tableau attente ;

 | R[5]=vraie ;

Finsi

Sinon

 | R[5]= faux ;

Finsion

Fin

Regle6(obj)

Début

- Si (un des enfants de obj à la balise <HR> comme fils)

- Mettre tous les fils dans le tableau attente ;

- Supprimer obj du tableau attente ;

- R[6]=vraie ;

- Finsi

- Sinon

- R[6]= faux ;

- Finsion

Fin

Regle7(obj)

Début

- Si (la couleur du fond du obj est différente de la couleur du fond d'un de ces enfants)

- Mettre fils dans le pool ;

- Supprimer obj du tableau attente ;

- R[7]=vraie ;

- Finsi

- Sinon

- R[7]= faux ;

- Finsion

Fin

Regle8(obj,seuil)

Début

- Si (obj a au moins un text node ou virtual text node est sa taille est inférieur au seuil)

- Mettre obj dans le pool ;

- Supprimer obj du tableau attente ;

- R[8]=vraie ;

- Finsi

- Sinon

- R[8]=faux ;

- Finsion

Fin

Regle9(obj, seuil){

Début

- Si (l'enfant de obj avec la taille maximale est inférieur au seuil)

- Mettre obj dans le pool ;

- Supprimer obj du tableau attente ;

- R[9]=vraie ;

- Sinon

- R[9]= faux ;

- Fsi

```

Fin
Règle10(obj)
Début
    Si (le frère précédent du obj n'est pas divisé)
        Mettre obj dans le pool ;
        Supprimer obj du tableau attente ;
        R[10]=vraie ;
    Finsi
    Sinon
        R[10]=faux ;
    Finsion
Fin

```

```

Regle11(obj)
Début
    Mettre tout les fils d'obj dans le tableau attente ;
    Supprimer obj du tableau attente ;
    R [11]=vraie ;
Fin

```

```

Regle12(obj){
Début
    Mettre obj dans le tableau pool ;
    Supprimer obj du tableau attente ;
    R [12]=vraie ;
Fin

```

L'intérêt de cette fonction est de savoir quel sont les règles applicable sur un nœud bien déterminé, elle a comme paramètre l'objet obj et la règle r.

```

AssocierRegleNoeud (obj, r)
Début
    Si (la règle r est applicable sur obj )
        Return vraie ;
    Finsi
    Sinon
        Return faux ;
    Finsion
Fin

```

C'est la fonction qui fait la detection des blocs visuels.

```

VIPSSegmentation()
Début
    taille =taille du ségment ;
    attente[0]=identifiant du segment ;
    Tantque(attente.length>0)
        Faire
            Appliquer les règle par leur priorité sur attente[0] ;
        Finfaire

```

Fin

V.2.3.3. La détection des séparateurs

C'est la fonction charger de la détection des séparateurs horizontaux ou verticaux, elle a comme paramètres le début d et la fin f de la page ou de sous-page

detectionSeparateur(d,f)

Début

Initialiser le tableau des séparateurs par les paramètres début et fin par d et f respectivement

Pour (tout les segment du pool)

Si (le segment est contenu dans le séparateur) ;
Divisez le séparateur ;

Finsi

Sinon

Si(le segment croise avec le séparateur)
Mettre à jour les paramètres du séparateur ;

Finsi

Sinon

Si(le segment couvre le séparateur)
Supprimer le séparateur ;

Fsi

Finsinon

Fisinon

Faire

Finfaire

Fin

C'est la fonction charger de la détection des blocs qui se trouvent au bord (superieur et inférieur) des séparateurs, elle a comme paramètre le tableau des séparateurs

detectionBordSeparateur(sep)

Début

Pour (tout les séparateurs détectés)

Faire

Détecter les blocs qui ont une bordure inférieure adjacente au séparateur ;
Détecter les blocs qui ont une bordure superieur adjacente au séparateur ;

Finfaire

Fin

C'est la fonction qui ce charge du calcule du poids de séparateur , elle a comme paramètre le tableau des séparateurs

AffecterPoidsSeparateur(sep)

Début

 Pour (tout les séparateurs détectés)

 Faire

 Affecter le poids suivant la différence entre les blocs visuels ;

 Finfaire

Fin

V.2.3.4. la construction de l'arbre sémantique

C'est la fonction charger de la construction de l'arbre sémantique du document HTML. Pour la création de la représentation cette partie utilise les mêmes fonctions que le module DOM.

ConstruireArborescence ()

Début

 Pour (tous les séparateurs)

 Faire

 Fusionner les segments qui se trouvent après et avant les séparateurs qui ont un faible poids ;

 Finfaire

Fin

V.3. Interface utilisateur

Notre outil comprend une interface graphique afin de permettre à tout utilisateur, quel que soit le niveau et sa pratique en informatique, extraire et consulter les structure hiérarchique (DOM, VIPS). Comme le montre la figure suivante l'outil VIPS est modélisé comme suit:

- Une interface pour la vue commande.
- Une interface pour la vue structurelle.
- Une interface pour la vue attribut.
- Une interface pour la vue document.

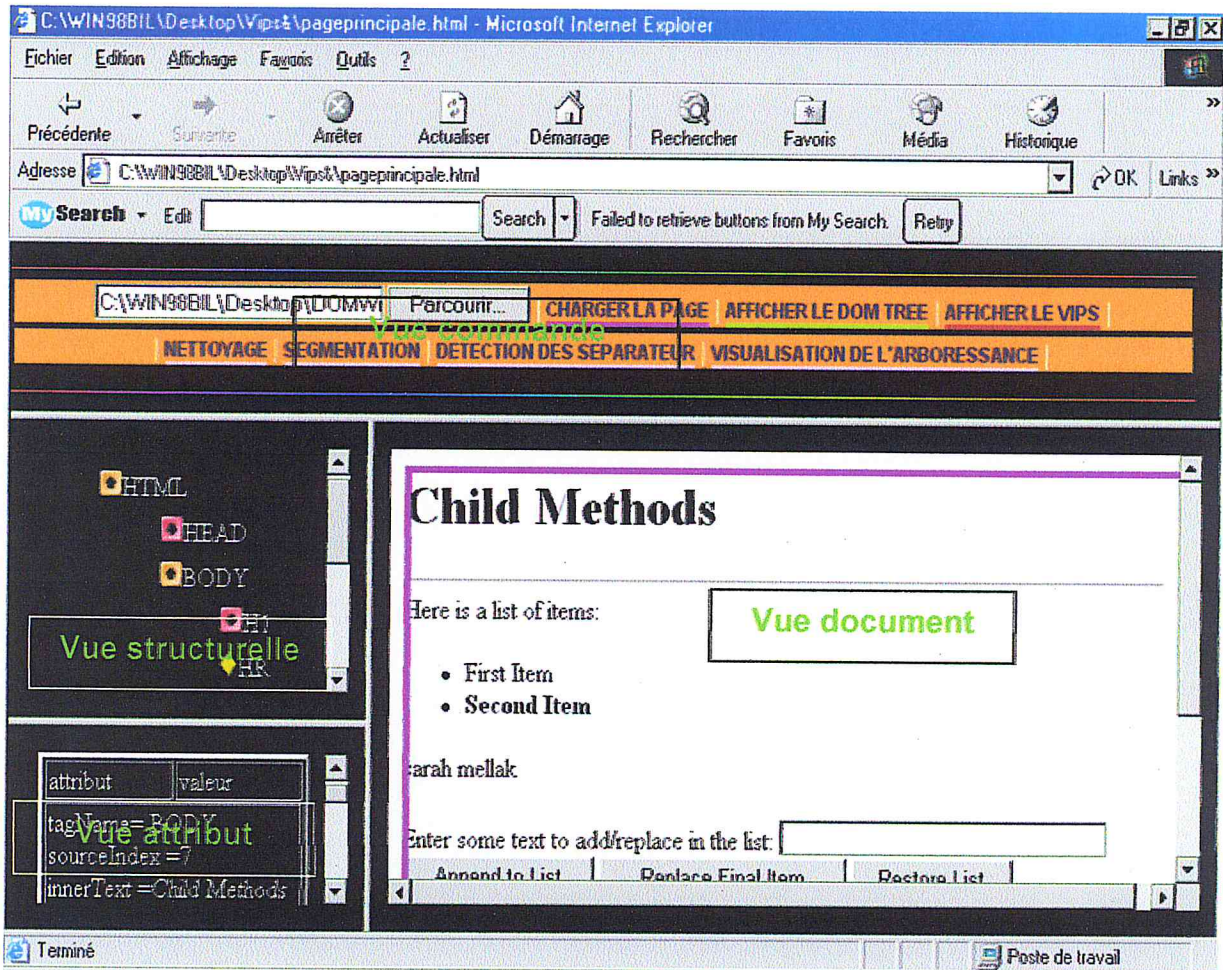


Figure 4.27 Vue globale de notre outil.

V.3.1 La vue commande

La vue commande est conçue pour que l'utilisateur manipule les diverses fonctionnalités du système. Elle permet les opérations de parcourir, de chargement de page, d'extraction et d'affichage du DOM et l'extraction et l'affichage du VIPS



Figure 4.28 Vue commande de notre outil.

V.3.2 La vue document

Cette vue permet de visualiser les changements faits au niveau de la vue structurale, par exemple quand l'utilisateur sélectionne un objet dans la vue structurale.

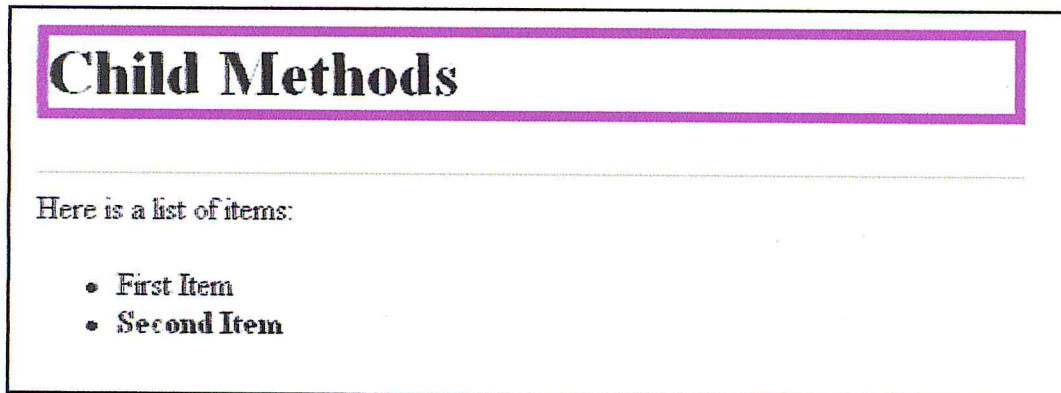


Figure 4.29 Vue document de notre outil.

V.3.3 La vue attribut

Cette vue a comme intérêt d'afficher les attributs visuels et leurs valeurs des objets (nœud ou segment) sélectionnés par l'utilisateur dans la vue structurale.

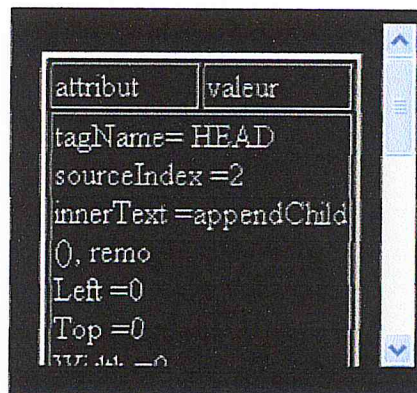


Figure 4.30 Vue attribut de notre outil.

V.3.4 La vue structurale

L'intérêt de cette vue est de donner à l'utilisateur une vue globale de la structure de son document, sous forme d'un arbre similaire à celui utilisé par les gestionnaires de fichier des systèmes d'exploitation, ou chaque objet (nœud ou segment) de la structure peut être déployé et il sera encadré dans la vue document.

Nous construisons cette vue en se basant sur l'organisation hiérarchique du document HTML. La racine de l'arbre dans le cas du DOM est l'élément "HTML", et dans le cas de VIPS est "VB1" qui représente le document en entier.

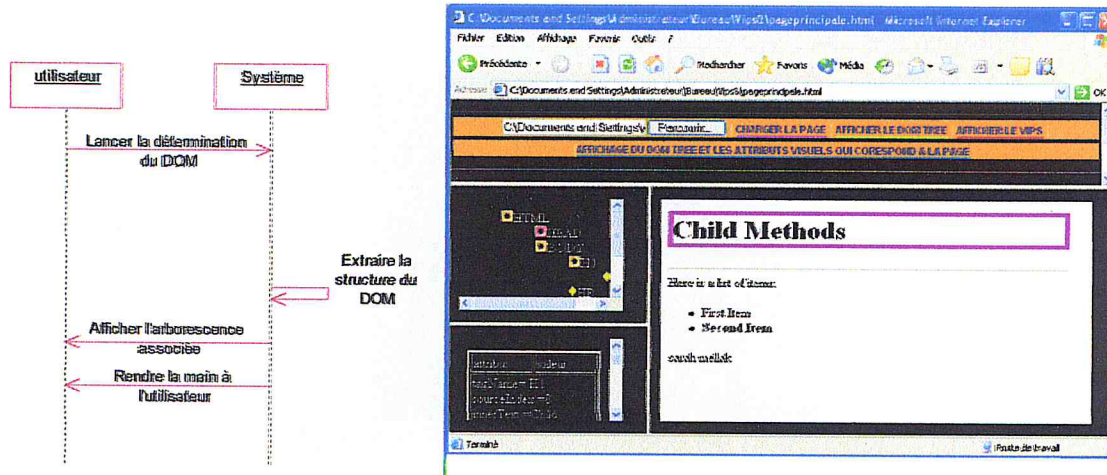


Figure 4.33 Visualisation des nœuds et de leurs attributs la partie DOM.

- Construction de l'arbre sémantique
L'utilisateur lance la construction de l'arbre sémantique pour qu'il puisse le visualiser.

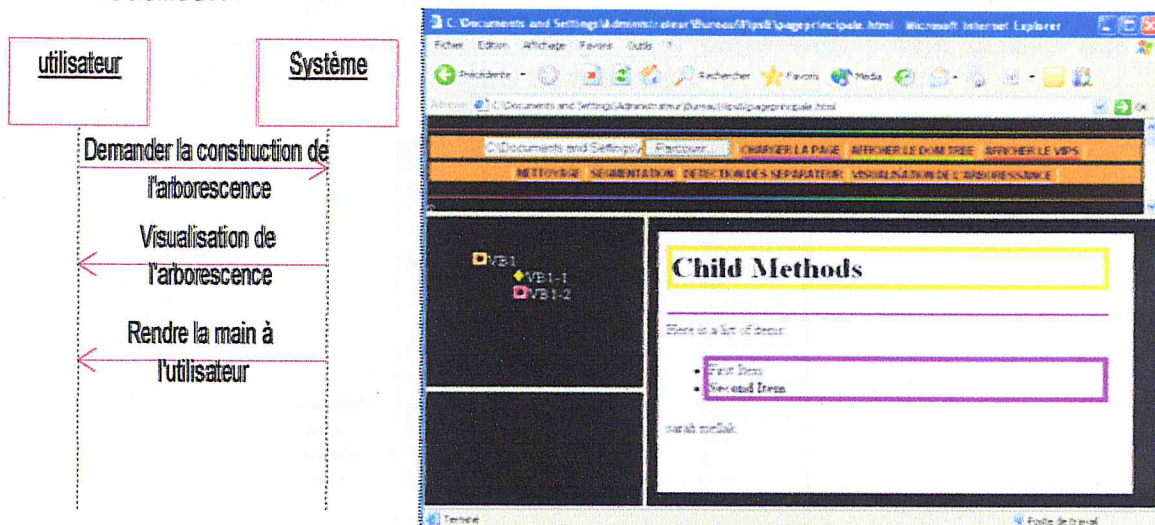


Figure 4.34 Construction de l'arbre sémantique.

VII. CONCLUSION

Nous avons présenté au cours de ce chapitre, la mise en œuvre d'un outil de segmentation et d'extraction du contenu sémantique d'une page web. Nous avons commencé par l'analyse des besoins qui nous a permis d'élaborer l'architecture de notre outil. Celui-ci est décomposé de trois modules : chargement de page, détermination du DOM, et détermination du VIPS.

Notre outil permet à l'utilisateur de visualiser les différentes hiérarchies (DOM, VIPS), accéder aux objets de celles-ci, et de voir les changements au niveau de la vue attribut et la vue document.

Conclusion générale

Au terme de notre étude, nous devons signaler que notre travail repose essentiellement sur la réalisation d'un outil pour la segmentation et l'extraction de la structure et le contenu sémantiques d'une page Web, en se basant sur les propriétés visuelles des nœuds. Le but de notre travail est l'enrichissement de l'application logicielle web avec des fonctions qui permettent

- ✓ Augmenter la qualité des résultats de recherche d'informations sur le net.
- ✓ Aider à faire une haute personnalisation suivant les besoins des internautes.
- ✓ Une bonne segmentation peut déterminer la relation entre le sujet de la page et les différentes informations exposées sur celle-ci.

La méthode adoptée VIPS (Vision-based Page Segmentation) présente plusieurs avantages par rapport aux autres méthodes en matière de temps de réponse et l'introduction de l'aspect visuel de l'information.

Nous avons pu à travers ce modeste travail, enrichir nos connaissances dans le domaine du webmining et de la segmentation des documents web pour implémenter l'algorithme VIPS. Quatre grandes étapes ont été réalisées: le nettoyage de données, l'extraction des blocs visuels, la détection des séparateurs, et la construction de l'arbre sémantique.

L'application développée en Javascript, présente l'avantage d'être conviviale et facilement intégrable dans un environnement web. Comme perspective de ce travail, nous recommandons de traiter les points suivants :

- ✓ Augmenter le temps de réponse car la nature du DOM ralentit le processus du parcours.
- ✓ L'intégration des notions comme le DOC (Degree of Cohérence) et le PDOC pour enrichir le processus de segmentation.
- ✓ Raffiner plus les règles heuristiques au niveau de l'extraction des blocs visuels pour qu'elles soient plus adaptées au document HTML.
- ✓ Ajouter des règles au niveau de la détermination du poids des séparateurs et l'intégration des techniques du Text mining.

Annexe A

UML - Unified Modeling Language -

I. PRESENTATION D'UML :

I.1. Historique [31]:

A partir de 1994, Rumbaugh et Booch (rejoins en 1995 par Jacobson) ont unis leurs efforts pour mettre au point la méthode unifiée (unified method 0.8), incorporant les avantages de chacune des méthodes précédentes.

La méthode unifiée à partir de la version 1.0 devient UML (Unified Modeling Language), une notation universelle pour la modélisation objet.

UML 1.0 est soumise à l'OMG (Object Management Group) en janvier 1997, mais elle ne sera acceptée qu'en novembre 1997 dans sa version 1.1, date à partir de laquelle UML devient un standard international.

Voici le récapitulatif des évolutions de ce langage de modélisation :

- En 1995: Méthode unifiée 0.8 (intégrant les méthodes Booch'93 et OMT)
- En 1995: UML 0.9 (intégrant la méthode OOSE)
- En 1996: UML 1.0 (proposée à l'OMG)
- En 1997: UML 1.1 (standardisée par l'OMG)
- En 1998: UML 1.2
- En 1999: UML 1.3
- En 2000: UML 1.4
- En 2003: UML 1.5

Cette méthode représente un moyen de spécifier, représenter et construire les composantes d'un système informatique.

I.2. Définition[31]:

UML représente l'état de l'art des langages de modélisation objet. Il fournit les fondements pour spécifier, construire, visualiser et décrire les artefacts d'un système logiciel. Pour cela, UML se base sur une sémantique précise et sur une notation graphique expressive.

UML permet de modéliser de manière claire et précise la structure d'un système indépendamment de tout langage de programmation.

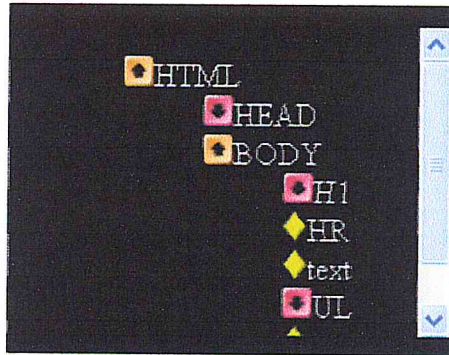


Figure 4.31 Vue structurelle de notre outil.

VI. TEST EST VALIDATION est et Validation

Dans cette partie nous allons tester quelques cas d'utilisation, qui sont les suivant

- chargement d'une page web
 Pour qu'un utilisateur charge une page web il faut qu'elle doit être sélectionnée en utilisant la fenêtre du parcours

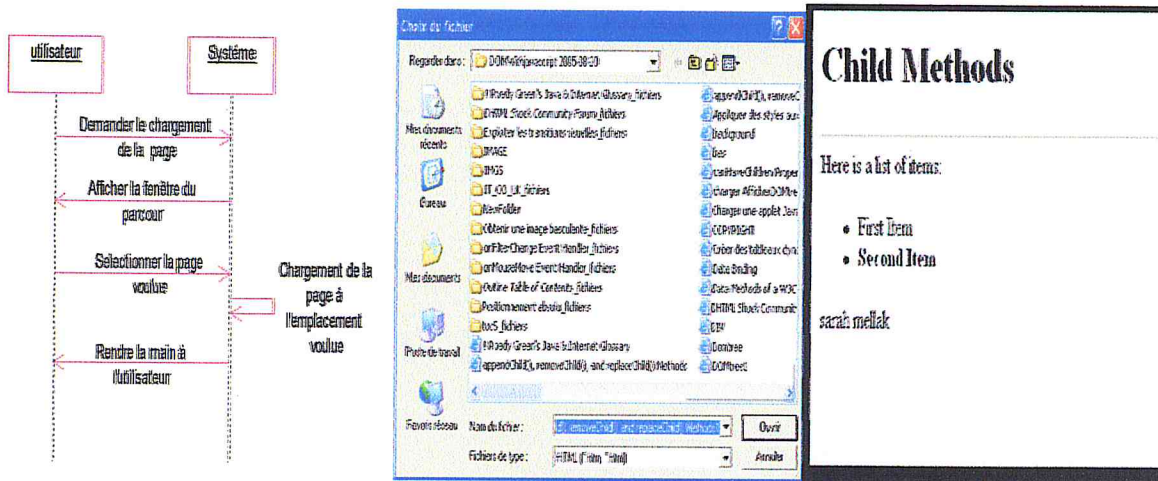


Figure 4.32 Chargement d'un nouveau document.

- La visualisation des nœuds et de leurs attributs la partie DOM.
 Pour la visualisation il faut que l'utilisateur sélectionne un nœud dans la vue structurelle.

I.3. Avantages et inconvénients d'UML [31]:

I.3.1. Les points forts d'UML:

- UML est un langage formel et normalisé :
 - Gain de précision.
 - Gage de stabilité.
 - Encourage l'utilisation d'outils.
- UML est un support de communication performant :
 - Il cadre l'analyse.
 - Il facilite la compréhension de représentations abstraites complexes.
 - Son caractère polyvalent et sa souplesse en font un langage universel.

I.3.2. Les points faibles d'UML :

- La mise en pratique d'UML nécessite un apprentissage et passe par une période d'adaptation : même si l'Espéranto est une utopie, la nécessité de s'accorder sur des modes d'expression communs est vitale en informatique. UML n'est pas à l'origine des concepts objets, mais en constitue une étape majeure, car il unifie les différentes approches et en donne une définition plus formelle.
- Le processus (non couvert par UML) est une autre clé de la réussite d'un projet : or, l'intégration d'UML dans un processus n'est pas triviale et améliorer un processus est une tâche complexe et longue. Les auteurs d'UML sont tout à fait conscients de l'importance du processus, mais l'acceptabilité industrielle de la modélisation objet passe d'abord par la disponibilité d'un langage d'analyse objet performant et standard.

II. LA MODELISATION [31]:

II.1. Qu'est-ce qu'un modèle?

La modélisation consiste à créer une représentation simplifiée d'un problème: le modèle. Grâce au modèle il est possible de représenter simplement un problème, un concept et le simuler. La modélisation comporte deux composantes:

- L'analyse, c'est-à-dire l'étude du problème.
- La conception, soit la mise au point d'une solution au problème.

Le modèle constitue ainsi une représentation possible du système pour un point de vue donné.

II.2. La modélisation UML :

Le métamodèle UML fournit une panoplie d'outils permettant de représenter l'ensemble des éléments du monde objet (classes, objets, ...) ainsi que les liens qui les relie.

Toutefois, étant donné qu'une seule représentation est trop subjective, UML fournit un moyen astucieux permettant de représenter diverses projections d'une même représentation grâce aux vues.

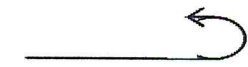
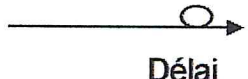
Une vue est constituée d'un ou plusieurs diagrammes. On distingue deux types de vues:

- Les vues statiques, c'est-à-dire représentant le système physiquement :
 - diagrammes d'objets ;
 - diagrammes de classes ;
 - diagrammes de cas d'utilisation ;
 - diagrammes de composants ;
 - diagrammes de déploiement.

- Les vues dynamiques, montrant le fonctionnement du système
 - diagrammes de séquence ;
 - diagrammes de collaboration ;
 - diagrammes d'états-transitions ;
 - diagrammes d'activités.

II.2.1. Diagrammes de cas d'utilisation :

Le diagramme de cas d'utilisation décrit la succession des opérations réalisées par un acteur (personne qui assure l'exécution d'une activité). C'est le diagramme principal du modèle UML, celui où s'assure la relation entre l'utilisateur et les objets que le système met en œuvre (voir figure C.1).

	<p>N'interrompt pas l'exécution de l'expéditeur. Le message envoyé peut être pris en compte par le récepteur à tout moment ou ignoré.</p>
	<p>Bloque l'expéditeur pendant un temps donné, en attendant la prise en compte du message par le récepteur.</p>

II.2.8. Diagramme d'états-transitions :

Le diagramme d'état (voir figure C.6) représente la façon dont évoluent ("cycle de vie") durant le processus les objets appartenant à une même classe. La modélisation du cycle de vie est essentielle pour représenter et mettre en forme la dynamique du système.

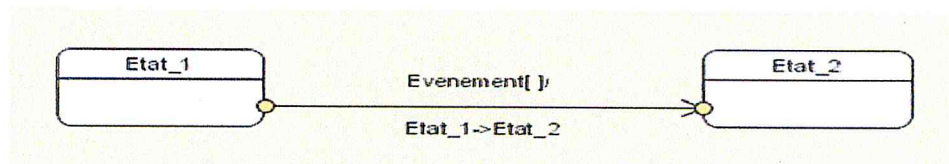


Figure C.6 : La notation des états, transition et événement.

II.2.9. Diagramme d'activités :

UML permet de représenter graphiquement (voir figure C.7) le comportement d'une méthode ou le déroulement d'un cas d'utilisation, à l'aide de diagrammes d'activités (une variante des diagrammes d'états-transitions).

Une activité représente une exécution d'un mécanisme, un déroulement d'étapes séquentielles. Le passage d'une activité vers une autre est matérialisé par une transition.

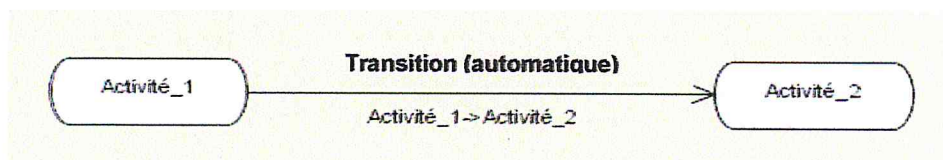


Figure C.7: La notation des activités et transition.

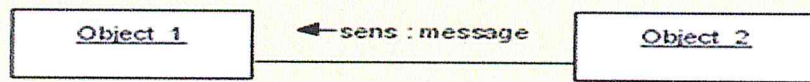


Figure C.4: Formalisme de base du diagramme de collaboration.

II.2.7. Diagramme de séquence :

Le diagramme de séquence représente la succession chronologique des opérations réalisées par un acteur : saisir une donnée, consulter une donnée, lancer un traitement ; il indique les objets que l'acteur va manipuler et les opérations qui font passer d'un objet à l'autre (figure C.5). On peut représenter les mêmes opérations par un diagramme de collaboration.

Diagramme de séquence et diagramme de collaboration sont deux vues différentes, mais logiquement équivalentes (on peut construire l'une à partir de l'autre), d'une même chronologie.

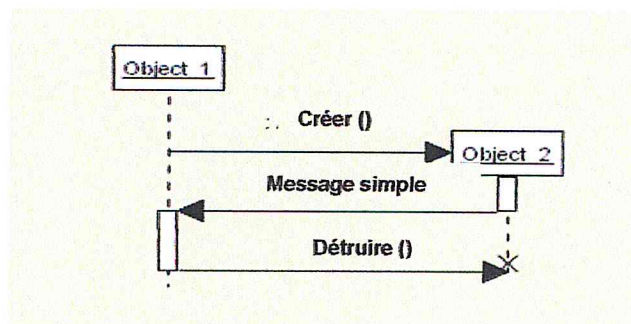


Figure C.5: Diagramme de séquence.

Le diagramme de séquence distingue 5 types de messages prédéfinis qui sont présentés dans le tableau suivant :

Tableau C.1 : Types de messages.

Type de message	Signification
→	Message simple.
→ \	Message asynchrone, pas de réponse attendue par l'émetteur.
→ X	Message synchrone, réponse nécessaire du destinataire.

II.2.3. Diagramme d'objets :

Ce type de diagramme UML montre des objets (instances de classes dans un état particulier) et des liens (relations sémantiques) entre ces objets. Ce diagramme sert essentiellement en phase exploratoire, car il possède un très haut niveau d'abstraction.

II.2.4. Diagramme de composants :

Les diagrammes de composants permettent de décrire l'architecture physique et statique d'une application en terme de modules (voir figure C.3) : fichiers sources, bibliothèques, exécutables, ... etc. Ils montrent la mise en oeuvre physique des modèles de la vue logique avec l'environnement de développement.

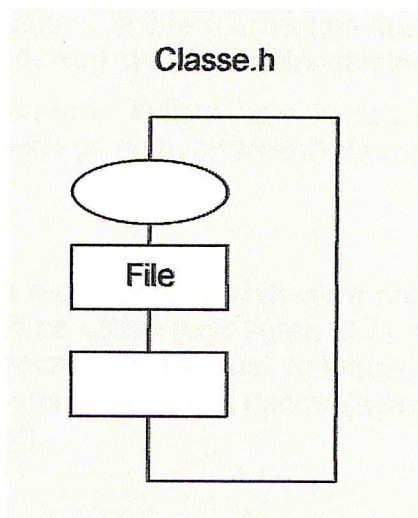


Figure C.3: Représentation d'un composant.

II.2.5. Diagramme de déploiement :

Les diagrammes de déploiement montrent la disposition physique des matériels qui composent le système et la répartition des composants sur ces matériels. Les ressources matérielles sont représentées sous forme de noeuds. Les noeuds sont connectés entre eux à l'aide d'un support de communication. La nature des lignes de communication et leurs caractéristiques peuvent être précisées.

II.2.6. Diagramme de collaboration :

Les diagrammes de collaboration montrent des interactions entre objets (figure C.4) (instances de classes et acteurs). Ils permettent de représenter le contexte d'une interaction, car on peut y préciser les états des objets qui interagissent.

Références bibliographiques

- [1] J. CLECH, S. HASSAS "Web Mining Système Multi-Agents" E.G.C. Lyon, Janvier 2003.
- [2] M. EIRINAKI, M. VAZIRGIANNIS "Web Mining for Web Personalization" Athens University of Economics and Business.
- [3] R.Kosala, H.Blockeel "Web Mining Research: A Survey" Department of Computer Science Katholieke University; Heverlee, Belgium.
- [4] J.Srivastava "Web Mining: Accomplishments & Future Directions" University of Minnesota, USA.
- [5] D.Goodman : "JavaScript Le guide de developpeur", Edition OEM,2002.
- [6] <http://www.w3c.org>.
- [7] Cours "DOM Primer Part 1", Oxford Brookes University 2002.
- [8] B Adelberg, "A tool for semi-automatically extracting structured and semi-structured data from text documents, In Proceedings of ACM SIGMOD" Conference on Management of Data, 1998, pp. 283-294.
- [9] N. Ashish, and C. A. Knoblock, "Semi-Automatic Wrapper Generation for Internet Information Sources", Proceedings of the Conference on Cooperative Information Systems, 1997, pp. 160-169.
- [10] Ashish, and C. A. Knoblock, "Wrapper Generation for Semi-structured Internet Sources", SIGMOD Record, Vol. 26, No. 4, 1997, pp. 8-15.
- [11] J.Hammer, H.Garcia-Molina, J.Cho, R.Aranha, and A.Crespo, "Extracting Semi-structured Information from the Web", Proceedings of the Workshop on Management for Semi-structured Data, 1997, pp. 18-25.
- [12] D. W. Embley, Y. Jiang, and Ng, Y.-K., "Record-boundary discovery in Web documents", Proceedings of the 1999 ACM SIGMOD international conference on Management of data, Philadelphia PA, 1999, pp. 467-478.
- [13] L. Page, S. Brin, Motwani, R. and T. Winograd, "The PageRank citation ranking: Bringing order to the web", Technical report, Stanford University, Stanford, CA, 1998.
- [14] J. Kleinberg, "Authoritative sources in a hyperlinked environment", Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms, Baltimore, MD, USA, 1998, pp. 668-677.
- [15] K. Bharat, and M. R.Henzinger, "Improved algorithms for topic distillation in a hyperlinked environment", Proceedings of the 21st ACM International

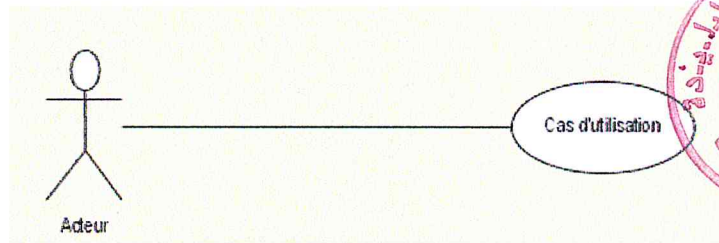


Figure C.1: cas d'utilisation.

Les cas d'utilisation peuvent être structurés et reliés par trois types principaux de relations :

- La relation d'utilisation : Une relation d'utilisation permet de décomposer un cas d'utilisation en sous-cas d'utilisation.
- La relation d'inclusion : le cas d'utilisation source comprend également le comportement de son cas d'utilisation destination.
- La relation d'extension : indique que le cas d'utilisation source étend (précise) les objectifs (le comportement) de cas d'utilisation destination.

II.2.2. Diagramme de classes :

Le diagramme de classes représente l'architecture conceptuelle du système : il décrit les classes que le système utilise (voir figure C.2), ainsi que leurs liens, que ceux-ci représentent un emboîtement conceptuel (héritage, marqué par une flèche terminée par un triangle) ou une relation organique (agrégation, marquée par une flèche terminée par un diamant).

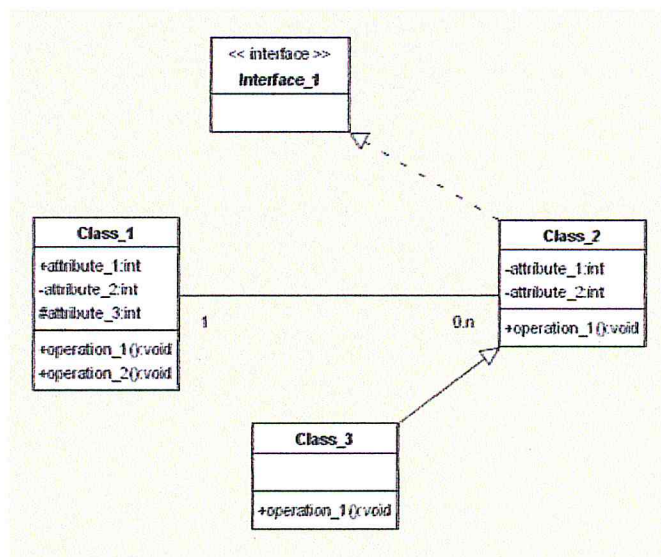


Figure C.2: Diagramme de classes.

Conference on Research and Development in Information Retrieval (SIGIR98), 1998, pp. 104-111.

[16] S. Chakrabarti, "**Integrating the Document Object Model with hyperlinks for enhanced topic distillation and information extraction**", the 10th International World Wide Web Conference, 2001.

[17] S. Chakrabarti, M. Joshi, and V. Tawde, "**Enhanced topic distillation using text, markup tags, and hyperlinks**", In Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval , ACM Press, 2001, pp. 208-216.

[18] E. Kaasinen, M. Aaltonen, J. Kolari, S. Melakoski, and T. Laakko, "**Two Approaches to Bringing Internet Services to WAP Devices**", Proceedings of 9th International World-Wide Web Conference, 2000, pp. 231-246.

[19] J. Chen, B. Zhou, J. Shi, H.-J. Zhang, and Wu, Q., "**Function-Based Object Model Towards Website Adaptation**", In Proceedings of the 10th International World Wide Web Conference, 2001.

[20] S. Chakrabarti, K. Punera, and M. Subramanyam, "**Accelerated focused crawling through online relevance feedback**", In Proceedings of the eleventh international conference on World Wide Web(WWW2002), 2002, pp. 148-159.

[21] M. L. Bernard, "**Criteria for optimal web design (designing for usability)**". 2002.

[22] S.-H. Lin, and J.-M. Ho, "**Discovering Informative Content Blocks from Web Documents**", In Proceedings of ACM SIGKDD'02, 2002.

[23] O. Buyukkokten, H. Garcia-Molina, and A. Paepche, "**Accordion Summarization for End-Game Browsing on PDAs and Cellular Phones**", In Proceedings of the Conference on Human Factors in Computing Systems, CHI'01, 2001.

[24] W. Wong, and A. W. Fu, "**Finding Structure and Characteristics of Web Documents for Classification**", In ACM SIGMOD Workshop on Research Issues in Data Mining and KnowledgeDiscovery (DMKD), Dallas, TX., USA, 2000.

[25] D. Buttler, L. Liu, and C. Pu, "**A Fully Automated Object Extraction System for the World Wide Web**", In International Conference on Distributed Computing Systems, 2001.

[26] Z. Bar-Yossef, and S. Rajagopalan, "**Template Detection via Data Mining and its Applications**", In Proceedings of the 11th International World Wide Web Conference (WWW2002), 2002.

[27] A. Rahman, H. Alam, and R. Hartono, "**Content Extraction from HTML Documents**", Proceedings of the First International Workshop on Web Document Analysis (WDA2001), 2001.

[28] M. Kovacevic, M. Diligenti, M. Gori, M. Maggini, V. Milutinovic "Recognition of Common Areas in a Web Page Using Visual Information: a possible application in a page classification" School of Civil Engineering, University of Belgrade, Serbia

[29] Y. Y.Tang, M.Cheriet, J. Liu, J. N. Said, and C. Y. Suen, "Document Analysis and Recognition by Computers, Handbook of Pattern Recognition and Computer Vision", edited by C.

[30] D.Cai, S.Yu, W.Ji-Rong, M.Wei-Ying," VIPS: A Vision based Page Segmentation Algorithm" *Microsoft Research Asia.

VIPS: A Vision based Page Segmentation Algorithm

[31] P.A. Muller, N. Gaertner "*Modélisation : objets avec UML* " Edition : Osman Eyrolles Multimédia - OEM (23 mars 2000).