Université Libre de Bruxelles
Faculté des Sciences Appliquées

Année académique 2002-2003

Mise en ligne d'informations relatives à l'analyse sémantique de la consultation de sites Web dont le contenu présente une évolution temporelle complexe et une mise en forme à la fois statique et dynamique.

Directeur :
Pr. E. Zimányi

Travail présenté par
Jean-Pierre Norguet
en vue de l'obtention du
Diplôme d'Etudes Approfondies

TABLE DES MATIÈRES

Table des figures

Remerciements

Un tout grand merci au promoteur de mon travail, Monsieur le Professeur Esteban Zimányi, pour sa patience, sa compréhension, son ouverture d'esprit, sa présence aux moments importants, ses critiques constructives et ses conseils judicieux.

Je remercie les membres de mon comité d'accompagnement Philippe Boutin et Thierry Massart ainsi que les membres du jury pour l'attention qu'ils porteront à ce travail.

Les journées de travail ont été rendues agréables par le bon esprit de mes collègues : Jean-Michel Dricot, Louis Jacomet, Inès Gam, Joël Cannau, Marie-Ange Remiche, Johnny Tsheke Shele, Elka Malinowski, Mohamed Minout, Pierre Stadnik, Didier Laurent, Jenny Scherrer, Henri McEuen, Natasha Van Der Heyden, sans oublier Olivier Samyn, que je remercie également pour son aide technique spontanée, aussi exceptionnelle que généreuse.

Je remercie les nombreux relecteurs de mes publications et autres documents de travail, ils se reconnaîtront.

Je remercie Gérard Materna et Jérémie Bury, dont les travaux de fin d'études se sont inscrits dans mes recherches et y ont été sources d'inspiration.

1. Introduction

1.1 Analyse de la communication et rétroaction

Dans toute communication entre deux émetteurs d'informations, il est important pour la qualité de la communication de connaître l'interaction entre les parties. Cela permet d'adapter la communication par rétroaction sur celle-ci (Figure 1). La plupart des théories de la communication soutiennent que cette rétroaction est bénéfique pour les deux parties [Mil73].

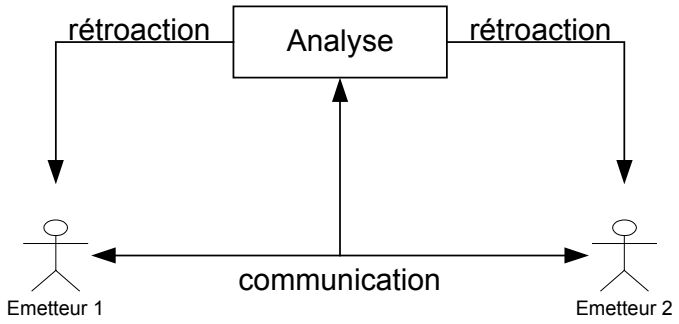


Figure 1 - Rétroaction sur la communication.

Lorsque la communication est unidirectionnelle, on parle de *diffusion* de l'information de l'émetteur vers le récepteur. Dans ce cas particulier, l'analyse de cette diffusion sera plutôt exploitée par l'émetteur afin d'adapter le message qu'il envoie aux attentes du récepteur (Figure 2).

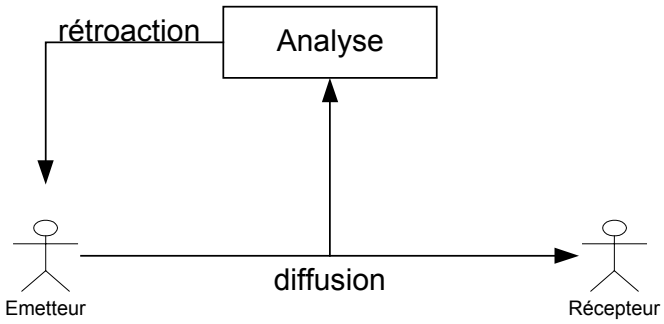


Figure 2 - Rétroaction sur la diffusion.

Un *média* est défini comme un canal de diffusion massive d'informations [Rob02]. Des exemples de média de notre société sont :

- la radio
- la télévision
- la presse
- la publicité
- le cinéma
- plus récemment, les sites Web d'Internet

Chaque média pratique d'une manière ou d'une autre l'analyse de sa diffusion pour rétroagir sur le contenu diffusé : modification, amélioration, extension, suppression, du fond et de la forme. La rétroaction est l'acte des rédacteurs et des techniciens, respectivement commis sur le contenu et la présentation de l'information diffusée.

1.2 Types de diffusion d'informations

La diffusion d'information peut être catégorisée en deux types : *push* et *pull*.
Prenons l'exemple d'un média très développé comme la télévision. L'information est diffusée selon un mécanisme de type *push* : l'ensemble des chaînes est transmis aux spectateurs, qui dans l'intimité de leur endroit de réception sélectionnent leurs programmes de vision, sans que la source de la diffusion ait connaissance de cette sélection (Figure 3).

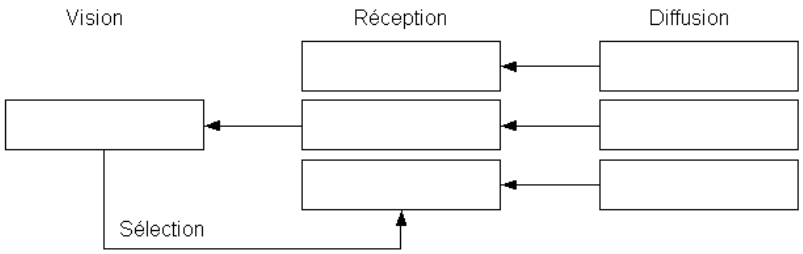


Figure 3 - Diffusion push.

Pour analyser ce que visionnent les téléspectateurs, il est nécessaire de leur demander des informations, par exemple sous forme de questionnaires et d'appareils auxiliaires, ou une combinaison des deux. Cette analyse est donc sujette à de nombreux facteurs d'incertitude et, puisqu'il est en pratique impossible de consulter la totalité des consommateurs de l'information diffusée, les résultats sont statistiquement extrapolés à partir d'échantillons. Il en résulte que l'analyse d'audience télévisuelle est à la fois *alambiquée* et *imprécise*.
La situation est similaire pour les autres médias traditionnels comme la presse et la radio.

Depuis une dizaine d'années, notre société voit l'apparition et le développement d'un nouveau média planétaire : Internet. Le *World Wide Web* en est devenu l'un des services les plus utilisés : des ordinateurs serveurs connectés au réseau mondial proposent des pages d'information consultables par n'importe quel client Web appelé navigateur ou *browser*. En comparaison avec la télévision, la diffusion d'informations sur le Web repose sur un mécanisme de type *pull*, dans lequel l'information de sélection est rapportée à la source de diffusion (Figure 4).

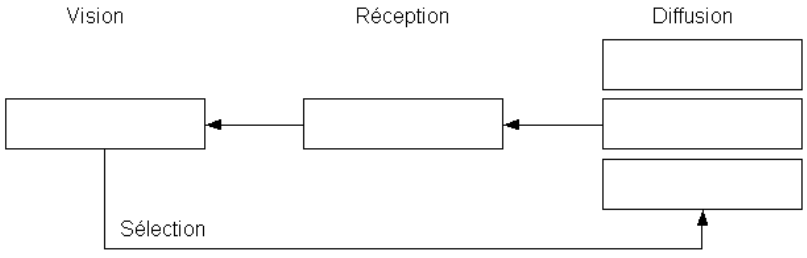


Figure 4 - Diffusion pull.

Lorsqu'un navigateur se connecte à un serveur Web, il envoie une requête à travers le réseau selon un protocole standardisé : HTTP (*Hyper Text Transfer Protocol*) [Ber96]. Le serveur traite cette requête et renvoie une page d'information. Ensuite, il garde une trace de la requête, des caractéristiques de la réponse (statut, taille) et des informations sur le poste client que le navigateur lui a fait parvenir. L'analyse de la consultation sur internet se révèle donc théoriquement *aisée* et *précise*.

De plus, le contenu diffusé sur Internet est digital, principalement au format texte ou hypertexte [Abi00]. Il est donc possible de le soumettre facilement à des traitements informatiques, au contraire du contenu vocal de la radio ou de la télévision.

Enfin, dans un souci d'exhausitivité, je citerai que l'Internet dispose également d'un mécanisme de diffusion push. Comme ce mécanisme est resté marginal, principalement pour des raisons de limitation de bande passante, je n'en tiendrai pas compte dans ce travail.

1.3 Historique de l'analyse d'audience Internet

En 1994, l'Internet était une technologie largement inconnue [Cof01]. Les premiers services de mise en ligne fournis aux consommateurs américains ont été CompuServe, Prodigy et America Online. A ce moment, la vitesse de connexion plafonnait à 9600 bauds. L'expérience des internautes était un mélange d'émerveillement et d'ennui. Voir s'afficher à l'écran une page renvoyée depuis l'autre bout de la planète était fantastique. Mais ce fantasme pâlissait lorsqu'ils se rendaient compte que chaque lien chargerait une autre page qui prendrait à chaque fois trois à cinq minutes de téléchargement. L'Internet n'était pas bien adapté à l'utilisation grand public car :

- la bande passante était insuffisante ;
- le contenu offert manquait d'organisation.

Par contre, les services commerciaux en ligne fonctionnaient correctement dans les limites de bande passante disponible, et leur contenu était bien organisé. Au milieu des années 90, les utilisateurs de ces services étaient logiquement plus nombreux que les utilisateurs d'Internet. Les revenus principaux de ces services

provenaient des abonnements des clients, qui payaient pour l'accès au contenu et pour la connection.
Dès lors, ces services ne demandaient pas un système de mesure d'audience externe : la mesure du nombre d'abonnements fournissait un excellent indicateur de performance. De simples sondages périodiques suffisaient à mesurer les parts de marché respectives.

En 1995, les acteurs du marché ont lancé leurs interfaces Web, telles que nous en connaissons aujourd'hui. Les ventes de PC et d'abonnements aux services commerciaux en ligne augmentèrent rapidement, principalement grâce à l'interface conviviale Windows proposée par America Online. Les mots "Internet", "World Wide Web" et "dot.com/point.com" sont sur toutes les lèvres.
La convergence de trois facteurs simultanés crée alors la demande d'un système de mesure d'audience de classe mondiale :

1. ce médium émergeant et en passe de croissance significative commence à intéresser les publicitaires ;
2. les avancées techniques permettent à de plus en plus de gens de se connecter à Internet, provoquant ainsi une croissance significative de l'audience ;
3. les investissements commençent à affluer dans les compagnies Internet, en particulier dans la Silicon Valley, ce qui contribue à la croissance du medium et à l'intérêt des autres médias pour celui-ci.

Aujourd'hui, la mesure d'audience Internet est utilisée à trois fins :

1. l'"auto-promotion" : il est important pour les organisations d'être en mesure de revendiquer sur base de sources objectives et impartiales la taille et la croissance de leur public ;
2. la prévision et la planification des ventes en fonction de la mesure de la consultation du site Web de l'organisation, qui est considéré comme une publicité génératrice de ventes ;
3. la "planification stratégique" : en connaissant les comportements des clients, leur interaction avec un site ou un groupe de sites, les responsables de site peuvent prendre des décisions qui augmentent considérablement la fluidité du traffic et l'*objectif du site* [Cof01] par rétroaction ;
4. le dimensionnement technique des machines serveurs.

A l'heure actuelle, plusieurs techniques existent :

1. la mesure électronique des ordinateurs d'un échantillon d'utilisateurs,

2. le sondage classique d'un échantillon d'utilisateurs,
3. l'analyse des traces de consultation stockées dans les fichiers de log des serveurs Web,
4. l'analyse des traces de consultation stockées par des serveurs publicitaires externes,
5. l'analyse des traces de consultation stockées par des serveurs de statistiques externes et indépendants.

Les deux premières techniques renseignent principalement sur les utilisateurs et leur comportement [Fas00]. Elles sont complémentaires aux trois dernières. Celles-ci renseignent plus précisément sur la consultation du site et seront détaillées dans le chapitre 2. Malgré cet avantage de précision, ainsi qu'une automatisation poussée, ces techniques ne donnent pas beaucoup de renseignements intéressants sur le *contenu* des pages consultées. L'objectif de ce travail est de pallier ce manque.

1.4 Présentation de l'exemple : le site Web de l'ULB

Tout au long de ce document, j'utiliserai comme exemple le site Web de l'Université Libre de Bruxelles [ULB03].
Les motivations du choix de ce site sont les suivantes :

- il existe réellement ;
- il est suffisamment complexe ;
- il est très représentatif des problèmes que l'on peut rencontrer en analyse d'audience.

Mes motivations personnelles dans ce choix sont les suivantes :

- il appartient à l'université où j'ai poursuivi mes études et obtenu mon diplôme d'ingénieur, son contenu m'est donc familier ;
- il appartient à l'université où je poursuis mes recherches, il est donc connu de mes collaborateurs ;
- le service informatique où mes recherches sont promues m'offre un accès privilégié aux responsables du site ;
- les propriétaires et rédacteurs du site ne sont pas contraints à une rentabilité commerciale et sont peu demandeurs d'analyse d'audience, ce qui libère mes recherches d'une contrainte.

Les caractéristiques du site ULB sont les suivantes :

- il met en ligne beaucoup de contenu : environ 100.000 pages statiques[1] ;
- ce contenu est très évolutif ;
- il diffuse des pages dynamiques de plusieurs technologies ;
- sa gestion requiert des compétences variées ;
- il est très accédé : environ 200.000 pages consultées par jour et 2.000.000 hits par jour[2] ;
- les fichiers mis en ligne se présentent sous plusieurs formats : texte, hypertexte et binaire ;
- le contenu est sémantiquement riche : beaucoup de sujets sont traités, les rédacteurs sont nombreux et variés.

Le public cible est varié et comprend entre autres :

- les étudiants internes,
- les étudiants externes,
- les chercheurs internes,
- les chercheurs externes,
- les enseignants,
- les membres du personnel,
- les entreprises,
- les journalistes,
- etc.

J'utiliserai fréquemment pour exemple un couple de deux sujets majeurs présents sur le site :

1. l'enseignement dispensé à l'ULB ;
2. les recherches scientifiques menées à l'ULB.

[1] J'ai obtenu ce chiffre en exécutant sur le serveur la commande find . | grep htm |wc -l

[2] La définition de ces grandeurs se trouve en section 3.1.

Figure 5 – Sujets.

L'objectif de la mise en ligne de ces sujets sur le site de l'ULB est de promouvoir auprès du public cible et via ce canal de diffusion les activités d'enseignement et de recherche de l'ULB ainsi qu'une certaine quantité d'informations associées que les rédacteurs du site jugent utiles. Cette information mise en ligne peut être consultée par le public cible ou par d'autres internautes, qui naviguent de page en page au gré de leur intérêt. La comparaison de l'audience obtenue par ces deux sujets permettra de prendre un certain nombre de décisions :

- dans la rédaction du site : par exemple mettre à jour le programme des cours au moment de l'année où il est le plus consulté ;
- dans la stratégie propre, indépendemment du site Web : par exemple favoriser dans des expositions scientifiques la promotion de la recherche scientifique à l'ULB.

1.5 Contenu de ce travail

Le chapitre 2 exposera les techniques existantes de mesure et d'analyse d'audience Internet. J'y passerai en revue les logiciels existants et leurs fonctionalités. Ensuite, je dresserai la liste des problèmes restants en msesure et analyse d'audience ; je mettrai en exergue la demande par les propriétaires de sites Web d'une analyse sémantique de l'audience et tenterai d'expliquer pourquoi cette fonctionalité majeure n'a jamais été implémentée jusqu'à présent. J'exposerai les diverses stratégies alternatives mises en place par les utilisateurs finaux pour satisfaire tant bien que mal ce besoin. Je décrirai également les solutions qui existent actuellement pour résoudre les autres problèmes. Je conclurai le chapitre en faisant le point sur les problèmes en suspens.

Le chapitre 3 tentera de répondre d'une manière nouvelle, plus générique et plus intuitive au besoin d'une analyse sémantique de l'audience Internet. Je définirai de nouvelles grandeurs audimétriques similaires aux grandeurs audimétriques

standard. L'exploitation de ces nouvelles grandeurs est plus intuitive et apporte une valeur ajoutée plus grande. J'expliquerai le défi technique que pose le calcul de ces nouvelles grandeurs et j'évoquerai plusieurs solutions ; je détaillerai celle qui à mon sens est la plus adaptée, élément par élément ; j'évaluerai enfin plusieurs améliorations potentielles.

Le chapitre 4 détaille la réalisation technique des éléments de la solution retenue au chapitre 3. J'y présente les technologies utilisées ainsi que le framework que j'ai conçu pour soutenir et coordonner l'implémentation de *WASA*, le système que j'ai développé pour stocker, traiter et mettre en ligne les informations décrites dans le chapitre 3.

Le chapitre 5 fait le point sur l'état d'avancement de ce travail et le positionne par rapport aux grands domaines de recherche que sont le data mining, l'intelligence artificielle et la recherche documentaire. J'y souligne les apports et les limitations des nouvelles grandeurs audimétriques. Partant de là, je propose d'étendre ces grandeurs pour prendre en compte plus de sémantique. Pour prendre en charge la complexité supplémentaire qui résulte de cette extension, je proposerai des pistes de solutions empruntées aux grands domaines de recherche préalablement cités. Enfin, je présenterai la piste principale que j'aurai sélectionnée pour faire l'objet de mes recherches futures : l'échantillonnage statistique, le modèle de distance sémantique dans les ontologies, l'indexation sémantique latente et les clusters électriques.

2 L'analyse d'audience

2.1 Les fichiers de logs

A chaque requête d'un navigateur au serveur Web, celui-ci stocke une ligne de log dans chaque fichier spécifié par son fichier de configuration `httpd.conf`. Une requête peut être adressée par exemple pour une page HTML, mais aussi pour chaque image mentionnée dans celle-ci.
Le serveur Web stocke le plus fréquemment cet historique des transactions dans 4 fichiers de logs distincts et complémentaires :
1. l'*access log* : adresse du client, nom d'utilisateur éventuel, date et heure de la requête, fuseau horaire, requête HTTP (méthode GET/POST/DELETE/PUT et URL), code d'erreur, nombre d'octets transférés au navigateur, 1 ligne par requête ;
2. le *referer log* : fichier consulté et adresse à partir de laquelle cette requête a été envoyée ;
3. l'*agent log* : navigateur et système d'exploitation utilisés sur le poste client ;
4. l'*error log* : les messages d'erreur éventuellement générés par le serveur.
Cette façon de répartir l'information dans les fichiers de logs s'appelle le Common Log Format. Il existe bien d'autres formats. Par exemple, le Combined Log Format est très répandu et regroupe l'information dans un seul fichier, ce qui permet une meilleure corrélation de l'information au détriment d'une rigidité de configuration supérieure et d'un espace de stockage plus important.

Format d'une ligne d'access_log :
`A.B.C D - [E:F:G:H I] "J K L" M N`
A = nom de la machine cliente
B = domaine de la machine cliente
C = pays de la machine cliente
D = nom d'utilisateur éventuel
E = date de la requête
F = heure de la requête

G = minutes de l'heure de la requête
H = secondes de l'heure de la requête
I = fuseau horaire du client
J = type de requête
K = fichier objet de la requête
L = protocole de la requête
M = code de réponse
N = nombre d'octets utiles transférés au client
Exemple :

```
crg-018.netra.canon.fr  -  -  [18/Aug/1998:19:39:00  +0200]
"GET /icons/jhe061.gif HTTP/1.0" 200 17175
```

Format d'une ligne de referer_log :

```
O -> P
```

Où :
O = adresse URL d'où la requête a été formulée
P = URI objet de la requête
Exemple :

```
http://search.yahoo.com/computer/society/internet/select.ht
ml -> /index.html
```

Une ligne d'agent_log peut quant à elle ressembler à ceci :

```
Mozilla/4.05 [en] (Win95; I)
```

d'où il est possible de déduire le nom du navigateur, son numéro de version et le système d'exploitation.

2.2 Besoins

Cette section, dans un langage plus spécifique au monde de l'entreprise, tente d'établir un parallèle entre les besoins commerciaux et la démarche scientifique précédemment introduite.

2.2.1 Organisations internationales

Durant les quelques dernières années, la gestion des sites Web d'organisations internationales s'est fortement concentrée sur la consolidation de leur infrastructure et de leur mode d'opération, ce qui leur a permis d'établir une plateforme commerciale fiable, extensible et au niveau de la taille de

l'organisation. Plus récemment, la maturation du commerce électronique, l'explosion du contenu Web et l'émergence de stratégie de gestion de la relation clientèle multi-canaux ont démarré une remise en question de ce qu'un site Web d'entreprise peut permettre en terme d'environnement de travail et de canal de communication. Le site Web évolue actuellement d'une plateforme technologique vers un canal de communication complet visant à la satisfaction de l'utilisateur. Une stratégie gagnante de communication Web doit établir les objectifs en terme d'expérience utilisateur : intégrité et accessibilité du contenu, facilité d'utilisation, etc. L'analyse de la performance qualitative de ce canal de communication requiert des capacités, des outils, des processus, des rôles et des mesures différentes de ceux utilisés pour la mise au point de l'infrastructure et du mode d'opération.

Les entreprises de pointes essayent déjà de combiner les outils existants pour remplir ce vide. Les analystes commerciaux utilisent les rapports des systèmes de gestion de contenu, de recherche, de catégorisation, de personalisation, et d'analyse d'audience et les traitent manuellement comme un ensemble.

Comme les sites Web font partie maintenant d'une stratégie de commerce électronique, le problème se pose de trouver le juste milieu entre un site Web fonctionnel et un site Web visuellement attractif. La production du site Web idéal, attractif, performant et continuellement mis à jour avec du contenu dynamique, implique de gérer un gros volume de pages Web hébergées sur de grandes grappes de serveurs Web, tout en faisant interagir un ensemble de développeurs, de groupes opérationnels et un grand nombre d'auteurs commerciaux, d'éditeurs et d'autres qui contribuent à la production de contenu.

[Met02] prévoit qu'à la fin 2003, 95% des entreprises internationales auront déployé un système de gestion de contenu et des coûts associés. Ceci requiert également des logiciels additionnels pour analyser la qualité, l'accessibilité et l'interêt de ce contenu.

2.2.2 Petites et moyennes entreprises

Les entreprises ou organisations de plus petite envergure n'ont pas toujours les moyens d'acheter des outils et de se payer des services d'analyse coûteux. Néanmoins, les questions qu'ils se posent restent les mêmes :

- Comment améliorer l'expérience de l'utilisateur en rendant le contenu plus attractif et plus accessible ?
- Comment calculer la performance et le retour sur investissement de la fourniture de contenu ?
- Qu'est ce qui intéresse les utilisateurs ?
- Quel est la corrélation entre le profil de l'utilisateur et ce qui l'intéresse ?
- A quoi bon faire évoluer un contenu qui n'est presque pas lu ?
- A quoi bon proposer un contenu qui ne répond pas à l'intérêt des visiteurs ?
- Quelle évolution doit suivre la ligne rédactionnelle d'un site Web pour augmenter la satisfaction des visiteurs ?

2.2.3 Sites Web personnels

Le problème se pose moins au niveau des sites personnels, qui ne contiennent que relativement peu de contenu, généralement subjectif et bien maîtrisé par son propriétaire.
De nombreuses solutions existent déjà pour y répondre.

2.3 L'environnement économique

Sur le marché, plusieurs secteurs s'unissent pour répondre à ces besoins : gestion de contenu, recherche/catégorisation, personalisation et analyse d'audience Web. Plusieurs tractations commerciales témoignent de cette tendance :

Fev-2003	Percussion Software (content management) et BuyStream (Web analytics) s'associent.
Jan-2003	Overture (pay-for-placement search) achète Keylime (Web analytics).
Nov-2002	Intelliseek (enterprise intelligence) et Inxight (recherche/classification) s'associent.
Nov-2002	iUpload (content management) achète WebSiteStory (Web analytics).
Oct-2002	Inxight (recherche/classification) achète WhizBang! Labs (information extraction from unstructured data).
Oct-2002	Keynote (Web site performance) achète Enviz (Web site effectiveness).
Jan-2002	Responsys (campaign management) achète NetAcumen (Web analytics).
Dec-2001	Ligth Speed (content management and delivery) et ensuite TropicalNet (content categorization) achètent I/Pro (Web analytics) et Teralytics (Web analytics).

Selon Aberdeen Group [Abe03], le marché de l'analyse d'audience Web s'est accru de 200% pour atteindre $400 millions de vente en 2000. Il a ensuite dimuné de 7% en 2001 et stagne en 2002. Après deux ans de ventes stagnantes, beaucoup de companies sont en train de réfléchir à comment redémarrer les ventes ou même à sortir du secteur.
Le secteur étant désespéré, les companies d'analyse d'audience tractent avec les concurrents directs et avec les companies de gestion de contenu, recherche/catégorisation, personnalisation et d'autres vendeurs e-business.
Au final, le nombre de vendeurs dans le domaine de l'analyse d'audience va diminuer. Certains vont faire faillite, certains vont être rachetés par d'autres, mais la plupart vont suivre les exemples repris dans le tableaux ci-dessus et vont s'associer avec des companies qui organisent et distribuent du contenu sur le Web. Actuellement, le marché est à nouveau en expansion et est attendu à grandir de $160 millions cette année, pour arriver à $463 millions en 2005.

2.4 Les produits existants

Il en existe une bonne centaine pour tous les prix. Tous ces produits apparaissent soit sous forme de logiciel, soit sous forme de fourniture d'un service applicatif. Ils se basent tous sur l'analyse des données de la consultation récupérée soit au niveau du serveur (logs ou sniffer réseau) soit au niveau du navigateur (marqueurs de pages).
L'évolution de ces produit s'est surtout faite au niveau de l'analyse, notamment en intégrant des techniques de data mining qui permettent de dévoiler des tendances cachées sous le volume des données. Par contre, il y a eu très peu d'évolution de la qualité des données récupérées.

Selon un rapport d'Aberdeen Group [Abe03] sur une étude de marché d'août 2002, 10 sur 12 des utilisateurs questionnés étaient contents de la solution dont ils disposaient. Mais cette satisfaction était difficile à obtenir, et un grand nombre ont essayé les produits de deux ou trois vendeurs différents. En cause, une très grosse différence entre les promesses et les résultats réels pour un grand nombre de vendeurs, qui combattent âprement pour survivre.
Les 5 critères les plus importants pour les utilisateurs sont les suivants :

1. précision et exactitude des résultats,
2. actualité et rapidité des résultats,

3. extensibilité,
4. coût total de déploiement,
5. reporting personnalisable et segmentation manuelle.

Les critères les mieux remplis par les produits sont les suivants :

1. reporting personnalisable et segmentation manuelle,
2. précision et exactitude des résultats,
3. extensibilité,
4. coût total de déploiement.

Par contre, l'actualité et la rapidité des résultats ne sont pas bien fournies par la plupart des produits. Cela suggère que les données récupérées au niveau du navigateur vont continuer à être populaires, parce qu'elles permettent des rapports rapides, sans devoir traiter ces quantités conséquentes d'informations que sont les fichiers de logs.

Concrètement, les résultats générés par les logiciels existants sont principalement :

- le nombre de visiteurs,
- le nombre de pages consultées en moyenne par visite,
- le nombre de consultations par page,
- le flux des visiteurs au sein des pages du site.

Des informations additionnelles peuvent être obtenues sur les postes clients qui se sont connectés :

- répartition géographique, dans une certaine mesure,
- navigateurs Web et systèmes d'exploitation utilisés pour afficher les pages.

2.5 Problèmes

2.5.1 Pertinence

Typiquement, ces logiciels génèrent un résumé laconique et trivial comme : *"1000 visiteurs par mois consultent en moyenne 5 pages par visite ; la page principale est la page la plus consultée"*. A côté de cela, ces logiciels génèrent une gigantesque quantité de résultats très détaillés. Ces résultats :

- sont peu intuitifs ;
- sont si nombreux qu'il faut passer un temps considérable à les dépouiller ;

- ont peu d'utilité, en particulier pour le processus de rétroaction illustré en Figure 10.

L'évolution de ces logiciels pendant ces dernières années a été de fournir une aide visuelle à travers une présentation améliorée sous formes de graphiques et tableaux et une interface de consultation intuitive et interactive. Cela simplifie l'étude des résultats, rend leur interprétation plus intuitive mais ne modifie en rien leur valeur intrinsèque.

2.5.2 Besoins insatisfaits

Le nombre de résultats qu'il est possible de générer à partir de l'information brute est presque infini. Dès lors, malgré la quantité de résultats générés par défaut, les desideratas des propriétaires de sites Web sortent souvent du cadre des possibilités qu'offrent ces logiciels [Tré98].
Une partie de ces desideratas est impossible à réaliser techniquement [Tré98]. Beaucoup de questions que se posent les propriétaires de sites Web sont inspirés de l'analyse d'audience classique et du marketing. Ces questions sont donc fortement "orientées utilisateurs" : quels sont leurs besoins ? Quels sont leurs profils sociaux ? Autant de questions aux réponses que les fichiers de logs ne permettent pas de trouver directement, et qu'il est difficile d'obtenir par des traitements informatiques simples.

2.5.3 La loi de Moore

Comme le rappelle avec raison l'article [Tuo02], la loi de Moore a été largement interprétée depuis sa formulation originale [Moo65] pour finalement refléter de manière générale l'évolution exponentielle des technologies de traitement de l'information.

La durée de traitement des informations de consultation, dont dépend la limite inférieure de la période de rafraîchissement des résultats, s'exprime par le rapport de la quantité de logs à traiter à la puissance de calcul de l'ordinateur responsable du traitement de l'information. Or, [Nor98] constate empiriquement que ces deux grandeurs sont approximativement sujettes à la même loi de Moore :

$$Durée\,de\,traitement\,(t) = \frac{Quantité\,de\,données}{Puissance\,de\,calcul} = \frac{C_1.M^t}{C_2.M^t} = C_3$$
$$< Rafraîchissement\,des\,résultats$$

Si l'on suppose que M est une constante, C_3 est une constante indépendante du temps. Ce résultat explique en partie pourquoi l'analyse de la consultation des sites Web repose depuis longtemps sur les mêmes principes. En effet, le développement de l'internet a augmenté considérablement le nombre des accès aux serveurs Web, donc la quantité d'informations à manipuler. Les possibilités de traitement ont donc été reléguées à l'arrière-plan par un compromis difficile entre précision des résultats et puissance de calcul.

2.5.4 La taille des fichiers de log

La navigation des visiteurs laisse des logs (traces) sur le serveur qui héberge le site. A partir de ces logs, des logiciels fournissent au propriétaire du site des informations quantitives sur l'audience obtenue par celui-ci. Cela représente beaucoup d'informations, temporelles, à manipuler, à croiser, à présenter.
Afin de déterminer un ordre de grandeur, supposons qu'une page HTML contient en moyenne 9 images ; un visiteur consultant une telle page aura provoqué après chargement complet de celle-ci 10 lignes de logs de chaque type sur le serveur. Si l'on considère qu'une ligne d'access log compte au moins une centaine de caractères et qu'il en va de même pour le referer log et l'agent log réunis, l'accès d'un visiteur à une page chargera en moyenne le disque du serveur de 2 Ko. Prenons le cas d'un site obtenant une audience moyenne de 50000 consultations par jour ; la taille des logs quotidiens de ce site atteindra 100 Mo par jour, soit plus de 30 Go par an. La quantité d'information est donc telle qu'il faut faire le tri parmi celle-ci et ne stocker que la partie la plus utile, réalisant le meilleur compromis possible entre précision de l'information, espace de stockage et durée du traitement.

2.5.5 Caches

La plupart des navigateurs Web disposent d'un système de *cache* qui consiste à stocker en mémoire vive ou sur le disque dur local de l'ordinateur les fichiers précédemment téléchargés. Plusieurs configurations existent et s'articulent

autour d'un paramètre principal : la durée d'utilisation sans confrontation avec le serveur (jamais, toujours, une fois par jour, etc.).
Lorsque le cache est activé sans confrontation avec le serveur, le navigateur utilise la version locale des fichiers sans consulter le serveur pour savoir si ces fichiers ont été modifiés. Dès lors, aucune trace de la consultation de tels fichiers n'est stockée sur le serveur, ce qui perturbe la mesure d'audience.
Lorsque le cache est activé avec confrontation avec le serveur, le navigateur envoie une requête HTTP au serveur en spécifiant la date de dernier téléchargement éventuel du fichier demandé. Dans un tel cas, le serveur Web compare la date de dernier téléchargement envoyée par le client à la date de dernière modification du fichier demandé. Si celle-ci est antérieure, il n'est pas nécessaire de renvoyer le fichier à travers le réseau. Le serveur renvoie donc uniquement un code HTTP `304` indiquant au navigateur qu'il peut afficher à l'utilisateur le fichier de cache local.

Les navigateurs ne sont pas les seuls à implémenter un tel système de cache. Les proxycaches[3] des organisations et des ISPs sont fréquents et poursuivent le même but : réduire autant que possible le trafic réseau.

Les pages HTML peuvent inclure des balises `PRAGMA` indiquant aux différents caches de ne pas fonctionner, ou de conserver les pages pendant une durée limitée.

Le caching avec confrontation avec le serveur a peu d'influence sur les logiciels de mesure d'audience. Les analyseurs de fichiers logs considèrent les codes 304 comme les codes 200, c'est-à-dire comme une consultation réussie. La seule différence réside dans le format de traçage : pour un code 304, le nombre d'octets transférés est remplacé par un tiret.

2.5.6 Multiplexage des adresses IP par les proxycaches

Comme l'illustre la **Error! Reference source not found.**, les proxycaches d'une organisation se positionnent dans une requête HTTP comme intermédiaires entre les navigateurs Web à l'intérieur de l'organisation et les serveurs Web de l'extérieur.

[3] Il ne faut pas confondre les firewall proxies SOCKS et les proxycaches. Ce sont deux fonctions différentes, qui peuvent par ailleurs être implémentées par un même logiciel.

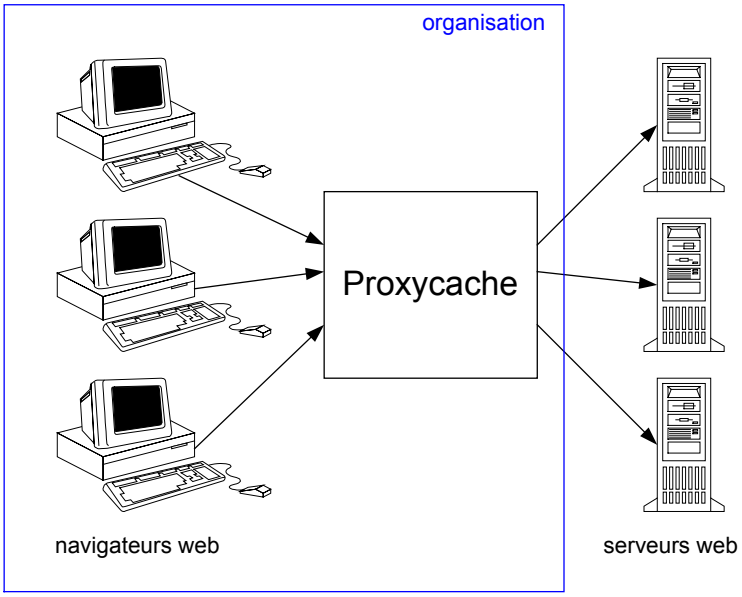


Figure 6 - Proxycache.

Comme mentionné à la section 2.5.5, un proxycache stocke les pages à l'intention des navigateurs de l'organisation configurés pour l'utiliser, ce qui permet de réduire le trafic réseau externe et d'ainsi préserver la bande passanrte Internet de l'organisation, qui est généralement une ressource critique.

Lorsqu'un proxycache transmet la requête d'un navigateur à un serveur, le serveur n'a pas connaissance de l'adresse IP du poste client. Tous les navigateurs utilisant un même proxycache sont donc vus par le serveur comme une seule et même machine, ce qui perturbe les logiciels de mesure d'audience qui calculent le nombre de visiteurs sur base des adresses IP. Dans des cas extrêmes ou l'organisation derrière le proxycache s'étend sur des zones géographiques différentes, la mesure de l'audience dans ces zones est perturbée.

2.5.7 Absence de fichiers de log

Certains ISPs hébergeant des sites Web ne fournissent pas aux propriétaires des sites hébergés l'accès aux fichiers de logs. Ce cas est fréquent chez les hébergeurs de sites personnels. Parfois les fichiers de logs ne sont pas générés

pour des raisons d'espace de stockage, ou de performance du serveur Web. Parfois les logs sont effacés après une période donnée pour des raisons d'espace de stockage.
Les logs sont l'information première indispensable à toute application de mesure d'audience. Toute période sur laquelle les logs sont manquants ne peut faire l'objet d'aucune mesure d'audience.

2.5.8 Résolution des répertoires

Les requêtes se terminant par un `/` ou faisant appel à un répertoire du serveur sont confiées à un module du serveur Web qui transforme l'URL avant de la desservir, en fonction de la configuration du serveur. Traditionnellement, le `/` de fin d'URL est remplacé par `/index.html`, ou bien si un tel fichier n'existe pas le contenu du répertoire est listé, ou encore la requête est rejetée.
Il n'est pas facile de retrouver l'opération réalisée par le serveur Web. D'autant plus que l'information nécessaire à cette détermination est volatile, même si elle est très stable.

2.5.9 Evolution temporelle du corpus

Avec le développement de l'intérêt pour le Web, le contenu des sites évolue de plus en plus rapidement : des pages sont modifiées, ajoutées et supprimées de plus en plus fréquemment. Les références faibles des fichiers de logs des serveurs Web ne retiennent pas l'information suffisante pour permettre une analyse tenant compte de cette dimension temporelle du contenu. D'où l'obsolescence quasi instantanée de la majorité des résultats actuellement obtenus. Par exemple, savoir qu'une page modifiée quotidiennement a été consultée un certain nombre de fois au cours d'une certaine période n'apporte pas beaucoup d'information sur l'intérêt porté par les visiteurs aux sujets qui y ont été successivement traités.

Figure 7 - Exemple de page Web évolutive.

2.5.10 Pages dynamiques

Les pages d'informations proposées par les sites Web modernes deviennent majoritairement dynamiques, dans le sens où les pages renvoyées sont composées après les requêtes des navigateurs, par exemple à partir de templates complétés à la volée par le serveur Web ou une extension de celui-ci [Abi00]. Les technologies à disposition des concepteurs de sites se multiplient : CGI, SSI, ASP, JSP, PHP, … Ici encore, les informations retenues par les serveurs Web se révèlent insuffisantes lorsque les requêtes envoyées par les navigateurs ne préfigurent aucunement du contenu de la page qui sera renvoyée. Nous disons alors que celle-ci est “dépendante de l’environnement” ou encore “dynamique variable”.

Figure 8 - http://webserv1.ulb.ac.be/searchdb/search (1).

Figure 9 - http://webserv1.ulb.ac.be/searchdb/search (2).

2.5.11 Programmes exécutés sur le poste client

Les applets Java et Flash sont des programmes téléchargés par le navigateur et exécutés en son sein. En général, les informations diffusées via ce genre de

programmes sont transférées de manière spécifique. Typiquement, seul le téléchargement du programme est tracé par le serveur dans les fichiers de logs.
Dans de rares cas, le programme sert uniquement d'interface de présentation multimédia ou interactive de l'information, cette dernière restant transférée via le canal classique (HTTP), ce qui génère les traces habituelles dans les fichiers de logs.

2.5.12 Résolution des adresses IP

Les fichiers de logs stockent les adresses IP des postes clients. A partir de ces adresses IP, il est parfois possible d'obtenir le nom de DNS (Domain Name Server), et partant de disposer d'informations géographiques. Plusieurs obstacles se présentent :

- certaines adresses IP ne sont pas associées à un nom de DNS ;
- l'association d'une adresse IP à un nom de DNS peut changer avec le temps ;
- les noms de domaines de plus haut niveau comme .com, .edu, .org , .net, etc. sont mondiaux et n'apportent aucune information géographique.

2.5.13 Méconnaissance de l'activité sur le poste client

L'activité de consultation stockée dans les fichiers de logs ne préfigure pas précisément du comportement de l'utilisateur assis devant son ordinateur. Si l'utilisateur ouvre plusieurs fenêtres de navigation, les utilise dans un ordre aléatoire, les ferme, quitte son poste de travail, le serveur Web n'en a aucune connaissance.
L'exploitation des fichiers de logs dans le but de modéliser de tels comportements des utilisateurs a donc peu de chances d'aboutir. On lui préférera des techniques orientées utilisateur comme la mesure électronique des ordinateurs ou le sondage classique d'un échantillon d'utilisateurs.

De plus, s'il est possible de savoir quelles pages ont été *affichées* dans les navigateurs Web des visiteurs, il est impossible de savoir dans quelle mesure elles ont été effectivement *lues* par ceux-ci. Cette différence est fondamentale.

Ce travail se base sur l'hypothèse que l'activité des postes clients est sujette à un effet macroscopique statistiquement normal.

2.5.14 Sémantique

Certaines desideratas des utilisateurs sont conceptuels [Tré98] : les propriétaires de sites Web veulent des résultats d'audience par rubriques, par sujets, etc., ce que les fichiers de logs ne permettent pas de trouver directement, et qu'il est difficile d'obtenir par des traitements informatiques simples. La mesure de l'audience obtenue par rubrique ou par concept[4] ne fait pas partie des fonctionnalités des logiciels d'analyse d'audience classique.

Pour une telle mesure – et l'on peut même ici pleinement parler d' "analyse" – il est nécessaire de prendre en compte le contenu des pages.

2.6 Solutions et workarounds

2.6.1 Rotation automatique des logfiles

Les logs d'une période traitée ne seront plus modifiés ; l'information que l'on peut en extraire sera donc toujours la même. Il est dès lors judicieux de la conserver dans une base de données et d'y rajouter de façon incrémentale les informations apparues depuis l'exécution précédente. Une possibilité est d'effectuer régulièrement une rotation (renommage) des fichiers de logs. Par exemple, quotidiennement.

2.6.2 Compression des logfiles

Les fichiers de logs prennent énormément de place sur les serveurs Web et dans les archives éventuelles ; il est donc être intéressant d'envisager une compression de ceux-ci. Etant donné que les fichiers de logs contiennent des informations extrêmement répétitives, les taux de compression sur ceux-ci sont excellents. Empiriquement, les facteurs de compression GNU ZIP sur ce type de fichiers s'étendent entre 10 et 50, c'est-à-dire :

$$taux\ de\ compression := \frac{taille - taille\ compressée}{taille} \in [90\%, 98\%]$$

[4] Ici , le mot "concept" s'entend au sens de "représentation mentale générale et abstraite" [Rob02].

Le taux de compression global moyen des logs d'un serveur Web est proche de 95%, ce qui est remarquable. Un serveur Web générant 30 Go de logs bruts par an nécessitera un espace disque annuel de 1.5 Go, ce qui est tout à fait acceptable.

2.6.3 Cookies

Les cookies sont de petits fichiers textes que le serveur Web envoie au client pour stockage sur son disque dur. L'utilisation de cookies – généralement via des fichiers images, HTML ou JavaScript, invisibles ou non, appelés "marqueurs de page" – permet de court-circuiter les mécanismes de cache et dès lors de :

- distinguer et compter précisément les visiteurs ;
- compter les requêtes de page ;
- analyser les visites a posteriori.

Cette technique n'est pas une panacée car un nombre non négligeable de navigateurs Web sont configurés pour refuser systématiquement les cookies.

2.6.4 Solution personnalisée

Si le propriétaire du site Web a du temps, des compétences et les possibilités techniques, il peut se programmer une solution personnalisée. Lourde, difficile à réaliser, à maintenir, coûteuse, cette solution répondra néanmoins certainement à ses besoins.

2.6.5 Clustering

Plusieurs stratégies alternatives sont possibles pour répondre à une question de type "quel sujet intéresse le plus les visiteurs d'un site ?". Pour supporter les cas d'utilisation présentés dans la section 1.4, l'ULB veut par exemple savoir si les visiteurs de son site s'intéressent plus à l'enseignement ou à la recherche scientifique.

Les rédacteurs peuvent également regrouper les pages Web qui concernent les cours dans un répertoire `/cours/` et les pages Web qui concernent la recherche dans un répertoire `/recherche/`. Ensuite, selon les possibilités du logiciel, définir deux *clusters* sur base de ces deux répertoires, et calculer l'audience obtenue pour chaque cluster. Plusieurs problèmes se posent :

- certains logiciels n'offrent pas cette possibilité de clustering ;
- l'unité de mesure est la consultation par page, sans tenir compte de la valeur sémantique des pages, càd la quantité d'informations traitant directement du sujet visé. Une page de 10 lignes peut traiter plus profondément d'un sujet qu'une page de 100 lignes, et vice-versa ;
- la délimitation des sujets dans les pages est floue : une page peut parler des deux sujets visés par la comparaison, ou peut ne faire que citer l'un des sujets comme exemple d'un domaine plus vaste comme par exemple "les missions de l'université".
- les pages sont réparties autrement dans l'arborescence du serveur, par exemple par faculté ;
- les pages sont dynamiques.

2.6.6 Application de la rétroaction

Pour un site Web, deux types de rétroaction sont possibles : technique et sémantique, comme l'illustre la Figure 10.

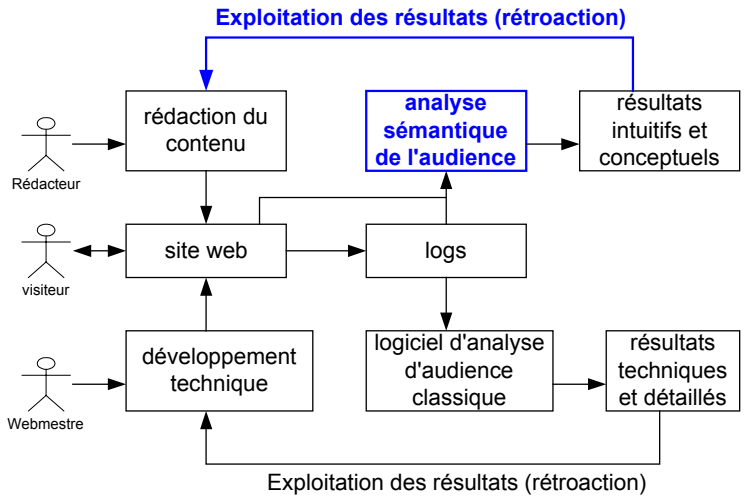


Figure 10 - Rétroactions technique et sémantique.

Cette section donne quelques lignes directrices pour exploiter les différents résultats de l'analyse d'audience dans les deux processus de rétroaction représentés à la Figure 10 :

- pour renforcer la qualité d'un message diffusé, l'effort d'adaptation du contenu doit être proportionnel à l'intérêt porté à celui-ci ;
- pour augmenter la quantité d'audience, pour attirer l'attention sur une partie du site, l'effort d'adaptation du contenu doit être inversément proportionnel à l'intérêt porté à celui-ci ;
- l'appel à la partie la moins consultée peut être adapté par un principe de vases communicants, c'est-à-dire par l'ajout et l'amélioration des hyperliens qui la référencent à partir de la partie la plus consultée.
- l'adaptation du contenu induit l'effet "bouche à oreille", par exemple les visiteurs comblés envoient des références par e-mail à leurs connaissances ou ajoutent spontanément des hyperliens sur leur site Web personnel.

2.6.7 Autres solutions alternatives

Selon la structure de son site, l'ULB peut consulter "manuellement" les résultats générés par le logiciel afin de les synthétiser dans un tableur en vue de les y interpréter intuitivement. Même pour un petit site, ce procédé est lourd, tout en étant source d'erreurs et de flous. De plus, cela demande des compétences particulières et hétérogènes.

Une autre solution est de poser la question en ligne aux visiteurs, ce qui s'apparente à du marketing direct. Sur internet, seule l'adresse IP du poste client est connue, et partant parfois une vague information géographique. Il faut donc se renseigner sur lui autrement, par exemple en lui demandant ses coordonnées, brisant ainsi son anonymat, ce qu'il est rarement prêt à accepter. Les possibilités de l'internet interactif permettent en revanche de lui poser la question online, lors de sa visite : par exemple, "vous intéressez-vous plutôt aux cours de l'ULB ou à la recherche scientifique ?". Les résultats sont le plus souvent partiels car peu de visiteurs prennent le temps ou font l'effort de répondre.

Enfin, du marketing global, comme le pratique NetValue, est coûteux et ne donne que des informations générales (génériques) et imprécises, de plus pas toujours adaptés à la spécificité du site étudié.

2.7 Considérations

L'analyse d'audience classique se heurte à un certain nombre de problèmes. De plus, [Mar93] suggère qu'une rétroaction sémantique, à l'usage du management de l'organisation, apporte une valeur ajoutée nettement supérieure ; il indique que les résultats nécessaires à une telle rétroaction doivent être conceptuels, intuitifs, résumés et qualitatifs.

Aucune des solutions classiques n'est vraiment conçue pour aider ce type de rétroaction. Or, il s'agit de prendre une décision importante et de haut niveau : réorienter la ligne rédactionnelle du site.

Par exemple, si l'ULB veut optimiser les effets de ses coûteux efforts d'adaptation du contenu de son site, doit-il focaliser ces efforts sur les cours ou sur la recherche ?

Les entreprises sont confrontées à des problèmes similaires, d'autant plus que leur site Web est généralement synonyme d'e-business, voire d'e-commerce [IBM03], et fait donc partie intégrante du processus de vente, c'est-à-dire de survie de l'entreprise. Une telle décision est donc capitale, tant pour l'entreprise que pour le canal de communication lui-même, un canal de communication coûteux et stratégique auquel le management global accorde autant d'importance et d'intérêt que le management du système d'informations.

3 Solution : la prise en compte du contenu

Des problèmes énumérés en section 2.5, je vais tenter de répondre à celui de la section 2.5.14. De par l'approche en profondeur que j'ai choisie, je m'attends à ce que ma solution résolve partiellement les autres problèmes. J'étudierai l'impact de la solution sur ceux-ci en section 3.7.
Cette solution va permettre d'obtenir des résultats intuitifs qu'il n'est pas possible d'obtenir à l'heure actuelle :

- par les technologies existantes,
- par les produits existants,
- par des astuces basées sur les produits existants.

Le principe de base de ma solution sera d'analyser l'audience non plus sur base des pages mais sur base du contenu de ces pages. Cela consiste à ramener l'ensemble des éléments que je viens de lister à leur dénominateur commun : le contenu diffusé. En focalisant la résolution sur ce seul dénominateur commun, l'ensemble de la problématique sera améliorée.
Pour commencer, je vais définir une série de grandeurs significatives et exploitables intuitivement par les propriétaires de sites Web dans le processus de rétroaction sémantique (Figure 10).

3.1 Les grandeurs audimétriques

Soit un site Web donné.
Soit C le corpus des pages (statiques et dynamiques) d_i affichées dans les navigateurs Web des visiteurs du site pendant la période d'observation $[t_1,t_2]$:

$$C([t_1,t_2]) := \{d_1, \ldots, d_N\}$$

Un tel corpus est constitué de deux types de pages :
Pages statiques : les pages d'information existant avant toute requête au serveur.
Pages dynamiques : les pages d'informations générées en temps réel par le serveur en fonction de la requête reçue.

Soit M_s le corpus des documents (pages) statiques ms_i mis en ligne à l'instant t :

$$M_s(t) := \{ms_1, \ldots, ms_n\}$$

Soit M_d le corpus des documents (pages) dynamiques md_i renvoyées par le serveur Web pendant la période [t1,t2] :

$$M_d([t1,t2]) := \{md_1, \ldots, md_n\}$$

La Figure 1 illustre les relations entre les corpus Ms, Md, Cs, Cd et C. La relation entre Ms et Cs n'est ni injective ni surjective. La relation entre Md et Cd est bijective. C=Cs U Cd.

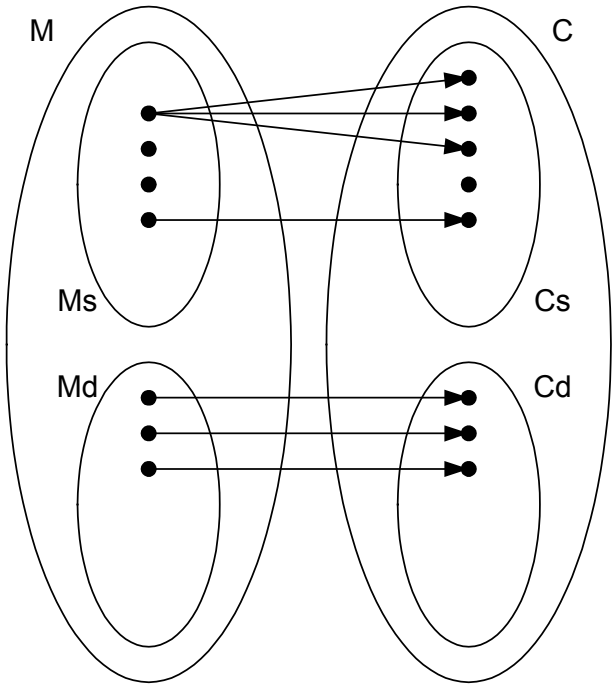


Figure 11 - Relations entre les corpus.

La mesure d'audience classique définit la consultation par site comme le cardinal de l'ensemble des documents du corpus C :

$$\text{Consultation } [t_1,t_2] := \#(C)$$

La mesure d'audience classique définit également la consultation par URI. L'expression "consultation par page" est rendue galvaudée par l'évolution temporelle des pages comme l'explique la section 2.5.9. Pour un site Web ne diffusant que des pages statiques, on a :

$$Consultation\,[t1,t2] = \sum_{URI} Consultation(URI,[t1,t2])$$

Enfin, la mesure d'audience classique définit le nombre de hits par site et par élément, selon une définition similaire à la consultation, où le corpus est élargi à l'ensemble des fichiers du site, y compris les fichiers binaires. Ces mesures servent à la rétroaction technique (Figure 10), par exemple pour le dimensionnement de la bande passante du site, pour l'optimisation de la taille des images, etc.

Je définis des grandeurs similaires pour chaque terme k_i dans les deux corpora. Soit $fréq_i$ la fréquence brute du terme k_i dans le corpus C et $fréq_{s\,i}$ la fréquence brute du terme k_i dans le corpus C_s.

$$\text{Consultation}(k_i,[t_1,t_2]) := fréq_i$$

$$\text{Mise en ligne statique}(k_i,t) := fréq_{s\,i}$$

Pour pouvoir calculer ces grandeurs, il faut pouvoir reconstituer à tout moment le "corpus des pages vues", càd l'ensemble des documents affichés dans les navigateurs Web des internautes connectés au site Web pendant une période donnée (dite "période d'observation").

Je définis la fréquence brute d'un terme k_i dans un corpus C comme la somme des fréquences brutes de ce terme dans les documents du corpus :

$$fréq_i := \sum_{d_j \in C} fréq_{i,j}$$

où la fréquence brute $fréq_{i,j}$ est définie par [Bae99] comme "la fréquence brute du terme k_i dans le document d_j, c'est-à-dire le nombre de fois que le terme k_i est mentionné dans le texte du document d_j".

Si un terme a été fortement consulté alors qu'il a été très peu mis en ligne, on peut conclure qu'il a fait l'objet d'un intérêt fort. Si à l'inverse, un terme a été peu consulté alors qu'il a été très intensément mis en ligne, on peut conclure qu'il a fait l'objet d'un intérêt faible. Sur base de ces considérations extrêmes, je tire comme principe général qu'un terme "a intéressé" d'autant plus un visiteur que ce terme a été consulté et d'autant moins qu'il a été mis en ligne. Je définis donc "l'intérêt" des visiteurs pour un terme comme le rapport direct de la consultation à la mise en ligne.

$$Intérêt\ statique(k_i, [t_1, t_2]) := \frac{Consultation(k_i, [t_1, t_2])}{\int_{t_1}^{t_2} Mise\,en\,ligne\ statique(k_i, t)dt}$$

Les informations nécessaires au calcul de ces grandeurs s'articulent en quatre dimensions :

- classique,
- temporelle,
- dynamique,
- lexicale.

Chaque dimension sera traitée par un sous-système logiciel spécifique détaillé en section 4.3. Passons ces quatre dimensions en revue.

3.2 Dimension classique

Le corpus des pages statiques et dynamiques constantes affichées est redondant. Entre deux mises à jour d'une telle page mise en ligne à une URI donnée, le contenu renvoyé à la requête de cette URI est constant. En d'autres termes, la consultation du contenu de tels documents peut être déterminée par multiplication de leur contenu et du nombre de consultations. Il n'est donc pas nécessaire de stocker chaque page renvoyée, il suffit de stocker une fois le

contenu, de déterminer sur quel période il est stable, et de connaître le nombre de consultations pendant cette période. Le nombre de consultations par période de temps peut-être déduite de l'information brute stockée dans les fichiers de logs.

La détermination de la stabilité, et plus généralement de l'évolution, du contenu est assurée par le sous-système WASA-CJ décrit ci-après. Le sous-système WASA-CA se charge d'analyser les fichiers de logs et d'en extraire les informations de consultation pertinentes :

- les requêtes aux pages statiques et dynamiques constantes, à l'exclusion des pages dynamiques variables ;
- les requêtes aux pages ayant un contenu, à l'exclusion des images et autres fichiers binaires ;
- les requêtes satisfaites, qui se marquent par un code de réponse HTTP 200 ("File Sent") ou 304 ("Use Cache") [Ber96], à l'exclusion des codes d'erreur ;
- l'URI de ces requêtes ;
- l'instant de ces requêtes, ou en fonction de la précision temporelle souhaitée, une sous-partie de cette information, par exemple seulement le jour, la semaine, le mois, …

Ne sont pas d'intérêt pour WASA :

- l'adresse IP du poste client,
- le nom d'utilisateur éventuel,
- le fuseau horaire du poste client,
- la méthode de requête HTTP (POST, GET, PUT ou DELETE),
- le nombre d'octets transférés,
- l'agent log,
- le referer log,
- l'error log.

Une application commerciale s'intéresserait également à l'adresse IP et répartirait les adresses en zones. Dès lors, tous les résultats pourraient être obtenus par zone ou par groupes de zones. Exemples de zones :

- les postes des développeurs Web qui testent en ligne les résultats visuels de leur travail,
- les robots Web,

- les pays (sur base du code ISO3166),
- les continents [Tré98],
- etc.

Le tri des adresses IP pourrait se faire sur base d'expressions régulières comme suit :

- France :=*.fr,
- Trafic interne ULB :=*.ulb.ac.be,
- Webadmin :=Webadmin*.domain.com,
- etc.

La réalisation technique de l'analyse des fichiers de logs est détaillée en section 4.3.1.

3.3 Dimension temporelle

3.3.1 Pages statiques

Par définition, le contenu de pages statiques est univoquement déterminé par la requête. C'est le contenu du fichier associé à l'instant de la requête à l'adresse "URI" spécifiée dans la requête. Ces informations concernant la requête (URI, instant) sont stockées dans les fichiers de log de type access et combined ; il est donc possible de déterminer quel fichier a été renvoyé. Si ledit fichier est *journalisé*, càd conservé avec ses méta-informations (nom, dates de début et de fin de mise en ligne, URI associée), nous disposons du chaînon manquant pour connaître le contenu renvoyé au navigateur et affiché par celui-ci. Nous appelons cela "ajouter une dimension temporelle à l'analyse d'audience sur internet".
Isolée, cette information permet donc de connaître a posteriori quel fichier a été mis en ligne en une URI donnée à un moment donné.

L'information d'évolution des pages statiques est évanescente, dans le sens qu'elle disparaît du système formé par le site Web au fur et à mesure du temps. Il faut donc veiller à ce que cette information soit en permanence récoltée et stockée par WASA. Il est prudent d'archiver régulièrement cette information fondamentale.

3.3.2 Journalisation

Les pages statiques mises en ligne, caractérisées par leur URI, leur contenu et leur timestamp (date de dernière mise à jour), peuvent subir plusieurs types de modifications significatives pour WASA, illustrées par le diagramme d'état de la Figure 12 :

1. modification de date sans modification de contenu,
2. modification de date avec modification de contenu,
3. création,
4. suppression.

Dans le cas particulier de la suppression, il ne reste aucune trace du fichier sur le serveur Web ; il manque donc l'information de l'instant de suppression, que l'on supposera égale à l'instant de la constatation de la disparition. Cette hypothèse n'affectera en rien les résultats puisque ceux-ci dépendent du croisement avec les consultations et que WASA ignore les consultations de fichiers manquants (erreur HTTP 404).

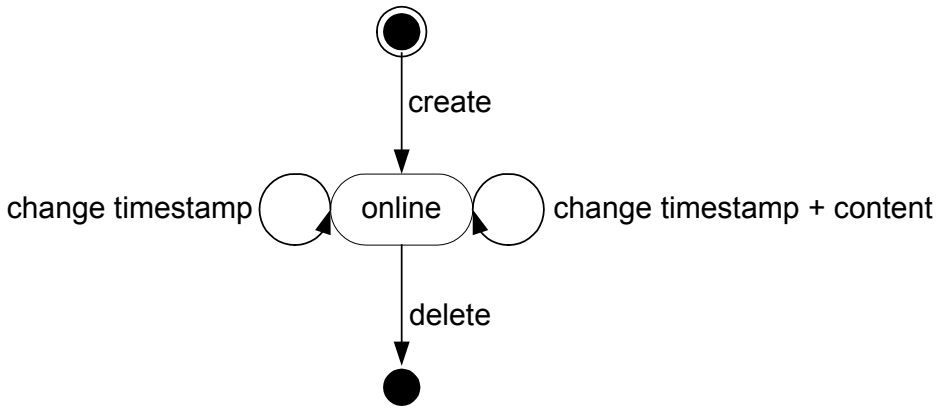


Figure 12 - Diagramme d'état d'un fichier mis en ligne.

3.4 Dimension dynamique

3.4.1 La problématique des pages dynamiques

Ces dernières années, l'informatique a beaucoup changé. Avec la démocratisation de l'accès à Internet et au World Wide Web, il est aisé d'accéder à des informations et même faire du commerce depuis n'importe où. Un grand nombre de technologies sont apparues pour combler la demande croissante d'applications Web rapides, légères, robustes, et surtout interactives. Dans un premier temps, l'interactivité et le contenu dynamique ont été délivrés par des programmes CGI. Bien vite, des solutions de scripting comme JSP, ASP et PHP ont fourni une interface Web aux composants utilisés pour gérer la logique des applications et l'accès aux sources de données, accélérant et facilitant ainsi le développement des applications. La complexité de ces applications Web et la diversité des technologies utilisées rendent la connaissance complète du corpus C impossible avec les outils actuels. En effet, le contenu des pages dynamiques est différent à chaque requête, en fonction des paramètres de la requête et de l'environnement à l'instant de la requête. A cause de cette dépendance à l'environnement et du fait que le mot "environnement" s'entend au sens très large, l'univocité entre la requête et le contenu renvoyé est le plus souvent perdue.

Le succès populaire et l'engouement commercial pour Internet ont été les moteurs de l'évolution de ce premier modèle statique du Web vers un modèle interactif, où le contenu des documents est généré selon les besoins. C'est l'avènement de l'Internet comme plateforme de déploiement d'applications Web.
L'architecture des applications Web découle de l'évolution de modèle client/serveur, qui est l'un des paradigmes les plus dominants des technologies de l'information. L'idée centrale de ce modèle client/serveur est de fournir une architecture d'application qui permette à un processus informatisé d'être partagé en plusieurs tâches moins complexes coopérant via un mécanisme de messagerie. La notion-clé du découpage du problème est de fournir des couches (niveaux) de fonctionnalités qui peuvent être écrites séparément et déployées sur plusieurs machines d'une manière efficace. Un exemple typique de découpage d'une application en couches fonctionnelles est le suivant :

- logique de présentation (presentation logic) : gère la façon dont l'utilisateur interagit avec l'application via une interface utilisateur (GUI) ;
- logique de l'application (business logic) : gère les règles de l'application (business rules) ;
- logique d'accès aux données (data access logic) : gère le stockage et la récupération des données.

Le découpage de la logique de l'application en couches permet de minimiser l'impact des modifications d'une couche, de rendre possible la réutilisation du code et d'augmenter sa fiabilité. Le modèle multiniveau (multicouche ou encore *multi-tier* en anglais) est un aboutissement logique du modèle client/serveur où :

- Le média utilisé par la logique de présentation est indépendant de la logique d'application ;
- la logique d'application est partagée et distribuée sur plusieurs machines ;
- la logique d'accès aux données peut supporter plusieurs bases de données ainsi que d'autres services comme des data warehouses, des systèmes "legacy", etc.

Cette solution est la plus flexible et la plus extensible.

La demande de contenu dynamique sur le web a transformé la programmation Web vers de nouvelles implémentations de l'architecture multiniveau. La couche de présentation forme le premier niveau qui inclut non seulement le navigateur Web mais aussi le serveur Web, qui est responsable de l'assemblage des données dans un format présentable. Le deuxième niveau est formé par la couche de l'application (business layer), qui consiste en un ensemble de script, de programmes ou de composants. Finalement, le troisième niveau fournit au deuxième les données dont il a besoin.
Bien que le langage HTML permette de créer des documents web qui sont des interfaces utilisateur délivrables par tout serveur Web et lisibles par tout navigateur, il reste un langage de description de données statiques par nature. CGI fut la première technologie à apporter un peu d'interactivité et de contenu dynamique au web. Bien vite suivie par des implémentations d'APIs spécifiques aux serveurs web, comme NSAPI et ISAPI. C'est ensuite qu'apparurent des solutions de script du côté serveur (server-side scripting) comme JSP et ASP, qui ont simplifié le développement d'applications web.

Après étude des techniques de programmation basées sur le server-side scripting, il ressort qu'une bonne partie du contenu sémantique des pages dynamiques est

déjà stocké dans un fichier template au format HTML. Seul le contenu vraiment dynamique est obtenu d'une base de données via le script. Une première approche consiste à extraire le contenu des templates : un compilateur (ou parser) traduit le langage source, constitué de l'HTML et du script, en un langage cible, constitué du seul HTML. Il suffirait alors de récupérer ce contenu et de le combiner avec l'information recueillie dans les fichiers de logs du serveur web.
Une première implémentation pour les JSPs basée sur JavaCC a montré les limitations de ce modèle :

- il faut écrire un compilateur pour chaque technologie de pages web dynamiques (ASP, JSP, PHP, etc.) ;
- il est impossible de récupérer ainsi le contenu généré par un programme CGI ou par une servlet Java ;
- la partie du contenu qui est extrait dynamiquement d'une base de données est inaccessible lors du déformatage ;
- l'avènement rapide de l'XML et de ses potentialités de mise en page via XSLT et CSS font évoluer les techniques de programmation vers des pages dynamiques ne contenant plus que du code qui génère les pages à partir de contenu provenant de fichiers XML et de bases de données.

La seule solution générique pour connaître à tout moment le contenu dynamique renvoyé aux navigateurs est de journaliser chaque page générée après la génération, et non plus avant. Je vais passer en revue les deux solutions étudiées dans le cadre de [Mat02].

3.4.2 Packet sniffer

Une première solution serait d'écrire un programme "renifleur de paquets" (packet sniffer) qui intercepte les paquets TCP/IP au niveau de la couche réseau et réalise le travail des couches réseaux supérieures pour reconstituer la page HTML qui a été générée par le serveur.
Les étapes qu'un tel programme doit réaliser sont les suivantes :

- stocker les paquets de manière efficace et rapide ;
- trier les paquets stockés pour obtenir des groupes de paquets correspondant chacun à une conversation entre le serveur et le navigateur, c'est-à-dire une requête HTTP suivie d'une réponse HTTP. Cela peut se faire sur base d'un groupe de paramètres qui identifient une conversation de manière univoque. Deux de ces paramètres sont spécifiés dans le protocole TCP et sont le port

du serveur et le port du client. Le paramètre supplémentaire est l'adresse IP du client, spécifiée dans le protocole IP ;

- reconstituer le flot de données en ordonnant ces paquets grâce à trois autres paramètres spécifiés dans le protocole TCP : le numéro du paquet, le numéro du prochain paquet et le numéro dans l'accusé de réception ;
- décortiquer le protocole HTTP pour séparer le document des messages et en-têtes HTTP.

Cette solution est "écologique", elle respecte l'environnement dans lequel elle est introduite. Elle est indépendante du type de serveur web. En contrepartie, elle demande beaucoup de ressources machine. En effet, elle doit stocker et traiter les paquets à l'aide de tris et traitements de chaînes de caractères. Or ces deux types d'opérations sont très coûteuses en terme de ressources.

Toute communication doit être traitée avant de connaître le type du fichier transféré. Si les images sont nombreuses, et c'est de plus en plus le cas dans les sites Web modernes, beaucoup de ressources de calcul sont perdues inutilement.

3.4.3 Module de serveur Web

Une deuxième solution est l'implantation dans le serveur Web d'un module "mouchard" interceptant le contenu de chaque page dynamique renvoyée et stockant celui-ci sur le disque avec sa méta-information (date de requête, format de la page renvoyée). Cette solution est dépendante du serveur Web, il faut donc écrire un module par serveur Web. Toutefois, il n'y a que deux serveurs Web vraiment répandus [Net03] :

- Apache : 62% des serveurs Web d'Internet ;
- Microsoft Internet Information Server : 27%.

Un autre désavantage de ce module est qu'il peut rendre le serveur instable si l'implémentation n'est pas suffisamment rigoureuse. Ce module sera exécuté dans le même processus que le serveur web : s'il bloque, le serveur peut bloquer aussi.

Par contre, un module peut accéder aux en-têtes HTTP du document, ce qui permet d'ignorer directement les fichiers binaires et les pages statiques, pour ne garder que les pages dynamiques.

Le module permet aussi d'intervenir sur le document avant qu'il ne soit envoyé au client, ce qui permet d'optimiser l'utilisation de la bande passante en compressant les documents pour les navigateurs qui le supportent.

3.4.4 Choix d'une solution

Au contraire de la technique de déformatage des templates, la solution du module d'extension du serveur Web est indépendante de la provenance des données et de la technologie utilisée pour générer la page.
Cette indépendance par rapport à l'évolution des techniques de programmation des applications Web confère à cette solution une grande pérennité.
Le module d'extension du serveur Web se situe suffisamment bas dans les couches de l'application web pour accéder aux pages déjà générées, mais suffisamment haut dans les couches du modèle réseau pour ne pas avoir à recomposer les données à partir des paquets TCP/IP.
C'est donc cette solution, baptisée *mod_trace_output*, qui a été retenue pour être implémentée et intégrée au protype de validation, comme détaillé et expliqué en section 4.3.3.

3.4.5 Pages dynamiques constantes

Il existe un cas particulier de page dynamique qui suggère une économie de traitement : les pages dynamiques constantes. Les requêtes successives à une URL associée à une page dynamique constante provoquent la génération de pages dont le contenu est constant sur une période donnée.
Je ne tiendrai pas compte de ce cas particulier car :

- Il est marginal de par l'évolution du contenu des sites Web décrite en section 2.5.9.
- Il peut être ramené au cas général des pages dynamiques.
- Les traitements redondants sont négligeables, en vertu des points précédents.
- Il est difficile à identifier par une méthode générique et durable à cause de la multiplicité des technologies de génération de pages dynamiques.

3.4.6 Volatilité des pages dynamiques

L'information du contenu des pages dynamiques variables est volatile, dans le sens qu'elle disparaît du système formé par le site Web aussitôt après sa génération. Il faut donc veiller à ce que cette information soit en permanence récoltée et stockée. Il est prudent d'archiver régulièrement cette information fondamentale.

3.5 Dimension lexicale

Les systèmes décrits en sections 3.3.2 et 3.4.3 permettent de collecter les documents et les informations nécessaires à la reconstitution complète du corpus C défini en section 3.1. Pour calculer les grandeurs audimétriques par terme définies en section 3.1, il faut pouvoir calculer le poids de chaque terme dans le corpus.

Les pages Web peuvent être textuelles mais sont le plus souvent semi-structurées au format HTML [Abi00]. L'avenir du Web semble indiquer l'émergence des formats XHTML et XML [W3C03]. Un compilateur pour ces formats permet de ramener aisément le cas des pages Web à celui des documents au format textuel simple.

3.5.1 Analyse lexicale

[Fox92] définit l'analyse lexicale comme "le processus de conversion d"un ensemble de caractères en un ensemble de mots ou d'occurrences" et "occurrence" comme "un groupe de caractères qui lorsqu'ils sont mis ensemble ont une signification".

L'analyse lexicale et l'indexation des documents peuvent être complétées par des algorithmes d'élimination des stopwords, de stemming et de clustering [Bae99]. Ces points seront respectivement abordés dans les sections 3.5.3, 3.5.4 et 3.6.

D'après [Fox92], la première décision à prendre lors de la conception d'un analyseur lexical pour un système d'indexation est la définition précise des termes valides dans le plan d'indexation, et *a complementario*, la définition des symboles séparateurs. Une définition simpliste serait que les termes d'indexation sont exclusivement constitués de lettres. En pratique, cette définition doit être revue pour tenir compte des éléments suivants :

- Les nombres : en général, la plupart des nombres ne sont pas de bons termes d'indexation. Néanmoins, les chiffres inclus dans des termes ne doivent être considérés comme des termes séparateurs, en particulier si le corpus a une vocation technique. Exemple : IBM Thinkpad *TP765L*. Nous adopterons une solution de compromis consistant à autoriser les chiffres dans les termes, à condition que ceux-ci contiennent également des lettres.

- Les traits d'union : faut-il séparer un terme à trait d'union en ses sous-parties ou le considérer comme un seul terme ? Plusieurs cas amènent à être traités différemment :
  - Dans le cas le plus général, la séparation d'un mot à trait d'union comme "abat-jour" peut amener une perte de précision.
  - Les mots sans trait d'union séparés en fin de ligne par un trait d'union entre deux syllabes doivent simplement perdre ce trait d'union.
  - Certains traits d'union font partie d'un nom : Jean-Pierre, F-16, MS-DOS.
- Les autres ponctuations : comme pour le trait d'union, certaines ponctuations peuvent être utilisées à l'intérieur de termes. Exemples : le système d'exploitation IBM *OS/2*, le fichier *COMMAND.COM*, la plateforme *.NET* de Microsoft.
- Les lettres capitales : la capitalisation des termes a généralement peu d'importance est ignorée par la plupart des systèmes existants.

3.5.2 Multilinguisme

Les sections suivantes traitent du multilinguisme dans les ensembles de documents. Pour des raisons de clarté, il convient de définir la notion de multilinguisme et de différencier différents niveaux de multilinguisme.
En effet, pour un ensemble multilingue de documents, plusieurs niveaux de granularité entrent en ligne de compte :

1. L'ensemble des corpora envisageables est totalement unilingue ou considéré comme tel ; la majorité des articles de la littérature scientifique anglophone adoptent cette hypothèse.
2. L'ensemble des corpora envisageables est multilingue et chaque corpus est unilingue.
3. L'ensemble des corpora envisageables est multilingue, chaque corpus peut être multilingue et chaque document est unilingue.
4. L'ensemble des corpora envisageables est multilingue, chaque corpus peut être multilingue, chaque document peut être multilingue et chaque paragraphe est unilingue.
5. L'ensemble des corpora envisageables est multilingue, chaque corpus peut être multilingue, chaque document peut être multilingue et chaque paragraphe peut être multilingue ; c'est la cas du World Wide Web.

Voici un tableau résumant les 5 niveaux de multilinguisme et indiquant le nombre de langues qu'il est possible de rencontrer dans chaque ensemble de termes.

Niveau	**Corpora**	**Corpus**	**Document**	**Paragraphe**
1	1	1	1	1
2	*	1	1	1
3	*	*	1	1
4	*	*	*	1
5	*	*	*	*

Le niveau de multilinguisme impacte la précision et la complexité des algorithmes de "stopword removal" et de "stemming". Ce travail traite du World Wide Web, niveau 5.
Pour des raisons de complexité et de performance, nous serons amenés à réaliser des hypothèses simplificatrices, à l'instar de la littérature. Je chiffrerai la perte de précision résultante et tenterai de la limiter par un choix judicieux des hypothèses.

La langue des documents d'un site Web n'est pas toujours spécifiée dans le document :

- un document texte ne fournit pas d'indication explicite ;
- un document semi-structuré (HTML) est censé inclure la balise `<HTML lang='fr'>` mais l'attribut `lang` n'est pas toujours présent.
- XHTML `<html xml:lang='fr'>` est prévu par le schéma XHTML [W3C03] mais de nouveau en pratique il reste un certain laxisme.
- dans un document multi-lingue, on pourrait spécifier un attribut `lang` par paragraphe : `<P lang='fr'>` ; cette pratique non standard et peu répandue permet une granularité plus fine.

Pour les documents (ou les paragaphes) dont la langue n'est pas spécifiée, il est possible d'appliquer des algorithmes de détermination automatique de la langue.

3.5.3 Stopwords removal

Peu après l'apparition de la recherche documentaire, [Luhn 1957] a identifié que beaucoup des mots les plus fréquents d'une langue font de piètres *termes d'indexation*. Des mots comme "le", "la", "de", "et" portent une sémantique trop large et donc non discriminante [Sal83, Van75]. De plus, l'ensemble des occurrences de tels termes représente une fraction importante des occurrences du

corpus. [Fra82] a déterminé que les 10 mots les plus fréquents en anglais représentent 20% à 30% des occurrences d'un document. L'élimination de ces termes le plus tôt possible dans le processus d'analyse lexicale améliore les performances de traitement et diminue la taille de l'index. Dans la plupart des cas, l'efficacité de recherche n'est pas impactée, ou très faiblement. A titre d'exhaustivité, je citerai comme contre-exemple le cas d'un site Web personnel dont il est souhaité savoir quelle est la proportion d'égocentrisme, mesurée par la proportion d'occurrences de termes comme "je", "j'", "moi", etc. Ce type de cas étant marginal, je n'en tiendrai pas compte. Une solution serait de développer un système modulaire permettant de "plugger" temporairement la prise en compte des stopwords.

Bien que l'élimination des stopwords est généralement acceptée comme un bienfait, il n'en reste pas moins une difficulté pratique, et ce déjà au stade de la conception. En effet, les listes de stopwords sont traditionnellement formés des mots les plus fréquents. Il reste à déterminer où mettre la limite.

Pour l'instant, je ne supprime pas les stopwords avant stockage des termes. Je ne tiens pas compte de la langue des documents. La raison est que je n'ai encore déterminé aucune utilité de tenir compte de ces deux points. Je suis néanmoins prêt à réagir si cela s'avèrait nécessaire, disposant de :

- une liste de stopwords anglais (voir annexe A) ;
- une liste de stopwords d'Eurovoc pour les 12 langues européennes [EUR03].

Comme nous l'avons vu en section 3.5.2, la langue d'une page d'un site Web n'est pas toujours bien définie. La détermination de la langue est importante car un terme peut être un stopword dans une langue tout en étant un terme significatif dans une autre. Exemple : en anglais, le terme 'kind' est considéré comme un stopword par la liste de [Fox92] (voir annexe A) tandis qu'en néerlandais, ce terme est un nom commun significatif : 'kind' signifie 'enfant'.

3.5.4 Stemming

L'algorithme de Porter est le plus connu. Il a été conçu pour l'anglais uniquement. Appliqué aux termes d'une autre langue, il peut réduire à une même racine des termes de familles différentes. Il y a donc là un risque de perte de précision sémantique.

Les algorithmes de stemming sont fondamentalement différents d'une langue à l'autre. Les développements dans ce domaine suggèrent qu'un système générique peut être mis en place sur base d'une grammaire spécifique [ISP03].

Porter est un algorithme de compression. D'après [Bae99], il n'est pas démontré que le gain en espace de stockage et en traitement ultérieur des termes réduits vaut l'investissement de performance dans l'application de l'algorithme à chaque terme stocké. Je n'appliquerai donc pas cet algorithme.

3.6 Clusters électriques

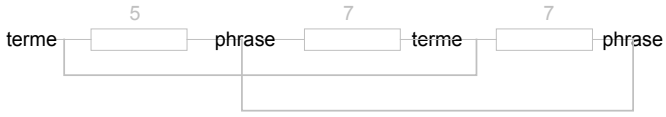
Pour satisfaire les besoins sémantiques futurs, j'ai étendu la théorie des clusters [Bae99] aux clusters électriques. L'innovation que je propose est une analogie entre d'une part la notion de distance sémantique entres les termes d'un document ou d'un corpus de documents (ou même d'une ontologie [Fen00]), et d'autre part les résistances électriques reliant les noeuds d'un circuit ouvert purement résistif. Cette analogie permet d'importer de l'électricité des règles qui régissent les circuits résistifs, ainsi que des méthodes de simplification et de calcul des circuits équivalents.

L'application de ce modèle à un document fonctionne comme suit. Chaque occurrence de terme est représentée par un point du circuit. Chaque point est relié par une résistance de 1 Ω aux points représentant les termes voisins. Exemple : "Cette phrase est courte" devient :

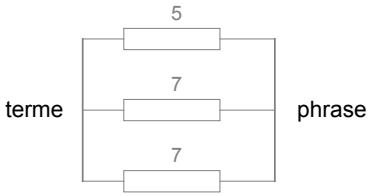
La distance sémantique entre deux termes d'un document se calcule par la transposition directe de la résistance équivalente entre les deux points associés dans le circuit électrique représentant le document. Dans l'exemple de cette phrase isolée, la distance sémantique entre les termes "phrase" et "courte" est de 2 Ω, au sens de la théorie des clusters électriques.

Une phrase isolée est représentée par un circuit électrique linéaire. Dans un document, il est fort probable que certais termes soient présents plusieurs fois. Si deux occurrences d'un même terme sont porteurs d'une même sémantique

précise, la distance sémantique entre ces deux occurrences est nulle par définition. La transposition à l'électricité est la liaison par une résistance de 0 Ω des deux points associés dans le circuit électrique. Autrement dit, les deux points sont en *court-circuit*. Exemple : La phrase "Un terme du début de cette phrase a la même sémantique qu'un terme à la fin de cette phrase." se modélise de la façon suivante :



Ce circuit pourrait être redessiné de la façon suivante :



Dans cet exemple-ci, la distance sémantique entre les termes "terme" et "phrase" est la résistance équivalente des trois résistances en parallèle, soit :

$$r(terme, phrase) = \frac{1}{\frac{1}{5}+\frac{1}{7}+\frac{1}{7}} = 2{,}06$$

Notons que dans cet exemple, seuls les termes "terme" et "phrase" sont court-circuités. En effet, dans ce modèle, seuls les termes porteurs d'une sémantique identique et précise se court-circuitent. Les termes polysémiques et les stopwords ne se court-circuitent pas. C'est booléen : ouvert ou fermé.
Une évolution de ce modèle relierait les occurrences de termes monosémiques par des résistances inversément proportionnelles au nombre d'occurrences du terme dans le document.

Il est également possible d'étendre ce modèle pour prendre en compte les hyperliens des documents hypertextes, ce que les modèles de clustering présentés dans [Bae99] ne font pas. Les termes contenus dans la balise HTML `<A>` se relient par une résistance de 1 Ω au document pointé par l'attribut `href` de cette même balise.

3.7 Conclusion

Tous les problèmes de détermination du corpus des documents affichés par les navigateurs ont été résolus.
Un certain nombre de questions restent en suspens concernant les améliorations possibles aux algorithmes d'analyse lexicale. D'après [Bae99], il y a actuellement controverse concernant les améliorations potentielles à la performance de recherche d'informations sur base de l'élimination des stopwords, du stemming et de la sélection des termes d'indexation. En fait, il n'existe aucune preuve concluante que de telles opérations textuelles apportent des améliorations tangibles en recherche d'informations. Dès lors, il se pourrait que les systèmes de recherche modernes n'utilisent pas du tout ces opérations textuelles. Un bon exemple de cette tendance est le fait que des moteurs de recherche Web indexent tous les mots du texte sans considération de leur nature syntaxique ou de leur rôle dans le texte. En vertu de l'analogie soulevée dans le cadre de [Bur02] entre l'analyse d'audience et la recherche d'informations, je n'exploite donc actuellement aucune de ces techniques.

Le système WASA résoud les problèmes des sections 2.5.1, 2.5.2, 2.5.9, 2.5.10, 2.5.14. Le traçage de l'output complet par mod_trace_output rend accessoire la présence des fichiers de logs et pourrait résoudre le problème exposé en section 2.5.7. Les autres problèmes, soit sont inhérents à la structure du Web, soit ont un impact réduit, notamment du fait que les grandeurs audimétriques par terme sont des grandeurs relatives et que dès lors les effets des facteurs perturbateurs se compensent.

4 Réalisation

Afin de valider les théories exposées au chapitre 3, j'ai réalisé systématiquement une implémentation baptisée WASA (pour Web Audience Semantics Analysis). Aujourd'hui, cette application 100% pure Java comporte 6500 lignes de code réparties en plus de 50 classes réparties en 10 packages. Elle a subi des tests avec succès pendant plus de 12 mois et a récolté pendant cette période les informations temporelles nécessaires à des tests sur des périodes suffisamment longues pour pouvoir servir de proof-of-concept.

L'architecture globale de l'application WASA est représentée à la Figure 13.

.

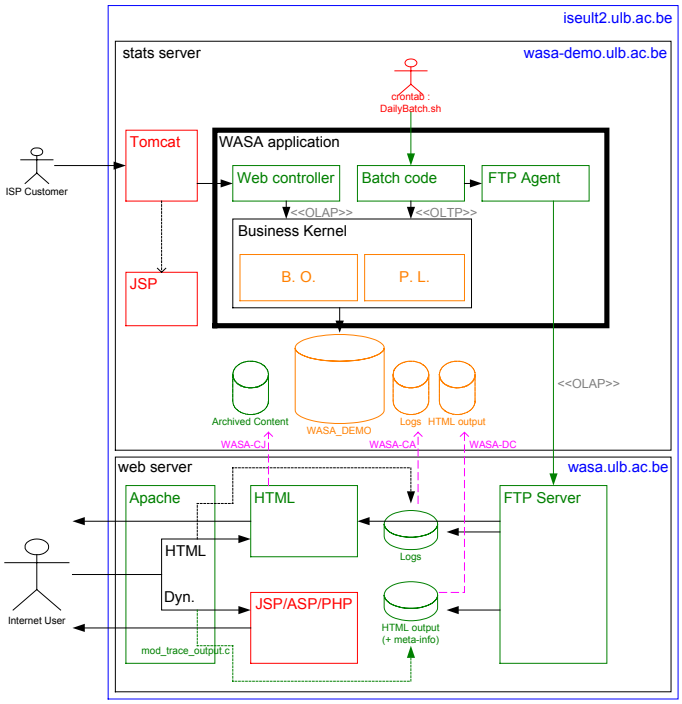


Figure 13 - Architecture globale de WASA.

4.1 Technologies

4.1.1 Java

L'implémentation est réalisée principalement en Java, afin de bénéficier des avantages suivants :

- Java possède des qualités intrinsèques favorables à un développement de qualité : langage simple, robuste, sécuritaire, mûr, orienté objet.
- Il existe de nombreuses libraires Java standard et third-party permettant de transférer les fichiers de logs et les pages Web d'un serveur distant à travers le réseau.
- La gestion intégrée des tâches multiples permet de gérer la consultation à distance des résultats de l'analyse sémantique par des propriétaires de sites Web multiples.

4.1.2 Java Server Pages (JSP)

L'interface du prototype se présente sous forme d'écrans JSPs s'affichant dans les navigateurs Werb des propriétaires de sites Web.
Les JSPs sont similaires à des pages Web HTML et y ajoutent la possibilité d'afficher un contenu dynamique. Cette technologie développée par Sun Microsystems [JAV03] propose un certain nombre de balises qui permettent au concepteur d'une JSP d'insérer des propriétés de JavaBeans [JAV03] et des éléments de contenu générés par du code Java [Wah00].

Pour qu'un serveur Web puisse desservir des JSPs, il faut lui adjoindre un composant auxiliaire spécifique appelé "serveur d'applications Java et JSP". Le prototype a été testé en fonctionnement au sein du logiciel Apache Jakarta Tomcat [JAK03]. Ce logiciel présente l'avantage d'être utilisable comme serveur Web indépendant et comme module de serveur Web Apache, ce qui rend très flexible son intégration à un environnement donné.

4.1.3 FTP

Les fichiers de logs, les pages statiques et les traces de pages dynamiques sont collectées par WASA grâce au protocole FTP.

FTP est l'acronyme de File Transfer Protocol, un protocole réseau standard qui permet le transfert de fichiers entre des ordinateurs, typiquement via l'Internet selon un mode client/serveur entre notre ordinateur et un serveur distant. Par exemple, il est possible de transférer des fichiers depuis un disque dur vers un serveur Web, ou d'y télécharger des programmes depuis un site de sharewares.
FTP est l'un des trois protocoles réseau les plus populaires d'Internet, avec HTTP (le surf sur le World Wide Web à l'aide d'un navigateur) et SMTP (l'envoi d'e-mails). Bien que le protocole HTTP fonctionne bien pour télécharger des pages HTML et les petits fichiers images associés, il n'a jamais été conçu pour transférer des gros fichiers. Avec FTP, nous pouvons télécharger de gros fichiers et reprendre le transfert après interruption, à partir d'où le transfert s'est arrêté. La transfert de fichiers par attachements à des e-mails est peu pratique et inefficace pour des gros documents. Pour le téléchargement de tels fichiers vers le serveur, FTP est la seule solution standard et ouverte. Cela fait de FTP une brique fondatrice d'Internet. Comme l'illustre la Figure 14, ces trois protocoles sont situés dans la couche applicative du modèle ISO en couches d'Internet.

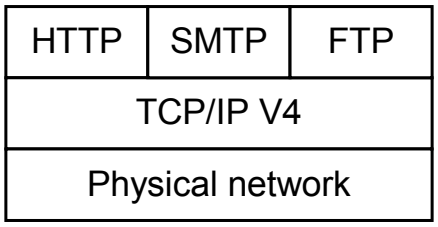


Figure 14 - Une vue simplifiée du modèle ISO en couches d'Internet.

Le document de référence RFC959, définit les principes, le contexte et les commandes du protocole FTP [Pos85]. Il est complété par les documents RFC1579 et RFC1738 de l'IETF.

4.2 WASA Framework

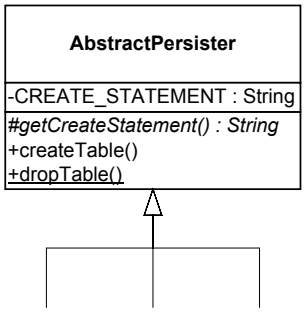
4.2.1 Base de données

WASA a été conçu pour fonctionner avec n'importe quel système de persistence. La classe `AbstractSqlPersister` permet d'utiliser n'importe quel SGBD compatible JDBC. Le système JDBC est expliqué en section 4.2.2.

Le support SQL diffère généralement d'un SGBD à l'autre, en particulier dans la syntaxe DDL. J'ai donc défini des balises représentant des types de données génériques. En fonction du SGBD auquel le système se connecte, les balises sont automatiquement remplacées par le type SQL approprié. La méthode `AbstractSqlPersister.createTable()` se charge de ce remplacement lors de la création des tables du schéma, sur base du préfixe de l'URL de connection JDBC.

Type de données/SGBD (préfixe de l'URL de connection JDBC)	Oracle 8i (jdbc:oracle)	MySQL (jdbc:mysql)	IBM DB2 UDB 7 (jdbc:db2)	Default (ANSI-SQL)
<LONG>	INTEGER	BIGINT	BIGINT	INTEGER
<STRING>	VARCHAR(1023)	TEXT	VARCHAR(1023)	VARCHAR(255)
<DATETIME>	CHAR(19)	DATETIME	TIMESTAMP	CHAR(19)

Les types standards SQL comme DATE sont acceptés par l'ensemble des SGBD.



Chaque objet business persistent est associé à une table du schéma et à une classe de persistence dérivant de la classe `AbstractSqlPersister`. Le nombre de tables peut être problématique lors d'une opération de JOIN SQL. En effet, lors de la programmation d'ordres SELECT joignant plus de deux tables MySQL, [Bur02] a constaté des problèmes rédhibitoires de performances. La solution d'indexer les clés de jointure n'apporte pas d'amélioration satisfaisante des performances. Deux solutions ont été avancées :

- adopter un SGBD professionnel comme IBM DB2 ou Oracle 8i (à tester) ;
- programmer la logique de jointure en Java : pénible mais efficace.

4.2.2 JDBC Persistence Layer

La volonté de se connecter à un nombre maximum de SGBD à partir du langage Java conduit naturellement au choix de JDBC.
JDBC est une marque déposée de Sun Microsystems désignant l'API Java de communication avec les SGBD. C'est pour cette raison que JDBC est souvent perçu et employé pour *Java DataBase Connectivity*. JDBC consiste en un ensemble de classes et d'interfaces Java permettant aux programmeurs qui les utilisent d'accéder virtuellement à n'importe quelle base de données, uniquement sur base de code Java. JDBC permet d'établir une connexion avec une base de données, de formuler des requêtes SQL plus ou moins complexes – éventuellement pré-compilées – et de traiter le résultat renvoyé par la base.
JDBC traite la diversité des protocoles de SGBD en permettant au programmeur de passer ses requêtes SQL sous forme de chaînes de caractères à un pilote sous-jacent. Toutes les fonctionnalités du SQL peuvent donc être exploitées.
Afin de limiter ou tout du moins de mieux cerner les incompatibilités, Sun Microsystems [JAV03] a instauré le label JDBC Compliant attribué aux pilotes de SGBD supportant la norme ANSI SQL2 de 1992.
Grâce à JDBC, il est ainsi possible d'écrire une application simple sans se soucier de la base à laquelle on s'adresse, ce qui évite de devoir écrire un code spécifique à l'accès d'une base Oracle, un autre pour accéder à une base Informix, et ainsi de suite pour chaque nouveau SGBD. La complexité est reportée sur le pilote ; il suffit d'utiliser l'API JDBC et de fournir le bon pilote pour que les spécifités de communication avec le SGBD deviennent transparentes. Le programme écrit peut donc se connecter virtuellement à n'importe quelle base tout en s'exécutant sur n'importe quelle plateforme, pour autant que l'application et le pilote soient conformes aux spécifications JDBC.

Il existe 4 types de pilotes :

1. Le bridge JDBC-ODBC (type 1) : l'API de connexion aux bases de données la plus utilisée est l'ODBC (Open DataBase Conectivity) établie par Microsoft. Largement adoptée, elle permet de connecter la plupart des SGBD d'un grand nombre de plateformes. L'inconvénient majeur de l'ODBC est qu'elle repose sur une interface écrite en C, ce qui induit naturellement des problèmes de sécurité, de robustesse, de stabilité et de portabilité. Ces facteurs rendent quasi inconcevable son intégration dans le langage Java. C'est pourquoi l'utilisation d'ODBC à partir de Java a été mise au point sous forme d'un pont logiciel appelé "bridge JDBC-ODBC", permettant ainsi de conserver certains avantages de JDBC. L'utilisation de ce pont ne dispense pas d'installer sur chaque poste client les pilotes ODBC. Son intérêt est donc de profiter de l'immense base existante de pilotes ODBC dans des applications autonomes, bien que la liste des pilotes JDBC est largement suffisante aujourd'hui.
2. Les pilotes natifs (type 2) : ce type de pilote réutilise le pilote natif du constructeur d'un SGBD et encapsule celui-ci dans des fonctions Java respectant l'API JDBC. Ainsi, toute requête JDBC d'une application utilisant un tel pilote est convertie en appel natif afin de communiquer directement avec le SGBD dans son protocole propriétaire.
3. Les pilotes trois-tiers (type 3) : ce type de pilote est à la fois plus souple et plus complexe à mettre en œuvre que les autres, car il se base sur une architecture trois-tiers au lieu du traditionnel modèle client-serveur. Un serveur middleware doit être mis en place entre le client final et le SGBD ; il joue à la fois de rôle de serveur vis-à-vis du premier et le rôle de client vis-à-vis du second. L'intégration client-serveur est donc double, permettant une souplesse nettement supérieure aux modèles classiques. Ce système ouvre de nombreuses possibilités au niveau du middleware : utilisation des pilotes de type 2, choix des langages et des protocoles, gestionnaire de sécurité supplémentaire, filtre ou observateur passif de requêtes SQL, implémentation d'une API de plus haut niveau, services complémentaires, accès répartis, optimisation de trafic, etc.
4. Les pilotes 100% pure Java (type 4) : ce type de pilote est idéal pour des applications 100% pure Java car il convertit directement les appels JDBC en protocole propriétaire au SGBD. Un tel pilote est généralement fourni par le constructeur du SGBD.

La couche de persistence de WASA se base sur un framework permettant de plugger n'importe quel pilote de type 4 ; ce framework a été testé avec succès pour les bases de données suivantes :

- MySQL 3.22
- IBM DB2 UDB 7.0
- Oracle 8i

4.2.3 ThreadConnectionManager

Une architecture typique d'une application persistente orientée objets est représentée à la Figure 3.

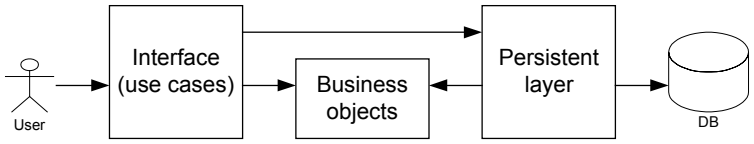


Figure 15 - Architecture logicielle classique.

Inévitablement, la couche première est responsable de la clotûre de la transaction. Pareillement, la couche de persistence est responsable des opérations CRUD [Nor02] des objets business et exploite à cette fin la connection au SGBD. Entre les deux, il n'y a pas de raison d'introduire à l'analyse la connaisance du concept de persistence dans les objets business. Cette connaissance est généralement introduite en phase de design, afin de permettre aux couches extrêmes l'utilisation partagée de la connection. Cela complique et allourdit le code. Cette gêne peut être évitée grâce à un système horizontal et centralisé de gestion des connections, basé sur les *threads* : le *ThreadConnectionManager* (TCM), dont l'intégration à l'architecture est représentée à la Figure 16.

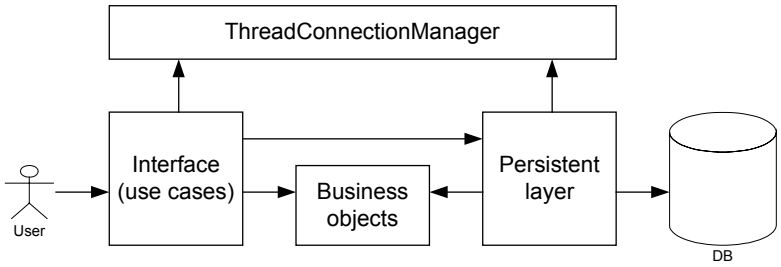


Figure 16 - Architecture avec TCM.

Le TCM attribue une connection par thread, donc par transaction business. L'appel à cette connection peut se faire depuis n'importe quel endroit du code sans qu'il soit nécessaire de disposer d'une référence à un quelconque objet :

```
Connection connection =
ThreadConnectionManager.getInstance().getConnection();
```

Le TCM gère l'ensemble des connections. A chaque appel de la méthode `getConnection()`, il renvoie la connection associée à la thread courante.
La fin de transaction et la cloture des connections est également gérée par le TCM.

Toute application utilisant ce type de ConnectionManager doit disposer au lancement du runtime de 4 paramètres, qui permettent au TCM de se connecter à la base de données :

1. le nom de classe du pilote JDBC,
2. l'URL de connection,
3. le nom dutilisateur,
4. le mot de passe.

Les deux principaux cas d'exécution utilisés dans l'application WASA sont :

1. autonome : exécution batch single-thread séquentiel ;
2. Web : les internautes sont les acteurs, l'exécution est multi-thread.

La récupération des paramètres se fait respectivement :

1. Par la ligne de commande, éventuellement via un fichier de configuration propre ;
2. Via le fichier de configuration `web.xml` de l'application Web. Dans le serveur d'applications

Les diagrammes de séquence des figures Figure 17 et Figure 18 illustrent l'utilisation du TCM dans ces deux cas d'exécution.

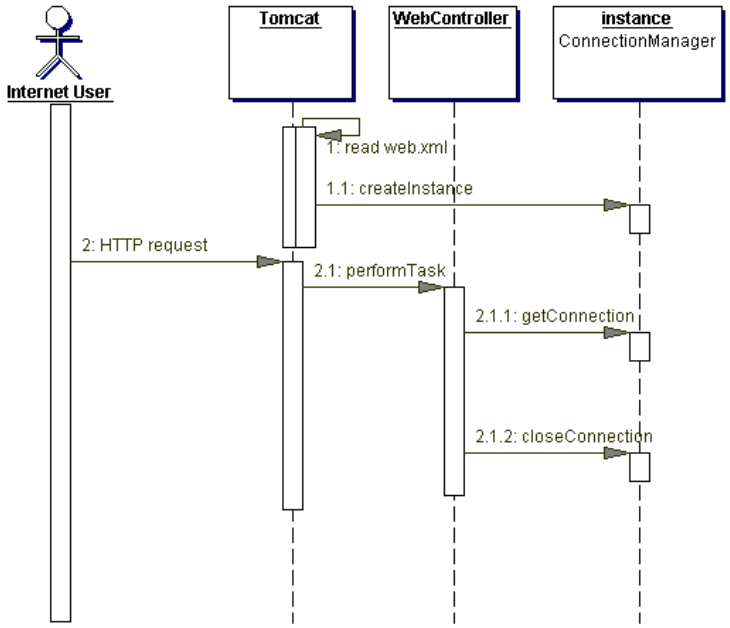


Figure 17- Utilisation du TCM dans une application Web

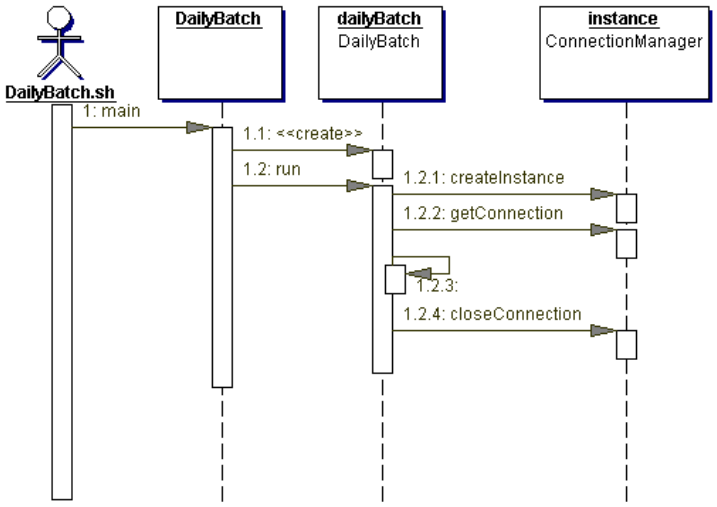


Figure 18 - Utilisation du TCM dans une application autonome

Avantages du TCM :

- Facilité et clarté d'appel de connection : ThreadConnectionManager.getInstance().getConnection() ;
- Création automatique de la connection par le TCM au premier appel de connection ;
- contexte d'appel implicite : pas de transport de connaissance par un paramètre de connection de méthode en méthode le long du thread d'exécution.

Désavantages du TCM :

- si le code appelant ne ferme pas la connection qu'il a ouverte, il y a risque de fuite des connections ; de plus, si le pilote n'impémente pas un système de time-out, ce qui est rare, il y a risque de saturation des connections du pilote ; ce problème pourrait être radicalement résolu par l'implémentation d'un système spécifique de "garbage collection" synchrone ou asynchrone ;
- les nouvelles threads créées en cours de process doivent gérer pareillement leur propre connection ; si nécessaire, une extension du TCM sera réalisée pour supporter la gestion des connections des threads enfants ;

- ne supporte actuellement qu'une seule base de données par exécution ; ceci pourra être étendu facilement si nécessaire.

Chaque désavantage a donc une solution possible en cas de nécessité. Le TCM a été conçu pour être facilement extensible à l'implémentation de telles solutions.

4.2.4 Gestion des erreurs

Le système de gestion des erreurs se base sur le mécanisme standard Java : les exceptions [JAV03].

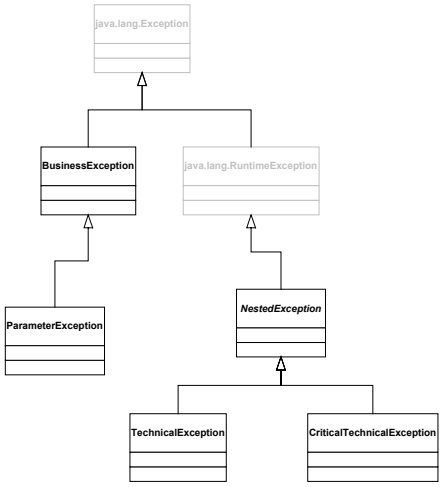


Figure 19 - Hiérarchie des exceptions WASA

Signification des différentes exceptions WASA :

- TechnicalException : apparaît lorsque le code rencontre une situation inconnue. Il peut essayer de la gérer. Exemple : valeur incongrue dans un traitement élémentaire.
- CriticalTechnicalException : apparaît lorsque le code rencontre une situation inconnue. Il ne va pas essayer de la gérer. Exemple : impossible de se connecter à la base de données.
- BusinessException : apparaît lorsque le code est apte à gérer une situation connue. Exemple : un traitement demandé par l'utilisateur ne peut être réalisé pour des raisons logiques.

- ParameterException : apparaît lorsque le code est apte à gérer une situation connue. Exemple : des paramètres de requête spécifiés par l'utilisateur sont non valides par rapport aux limites de service imposées par l'application.

Les exceptions TechnicalException et CriticalTechnicalException dérivent de la classe abstraite NestedException, qui elle-même dérive de java.lang.RuntimeException ; elles ne doivent donc pas être déclarées dans la signature des méthodes susceptibles de les lancer.
La fonctionnalité apportée par une NestedException est l'encapsulation d'une exception ne dérivant pas de java.lang.RuntimeException, ce qui permet de lancer n'importe quelle exception sans devoir la déclarer explicitement, ni dans les méthodes susceptibles d'en lancer une, ni dans les méthodes dépendantes de celle-ci jusqu'au plus haut niveau.
Avantages :

- cela allège le code ;
- cela évite la connaissance d'informations techniques dans le code business ;
- l'attention est concentrée sur les exceptions importantes dans le fonctionnement du programme.

4.3 Les 4 sous-systèmes

Pour assumer ses tâches, WASA est divisé en quatre sous-systèmes :

- WASA-CA (Consultation Analysis)
- WASA-CJ (Content Journalling)
- WASA-DC (Dynamic Content)
- WASA-LA (Lexical Analysis)

Ces quatre sous-systèmes sont décrits en détail ci-dessous. Chacun de ces sous-systèmes est responsable de la collecte (et dans le futur de l'exploitation) des informations. En rapport aux quatre sous-systèmes, ces informations s'articulent en quatre dimensions :

- Classique (WASA-CA)
- Temporelle (WASA-CJ)
- Dynamique (WASA-DC)
- Lexicale (WASA-LA)

Passons ces quatre dimensions en revue, en regard des sous-systèmes associés.

4.3.1 WASA-CA

Les informations nécessaires aux sous-systèmes se trouvent sous forme de fichiers de logs stockés sur le serveur Web qui héberge le site cible. La propriété `logDirectoryPath` de la table WebSite indique le répertoire distant dans lequel se trouve les fichiers de logs. La méthode `LogDirectory.downloadLogFiles()` rapatrie les fichiers non encore rapatriés de par le passé, dont la liste est stockée en local dans le fichier `~wasa/var/<WebSiteName>_downloadedLogFiles.log`. Le système de récupération à distance des fichiers est détaillé en section 4.1.3.

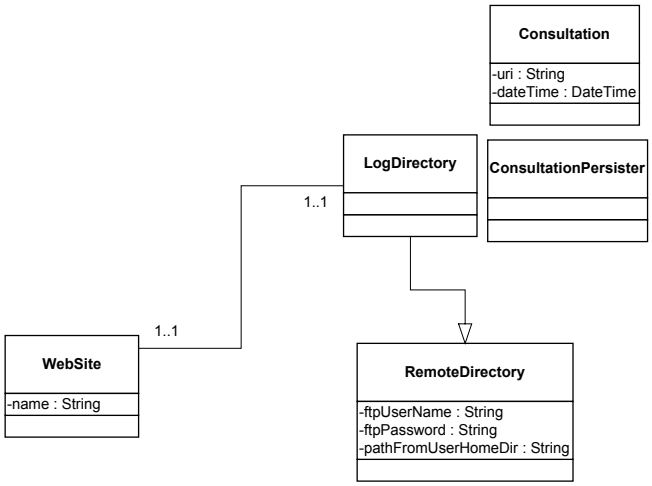


Figure 20 - Modélisation UML du système de gestion des logs.

Ensuite, la méthode `LogDirectory.analyzeLogFiles()` extrait les informations de consultation spécifiée en section 3.2 depuis les fichiers de logs vers la base de données, dans la table `Consultation`. Le système de parsing des lignes de logs est basé sur la librairie d'expressions régulières Jakarta ORO [JAK03].

Une classe SemanticsContainerPolicy attachée à la classe WebSite permet de sélectionner ou de rejeter un fichier associé à une URL comme porteur d'une

sémantique sur base de son extension. Par exemple, par défaut les fichiers `.html, .htm, .xhtml, .txt` sont considérés comme porteurs de contenu.

4.3.2 WASA-CJ

Le principe de l'opération est de comparer l'état du contenu du WebSite actuellement connu dans WASA (A) avec le contenu réel (B). Soit C = A U B l'ensemble des fichiers au moment de la comparaison. Pour chaque URI de C on compare les timestamps de tA et de tB, et éventuellement les contenus cA et cB :

TA	tB	cA	cB	Action	Download
X	-			DELETE	
X	Y	X	X	REFRESH	
X	Y	X	Y	UPDATE (content)	X
-	X			INSERT	X
X	X				

[Rob02] définit un *journal* comme « un écrit portant la relation quotidienne des événements ». C'est ce qui convient dans ce cas-ci.
La mise à jour du journal devrait idéalement être faite le plus tôt possible après chaque action sur le contenu du site afin de traiter le plus précisément possible l'évolution du contenu affecté par l'action DELETE, la suppression d'un fichier ne laissant aucune trace du moment de cette action. L'idéal est de journaliser juste après chaque mise à jour. En pratique, en fonction de l'évolutivité du contenu du site, une journalisation horaire ou quotidienne peut être "raisonnablement précise".
La tâche de tenir le journal du contenu d'un site (modélisé par la classe WebSite) est déléguée à la classe ContentJournal. Le ContentJournal gère la liste des traces des pages statiques ayant été mises en ligne sur le site Web. Il parcourt quotidiennement le répertoire de contenu (ContentDirectory) du site, récupère les pages statiques et met à jour les méta-informations dans la base de données. Les méta-informations stockées par WASA sont des objets de la classe `FileTrace`, caractérisés par une URI, une date de début de mise en ligne, une date de fin de mise en ligne et l'association à une archive de fichier.

4.3.3 WASA-DC

WASA-DC inclut le module mod_trace_output, dont l'implémentation est détaillée dans [Mat02]. Cette implémentation en C fonctionne pour le serveur Web Apache 1.3.23+ sur un système d'exploitation de type UNIX/POSIX, ce qui est le plus courant pour Apache. Les versions Apache 2.0 et supérieures implémentent un système de modules différent et ne sont pas supportées par mod_trace_output. Les versions inférieures à Apache 1.3.23 n'implémentent pas le système de filter_callback et ne sont pas supportées non plus par mod_trace_output.

mod_trace_output s'immisce dans le cycle du traitement des requêtes, de manière à obtenir le résultat du traitement par les autres modules, c'est-à-dire les données qui vont être envoyées au navigateur.
Deux types de support de stockage sont fournis :

- dans des fichiers sur le disque dur du serveur ;
- dans une base de données MySQL, locale ou distante.

Il y a aussi une option de compression au format gzip tant pour les fichiers que pour les enregistrements en base de données.

La méthode `DynamicOutput.downloadTraceFiles()` assure la récupération des fichiers tracés. Les fichiers sont ensuite analysés par la méthode `DynamicOutput.analyseTraceFiles()`.

4.3.4 WASA-LA

Les fichiers statiques récoltés par le sous-système WASA-CJ subissent l'analyse lexicale puis sont listés dans le fichier `~wasa/var/<WebSiteName>_analysedStaticFiles.log`.
Les fichiers de trace récoltés par le sous-système WASA-DC subissent l'analyse lexicale puis sont listés dans le fichier `~wasa/var/<WebSiteName>_analysedDynamicFiles.log`.

L'analyse lexicale consiste en la séparation des termes des documents et l'insertion des occurrences dans la base de données, dans la table Occurrence, dont les deux champs sont des clés secondaires pointant vers les clés primaires des tables FileTrace et Word. La table Word est assistée par un cache Java en

mémoire vive accélérant les opérations sur les données de cette table, en particulier les opérations de jointure.

4.3.5 Le calcul des grandeurs audimétriques

Une première transformation de l'expression mathématique de la consultation repose sur les constatations suivantes :

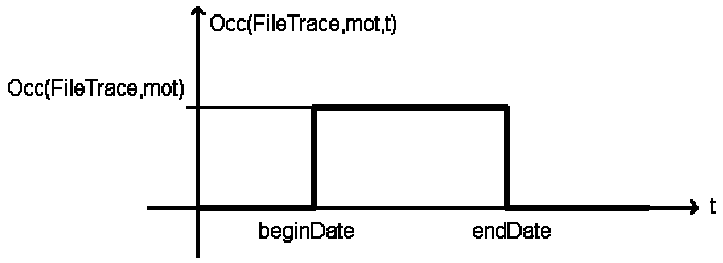
- le nombre d'occurrences vues dans des pages statiques peut être exprimé en fonction des Consultations, des FileTraces correspondantes et des occurrences dans ces FileTraces.
- le nombre d'occurrences vues dans des pages dynamiques est directement obtenu par le nombre d'occurrences dans les DynamicContentTrace (le nombre de consultations d'une DynamicContentTrace étant par définition exactement égal à 1).

$$\Rightarrow i(t_1,t_2,mot) = \sum_{Consultation=t_1}^{t_2} Occ(FileTrace(Consultation),mot) + \sum_{DynamicContentTrace=t_1}^{t_2} Occ(DynamicContentTrace,mot)$$

Les méthodes nécessaires dans WASA pour calculer cette grandeur sont :

- Collection:Consultation ConsultationPersister.getConsultationsInPeriod(WebSite,t1,t2)
- FileTracePersister.getFileTrace(WebSite,uri,dateTime) où uri et dateTime sont obtenus par interrogation d'un objet Consultation
- ArchiveFilePersister.getDynamicContentTraceInPeriod(WebSite,t1,t2)
- OccurrencePersister.getNumberOfOccurrences(ArchiveFile,Word)

La mise en ligne peut être réexprimée sur base d'une propriété des FileTraces : leur domaine d'existence sur une période de temps limitée fait que le nombre d'occurrences de tout mot est nul en dehors de cet intervalle de temps et constant dedans :



La valeur de l'intégrale de cette grandeur sur une période [t1,t2] vaut donc :

$$\int_{t_1}^{t_2} Occ(FileTrace, mot, t)dt = Occ(FileTrace, mot).Durée\ de\ présence(Filetrace, t_1, t_2)$$

Dès lors, pour des FileTraces dont le domaine temporel d'existence est disjoint de la période [t1,t2], la valeur de cette intégrale est nulle. Cela apporte une simplification intéressante en terme de calcul puisqu'il n'est dès lors plus nécessaire d'instancier tous les objets FileTrace. Soit FileTrace[t1,t2] l'ensemble des FileTraces dont le domaine d'existence n'est pas disjoint de l'intervalle de temps [t1,t2], nous avons :

$$p(t_1, t_2, mot) = \sum_{FileTrace[t_1,t_2]} Occ(FileTrace, mot).Durée\ de\ présence(FileTrace, t_1, t_2)$$

Les méthodes nécessaires dans WASA pour calculer cette grandeur sur base de cette expression sont :

- long FileTrace.getDuration(t1,t2)
- Collection:FileTrace FileTracePersister.getFileTracesCrossingPeriod(WebSite,t1,t2)
- OccurrencePersister.getNumberOfOccurrences(FileTrace,Word)

5 Conclusion

L'objectif global de ces recherches est de développer de nouvelles méthodes d'analyse de la consultation des sites Web, dans le contexte présenté au chapitre 1.

Le chapitre 2 expose les techniques et logiciels existants, ainsi que les résultats qu'ils obtiennent. Au fil des années, les fonctionnalités de ces logiciels se sont révélées de plus en plus insuffisantes, alors qu'elles évoluaient de moins en moins. J'explique que ce phénomène trouve son origine dans les problèmes techniques posés par l'évolution du Web comme l'augmentation exponentielle de la consultation, l'évolution temporelle du contenu statique et dynamique et l'émergence constante de nouvelles technologies serveur. Une autre conséquence des ces problèmes techniques est que la demande d'une analyse sémantique de l'audience Internet n'a pas trouvé de réponse satisfaisante. Des solutions alternatives incomplètes ou superficielles ont été temporairement utilisées.

Dans le chapitre 3, je me suis fixé comme objectif de développer une solution audacieuse traitant ces problèmes en profondeur. Le principe de ma solution est de collecter pendant une période dite "d'observation" l'information nécessaire et suffisante à une analyse sémantique de l'audience des sites Web. Cette information s'articule en quatre dimensions principales :

- la dimension "classique" : les traces des consultations du contenu statique par les visiteurs du site Web au cours de la période d'observation,
- la dimension "temporelle" : le journal de l'évolution du contenu statique mis en ligne sur le site au cours de la période d'observation,
- la dimension "dynamique" : les archives des pages dynamiques générées et renvoyées au cours de la période d'observation [Mat02].
- la dimension "lexicale" : la réduction des archives de pages semi-structurées [Abi00] à leur contenu textuel, puis à une liste d'occurrence de termes [Fox92].

Le mécanisme de récupération des informations exploite les conclusions de [Nor98] et se base sur des composants technologiques que j'ai passés en revue dans [Nor03].

Le sous-ensemble d'informations formé par les trois premières dimensions permet de retrouver avec précision l'ensemble du contenu du site affiché dans les navigateurs Web des visiteurs pendant la période d'observation, quelques soient les technologies (CGI, ASP, PHP, JSP, etc.) mises en oeuvre sur le serveur Web hébergeant le site et quelque soit la fréquence de mise à jour du contenu de celui-ci. En y associant la dernière dimension, il est possible de calculer pour chaque terme des grandeurs similaires aux grandeurs audimétriques standards (présence, diffusion, audience, intérêt, affinité [Med03]). L'analyse globale de l'ensemble des termes et des grandeurs associées permet de prendre des décisions pour réorienter la ligne rédactionnelle du site Web en modifiant la priorité et l'importance relative des différents concepts véhiculés par le site, créant ainsi un processus de rétroaction sur le contenu plus efficace [Mar93].

Cette solution générique et inédite pallie les manques technologiques décrits dans la section 2.5 et génère des résultats plus intuitifs que les produits existants. Sur base des méthodes de développement logiciel préconisées dans [Nor02], j'ai validé ma théorie en implémentant un logiciel prototype fonctionnel doté d'une interface Web, permettant à des propriétaires de sites Web cobayes de tester à distance les résultats obtenus. L'implémentation de ce prototype est détaillée au chapitre 4.

A présent, je vais m'atteler dans la seconde partie de ces recherches à prendre en compte la dimension "sémantique". Comme le signale [Sch02], la plupart des termes peuvent être interprétés de plusieurs manières. Ainsi, un terme polysémique ne communique pas forcément le sens attendu. Une grandeur calculée par terme représente l'audience obtenue par une idée potentiellement différente de celle que l'utilisateur projette dans le terme.

Figure 21 - Exemple de terme polysémique dans une page Web.

Le marquage sémantique de l'information, consistant à indiquer dans les documents la juste interprétation des termes ambigus, est une solution irréaliste

car trop lourde à mettre en oeuvre pour la masse d'information traitée, du moins pour l'être humain [Gro02]. J'étudierai dans quelle mesure l'ordinateur peut assumer cette tâche.

A l'inverse, une sémantique donnée peut être caractérisée par plusieurs termes, leurs variations grammaticales et leurs synonymes, voire des locutions, auxquels cas il peut être plus précis de ne pas se limiter à un seul terme. J'étendrai donc les grandeurs audimétriques par termes à des grandeurs similaires par concepts, c'est-à-dire par entités conceptuelles porteuses d'une sémantique significative. Les concepts seront judicieusement caractérisés par des groupes de termes interprétés. La tâche de l'ordinateur sera à la fois de faire tendre à l'univocité la sémantique portée par la liste des termes caractéristiques et de suggérer d'autres termes pour étendre cette liste [Cro92]. Pour ce faire, l'ordinateur exploitera les liens relevés entre les termes dans des vocabulaires électroniques [Lan00] comme les thesauri [Wil98] et les ontologies [Fen00]. J'exploiterai trois types de vocabulaires :

- construits automatiquement à partir des documents collectés [Sri92],
- construits automatiquement à partir des associations indiquées par l'humain au système [Har92],
- externes, c'est-à-dire existant dans le monde extérieur au système, spécifiques à un domaine précis ou génériques comme WordNet [WOR03], EuroWordNet [EWN03] et CYC [CYC03].

L'interprétation sémantique des concepts analysés restera sous responsabilité humaine assistée par l'ordinateur, selon une répartition des tâches prônée par [Ait87].
Cette amélioration donnera des résultats à la fois plus intuitifs et plus précis.

De plus, j'ai constaté lors de ma participation à [Bur02] qu'un sous-ensemble de l'information recoltée dans l'observation sus-décrite d'un site Web permet l'implémentation facile d'un moteur de recherche local. De là, j'ai cerné une analogie opportune entre l'analyse sémantique de l'audience des sites Web et la recherche documentaire, dont l'application la plus populaire est la recherche d'informations sur le Web [Bae99]. Les éléments fondamentaux d'une recherche documentaire, à savoir une requête et une collection cible de documents, se retrouvent dans ma problématique respectivement sous forme d'un concept et de l'ensemble des documents affichés par les navigateurs connectés à un site Web pendant une période d'observation.

Dès lors, une étude sera réalisée pour déterminer les techniques utilisées dans la recherche documentaire qui pourront être exploitées après transposition dans le contexte de l'analyse sémantique de l'audience des sites Web, qui se distingue par :

- un traitement de l'information d'une complexité supérieure,
- la contrainte forte de gestion multilingue des documents,
- l'absence de contraintes temps réel, obstacle majeur de la recherche d'informations sur le Web.

D'après R. Baeza-Yates [Bae99], ce dernier point ouvre la voie à l'expérimentation des techniques d'approfondissement du traitement de l'information comme la reformulation de la requête [Qiu93], l'exploitation des co-occurences de termes dans les pages [Xu96], la prise en compte des volumes de citations [Bri98], technique à la base du succès du moteur de recherche Google, la classification automatique des documents [ACM03], etc.

Pour obtenir de bonnes performances de calcul, j'envisage de faire appel à deux techniques simplificatrices :

- l'échantillonnage statistique, consistant à sélectionner judicieusement un nombre limité de pages dynamiques et à extrapoler les résultats obtenus à partir de ces pages, à la manière d'un sondage public ;
- l'indexation sémantique latente [Fur88], théorie considérant que les idées d'un document sont plus liées aux concepts décrits dans celui-ci que les termes du document. En pratique, chaque document est associé à des concepts dans un espace de dimension inférieure, dans lequel les opérations seront transposées pour obtenir des résultats à la fois plus utiles et moins gourmands en calcul.

De telles simplifications introduisent une imprécision négligeable. En effet, [Bro99] pose l'hypothèse fondamentale et largement supportée que l'information est un phénomène subjectif, un produit de l'interprétation du texte. Le modèle transactionnel de lecture postule que la négociation active entre les lecteurs et les textes produit de la *signification* [Str90]. La lecture n'est pas une réception passive de signification projetée par une référence neutre et fixe, ou stable, mais un processus actif de contextualisation des mots à l'intérieur d'un ensemble de définitions relatives et concurrentes qui renseignent le moment d'expression [Jac97]. En d'autres termes, ce ne sont pas les chaînes de caractères qui sont significatives mais leur interprétation par un agent cognitif [Ber95]. Il y a donc un grand écart entre la connaissance du contenu affiché dans les navigateurs Web

des visiteurs d'un site et la signification générée dans l'esprit des visiteurs par la diffusion des informations du site.

[Bro99] suggère comme approche à la manipulation de ce type de données le modèle de distance sémantique entre les termes. J'évaluerai les différentes techniques de calcul de distance sémantique applicables aux ontologies externes comme WordNet. J'évaluerai également les techniques similaires applicables aux ontologies construites automatiquement à partir des documents. Dans ce domaine, les techniques qui me semblent les plus intéressantes sont les techniques de clustering. Néanmoins, en l'état actuel, elles ne me semblent pas suffisantes car elles perdent de la sémantique en ne tenant pas compte :

- de toute l'information structurelle disponible dans les documents ;
- des hyperliens présents dans les documents hypertextes, ce qui s'explique aisément par le fait que ces techniques ont été conçues pour des documents textuels.

L'évolution que je propose en section 3.6 se base sur une analogie entre d'une part la distance sémantique entre les termes d'un document et d'autre part les résistances électriques entre les noeuds d'un circuit.

Au final, en combinant toutes ces techniques, je compte obtenir un système capable de calculer pour chaque concept diffusé par un site Web des grandeurs audimétriques les plus intuitives possibles.

Pour chaque technique, je prendrai en compte la question complexe des associations ternaires multiples posée par les sites Web multilingues, cas fréquent en Europe et très peu abordé dans la littérature internationale, majoritairement anglophone [Flu96, Bae99].

Au fur et à mesure de mes explorations théoriques, je compléterai mon logiciel prototype – conçu pour être aisément extensible – avec une implémentation des hypothèses retenues afin de valider ces dernières. L'étalon de mesure sera apporté par la mesure d'audience classique, capable de produire une analyse sémantique dans un cas très particulier. En effet, certains sites présentent naturellement un groupement sémantique de pages dont il suffira de mesurer l'audience standard [Ras92].
Enfin, j'apprécierai les performances de chaque technique en l'appliquant à l'analyse de l'audience de sites Web extrêmement consultés, de l'ordre de

100.000 accès quotidiens aux pages, et au contenu volumineux, tels le site Web de l'Université Libre de Bruxelles et d'autres sites auxquels j'aurai accès grâce à des relations privilégiées avec le Service Informatique et Réseaux de l'Ecole Centrale Paris.

Glossaire

Dans ce document et à travers la littérature, différentes conventions de langage sont utilisées, des abus ou commodités de langage risquent certaines ambiguïtés selon les contextes, beaucoup de définitions sont controversées ou galvaudées, etc. Ce glossaire regroupe une série d'expressions pouvant porter à confusion et précise le ou les sens utilisés dans ce document.

Concept : un mot largement signifié :

- [Rob02] concept := "représentation mentale générale et abstraite d'un objet" (cette définition y est monosémique).
- [Rob02] conception := "ensemble de concepts".
- [Zha02] concept := "terme représentant un sujet pertinent" ; pour éviter tout confusion, je n'utilise pas cette définition, à la fois galvaudée et tautologique.
- [Zha02] concept space := "terme(s) et relations sémantiques représentant un sujet pertinent".
- [Bae99, p169] concept := "groupement en un composant unique d'indexation de noms qui apparaissent les uns auprès des autres dans le texte" ; $concept_i := \{k_1, \ldots, k_n\}$
- [Bae99, p171] concept := "unité sémantique de base communiquant une idée" ; "habituellement, un terme de thesaurus est utilisé pour dénoter un concept".

Consultations : nombre de requêtes à des pages HTML reçues par un serveur Web ; cela exclut les images et autres fichiers binaires.
Corpus : ensemble, collection de documents.
Hits : nombre de requêtes au sens large reçues par un serveur Web ; cela inclut les pages HTML, les fichiers images, les autres fichiers binaires, les autres documents en ligne, . .
Média : canal de diffusion massive d'informations.
Mise en ligne : une information est mise en ligne sur le Web lorsqu'elle est consultable à partir d'un navigateur Web connecté à l'Internet.
Page Web : fichier textuel ou hypertextuel mis en ligne sur un site Web
Page statique : page d'information existant avant toute requête au serveur Web.

Page dynamique : page d'information générée en temps réel par le serveur en fonction de la requête reçue.
Page dynamique constante : page dynamique dont le contenu est constant et univoquement déterminé par son URL sur une période donnée.
Page dynamique variable : page dynamique dont le contenu n'est pas univoquement déterminé par son URL, c'est-à-dire est fonction du temps, de l'utilisateur ou d'autres paramètres.
Pages prélevées : voir consultations.
Période d'observation : période pour laquelle le système d'analyse d'audience dispose d'informations référentielles suffisantes pour fournir l'analyse demandée.
Sémantique : relatif aux phénomènes signifiants dans le langage, relatif à la signification, au sens ; analyse sémantique := analyse de contenu ; champ sémantique := ensemble de mots et de notions qui se rapportent à un même domaine conceptuel ou psychologique.
Terme : chaîne de caractères portant ou non une signification.
Taxonomie, ou taxinomie : classification.
Topologie : structure dans un ensemble, géométrie de situation.

Abréviations

ADT Abstract Data Type
ANSI American National Standards Institute
API Application Programming Interface
AWT Abstract Window Toolkit
CGI Common Gateway Interface
CLF Common Log Format
CORBA Common Object Request Broker Architecture
DBMS DataBase Management System
DMZ DeMilitarized Zone
DNS Domain Name Server
ECP Ecole Centrale Paris
FTP File Transfer Protocol
GIF Graphics Interchange Format
GNU GNU is Not Unix
GUI Graphic User Interface
HTML HyperText Markup Language
HTTP HyperText Transfer Protocol
IHM Interface Homme-Machine
IP Internet Protocol
ISP Internet Service Provider
JDBC Java DataBase Connectivity
JDK Java Development Kit
JFC Java Foundation Classes
JIT Just In Time
JVM Java Virtual Machine
ODBC Open DataBase Connectivity
PIC Position Independent Code
RGB Red Green Blue
RMI Remote Method Invocation
SGBD Système de Gestion de Bases de Données
SGBDR Système de Gestion de Bases de Données Relationnelles
SMB Service Message Broking
SQL Sructured Query Language

SSI Server Side Include
SSJ Server Side Java
PC Personal Computer
TCM ThreadConnectionManager
TCP Transfer Control Protocol
ULB Université Libre de Bruxelles
UML Unified Markup Language
URI Uniform Resource Identifier
URL Uniform Resource Locator
WASA Web Audience Semantics Analysis
WORA Write Once Run Anywhere
WWW World Wide Web

Références

[Abe03] Aberdeen Group, *Aberdeen Reports on Enterprise Disquiet in the Land of Web Analytics*, 2003, http://www.dmreview.com/master.cfm?NavID=198&EdID=6610.

[Abi00] S. Abiteboul, P. Buneman, D. Suciu, *Data on the Web, From Relations to Semistructured Data and XML*, 2000, Morgan Kaufmann Publishers, ISBN 1-55860-622-X.

[ACM03] http://www.acm.org/class/, ACM Computing Classification Systems.

[Ait87] J. Aitchison, A. Gilchrist, *Thesaurus Construction - A Practical Manual*, 2nd edition, 1987, London, ASLIB.

[Bae99] R. Baeza-Yates, B. Ribeiro-Neto, *Modern Information Retrieval*, 1999, Addison-Wesley, ISBN 0-201-39829-X.

[Ber96] T. Berners-Lee et al., *Hypertext Transfer Protocol – HTTP/1.0, Request for Comments: 1945*, IETF Network Working Group, 1996.

[Bri98] S. Brin, L. Page, *The anatomy of a large-scale hypertextual Web search engine*. In Proc. of the 7th Int. WWW Conference, Brisbane, Australia, April 1998.

[Bud99] A. Budanitsky, G. Hirst, *Semantic distance in WordNet : an experimental, application-oriented evaluation of five measures*, 1999.

[Bur02] J. Bury, *Analyse et conception d'un moteur de recherche*, Travaux de Fin d'Etudes de la Faculté des Sciences Appliquées, Université Libre de Bruxelles, 2002.

[Cof01] S. Coffey, *Internet Audience Measurement, a practitioner's view*, http://jiad.org/vol1/no2/coffey/, 2001.

[Cro92] C. J. Crouch, B. Yang, *Experiments in automatic statistical thesaurus construction. In Proc. of the ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 77-88, Copenhagen, Denmark, 1992.

[CYC03] http://www.cyc.com/, site web de CYC.

[EUR03] http://europa.eu.int/celex/eurovoc/, site Web du thesaurus Eurovoc.

[EWN03] http://www.illc.uva.nl/EuroWordNet/, site web d'EuroWordNet.

[Fas00] Fast, *Principles of Online Media Audience Measurement*, http://www.fastinfo.org/measurement/pages/index.cgi/audiencemeasurement, 2000.

[Fen00] D. Fensel, *Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce*, 2000, Springer, ISBN 3-540-41602-1.

[Flu96] C. Fluhr, *Multilingual Information Retrieval*. In R. A. Cole et al., Survey of the State of the Art in Human Language Technology, 1996, Cambridge University Press, ISBN 0-521-59277-1, http://cslu.cse.ogi.edu/HLTSurvey/HLTSurvey.html

[FNR01] J. P. Norguet, *Mise en ligne d'informations relatives à l'analyse sémantique de la consultation de sites Web dont le contenu présente une évolution temporelle complexe et une mise en forme à la fois statique et dynamique*, Formulaire de candidature au mandat d'aspirant FNRS, 2001.

[FNR03] J. P. Norguet, *Mise en ligne d'informations relatives à l'analyse sémantique de la consultation de sites Web dont le contenu présente une évolution temporelle complexe et une mise en forme à la fois statique et dynamique*, Formulaire de candidature au renouvellement du mandat d'aspirant FNRS, 2003.

[Fra82] W. Francis, H. Kucera, *Frequency Analysis of English Usage*, New York, Houghton Mifflin, 1982.

[Fox92] C. Fox, *Lexical analysis and stoplists*, In W. B. Frakes and R. Baeza-Yates, editors, Information Retrieval: Data Structures & Algorithms, pages 102-130. Prentice Hall, Englewood Cliffs, NJ, USA, 1992, ISBN 0-13-463837-9.

[Fur88] G. W. Furnas et al., *Information retrieval using a singular value decomposition model of latent semantic structure*. In Proc. Of the 11th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 465-480, 1988.

[Gro02] C. Grover, E. Klein, A. Lascarides, M. Lapata, *XML-Based NLP for Analysing and Annotating Medical Language*, Proc. of the Second Workshop on NLP and XML, Coling, Taipei, 2002.

[Har92] D. Harman, *Relevance feedback and other query modification techniques*. In W. Frakes and R. Baeza-Yates, editors, Information Retrieval: Data Structures & Algorithms, pages 241-263. Prentice Hall, Englewood Cliffs, NJ, USA, 1992, ISBN 0-13-463837-9.

[IBM09] http://www.ibm.com/, site Web de la compagnie IBM.

[ISP03] http://www.gnu.org/software/ispell/ispell.html, site Web du logociel Ispell.

[JAK03] http://jakarta.apache.org/, site Web du projet Apache Jakarta.

[JAV03] http://java.sun.com/, site Web de Sun Microsystems dédié au langage et aux technologies Java.

[JDN02] *Dossier audience sur internet*, Journal du Net Solutions, mars 2002, http://solutions.journaldunet.com/0203/020311_bnpparibas.shtml.

[Lan00] E. Lanzi, P. Harpring, *Introduction to Vocabularies: Analyzing and Recording Information*, J. Paul Getty Trust, 2000.

[Luh57] H. P. Luhn, *A Statistical Approach to Mechanized Encoding and Searching of Literary Information*, 1957, IBM Journal of Research and Development.

[Mar93] J. G. March, H. A. Simon, H. S. Guetzkow, *Organizations*, 2nd edition, 1993, Cambridge Mass. Blackwell, ISBN 063118631X.

[Mat02] G. Materna, *Extraction par déformatage du contenu des pages Web dynamiques semi-structurées*, 2002, Travaux de Fin d'Etudes de la Faculté des Sciences Appliquées, Université Libre de Bruxelles, http://cs.ulb.ac.be/publications/MT-02-02.pdf.

[Med03] http://www.media-institute.com/html-fr/IG/glossaire.php3, glossaire du Media Institute.

[Met02] Meta Group, Enterprise Site Management: Gaining Insight Through Content Analysis, 2002, http://www.watchfire.com/resources/meta-group-brief.pdf.

[Mil73] G. A. Miller, Communication, language, and meaning; psychological perspectives, New York : Basic Books, 1973, ISBN 0465012833.

[Moo65] G. Moore, *Cramming more components onto integrated circuits*, volume 38, number 8 (19 April 1965), ftp://download.intel.com/research/silicon/moorespaper.pdf.

[Net03] http://news.netcraft.com/archives/web_server_survey.html, Netcraft web server survey, May 2003.

[Nor98] J. P. Norguet, *Mise en lignes d'informations statistiques relatives à des serveurs Web*, Travail de Fin d'Etudes de la Faculté des Sciences Appliquées, Université Libre de Bruxelles, en collaboration avec l'Ecole Centrale Paris, 1998.

[Nor02] U. Wahli, A. Matthews, P. C. Lapido, J. P. Norguet, *WebSphere Version 4 Application Development Handbook*, 2002, Prentice Hall, ISBN 0-13-009225-8.

[Nor03] J. P. Norguet, *Java FTP Client Libraries Reviewed*, April 2003, JavaWorld Magazine, http://www.javaworld.com/javaworld/jw-04-2003/jw-0404-ftp_p.html.

[Pos85] J. Postel, J. Reynolds, *File Transfer Protocol (FTP), Request for Comments: 959*, IETF Network Working Group, 1985.

[Qiu93] Y. Qiu, H. P. Frei, *Concept based query expansion*. In Proc. of the 16th ACM SIGIR Conference on Research and Development in Information Retrieval, pages 160-169, Pittsburgh, PA, USA, 1993.

[Ras92] E. Rasmussen, *Clustering Algorithms*. In W. B. Frakes and R. Baeza-Yates, editors, Information Retrieval: Data Structures & Algorithms, pages 419-442. Prentice Hall, Englewood Cliffs, NJ, USA, 1992, ISBN 0-13-463837-9.

[Rob02] P. Robert, *Le petit Robert 1, dictionnaire alphabétique et analogique de la langue française*, 2002.

[Sal83] G. Salton, M. McGill, *Modern Information Retrieval*, 1983, New York, McGraw-Hill.

[Sch02] G. Schreiber, *Convergence of Web Services, Grid Services and the Semantic Web for delivering e-Services*, 2002, Proceedings of the Diffuse Final Conference, Brussels, 2002, http://www.diffuse.org/event3.html.

[Sri92] P. Srinivisdan, *Thesaurus Construction*, In W. B. Frakes and R. Baeza-Yates, editors, Information Retrieval: Data Structures & Algorithms, pages 161-218. Prentice Hall, Englewood Cliffs, NJ, USA, 1992, ISBN 0-13-463837-9.

[Tré98] F. Trézal-Mauroz, *Etude de marché : desideratas en matière d'audience internet pour les sites renault.com/.fr*, Renault, 1998.

[Tuo02] I. Tuomi, *The Lives and Death of Moore's Law*, 2002, http://www.firstmonday.dk/issues/issue7_11/tuomi/

[ULB03] http://www.ulb.ac.be/, site Web de l'Université Libre de Bruxelles.

[Van75] C. J. Van Rijsbergen, *Information Retrieval*, 1975, London, Butterworths.

[W3C03] http://www.w3c.org/, site Web du consortium W3.

[Wah00] U. Wahli, M. Fielding, G. Mackown, D. Shaddon, G. Hekkenberg, *Servlet and JSP Programming*, 2000, IBM Redbooks, ISBN 0-7384-1608-8.

[Wil98] L. Will, *Thesaurus principles and practice*, Museum Documentation Association, 1998.

[WOR03] http://www.cogsci.princeton.edu/~wn/, WordNet, a lexical database for the English language.

[Xu96] J. Xu, W. B. Croft, *Query expansion using local and global document analysis. In Proc. ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 4-11, Zurich, Switzerland, 1996.

[Zha02] J. Zhang, J. Mostafa, H. Tripathy, *Information Retrieval by Semantic Analysis and Visualization of the Concept Space of D-Lib Magazine*, 2002, D-Lib Magazine, http://www.dlib.org/dlib/october02/zhang/10zhang.html.

[Zim97] E. Zimányi, C. Parent, S. Spaccapietra, A. Pirotte, *TERC+: A temporal conceptual model*, In Proc. Int. Symp. on Digital Media Information Base, DMIB'97, Nara, Japon, November 1997.

Annexes

A. Liste de stopwords

Voici une liste de stopwords de la langue anglaise. Cette liste est issue de [Fox92]. Pour plus d'informations sur l'utilisation des stopwords, consulter la section 3.5.3.

a, about, above, across, after, again, against, all, almost, alone, along, already, also, although, always, among, an, and, another, any, anybody, anyone, anything, anywhere, are, area, areas, around, as, ask, asked, asking, asks, at, away, b, back, backed, backing, backs, be, became, because, become, becomes, been, before, began, behind, being, beings, best, better, between, big, both, but, by, c, came, can, cannot, case, cases, certain, certainly, clear, clearly, come, could, d, did, differ, different, differently, do, does, done, down, downed, downing, downs, during, e, each, early, either, end, ended, ending, ends, enough, even, evenly, ever, every, everybody, everyone, everything, everywhere, f, face, faces, fact, facts, far, felt, few, find, finds, first, for, four, from, full, fully, further, furthered, furthering, furthers, g, gave, general, generally, get, gets, give, given, gives, go, going, good, goods, got, great, greater, greatest, group, grouped, grouping, groups, h, had, has, have, having, he, her, here, herself, high, higher, highest, him, himself, his, how, however, I, i, if, important, in, interest, interested, interesting, interests, into, is, it, its, itself, j, just, k, keep, keeps, kind, knew, know, known, knows, large, largely, last, later, latest, least, less, let, lets, like, likely, long, longer, longest, m, made, make, making, man, many, may, me, member, members, men, might, more, most, mostly, mr, mrs, much, must, my, myself, n, necessary, need, needed, needing, needs, never, new, newer, newest, next, no, nobody, non, noone, not, nothing, now, nowhere, number, numbered, numbering, numbers, o, of, off, often, old, older, oldest, on, once, one, only, open, opened, opening, opens, or, order, ordered, ordering, orders, other, others, our, out, over, p, part, parted, parting, parts, per, perhaps, place, places, point, pointed, pointing, points, possible, present, presented, presenting, presents, problem, problems, put, puts, q, quite, r, rather, really, right, room, rooms, s,

said, same, saw, say, says, second, seconds, see, seem, seemed, seeming, seems, sees, several, shall, she, should, show, showed, showing, shows, side, sides, since, small, smaller, smallest, so, some, somebody, someone, something, somewhere, state, states, still, such, sure, t, take, taken, than, that, the, their, them, then, there, therefore, these, they, thing, things, think, thinks, this, those, though, thought, thoughts, three, through, thus, to, today, together, too, took, toward, turn, turned, turning, turns, two, u, under, until, up, upon, us, use, used, uses, v, very, w, want, wanted, wanting, wants, was, way, ways, we, well, wells, went, were, what, when, where, whether, which, while, who, whole, whose, why, will, with, within, without, work, worked, working, works, would, x, y, year, years, yet, you, young, younger, youngest, your, yours, z.

B. Configurations matérielle et logicielle

Voici la liste des ordinateurs et logiciels utilisés pour le développement du prototype WASA décrit en section 4.

- PC de développement :
  - Nom : bellini6.ulb.ac.be (alias wasa-dev.ulb.ac.be)
  - Adresse IP : 164.15.78.26
  - Processeur : Pentium IV 1.5 GHz
  - Mémoire RAM : 1 Go
  - Capacité de stockage : 2 x 60 Go
  - Système d'exploitation : Microsoft Windows 2000
  - Logiciels :
    - IBM Visual Age for Java 4.0
    - UltraEdit 7.0
    - Visio Professional 5.0
    - Together 5.5 ControlCenter
- PC serveur :
  - Nom : iseult2.ulb.ac.be (alias wasa-demo.ulb.ac.be, alias wasa.ulb.ac.be)
  - Adresse IP : 164.15.78.28
  - Processeur : Pentium II 400 MHz
  - Mémoire RAM : 128 Mo
  - Capacité de stockage : 10 Go
  - Système d'exploitation : Linux 2.4.? Debian 2.2

    - Logiciels :
        - MySQL server 3.22 : bases de données
        - Apache server 1.3.24 : http://wasa.ulb.ac.be
        - ProFTPd server : ftp://iseult2.ulb.ac.be
        - IBM JDK 1.3
- PC home office :
    - Nom : ys
    - Adresse IP : 192.168.0.29 (intranet)
- Processeur : Pentium MMX 200 MHz
    - Mémoire RAM : 256 Mo
    - Capacité de stockage : 12 Go
    - Système d'exploitation : Microsoft Windows 98
    - Logiciels :
        - IBM Visual Age for Java 3.5 (4.0 dès que possible)
        - IBM WebSphere Studio 3.5
        - Visio Professional 5.0
- Serveur Sun Enterprise :
    - Nom : informa1.ulb.ac.be (alias eao1(eoa1?).ulb.ac.be)
    - Adresse IP : 164.15.78.109
    - Processeur : Sun SPARC
    - Mémoire RAM :
    - Capacité de stockage : 20 Go
    - Système d'exploitation : Sun Solaris
    - Logiciels :
        - Oracle 8.1.7.1
- Serveur web de l'ULB :
    - Nom : resu1.ulb.ac.be (alias www.ulb.ac.be)
    - Adresse IP : 164.15.59.200
    - Processeur : Sun
    - Mémoire RAM :
    - Capacité de stockage : 100 Go
    - Système d'exploitation : Solaris
    - Logiciels :
        - Apache Server 1.3.14