Université Paris 7
Institut Gaspard Monge

Un modèle hypertexte de traitement de langues naturelles

Pierre-Yves FOUCOU

Thèse de doctorat d'informatique fondamentale

Directeur: Maurice Gross
Novembre 1996

Jury : Maurice Gross
Gaston Gross
Éric Laporte
Dominique Perrin
Jean-Marie Rifflet

Un modèle hypertexte de traitement de langues naturelles

Résumé : La langue naturelle apparaît comme une somme de micro-problèmes dont la description systématique conduit à des ordres de grandeur difficilement manipulables par les moyens informatiques usuels.

Nous présentons un modèle de traitement des langues naturelles basé sur l'initiative World Wide Web. Nous intégrons dans un environnement d'expérimentation homogène, les divers aspects de la maintenance des données linguistiques : dictionnaires électroniques, descriptions syntaxiques et corpus.

La formalisation des nombreuses opérations possibles à partir des outils de traitements automatiques existants permet d'optimiser les requêtes et ouvre des perspectives pour la validation manuelle des résultats.

Mots-clés: lexique-grammaire, dictionnaires électroniques, corpus, traitements automatiques, W_3, hypertexte, Internet, service, adressage,

Hypertext model of natural language processing

Abstract : Natural language can be considered as a collection of many small problems. The systematic studies lead to large amount of data, hard to handle with common computers.

We present a model of natural language processing, based on World Wide Web initiative. We integrate in a homogeneous environment, various aspects of linguistic resources management : electronic dictionaries, syntactic structures, and corpora. Formal description of requests makes optimization possible and encourages manual validation of results.

Keywords: lexicon-grammar, electronic dictionaries, corpus, natural language processing, W_3, hypertext, Internet, service, adressing scheme, collaborative work

Table des matières

Présentation

Le champ de la recherche sur la langue s'oriente de plus en plus vers une description détaillée des données. La langue naturelle apparaît comme une somme de micro-problèmes dont la description, lorsqu'elle est envisagée, donne lieu à des factorisations plus ou moins pertinentes, implicites, intuitives, généralisables ...

La description systématique de la langue, sur des bases syntaxiques, engagée depuis un quart de siècle au Laboratoire d'Automatique Documentaire et Linguistique, a produit une quantité d'informations considérable: les dictionnaires électroniques pour le français de 90,000 mots simples et 100,000 mots composés (formes canoniques).

Au niveau syntaxique, 5,000 verbes simples, soit 12,000 emplois, sont répertoriés dans une cinquantaine de tables où sont décrites les différentes structures de phrases simples dans lesquelles ils sont acceptés. Des listes de verbes supports de noms prédicatifs, d'adjectifs, d'adverbes, de phrases figées augmentent ce lexique-grammaire du français, et servent de base à des travaux sur d'autres langues.

La complexité et la diversité des phénomènes étudiés conduisent à des ordres de grandeur difficilement manipulables sur des machines courantes, par des applications plus ou moins automatiques coûteuses en développements et en maintenance.

Nécessité d'expérimentation

Pour fournir un environnement cohérent d'expérimentation des traitements automatiques de la langue naturelle, nous présentons un système hypertexte, basé sur W_3[1], offrant une facilité d'accès et une souplesse de représentation, tout en permettant une structuration très forte de données hétérogènes.

Une structure hypertexte permet d'associer ces *ressources* logiques et leur description explicite textuelle ou graphique, facilitant une consultation sélective des données. Selon ses besoins, le lecteur *active* les liens hypertextes que ce soit sur un document précis ou sur une requête de recherche dynamique.

L'unification des accès aux moyens informatiques mis en oeuvre pour ces traitements complexes est un des points essentiels qui pousse à s'investir dans l'initiative W_3. Contrairement aux environnements constructeurs auto-limités, peu

1. World Wide Web

documentés et onéreux, l'essor de l'initiative W_3 engendre des collaborations de domaines habituellement sans relation.

L'intense activité développée autour du *Consortium* W_3 et de la *communauté Web* où la normalisation est souvent menée de front avec le développement de logiciels ou de librairies permettant de profiter rapidement des dernières évolutions.

W_3 : quelques bases

Un utilisateur peut accéder aux informations via une unique application (client W_3), indépendamment des contingences matérielles ou logicielles qui rendent habituellement leur distribution et leur exploitation difficiles. La notion de fichier, propre aux systèmes informatiques, laissant la place à celle de document, destiné aux utilisateurs humains.

Les documents sont rédigés selon la norme SGML, plus précisément HTML, qui sert de format pivot pour les échanges. La mise en valeur typographique du contenu et les liens hypertextes seront rendus de façon *équivalente* quels que soient le réseau, le système ou les logiciels employés ... Un mécanisme d'adressage *universel* identifie le site (machine serveur) et le protocole à utiliser pour la consultation d'une ressource.

Rédaction

L'hypertexte, et W_3 en particulier, constitue un outil indispensable dans un domaine où la complexité des interactions entre les structures théoriques employées, rend problématique toute normalisation en aval. Il offre un cadre de présentation extrêmement tolérant aux approximations des spécifications manuelles et favorise un accès raisonné aux outils de traitement automatique. Il ne s'agit pas simplement d'appliquer des convertisseurs sur des données existantes en constatant les limites de W_3, mais de les adapter pour tirer bénéfice des caractéristiques d'un système documentaire: normaliser les informations collectées, formaliser les interdépendances, offrir des vues précises, croisées, composites. Cela passe bien sûr par le développement de fonctions de manipulation et de présentation des informations linguistiques.

Cette base théorique, première étape du traitement des langues naturelles, doit être étendue à des unités plus difficilement cernables comme les mots dérivés, les noms propres, les sigles ou abréviations, pour lesquels l'utilisation de corpus est très instructive. La mise à disposition de textes en quantité significative sera aussi facilitée par W_3.

Organisation d'un serveur

L'organisation pragmatique des moyens disponibles permet à un utilisateur depuis son ordinateur d'accéder une puissance de calcul et des espaces de stockage des quelques machines Unix sur lesquelles nous avons organisé les différents

services. Ces moyens sont limités en regard des performances des machines haut de gamme actuelles, mais suffisent largement si on répartit leur charge de façon pertinente.

L'intérêt de l'hypertexte est de pouvoir expliciter les nombreux liens entre les ressources, de façon totalement transparente pour l'utilisateur. La définition du système d'adressage sera la partie la plus délicate de la gestion des serveurs. L'interdépendance très forte des ressources et la relation directe texte/référence impliquent des méthodes de développement adaptées, entre rédaction et programmation.

Production automatique

Interfacer des traitements automatiques sous W_3 oblige en à reconsidérer le processus complet et permet de définir formellement les différentes séquences opérations.

Un système d'analyse s'appuie sur des dictionnaires utilisant des codes flexionnels, syntaxiques ou sémantiques, des grammaires à portée locale ou générale. La production d'un hypertexte résultat permet par exemple de relier un processus de levée d'ambiguïté au texte source dans le corpus ainsi qu'à une description de la portée des règles employées. Ce résultat sera alors accessible, évitant de relancer l'analyse à chaque fois que l'on cherche à évaluer la pertinence d'une règle sur tel ou tel texte.

W_3 généralise toutes les opérations de transfert de données maintenant usuelles sur Internet (mail, ftp, gopher, news, ou wais), ce qui ouvre des possibilités de maintenance coopérative répartie. La possibilité pour l'utilisateur de dériver ses propres documents à partir des informations parcourues, permet un développement cumulatif du *système d'informations*.

Veille technologique

La vitesse de développement des diverses normes, logiciels ou librairies concernant W_3 rend problématique tout positionnement. Le lecteur voudra bien considérer les informations techniques (numéro de version, couverture d'une norme) comme largement sujettes à caution à brève échéance. Cependant les concepts clés sur lesquels nous nous appuyons resteront à coup sûr l'intégration de l'existant, l'application tolérante des normes et la simplicité d'utilisation.

Jusqu'à présent, il n'y a pas eu à déplorer de schisme ou d'anarchie notable, même si une dérive vers le tout multimédia induira et peut-être induit déjà un coût matériel de connexion assez important. Même si les aspects médiatiques procèdent un peu de la confusion des genres, les progrès accomplis par la communauté W_3 rendront l'utilisation des informations que nous présentons de plus en plus facile. L'exemple le plus parlant sera donné Figure 1.4. Les possibilités de formatage d'une table du lexique-grammaire directement en HTML, sont aujourd'hui utilisables, ce qui n'était pas évident il y a six mois.

Représentation papier d'hypertexte

Il est clair que pour présenter les arguments qui plaident en faveur d'un développement d'hypertexte sous W_3, une présentation papier n'est pas des plus convaincantes. La facilité d'accès et d'utilisation pour un utilisateur λ est difficilement montrable! Il est malgré tout probable que le lecteur en aura déjà fait l'expérience par lui même.

L'avantage décisif de l'hypertexte est de favoriser les références plutôt que la recopie. Dans ce texte, nous avons donc cherché un équilibre entre les adresses, qui seront mentionnés en bas de page, et les copies écran de documents quand leur lisibilité permettra d'illustrer notre travail.

Les liens mentionnés concerneront des documents de référence incontournables pour le suivi de la normalisation et des spécifications techniques, ou l'approfondissement d'un point abordé dans le texte. Ces informations n'ont pas encore le crédit des références bibliographiques usuelles, mais leur pérennité a fait l'objet de vérifications particulières (voir Annexe C).

Exemples

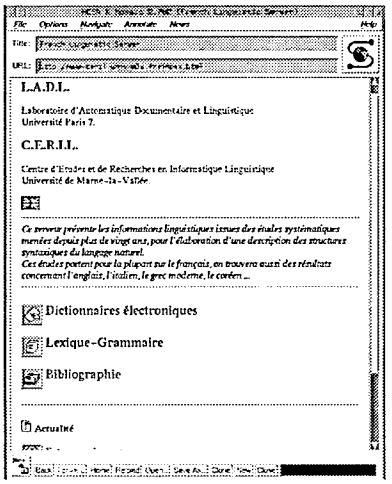


FIG. 0.1 – *Le serveur LADL-CERIL via Mosaic*

Les quelques exemples suivants illustrent la diversité des types documents accessibles:

http://www-ceril.univ-mlv.fr/LexG/f21 : une table du verbe support *faire* ([GS78]).
http://www-ceril.univ-mlv.fr/LexG/4 : les entrées de la table 4 ([Gro75]).
http://www-ceril/Delaf5.html : recherche d'un mot dans le DELAF (version 5).
http://www-ceril/Texte/DELAF/N:ms : recherche des noms masculins singuliers dans un texte.
http://www-ceril/GrammaireLocale/hata.gif : une grammaire locale du coréen.
http://www-ceril/Bibliographie/ : des travaux sur le lexique grammaire.

Chapitre 1

Hypertexte et système d'information

Nous présentons ici les concepts de base de W_3 qui guideront nos choix pour la présentation des données linguistiques.

Les spécifications techniques plus détaillées sont accessibles via le serveur du `consortium`[1] W_3.

1.1 Principes généraux de W_3

Le projet W_3 a été lancé au `CERN`[2] en 1991 ([BLa93]), puis repris et développé par de nombreuses institutions universitaires ou commerciales, dans le but de faciliter l'interconnexion de systèmes documentaires par réseaux informatiques. Il s'agissait principalement d'intégrer un grand nombre d'informations, souvent techniques, disponibles sur Internet. Ce domaine, connu sous le sigle de WAIS (Wide Area Information Systems), nécessitait auparavant une solide connaissance en informatique pour espérer extraire la moindre information valable. Le nom, comme le nombre des opérations nécessaires à l'identification, la recherche, le rapatriement puis le traitement d'une information présente sur le réseau, en ont découragé plus d'un.

Pour mettre de l'ordre dans tout cela, l'idée a été de doter les informations d'un identificateur universel clair (URL) et de les relier entre elles de façon lisible dans une structure hypertexte (`HTML`). W_3 s'appuie sur des normes existantes largement répandues. *Internet* pour les protocoles d'échanges de données sur un réseau mondial et SGML (Standard Generalized Markup Language) pour la structuration des documents

Une coopération mondiale a donc pu très vite se développer pour mettre dans le domaine public des outils d'administration d'informations (serveurs générateurs, convertisseurs, indexeurs, ...) et des logiciels de consultation robustes

1. URL: http://www.w3.org/
2. URL: http://www.cern.ch/

(clients W_3).

A l'heure actuelle, des clients W_3 fonctionnent sur la plupart des systèmes d'exploitation: PC sous Windows ou non, MacIntosh, station Unix, NeXT ... Il est bien sûr préférable d'être connecté à un réseau, les hypertextes HTML sur disques locaux ayant un intérêt bien limité.

1.2 Structure des documents hypertexte

L'hypertexte apparaît comme l'interface la plus conviviale pour explorer efficacement de grandes quantités de données peu structurées, notamment en autorisant plusieurs vues concurrentes des mêmes objets.

> Un **lien hypertexte** est constitué d'un point d'ancrage visible dans le document courant et d'une référence à un document associé.
>
> Les **points d'ancrage** sont facilement identifiables, généralement mis en valeur typographiquement (gras ou soulignés) par le logiciel de visualisation.

Les logiciels existants comme *HyperCard, Rtfd, Framemaker, KMS* ... ([BD92a]) s'avérant trop lourds ou trop coûteux, en tout cas peu adapté et peu `standard`[3], l'accent a été mis sur la facilité d'accès et de développement. Une interface homogène, le client W_3, permet de consulter des **documents**, c'est à dire des données compréhensibles par un utilisateur humain : textes, images, son ...

La qualité d'un document hypertexte s'évalue en fonction des choix de parcours fournis au lecteur à chaque étape. La sémantique des liens est a priori libre. Par exemple, on peut relier chaque section d'un document à la table des matières, ou encore chaque mot d'un index à sa position de le texte, organisant ainsi différentes types de lectures non linéaires. Des extensions plus complexes permettent de relier une référence bibliographique à un document électronique stocké quelque part ailleurs sur le réseau, ou encore de lier un bulletin météo à une image satellite mise à jour périodiquement.

Toutes ces caractéristiques de la rédaction hypertexte seront abordées par la suite, il convient de relever tout de même qu'il s'agit de rédaction, et comme *cliquer* a peu de synonymes ou de paraphrases, on évitera l'écueil classique (Figure 1.1) qui agace inévitablement n'importe quel lecteur.

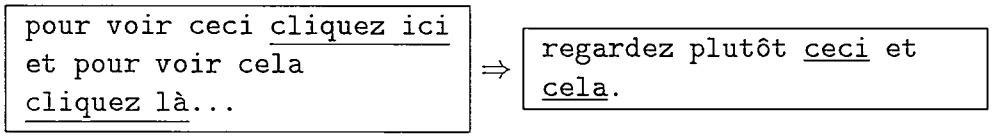


FIG. 1.1 – *Rédiger n'est pas cliquer*

3. URL: http://www.w3.org/hypertext/Standards/Overview.html

L'organisation d'un hypertexte doit en tout état de cause préserver un certain équilibre entre les différentes parties. Chaque point méritant d'être explicité doit renvoyer à un autre document pour conserver des dimensions acceptables, en général de l'ordre d'une page écran. L'explicitation d'un terme très complexe pourra ainsi nécessiter plusieurs niveaux de documentation.

L'organisation interne et la localisation géographique du système documentaire doit être transparente pour l'utilisateur, cependant la lecture se faisant par transfert d'informations sur le réseau, la taille des documents influe sur le temps de transmission et donc sur la patience du lecteur!

1.2.1 HTML

Le marquage de la structure des documents se fait selon la norme SGML[4] . Les différents types de balisage sont illustrés Figure 1.2. Ce marquage un peu verbeux a l'énorme avantage de rester très lisible quels que soient les avatars infligés par les moyens informatiques usuels.

Une DTD (Définition de Type de Document) SGML normalise les balises, ou tags, utilisables dans un document HTML.

```
<aTAG> text entre une balise ouvrante et une fermante ... </aTAG>
<aTAGwithAttr attribut="valeur"> ...contenu... </aTAGwithAttr>
<anEmptyTAG>
<aNonEmptyTAG> marquage fermé par un autre tag ouvrant <anotherTAG>
\&anEntity;
```

FIG. 1.2 – *Syntaxe de marquage SGML*

Le marquage HTML[5] est suffisamment raffiné, une cinquantaine de balises environ, pour formater des documents contenant du texte, des images, des liens, ...

Parmi les plus utiles, on peut baliser:

Structure logique : `<TITLE>`, `<Hn>`, `<UL>`, `<OL>`, `<DL>`

Structure physique : `<P>`, `<BR>`, `<HR>`, `<IMG SRC=...>`

Effets typographiques : `<B>`, `<I>`, `<TT>`

Sémantique : `<ADDRESS>`, `<CITE>`, `<STRONG>`

Structure hypertexte : `<A NAME="refinterne" HREF="http://...">`

Fonctions Interactives : `<FORM>`, `<ISINDEX>`, `<ISMAP>`

4. URL: http://www.sil.org/sgml/sgml.html
5. URL: http://www.utoronto.ca/webdocs/HTMLdocs/NewHTML/htmlindex.html

Mise en page complexe : `<TABLE>, <FRAME>`

Commentaire : `<!--` des commentaires `-->`

Exemple : La Figure 1.3 montre un marquage possible d'une forme fléchie issue du DELAF[6] , sa représentation formatée est donnée Figure 2.6.

```
   <H2>Résultat de la consultation du
        <A HREF="/Dictionnaires/Delaf5.html">DELAF v5</A>
   </H2>
<HR>
  <H3>[élargis]</H3>
<DL>
  <DT>élargir <DD> Verbe  Participe passé,  masculin pluriel
  <DT>élargir <DD> Verbe  Indicatif Présent,  1ère personne du singulier
  <DT>élargir <DD> Verbe  Indicatif Présent,  2ième personne du singulier
  <DT>élargir <DD> Verbe  Indicatif Passé simple,  1ère personne du singulier
  <DT>élargir <DD> Verbe  Indicatif Passé simple,  2ième personne du singulier
  <DT>élargir <DD> Verbe  Impératif Présent,  2ième personne du singulier
</DL>
```

FIG. 1.3 – *Présentation d'une forme fléchie*

Les balises réservées pour la définition de menus ou de champs de saisie seront décrites Section 3.2.
Les effets typographiques complexes du type indice ou exposant ont été ajoutés dans la version 3.2 de HTML (mai 1996), qui incorpore aussi des fonctionnalités de programmation développées par Netscape[7] (`<SCRIPT>, <APPLET>`).

Le codage des caractères est spécifié par la norme ISO-8859. Ce codage a été adopté comme standard par Microsoft ou Xwindow, mais, par exemple, pas par NeXT[8] . En tout état de cause, les clients W_3 sur d'autres systèmes seront capables d'afficher correctement les codes correspondants. C'est par exemple le cas d' OmniWeb[9] sous NeXTStep.

De plus la définition d'entités permet un codage 7-bits en solution de repli pour la rédaction, au détriment de la lisibilité des documents sources.
Exemple: è = `&agrave;` é = `&eacute;` è = `&egrave;` ...
donc déjà = `d&eacute;j&agrave;`
La liste complète des entités fait partie de la norme HTML.

La partie 1 de la norme ISO-8859, souvent nommée *ISO-latin-1*, concerne les langues "occidentales", donc le français. A noter que les diagrammes soudés œ et Œ ainsi que les caractères ÿ et Ÿ ne sont pas définis.

6. URL: http://jobim.univ-mlv.fr/dyn/delaf5.pl/élargis
7. URL: http://www.netscape.com/
8. URL: http://www.next.com/
9. URL: http://www.omnigroup.com/Software/OmniWeb

Pour dépasser ces limites, l'intégration des travaux de normalisation `Unicode`[10] est d'ores et déjà prévue.

1.2.2 Rendu

W_3 est largement orienté vers la consultation. De plus en plus de logiciels permettant la rédaction émergent, mais à l'heure actuelle aucun n'est complètement satisfaisant, peut-être parce que chaque domaine a des besoins spécifiques.

Un document HTML est principalement destiné à être visualisé sur écran dans un hypertexte, donc pas à être imprimé!

Lors de la rédaction, les contraintes de formatage doivent être relâchées puisque l'affichage des différentes balises n'est pas strictement normalisé. Comme le montre la Figure 1.4, il s'agit plus de directives incitant au bon sens que d'obligations à respecter précisément. La normalisation évoluant sans cesse, les visualiseurs un peu datés peuvent être complètement inutiles si les rédacteurs suivent le dernier cri du formatage.

La création de documents est souvent faite par conversion depuis des formats standards comme RTF, LaTeX ou simplement ASCII. Une connaissance minimale des principes (simples) de HTML est malgré tout requise pour éviter les pertes d'informations.

1.2.3 Autres types de contenu

La norme MIME (Multipurpose Internet Message Extension `RFC 1521`[11]) décrit les conventions permettant de spécifier les associations entre les formats de documents, leurs dénominations habituelles et les applications susceptibles de les gérer.

On distingue 5 types de base: `text`, `image`, `audio`, `video`, `application`.

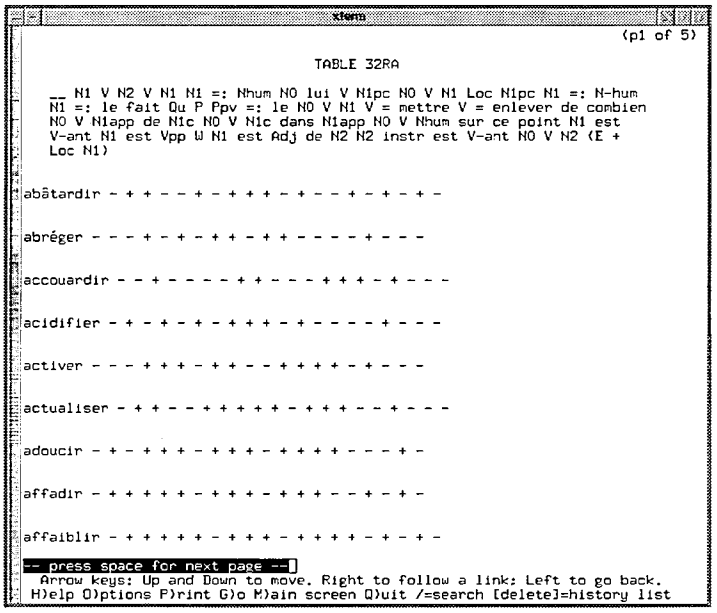
Existent aussi les types `multipart` et `message`, qui concernent surtout les systèmes de messagerie électronique.

On précise ensuite le sous-type parmi les formats universellement répandus (voir Table 1.1).

Un client W_3 n'a bien entendu pas vocation à gérer tous les types de formats possibles et imaginables, il fera donc appel à un logiciel externe pour traiter les données transmises, ou proposera une sauvegarde sur disque.

Des conventions privées d'appellation, i.e. non enregistrées par l'IANA[12]. peuvent être utilisées en préfixant le type souhaité par `X-`.
Cela nous servira par exemple à typer `x-graph` les automates représentant des grammaires locales (voir Figure 4.3) .

10. URL: http://www.stonehand.com/unicode/glosscnt.html
11. URL: ftp://ftp.ibp.fr/pub/rfc/rfc/rfc1521.txt
12. voir Appendix A.



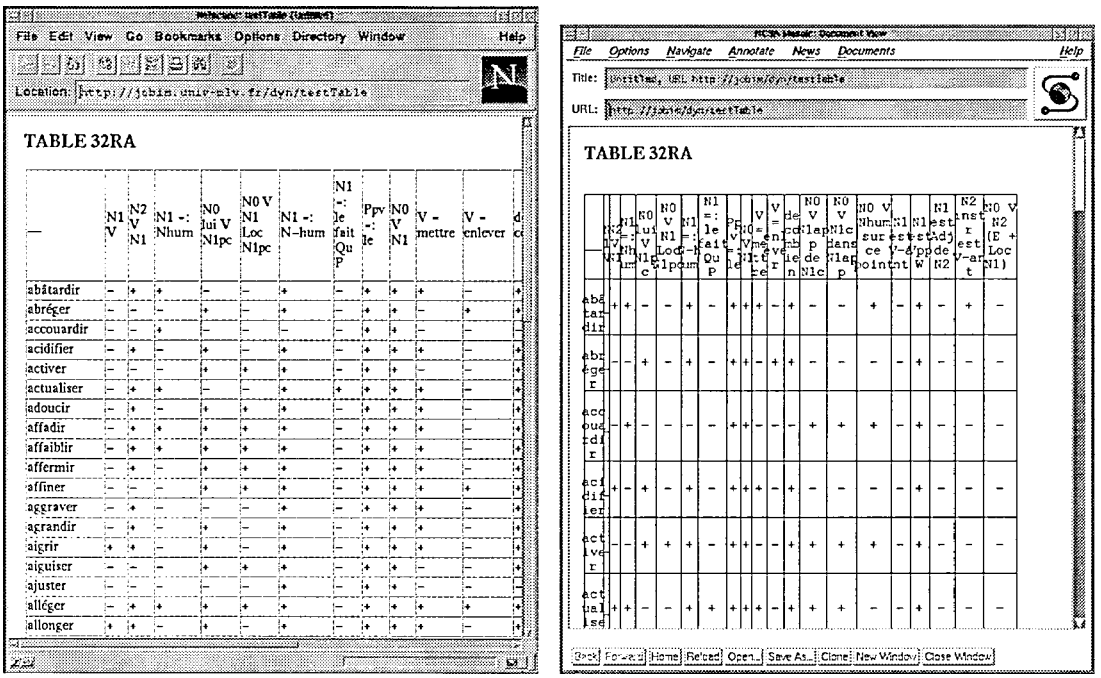


FIG. 1.4 – *Une table du lexique-grammaire vue sous OmniWeb, Netscape ou Lynx*

Content-Type	Suffixes usuels	Application associée
text/plain	txt	Editeur
text/html	html shtml	Client W_3
text/x-dvi	dvi	TEX output viewer
application/pdf	pdf	Adobe formats
application/postscript	eps ps	...
application/x-tar	tar	archiveur
application/zip	zip	compacteur
application/gzip	gz	...
audio/basic	au snd	
audio/x-wav	wav	
image/gif	gif	interne ou
image/jpeg	jpeg jpg	image viewer
image/tiff	tiff tif	...
video/mpeg		

TAB. 1.1 – *Quelques types MIME*

1.3 Mécanismes de transfert d'informations

1.3.1 Modèle client/serveur

La machine serveur regroupe et structure les informations pour offrir les ressources demandées. L'organisation, l'implémentation et la maintenance du système de ressources sont entièrement à la charge du serveur. Ces aspects seront développés Section 3.3.

Le comportement du logiciel client W_3 est schématisé Figure 1.5. Il répète les 3 étapes suivantes :

1. Initialisation de la connexion vers le serveur selon le protocole choisi,
2. Récupération du document pour affichage,
3. Fin de connexion.

Dans un document HTML, certaines balises contiennent les URL d'images incluses (`<IMG>`) ou d'images de fond (`<BODY BACKGROUND>`). Pour un formatage complet, le client doit réïtérer les 3 étapes ci dessus pour chaque ressource à inclure.

L'avantage est que cette distinction étant faite, on peut choisir d'activer l'inclusion des images seulement si elles sont essentielles au document. Les images ayant une taille 10 à 100 fois supérieure à celle des documents textuelles, ce niveau de lecture accélérera considérablement la navigation.

La version 3.2 d'HTML a étendu cette inclusion des ressources pour les sous-documents (`FRAME`) et les programmes `Java`[13] (`APPLET`).

13. URL: http://www.javasoft.com/

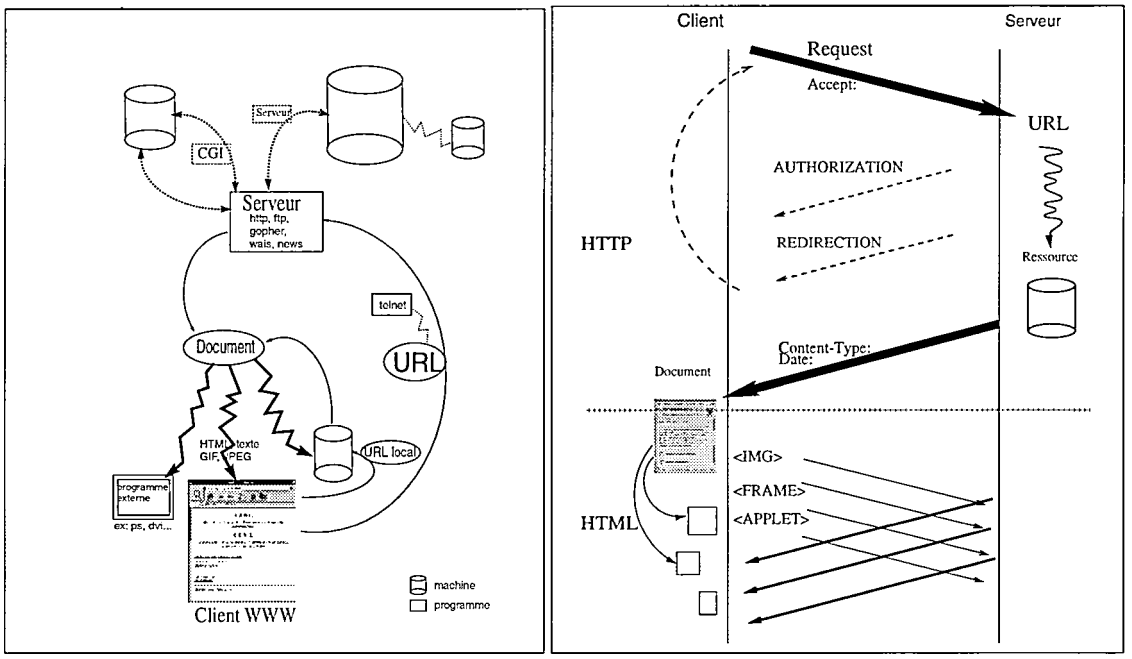


FIG. 1.5 – *Requêtes induites par un document*

W_3 intègre la plupart des protocoles d'échanges de données d'Internet (voir Table 1.3), et définit un protocole plus spécifiquement adapté: HTTP.

W_3 privilégie des échanges en mode non connecté, ce qui réduit les possibilités d'erreurs et simplifie la mise en oeuvre (voir Section 3.1). La tendance actuelle est d'essayer de lever les contraintes des services **sans état** pour limiter les opérations de connexion/déconnexion dans les requêtes répétitives.

Au niveau du protocole HTTP[14] , parmi les méthodes initialement prévues dans la norme, seules GET et POST sont utilisées (voir annexe D). Beaucoup des fonctions normalisées ne sont pas respectées par la majorité des agents :

- Le champ `Accept`, limitant les types de documents souhaités par les clients, n'est souvent pas pris en compte. Le lecteur averti pourra tester une requête avec `image/gif` ou `text/plain` seul.
- L'algorithme de négociation des paramètres de qualité de transfert, initialement prévu pour la gestion de document multimédia, était trop abstrait pour avoir donné lieu à des implémentations réelles.

HTTP[15] devrait aussi évoluer pour renforcer les capacités d'identification et d'authentification dans tout ce qui concerne la facturation des services. Par exemple Netscape utilise un schéma d'adressage `https://` pour identifier le protocole *sécurisé*.

14. URL: http://www.w3.org/hypertext/WWW/Protocols/HTTP/References.html
15. URL: http://www.w3.org/pub/WWW/Protocols/HTTP-NG/

1.3.2 Adresse des *ressources*

Un (ou une) URL (Uniform Resource Location) permet de spécifier:
le document, le serveur qui le fournit et le protocole pour le transférer. La figure 1.6 décrit schématiquement la syntaxe des URL.

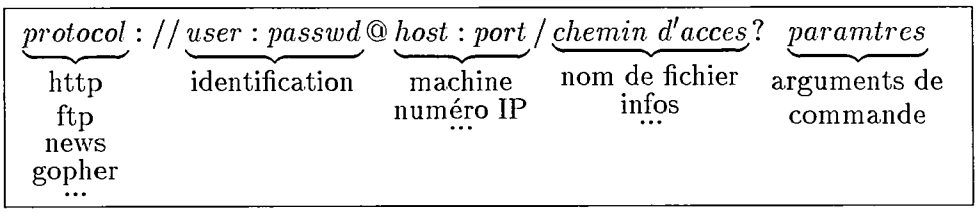


FIG. 1.6 – *URL = chaine de caractères identifiant une ressource*

Cette approximation des concepts d'adressage sous W_3 est suffisante pour l'utilisation qui nous intéresse, nous laisserons de côté les discussions sur la généralisation de l'adressage par `URN`, `URI`, `URC` ...[16] , respectivement pour nom, identificateur et citation de documents.

A noter simplement que selon le protocole utilisé, le schéma d'adressage change. Par exemple, une URL `mailto:` ne contient pas de chemin d'accès ni de paramètres.

Dans le schéma d'adressage `http:`, l'interprétation de la partie *fichier* est totalement opaque au niveau du serveur, contrairement au schéma `ftp:` où elle reflète la structure hiérarchique des répertoires.

Les constructions d'URL normalisées à partir de documents HTML seront présentées plus en détail Section 3.2.

1.3.3 Internet, W_3, Web ...

Internet trouve ses origines dans `ARPANET`, réseau expérimental américain créé au début des années 1970 pour interconnecter des centres de recherche de l'Advanced Research Projects Agency (ARPA). Aujourd'hui, `Internet`[17] est un curieux mélange d'organisations gouvernementales, d'entreprises, d'associations à but plus ou moins ouvertement lucratif, chargées des divers aspects de l'infrastructure et de l'administration du réseau: NSFNet (National Science Fundation), MILNET (pour la partie militaire) ...

Internet recouvre un ensemble de protocoles, de niveaux divers. Le protocole de base[18]est tout simplement `Internet Protocol`, souvent abrégé `IP`. C'est ce niveau qui définit la partie *host* des URL (Figure 1.6). Chaque machine connectée par `Internet` est identifiée par un numéro. Ce numéro est formé de 4 champs,

16. URL: http://www.w3.org/pub/WWW/Addressing/#terms
17. URL: http://www.isoc.org/home.html
18. au niveau réseau du célèbrissime modèle ISO/OSI

Symbole	Domaine	Administration
COM	US Commercial	Network Solutions Inc.
EDU	US Educational	..
GOV	US Government	..
NET	Network	..
ORG	Non-Profit Organization	..
INT	International	ARPA/CSTO
MIL	US Military	DDN NIC
ARPA	Old style Arpanet	
NATO	Nato field	
AD	Andorra	$\emptyset$
AF	Afghanistan	Reserved by Asia Pacific NIC
DZ	Algerie	Algerian Unix User Group AlUUG
	...	
BE	Belgium	Katholieke Universiteit Leuven
	...	
CA	Canada	University of British Columbia
	...	...
FR	France (French Republic)	INRIA
GF	French Guiana	$\emptyset$
PF	French Polynesia	$\emptyset$
TF	French Southern Territories	$\emptyset$
	...	...
UK	United Kingdom	University College London
US	United States	University of Southern California
	...	...

TAB. 1.2 – *Quelques domaines mondiaux*

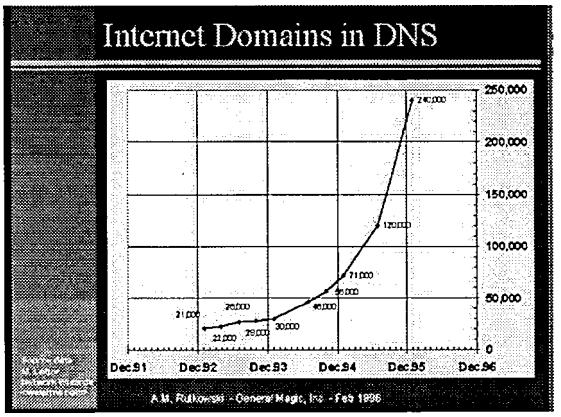
notés sous forme décimale, séparés par des points.
Exemples :

198.137.241.30 pour `whitehouse.gov`
193.105.19.152 pour `www-ceril-,univ-mlv.fr` ...

Ces numéros `IP` étant peu clairs, un service de noms s'est développé pour associer, de façon distribuée, des noms plus évocateurs aux machines. Ce service, connu sous le sigle `DNS`[19] (pour Domain Name System), est hiérarchisé par niveaux géographiques ou économiques : mondial, régional, local ...

Le `Departement de la défense américain`[20] gère le niveau global donc délègue l'administration des sous-domaines au `niveau mondial`[21] . Les organismes choisis (NIC) font de même. La Table 1.2 donne quelques exemples de domaines mondiaux.

19. URL: http://www-igm.univ-mlv.fr/foldoc?DNS
20. URL: http://www.nic.ddn.mil/LIBRARY/whatisit.html
21. URL: http://www.nic.fr/Statistiques/world/top-domains.html



...
barclay.fr
chargeurs.fr
...
danone.fr
decylog
evian
fnac
gaumont
gie-cartes-bancaires
mccann
orangina
palais-festivals-cannes
seita
tf1
toplog
...

FIG. 1.7 – *Service de Noms*

Ce service de noms, qui n'était au départ qu'utilitaire, attise aujourd'hui les convoitises. En effet, les URL mentionnant explicitement le nom de la machine, chacun veut se voir apparaître avec le moins d'informations parasites possibles. Donc il faut être rattaché au domaine le plus haut, même si sa présence réelle sur *Internet* se limite à quelques pages sur une machine commune.

Le domaine `.fr` compte aujourd'hui plus de `10,000 noms de sous-domaines`[22] , évidemment regroupés en 100 fois moins de sous-domaine IP.
La Figure 1.7 donne quelques exemples de noms symboliques, sous-domaines rattachés à un unique fournisseur d'accès à *Internet*: `imaginet.fr`. Le graphique montre l'évolution exponentielle du nombre de domaines DNS depuis cinq ans.

Par dessus `IP`, viennent s'ajouter d'autres protocoles. Au niveau transport, le plus connu parce que le plus usité chez les développeurs Unix est `TCP` ([Rif90]). L'acronyme *TCP/IP* a d'ailleurs joué pendant vingt ans le même le rôle que *e-mail* hier, qu'*Internet* aujourd'hui, et que *Web* demain, c'est à dire l'évocation floue de tout ce qui s'y raccroche.

D'un point de vue technique, l'essor d'`Internet` comme standard au niveau réseau peut paraître un handicap puisque ses performances sont limitées par des caractéristiques trop lourdes au vu des équipements physiques assurant aujourd'hui les transmissions. Cette rusticité permet un coup de raccordement faible donc garantit un avenir serein, même si l'idée d'un `Internet nouvelle génération`[23] fait de plus en plus son chemin.

Les spécificités des protocoles (Figure 1.3) sont largement masquées à l'utilisa-

22. URL: http://www.nic.fr/Procedures/
23. URL: http://www-igm.univ-mlv.fr/foldoc?Internet+Protocol+Version+6

	Type de connexion	Port TCP	Type d'informations	Commandes
ftp	session	21	arborescence de répertoires et de fichiers	PUT, GET, DIR
gopher	requête	70	hierarchie de documents, répertoires	Parcours
wais	requête	210	base de donneés documentaires	Recherche indexées
W_3	requête	80	ressources reliées par hypertexte	Parcours ...
telnet, rlogin	session	23	programmes interactifs en mode texte	
Xwindow (X11)	requête	6000	interface graphique	∞
e-mail	diffusion	25	messages entre utilisateurs	POST
news	diffusion	119	messages aux groupes	POST
archie, netfind	requête	–	index des sites `ftp` ou des utilisateurs d'*Internet*	recherche

TAB. 1.3 – *Applications liées aux protocoles usuels sur* TCP/IP

teur par le logiciel client W_3 qui se charge d'établir ou de simuler des connexions pour produire un document HTML.

Les informations demandées sont pour la plupart publiques et ne nécessitent pas d'autres paramètres que ceux donnés par l'URL. Tout au plus, pour les sites à accès restreint, l'utilisateur pourra être invité à entrer un mot de passe, mais on est loin des temps héroïques où les transferts de fichiers via FTP, la consultation des *newsgroups* ou encore la souscription d'un abonnement à une liste de diffusion (*mailing-list*), étaient des opérations relevant un peu de la magie.

1.3.4 Tolérance et qualité de service

La simplicité des standards couvrant l'initiative W_3 s'accompagne d'un parti pris de tolérance quant aux vérifications de conformité. Autant que possible, l'esprit l'emporte sur la lettre, souvent au bénéfice de l'utilisateur.

C'est particulièrement salutaire pour les rédacteurs novices qui en s'approchant du strict minimum de HTML, voient leur document correctement interprété par les visualiseurs.

Cela peut être pratique pour les clients qui rechignent à taper des informations redondantes comme `http://www....` ou `ftp://ftp....`,
Netscape leur fournit une *résolution transparente* des schémas d'adressage.
Le problème est ici la mise en défaut de la construction des URL relatives (voir Section 3.2) :
L'utilisateur constatant que l'URL `www.un-serveur.dom/` est accessible, va insé-

rer des liens dans un de ses propres documents, sans préciser `http://`.
Ce qui était redondant devient ambigu.
Par exemple, si dans `http://www.local.dom/d1/son-document.html`,
on active un lien vers `www.un-serveur.dom/`
on cherchera en fait `http://www.local.dom/d1/www.un-serveur.dom/` qui sera sans aucun doute une erreur.

D'un point de vue plus technique, l'encodage hexadécimal pour le nommage des URL n'est que rarement nécessaire, et ne pose encore des problèmes pour les caractères 8bits qu'aux machines dont la configuration date un peu.
Par exemple: `http://www-ceril.univ-mlv.fr/Actualité/`
induit quelques erreurs `http://www-ceril.univ-mlv.fr/Actualit` non trouvé.

1.4 Navigation

Le terme de **navigation** est communément employé pour illustrer la fluidité[24] des parcours possibles (ou virtuels?), par activation manuelle des liens.

Les liens W_3 sont unidirectionnels: rien n'indique si un document est pointé par d'autres documents.
Si un document n'est pas accessible, l'activation du point d'ancrage produira un message d'erreur (voir annexe). Ce principe apporte énormément de souplesse et de modularité mais mène parfois à des impasses inattendues, temporaires ou rédhibitoires.

1.4.1 Fonctionnalités d'un client

Les clients W_3 sont des applications destinées à afficher les documents, HTML ou autres (voir section 1.2.3).

Sur MacIntosh ou Windows, *Netscape Navigator* et *Internet Explorer* se partagent le marché. Pour Xwindow, *Mosaic* , *Arena* sont plus limités mais fournissent les programmes sources. Pour NeXTStep, *OmniWeb* a rattrapé son retard sur la normalisation HTML. *SpiderWoman* ou *NetSurfer* offre une alternative plus limitée.

Ils offrent bien sûr de nombreuses `fonctionnalités`[25] pour chercher, mémoriser des chemins à travers le réseau.

Liste de liens prédéfinis d'abord pour trouver l'aide en ligne du logiciel client lui même, plus quelques sites recueils de liens incontournables.

Historique des parcours hypertexte avec la possibilité de revenir en arrière. Cet historique est présenté de manière linéaire avec une détection des cycles d'exploration lorsque c'est possible.

Marque-page, Signet (*bookmarks*) La plupart des logiciels permettent de mémoriser des adresses de documents visualisés.
De plus en plus la gestion de ces marque-pages est fournie en interne pour ordonner et mettre à jour ces informations précieuses.

Cache Mémorisation locale et temporaire des documents distants pour limiter les temps de transfert réseau.

E-mail, news Gestion directe d'applications liées aux protocoles *Internet*, autres que la visualisation de page HTML.

Parmi les extensions souhaitables on peut imaginer l'exploration complexe pour vérification d'intégrité ou anticipation des parcours, mais ces fonctionnalités consommerait trop de bande passante pour être efficaces, compte tenu des

24. On trouve aussi *surfer*, que les Quebécois traduisent par *butiner*.
25. URL: http://browserwatch.iworld.com/

performances actuelles des réseaux. Mais cela serait une généralisation utile des *robots* indexeurs ([Fie94, Ric94]).

Une possibilité d'édition *WYSIWYG* HTML avec appels au serveur pour prendre en compte les modifications locales et distantes favoriserait l'édition répartie et le travail en groupe ([DS96]).

A l'opposé une tendance voudrait spécialiser les clients dans la présentation de documents formatés d'une qualité bien supérieure à ce que permet HTML (`HyperTeX`[26] ou `Acrobat`[27]), avec intégration de ressources audio, video, réalité virtuelle ...

1.4.2 Contenu des serveurs

La partie *protocole* d'une URL (voir Figure 1.6) donne des indications sur la nature des informations potentiellement disponibles. Schématiquement tout ce qui n'est pas `http://` n'est pas directement pensé pour un service documentaire.

On trouve principalement des archives de fichiers, anciennement `ftp` ou `gopher` avec quelques sucres HTML comme entêtes de présentation.

Il est par ailleurs dommage que ces archives FTP usuelles ne soient pas plus utilisées, car elles sont beaucoup plus simples à mettre en oeuvre et suffisent largement à la diffusion de documents statiques, pour des sites qui de toutes façons ne s'investiront pas dans le développement spécifique des services W_3.
Nous utiliserons par exemple : `ftp://www-ceril.univ-mlv.fr/`

On trouve aussi des interfaces sommaires de bases de données pré-existantes, ou encore de la documentation, des manuels en ligne issus de documents existants convertis en HTML avec un besoin flagrant de post-édition.

La possibilité pour l'utilisateur de conserver la trace des documents parcourus, constitue une première étape dans la rédaction de documents hypertextes. Un recueil de liens intéressants, thématiquement cohérents, est en tant que tel une information précieuse. Les marque-pages rassemblés *à la main* ou issus de recherches automatiques ont donc tendance à proliférer sans forcément présenter un intérêt pour les utilisateurs autres que leur propriétaire.

Autre travers souvent rencontré, les listes de ressources disponibles par ailleurs, le plus souvent de façon payante. Dans le domaine linguistique, le `Trésor de la Langue Française`[28] ou `Linguistic Data Consortium`[29] sont deux exemples représentatifs de cette stratégie de plus en plus en vogue.

Les documents HTML natifs ont fait leur apparition mais, souvent destinés à être clinquants, ils n'apportent que peu d'informations intéressantes, ce qui se comprend si on considère la difficulté d'utilisation des quelques éditeurs *WYSIWYG* HTML existants.

26. URL: http://xxx.lanl.gov/hypertex/
27. URL: http://www.adobe.com/
28. URL: http://humanities.uchicago.edu/ARTFL/
29. URL: ftp://www.cis.upenn.edu/pub/ldc_www/ldc_catalog.html#price

Même si les progrès sont constants, la rédaction d'hypertextes, structurés pour offrir des vues parallèles ou concurrentes d'informations complexes, est un domaine en devenir. Les outils de manipulation de structures de dimension supérieure à 2 restent peu pratiques.

Les ressources issues de travaux coopératifs sont encore rares et le plus souvent basés sur des échanges d'informations par *e-mail* (Voir Section 3.1.1) puisque les interfaces des clients W_3 restent largement spécialisés dans la visualisation.

1.4.3 Identifier les services intéressants

Pour s'y retrouver dans le nombre sans cesse en augmentation de serveurs W_3, un utilisateur novice ou non, dispose d'une foule de solutions plus ou moins évidentes et efficaces.

Navigation aléatoire Tout site un tant soit peu développé, tout comme un logiciel client W_3, recèle un recueil de liens considérés comme intéressants, connexes d'où l'on peut démarrer une exploration du réseau.

En choisisant un point d'entrée au hasard on peut errer indéfiniment vers des choses inattendues, à défaut d'être intéressantes.
Certains serveurs vont jusqu'à définir une URL constante pointant aléatoirement vers une de leurs propres ressources.

Sites célèbres Une connaissance minimale de l'adressage IP (voir figure 1.2) permet de reconstruire les noms Internet des serveurs suivant l'organisme ou l'entreprise recherchée, en rajoutant `www.` devant. Ces conventions ne varient que très peu, `web.` pouvant parfois se substituer à `www.` ou les noms des sites utilisant - comme séparateur,

Quelques exemples:

- Université de Stanford ⟶ `www.stanford.edu`
- MIT ⟶ `www.mit.edu`
  Attention ce serveur est celui des étudiants du Massachusetts Institute of Technology, le serveur institutionnel étant `http://web.mit.edu`.
- Université de Paris 7 ⟶ `www.jussieu.fr`
  moins trivial à trouver pour des raisons historiques.
- LADL-CERIL ⟶ `www-ceril.univ-mlv.fr` notre modeste serveur.
- l'ONU ⟶ `www.un.org` le nom du site venant bien sûr de l'anglais, `www.onu.org` n'existe pas.

Outils Internet classiques (voir Section 1.3.3) Comme `archie` qui donnent accès aux index des sites archives FTP, essentiellement donc pour rechercher le nom d'un fichier ou d'un programme. L'accès n'est pas directement fourni par les clients W_3 classiques. Il faut passer par un serveur HTTP qui assure l'interface, comme c'est le cas à l' `UREC`[30]

30. URL: http://web.urec.fr/archie.html

WAIS ou gopher procèdent du même esprit, un peu compliqués à manipuler et à interpréter, mais plus orientés vers le contenu des fichiers.

Listes de liens thématiques ou géographiques Issues d'indexation automatique et/ou de recensement manuel. La deuxième solution présentant les informations les plus cohérentes Cependant les déclarations coopératives, où les responsables de serveurs déclarent eux même leurs pages amènent encore souvent sur des serveurs sans contenu ou en travaux.

`Virtual Library Project`[31] Lancé à ses débuts par le Consortium W_3, chacun peut dériver un sous domaine plus spécialisé d'un domaine déjà couvert, en déclarant ses ajouts au niveau directement supérieur.
Les sites volontaires pour collaborer offrant des qualités de service très inégales, les recherches dans la *Bibliothèque Potentielle* peuvent être parfois fastidieuses, mais les informations sont souvent pertinentes et de qualité.

L'arborescence des thèmes documentés via W_3 est colossale et ne cesse de se développer!

Indexeurs aveugles basés sur des *robots* explorateurs

- `Alta Vista`[32] recense plus de 30.000.000 pages, reste à faire le tri!
  Moins impressionnant mais performant `Lycos`[33] , `OpenText`[34] ...
- `Harvest`[35] orienté vers l'indexation répartie des documents, les serveurs W_3 devenant serveurs `Harvest`.
  La mise en place et la maintenance des logiciels étant assez coûteuse au niveau matériel, la généralisation, par ailleurs souhaitable, de ce type d'architecture est peu probable, comme ce fut déjà le cas avec WAIS (voir Table 1.3)
- `Yahoo`[36] , `WebCrawler`[37] , `Aliweb`[38] fournissent un index des thèmes pour réduire le domaine des recherches.
  `Lokace`[39] est un des rares indexeurs pour la documentation en français.

Meta-Index et Meta-Listes Vu le nombre grandissant des sites se chargeant de lister et d'indexer des documents, il était naturel que des listes de ces sites se mettent en place!
Par exemple:

- `Configurable Unified Search Engine`[40] Limité à l'interface.

31. URL: http://www.w3.org/hypertext/DataSources/bySubject/Overview.html
32. URL: http://altavista.digital.com/
33. URL: http://www.lycos.com/
34. URL: http://index.opentext.net/
35. URL: http://harvest.cs.colorado.edu/
36. URL: http://www.yahoo.com/
37. URL: http://www.webcrawler.com/
38. URL: http://www.nexor.co.uk/public/aliweb/doc/history.html
39. URL: http://lokace.iplus.fr/
40. URL: http://pubweb.nexor.co.uk/public/cusi/cusi.html

- `w3catalog`[41] concaténation de quelques listes (*Yahoo, VL* entre autres), mais moteur d'indexation pas très fin puisque si on recherche `index` on retrouve tous les fichiers `index.html`, qui est une convention quasi universelle!!

La publicité pour essayer d'intéresser les clients, se fait par déclaration aux divers acteurs sur *Internet*: annonces par `e-mail` à un groupe d'utilisateurs connus, une liste de diffusions (voir section 1.4.3), un groupe de `news`, ou des déclarations explicites aux sites recueils de liens. Il existe même un *meta-déclarateur* : `Meta registry`[42] , qui recense les pages où l'on peut faire enregistrer ses URL.

Ces déclarations peuvent concerner le serveur comme représentant "institutionnel" comme ce sera le cas avec les données linguistiques pour le serveur LADL-CERIL.
Cela concernera aussi des services annexes (Voir Section 3.1.1).

Exemple d'interrogation

La mise en pratique des suggestions ci dessus permet d'esquisser un tableau du domaine linguistique sur W_3. Celui ci étant par essence très mouvant, nous nous limiterons à quelques grandes lignes.

A partir des recueil de liens essentiels:

- Le projet Virtual Library a suscité quelques pages intéressantes à des titres divers, des dates conférences aux corpus: `Linguistic`[43], `Applied Linguistics`[44] , `World Litterature`[45]
- citons aussi `Association for Computational Linguistics NLP/CL Universe`[46] et la page du `MIT`[47] .

La documentation est évidemment le point fort de W_3, on trouve de plus en plus facilement articles, actes de conférences, rapports techniques de laboratoires de recherche ou d'entreprises.
`Computation and Language E-Print Archive`[48] est le site absolument incontournable pour toute bibliographie. Il regroupe tout ce que les auteurs ou les éditeurs du domaine ont bien voulu lui transmettre et offre un mécanisme de recherche très performant. Ces services étaient auparavant disponibles via *e-mail.*

Le courrier électronique à vingt ans d'avance sur W_3, il existe beaucoup d'autres (> 50) `listes de diffusion`[49] , plus ou moins sérieuses ou scienti-

41. URL: http://cuiwww.unige.ch/w3catalog
42. URL: http://www.submit-it.com/
43. URL: http://www.emich.edu/ linguist/www-vl.html
44. URL: http://www.bbk.ac.uk/Departments/AppliedLinguistics/VirtualLibrary.html
45. URL: http://sunsite.unc.edu/ibic/IBIC-World-Lit.html
46. URL: http://www.cs.columbia.edu/ radev/cgi-bin/universe.cgi/
47. URL: http://www.ai.mit.edu/projects/iiip/nlp.html
48. URL: http://xxx.lanl.gov/cmp-lg
49. URL: http://www.ling.rochester.edu/lists.html

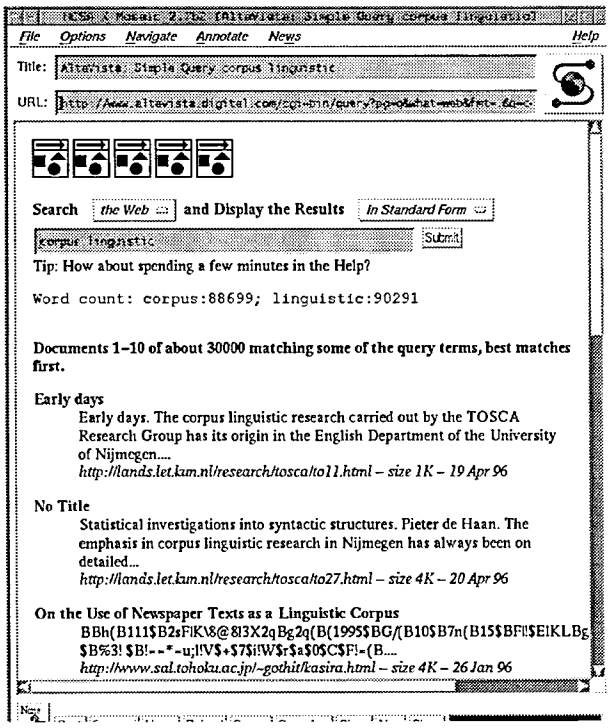


FIG. 1.8 – *Exemple de résultat cabalistique par* `Altavista`

fiques, plus ou moins active dans le domaine linguistique. Leurs buts sont essentiellement l'annonce d'événements et les discussions attenant à leur centre d'intérêt déclaré: `Corpora`[50] , `LINGUIST`[51] , `LN`[52] (pour langage naturel en français),

Pour des références bibliographiques, on peut aussi consulter les catalogues des grandes bibliothèques, par exemple celui de la `BNF`[53] même si son interface de recherche `telnet` date un peu.

W_3 *regroupant* tout et n'importe quoi, les recherches au hasard sur des mots clés que l'on croit spécifiques à un domaine peuvent s'avérer surprenantes.

On se rend compte que `NLP`[54] n'est pas toujours ce que l'on croit, et que `Corpus`[55] est aussi, et surtout, Corpus Christi ...

Les listes thématiques de `Yahoo` ou `WebCrawler`, sont pour cela plus claires que les recherches *avancées* d'`AltaVista`. La Figure 1.8 montre un résultat de requête pertinent mais incluant un document japonais dont l'encodage est évidemment problématique.

50. URL: http://www.hd.uib.no/corpora/
51. URL: http://www.emich.edu/ linguist/
52. URL: http://www-ceril.univ-mlv.fr/ln/
53. URL: telnet://opale02.bnf.fr/
54. URL: http://www.nlpinfo.com/humor/glossary/index.htm
55. URL: http://www.texas.org/

1.4.4 Interprétation des erreurs

Un client standard W_3 masque l'essentiel des paramètres de communication. Les causes d'erreur peuvent être liées au réseau, à l'ensemble client/serveur, ou à la configuration du serveur, mais n'induisent souvent qu'une notification très générale.

Un rapide retour sur les informations présentées dans ce chapitre permet de se faire une idée précise du problème rencontré. Une fois que l'on s'est assuré des problèmes matériels ou physiques, donc que sa machine était bien branchée, quelques programmes utilitaires courants permettent d'avoir des renseignements sur la configuration et l'état du réseau.

Pour trouver si un nom correspond bien à une machine (voir Table 1.2) la résolution fait appel à une machine serveur DNS :

- Pas de réponse indique un problème interne au DNS.
- Une réponse négative signifie que le nom n'existe pas:

```
# nslookup www-ceril.univ.mlv.fr
  can't find www-ceril.univ.mlv.fr: Non-existent domain
```

- Une réponse valide retourne le numéro IP correspondant au nom demandé :

```
# nslookup www-ceril.univ-mlv.fr
Name:    wisefull.univ-mlv.fr
Address:  193.55.44.154
Aliases:  www-ceril.univ-mlv.fr
```

La commande `ping` permet de tester si la machine distante répond.
On obtient une réponse du type:

```
PING wisefull.univ-mlv.fr: 64 byte packets
64 bytes from 193.55.44.154: icmp_seq=0. time=2. ms
64 bytes from 193.55.44.154: icmp_seq=1. time=1. ms
64 bytes from 193.55.44.154: icmp_seq=2. time=1. ms
...
```

En cas de non réponse on peut visualiser le chemin suivi par les données entre 2 sites Internet avec la commande `traceroute`.
Par exemple, la Figure 1.9 montre par où passe l'information pour aller de Marne-La-Vallée à l'ONU.

Si le serveur est incapable de fournir le document adressé : les codes d'erreur HTTP indique clairement la cause du problème URL erronée ou problème d'intégrité, (voir annexe D).

```
    30 hops max, 40 byte packets
 1  rd2.univ-mlv.fr (193.55.44.2)  4 ms  3 ms  3 ms
 2  192.168.1.1 (192.168.1.1)  5 ms  5 ms  5 ms
 3  renatif.univ-mlv.fr (192.134.103.21)  6 ms  5 ms  5 ms
 4  193.55.47.3 (193.55.47.3)  5 ms  6 ms  5 ms
 5  fontenay.rerif.ft.net (193.48.74.33)  8 ms  8 ms  8 ms
 6  danton1.rerif.ft.net (193.48.53.57)  13 ms  18 ms  24 ms
 7  stlambert.rerif.ft.net (193.48.53.49)  11 ms  16 ms  11 ms
 8  stamand1.renater.ft.net (192.93.43.115)  12 ms  13 ms  11 ms
 9  rbs1.renater.ft.net (192.93.43.186)  11 ms  15 ms  16 ms
10  raspail-ip.eurogate.net (194.206.207.18)  11 ms  11 ms  13 ms
11  Reston.eurogate.net (194.206.207.5)  269 ms  258 ms  264 ms
12  204.59.144.199 (204.59.144.199)  278 ms  275 ms  275 ms
13  Hssi4-0.BR1.TCO1.Alter.Net (137.39.103.17)  270 ms  269 ms  264 ms
14  * Hssi4-0.CR1.DCA1.Alter.Net (137.39.100.77)  246 ms  277 ms
15  101.Hssi4-0.CR1.NYC1.Alter.Net (137.39.30.6)  288 ms  274 ms  270 ms
16  Fddi0-0.CR2.NYC1.Alter.Net (137.39.33.227)  279 ms  297 ms *
17  Hssi4-0.New-York3.NY.Alter.Net (137.39.100.5)  437 ms  279 ms  439 ms
18  Fddi0-0.New-York2.NY.Alter.Net (137.39.126.3)  296 ms  428 ms *
19  UN-gw.ALTER.NET (137.39.208.34)  291 ms *  283 ms
20  unsvc.un.org (157.150.192.234)  262 ms *  283 ms
```

FIG. 1.9 – *Traceroute to www.un.org (157.150.195.19)*

Seuls un ou plusieurs essais ultérieurs permettront de juger du rétablissement ou de l'existence du service. Des requêtes par des canaux différents (essentiellement e-mail) peuvent aussi accélérer la prise en compte et la résolution du problème rencontré.

En tout état de cause, avec les performances des réseaux actuels, il est très rare de rencontrer des erreurs malignes lors de la transmission des données. Si le document arrive illisible, ce sera plutôt la configuration du client qu'il faut incriminer et notamment la définition des traitements externes (Voir Section 1.2.3).

Chapitre 2

Structuration des données linguistiques

2.1 Vers un service de références

La mise en place d'un service cohérent et utile passe par le partage des compétences linguistiques et informatiques. Le travail *d'édition et de publication* des dictionnaires, des tables du lexique grammaire sur W_3 doit fournir un service de référence homogène et permettre de relier naturellement les études linguistiques à leur compilation en vue de traitements automatiques.

2.1.1 Documentation

La rédaction des descriptions textuelles prend une place importante dans l'enrichissement du serveur, et passe, à terme, par l'intégration des articles, thèses ou livres décrivant les résultats des différents travaux présentés. La bibliographie (voir Figure 2.1) est la première étape dans la collecte des données.

L'auteur est bien évidemment le mieux placé pour juger de la pertinence des documents disponibles, ainsi que pour évaluer, sur le fond comme sur la forme, l'intérêt de la présentation de ses sources sur W_3. Il en va de même pour les corrections à y apporter, que les erreurs soient d'origine ou dues aux traitements postérieurs.

Les solutions actuelles pour produire des documents intéressants sous W_3 ajoutent à l'aspect rédaction, des préoccupations encore très techniques liées à l'environnement du serveur. Les auteurs d'études linguistiques s'avèrent en règle générale fort dépourvus dans le domaine informatique. L'expérience montre que l'implication n'est pas non plus très importante parmi les informaticiens, même si la tendance au tout W_3 fait son chemin.

La présentation de ces données nécessite un examen minutieux, pour identifier les références internes, les exemples qui mériteront l'extraction de liens hypertexte. Une procédure semi-automatique transformant les données devra tenir compte des conflits entre le format interne utilisé sur le serveur et les conven-

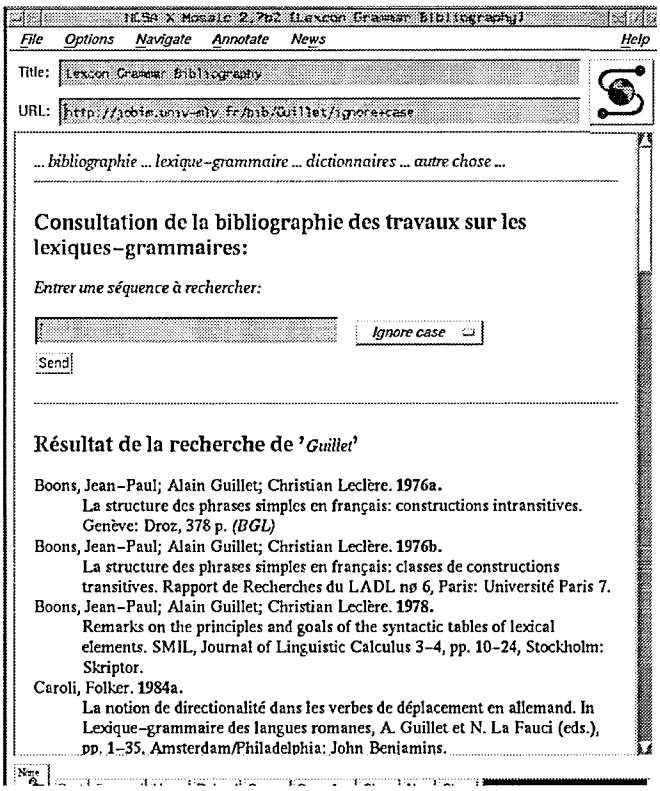


FIG. 2.1 – *Bibliographie LADL, CERIL ...*

tions adoptées par chaque rédacteur. Ces conventions de marquage spécifiques: abréviations, séparateurs, propriétés, métalangage dépendent actuellement plus de contingences matérielles, que de choix formellement fondés.

Outre des facilités d'accès aux documents, la structuration des données linguistiques permet de fournir des mécanismes de recherche qui ne sont pas disponibles avec les outils d'analyse qui intègrent ces données (voir Figure 4.5).

L'accès homogène à ces ressources permettra d'évaluer la pertinence des spécifications et classifications théoriques. Cela incitera à développer des fonctionnalités de traitements de ces données de base, pour définir et vérifier les notations, croiser les informations et fournir un adressage précis (Voir modules Perl).

2.1.2 Marquage

Pour concilier les aspects documentaires et formels certaines balises seront réservées pour la structuration, la sémantique variant selon les contextes. Les relations seront principalement explorées d'après les marqueurs suivants:

- Les références internes `<A NAME="identificateur">`
- Les titres: la structure des documents HTML en prévoit 6 niveaux:
  `<H1>titre de niveau 1</H1>`

- Les liens hypertextes `<A HREF="URL associée">point d'ancrage</A>`
- Les listes :

```
<DL>  <!-- Recherche de ''phrases simples'' -->
<DT>Boons, Jean-Paul; Alain Guillet; Christian Leclère.  <B>1976a.</B>
    <DD> La structure des phrases simples en français: constructions
     intransitives. Genève: Droz, 378 p. <I>(BGL)</I>

<DT>Boons, Jean-Paul; Alain Guillet; Christian Leclère.  <B>1976b.</B>
  <DD> La structure des phrases simples en français: classes de
       constructions transitives. Rapport de Recherches du LADL nø 6,
       Paris: Université Paris 7.

<DT>Giry-Schneider, Jacqueline.  <B>1987.</B>
  <DD> Les prédicats nominaux en français. Les phrases simples à verbe
       support. Genève: Droz.<I>(Giry87)</I> </DL>
...
```

Ces marquages associent simplement les codes couramment utilisés à leur signification. Il convient de noter que la relation `<DT> / <DD>` est locale à une ressource, l'association est bidirectionnelle. Les mécanismes de recherche en sont facilités d'autant.

Ce n'est pas le cas de la relation hypertexte entre le point d'ancrage et la ressource associée. *Remonter* la relation d'un document vers ses références nécessiterait un parcours de tous les documents potentiellement pertinents, pour y rechercher les points d'ancrage. Une solution simple consiste à spécificier un lien inverse, du document pointé vers le point d'ancrage: `<A LINK= ... >`

Les données doivent être stables et fortement reliées car chaque changement de structure impliquera de modifier deux documents. Cette contrainte peut être allégée par une organisation modulaire du serveur qui pourra explorer automatiquement et périodiquement l'ensemble des documents et mettre à jour les liens. L'intégrité de l'hypertexte variera alors en fonction des fréquences de modification des données et de mise à jour. L'organisation actuelle des données électroniques est schématisée Figure 2.2.

2.1.3 Problèmes ouverts

La normalisation telle que nous l'esquissons ici n'épuise pas tous les problèmes. Le plus difficile étant probablement celui très général de la documentation multilingue. Elle intervient principalement à deux niveaux, d'abord dans les études comparées où les données comme les exemples ne suivent que rarement des structures parallèles.

Deuxième aspect, la traduction des descriptions pour des locuteurs non natifs. A l'heure actuelle, l'organisation du serveur privilégie le français pour limiter l'imbrication des liens hypertextes aux relations utiles pour une exploitation automatique.

D'un point de vue plus pratique, la coexistence des polices de caractères n'est pas forcément encore très simple. W_3 permet déjà de prendre en compte la plupart des langues et accélére la normalisation d'un jeu de caractères universel (Voir

Section 1.2.1). Les polices doivent être installées sur les machines clients, on peut ainsi obtenir, toujours via une interface homogène, les deux exemples de la Figure 2.3 issus des tables ADJN [Skl93] et AWS [Nam94].

Pour le grec, les deux premières colonnes apparaissent en *ISO-latin-7*, qui est l'encodage standard grec, et les autres en *ISO-latin-1*.

Pour le coréen, les logiciels doivent être adaptés pour afficher des fontes 16bits, dont la normalisation reste à faire.

A l'usage, un vrai système documentaire doit pouvoir définir ses propres objets, syntaxiques ou sémantiques. Sans prétendre atteindre un niveau de généralité absolu, type `TEI`[1] (Text Encoding Initiative), l'utilisation de marqueurs prédéfinis permettrait de formaliser plus précisément l'emploi des symboles (ex :`<Nhum>` ou `<Vmvt>`). Les balises SGML, non définies par HTML, restent compatibles avec les logiciels client W_3, elles sont en général ignorées à l'affichage. L'utilisation des commentaires HTML remplit une fonction identique.

On pourra donc enrichir les informations contenues dans un document pour des usages spécifiques tout en conservant les fonctionnalités d'accès de W_3.

1. URL: http://etext.virginia.edu/TEI.html

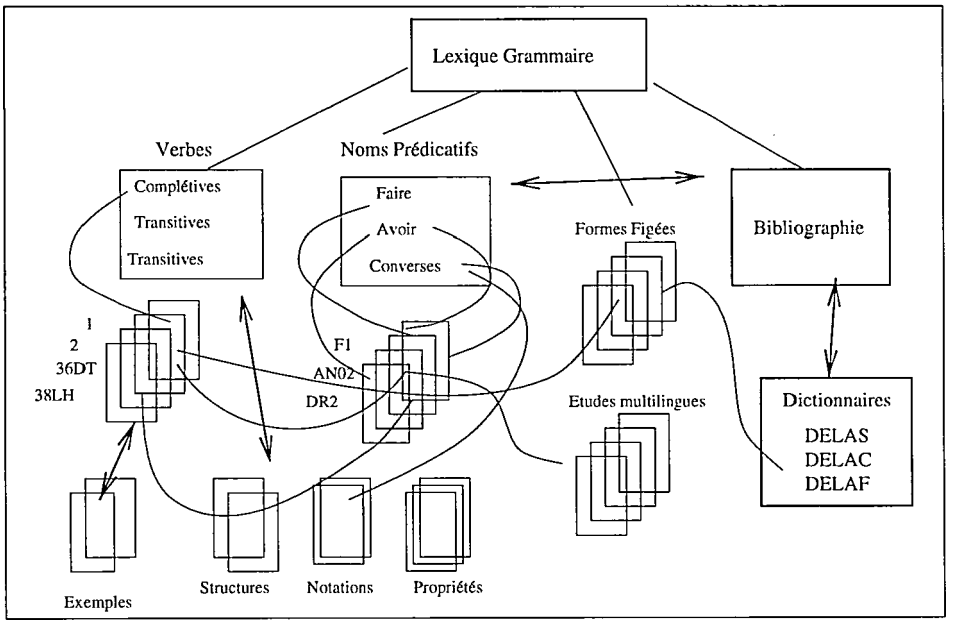


FIG. 2.2 – *Organisation du lexique grammaire*

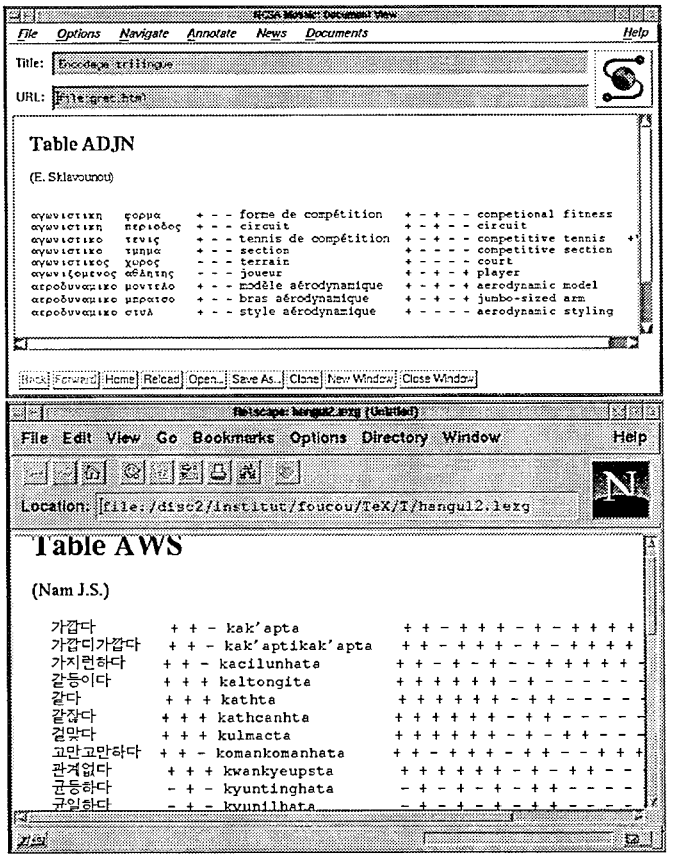


FIG. 2.3 – *Exemples de rendu multilingue*

2.2 Les dictionnaires électroniques

Les dictionnaires électroniques du LADL ont une couverture extrêmement large et contiennent des informations linguistiques spécifiées précisément ([CS90]).

2.2.1 Les codes

La gestion de ces données amène naturellement à coder et abréger les propriétés des formes étudiées. Ces informations étant à la base de tous les traitements des langues naturelles, il est nécessaire de pouvoir s'y référer constamment. Ces informations sont aussi importantes pour les lexicographes en vue d'une unification des formats des dictionnaires (`DELAS`,`DELAF`, `DELAC`[2]). Cela permettra à terme de comparer les solutions choisies dans différentes langues. Les catégories grammaticales por le français sont les suivantes :

Mots simples	Mots composés
A = Adjectif	NA = Nom Adjectif
N = Nom	NAN = Nom Adjectif Nom
V = Verbe	VN = Verbe Nom
DET = Déterminant	NDN = Nom Déterminant Nom
ADV = Adverbe	...
PRO = Pronom	
PREP = Préposition	
INTJ = Interjection	
CONJS = Conjonction de subordination	
CONJC = Conjonction de coordination	
PFX = Préfixe	
XINC (ou XI ou X) = Partie de composé	

Pour passer des formes canoniques au formes fléchies, les mots simples réfèrent à 350 classes flexionnelles dont 150 verbales. Le format interne utilisé pour représenter les classes peut ressembler à la Figure 2.4.

```
.V1     inf/pr(oir)          %av-oir   E  av-  E-  e-  au-  ai-  ay-
!       part/pr(-1yant)
!       part/pa(-2eu,-2eue,-2eus,-2eues)
!       ind/pr(-2ai,-2as,-2a,ons,ez,-2ont)
...
.N.A.           (:ms,:fs,:mp,:fp)    %Modèle
.N.A1   =  (0,-,s,-)                 %ballon,ballons
.N.A2   =  (0,-,0,-)                 %engrais,engrais
...
.DET1   =  Dind_UN(un,une,des,des)
.DET2   =  Dind_DU(du.de_l,de_la.de_l,des,des)
...
```

FIG. 2.4 – *Schéma de classes flexionnelles*

2. URL: http://www-ceril.univ-mlv.fr/Dictionnaires/

Le système INTEX, que nous évoquerons au chapitre 4, dans sa version 3.4, normalise une représentation des flexions sous forme de transducteurs.
Exemple de représentation (non graphique) de la flexion `N1.nfa`:

```
 3 4
\%<E>\%<E>/:ms\%s/:mp\%
: 0 3 0 4 -1
t -1
: 1 2 -1
: 2 2 -1
f
```

2.2.2 Ordres de grandeur

Le dictionnaire électronique des mots simples (DELAS) contient 90000 entrées canoniques. Le dictionnaire électronique des mots composés (DELAC): 90 000 noms, 15 000 constructions *être Prep N*, 8000 adverbes, 500 Conjonctions.

Quelques exemples:

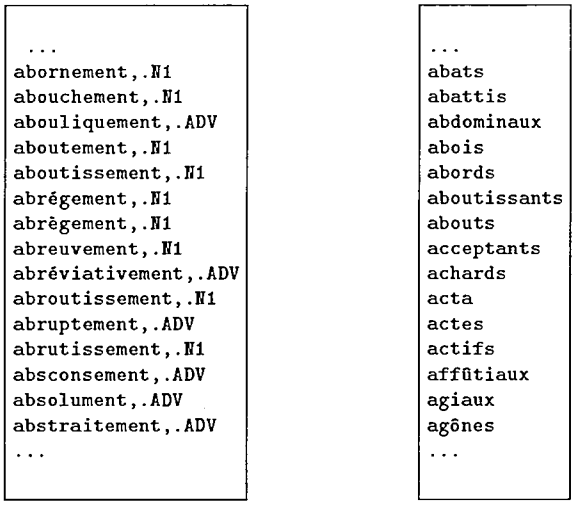
```
a,.N2+[LET]+z1
a,.XI+{~_contrario}+z1
à,.PREP+z1
aa,.N2+z3
aabam,.N2S+z3
aalénien,.A41+z3
aalénien,.N1+z3
ab,.XI+{~_intestat}+z1
abaca,.N1+Conc+[Vég]+z3
abacule,.N1+z3
abaissable,.A31+z2
abaissant,.A32+z2
abaisse,.N21+z3
abaissé,.A32+z1
abaissée,.N21+z3
abaissement,.N1+z2
abaisser,.V3+t+'11'+z1
...
```

```
Marché   commun;NA:ms/--;le
marche   antinucléaire;NA:fs/-+;une
         arrière;NA:fs/-+;une
         aux/flambeaux;NAN:fs/-+;une
         avant;NA:fs/-+;une
         d'approche;NDN:fs/-+;une
         d'harmonie;NDN:fs/-+;une
         dansante;NA:fs/-+;une
         de/la/légion;NDN:fs/--;la
         de/la/paix;NDN:fs/-+;une
         funèbre;NA:fs/-+;une
         ...
ouvre    -boîte;VN:mp/-+;un
         -boîte;VN:ms/-+;un
         -bouteille;VN:mp/-+;un
         -bouteille;VN:ms/-+;un
```

La taille de la version texte du dictionnaire des mots simples est inférieure à 2.5Mo. sa lecture linéaire prend moins de 3 secondes sur une machine NeXT tout à fait ordinaire. Ce qui est un temps de réponse largement acceptable pour un document W_3. Pour les formes fléchies ou les noms composés les tailles subissent un facteur 10, il faudra donc optimiser les mécanismes de recherche. Ces programmes ne prétendront bien sûr pas faire de la confrontation des dictionnaires à du corpus, on utilisera alors des mécanismes spécifiques d'indexation à base d'automates finis ([Rev91]).

2.2.3 Recherches diverses

Un avantage de la structuration des données à un niveau très bas est que l'on peut multiplier les points d'entrée. La Figure 2.5 montre les pages permettant de



```
...
abornement,.N1
abouchement,.N1
abouliquement,.ADV
aboutement,.N1
aboutissement,.N1
abrégement,.N1
abrègement,.N1
abreuvement,.N1
abréviativement,.ADV
abroutissement,.N1
abruptement,.ADV
abrutissement,.N1
absconsement,.ADV
absolument,.ADV
abstraitement,.ADV
...
```

```
...
abats
abattis
abdominaux
abois
abords
aboutissants
abouts
acceptants
achards
acta
actes
actifs
affûtiaux
agiaux
agônes
...
```

TAB. 2.1 – *Recherches diverses sur les mots simples*

vérifier la conjugaison des verbes et la flexion des noms ou adjectifs du DELAS.

A noter que le DELAC ne contient pas de verbes composés qui relèvent plutôt de la description syntaxique et seront donc à chercher parmi les `formes figées`[3] du lexique grammaire. L'accès au DELAC est illustré Figure 4.2.

Contrairement à une version compilée, on peut fournir des consultations transversales du type:

"Quels sont les mots se terminant par *ment*?"

Il y en a en tout 4282! Voir les premières entrées Table 2.1.

On est très tenté de multiplier les explorations:

"Quels sont les noms pluriels?" On en trouve 454. (Table 2.1)

Exemple: `http://www-ceril.univ-mlv.fr/dyn/delafGrep/,.NP2`

"Quels sont les épicènes? les palyndromes?" ...

Les fonctions de recherche dans ces dictionnaires et de présentation en clair des informations utilisent la puissance de *expressions régulières* Perl, ce qui permet une infinité de requêtes, en tout cas beaucoup plus que la commande `egrep`.

Les résultats sont évidemment filtrés selon les clients, il ne s'agit pas d'offrir la totalité des dictionnaires aux utilisateurs. Outre les problèmes de confidentialité, si un utilisateur cherche `zébus` dans le DELAF, il ne souhaitera probablement pas attendre une heure, en recevant 3 millions d'octets précédant la réponse qu'il attend.

Actuellement la maintenance de ces dictionnaires ne passe pas une interface W_3, il s'agit de copies datées, transférées manuellement sur le serveur. L'établissement de versions de référence consultables et modifiables reste à faire.

3. URL: http://www-ceril.univ-mlv.fr/LexiqueGrammaire/Francais/

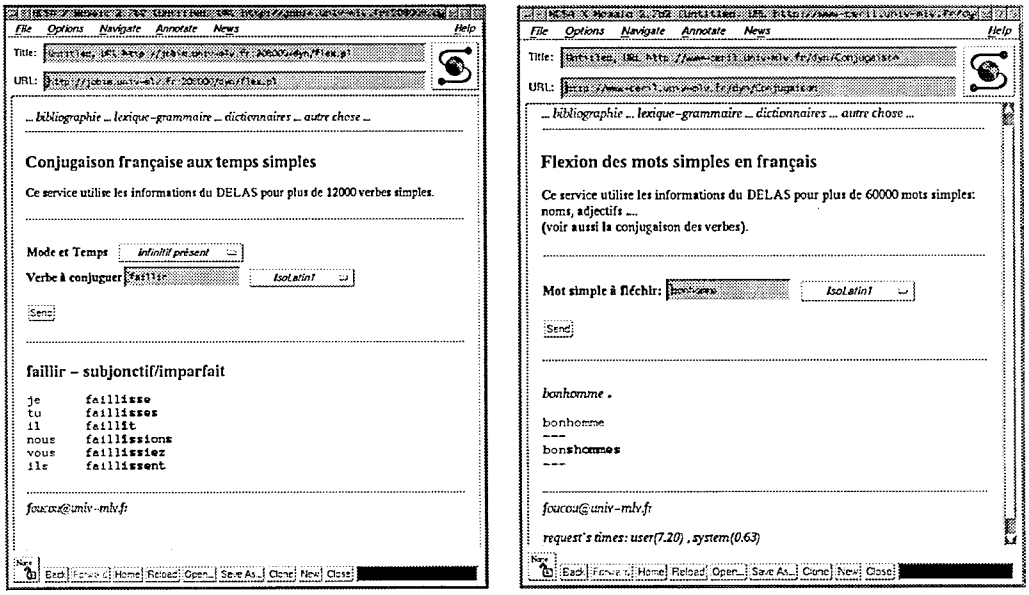


FIG. 2.5 – *Flexion des mots simples du DELAS*

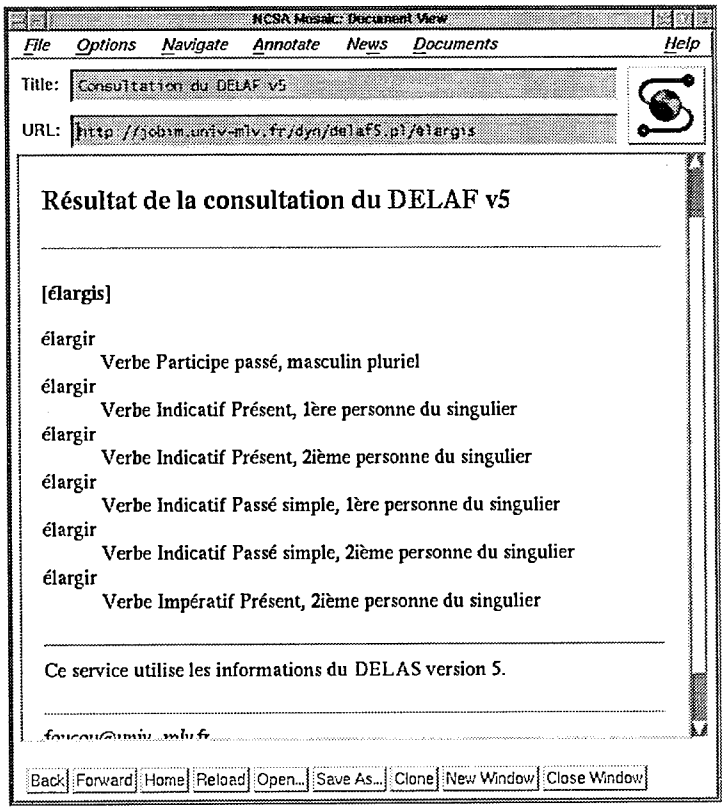


FIG. 2.6 – *Recherche de forme fléchie*

2.3 Lexique Grammaire

Une structuration hypertexte via W_3 permet de concilier les exigences du recensement des phénomènes linguistiques avec une formalisation minimale nécessaire à leur documentation [Fou95]. Il ne s'agit en aucun cas de refondre le cadre théorique du lexique grammaire dans un moule précontraint ([Aa93]), mais de permettre une consultation voire une rédaction plus aisée que le système D actuellement en vigueur, pour s'intégrer dans un environnement complet d'outils d'analyse automatique.

2.3.1 Méthodologie

Le lexique-grammaire recense, en grandeur réelle, les structures syntaxiques du français, sur des bases distributionnelles et transformationnelles ([Gro75]). La phrase élémentaire, schématiquement un verbe et ses arguments, est l'unité minimale d'étude. Les transformations syntaxiques reliant deux structures permettent de caractériser les comportements des éléments lexicaux. Les descriptions sont faites, de façon systématique, sur la base de l'intuition du linguiste, qui juge de l'acceptabilité des structures étudiées. Ce jugement se doit d'être reproductible, il distingue de façon binaire les structures du "français standard" des structures interdites. Pour des raisons théoriques et pragmatiques, l'attestation vient souvent plus de la construction d'exemples que du recours au corpus.

Le nombre de phénomènes linguistiques observés implique un système de notation compact. La représentation matricielle donne une vue synthétique des propriétés syntaxiques notées dans les entête de colonnes, pour les éléments lexicaux de chaque ligne (voir Figure 2.7). A chaque intersection, une marque (+ ou -) indique l'acceptabilité de la structure.

Le principe d'économie dans la description des structures examinées amène à toutes sortes de factorisations des informations. La maîtrise de la terminologie et les symboles utilisés devient alors essentielle à l'interprétation des informations regroupées dans le lexique grammaire ([Bès94b]). Cette nomenclature non triviale est en étroite relation avec les explications théoriques et les exemples contenus dans les commentaires des tables. En ce sens il ne s'agit pas d'un formalisme strict et l'aspect documentaire joue un rôle important.

2.3.2 Dimensionnement

Le regroupement des éléments par tables obéit à un souci de répartition homogène des données. La cohérence et l'homogénéité de la classification dépend du choix des domaines d'exploration. On distingue principalement: verbes simples, verbes supports/noms prédicatifs, formes figées, adjectifs, adverbes.

Les chiffres du tableau 2.2 recensent les données linguistiques actuellement disponibles sur le serveur W_3. Ces ordres de grandeur sont aujourd'hui très facilement manipulables du point de vue documentaire. Les tables syntaxiques dans

	tables	entrées (lignes)	propriétés ≠ (colonnes)	Exemples
Phrases simples & complétives	60	10000	400	12000
Noms Prédicatifs	50	10000	100	-
Adjectifs	10	3000	∅	-
Formes figées	50	30000	50	*(utile?)*

TAB. 2.2 – *Le lexique grammaire du français sous forme électronique*

leur version ASCII occupent moins de 20 Mo, une recherche globale d'un motif simple ne nécessitera pas plus de quelques secondes.

La récupération sous forme électronique ou la saisie manuelle de travaux menés il y a vingt ans reste encore à faire, tout comme l'intégration des études plus récentes. L'interdépendance des phénomènes est très forte et croît à mesure que de nouvelles études composent les propriétés des classes de base: insertion d'adverbes dans les structures simples, caractérisation de noms appropriés, de structures de l'adjectif ... Le modèle hypertexte proposé ici permettra de mieux cerner la couverture actuelle du lexique grammaire, mais aussi de relier des lexiques grammaires comparés ([Lab88, Skl93, Küb95a, LK92]), ou encore d'intégrer des classes objets dans les propriétés distributionnelles ([Gro94]).

2.3.3 Représentation par table

La faisabilité même d'une description de la langue en grandeur réelle n'est pas un des moindres résultats du lexique grammaire, mais son utilisation pour des traitements automatiques en exige une connaissance très spécialisée. Le premier système informatique gérant lexique grammaire ([Vas82]) avait pour but principal le stockage et l'impression des tables, ou l'indexation des entrées. L'organisation extrêmement centralisée n'en favorise pas la consultation, la maintenance ou l'enrichissement. En regard des outils informatiques actuels, tant au niveau matériel que logiciel, on peut envisager de dépasser ce stade artisanal de la base de données.

La classification étant le but même des recherches, l'examen de nouvelles propriétés peut conduire à dédoubler ou regrouper des entrées. Du fait de la diversité et des ambiguïtés des données, toute factorisation des formes canoniques mène à des classes non disjointes. Il paraît donc peu intéressant de formaliser des attributs ou des schémas d'héritage suffisamment stables pour organiser de ces informations linguistiques selon les modèles standards de base de données relationnelle ou orientée objet [San96].

L'utilisation d'automates et de transducteurs apparaît assez naturelle, la factorisation des spécifications pouvant être particulièrement efficace. De nombreux traitements automatiques peuvent être définis formellement à partir des opérations d'union et d'intersection. La représentation graphique des automates constitue un formalisme très lisible pour un nombre réduit d'étiquettes, on peut ainsi

```
Table 32R2
_                N N N N N N N N N N N N
_                0 0 0 1 0 0 1 1 1 1 0 1
                 = = b V e V = = = = V e
                 : : é   n   : : : :   s
                     n                 N t
                 N V é   m     N N l V 1
                 n - f   o     h - e -   V
                 r n i   u     u h     n a -
                     c   v     m u f   v p p
                     i   e       m a   e c
                     a   m             i   H
                     i   e             t
                     r   n                 V
                     e   t               Q -
                                         u n
                                         P i
                                           n
                                           s
                                           t
                                           r
abdiquer         - - - - - + - + - - - -
abolir           - - - - - - - + + - - +
accaparer        - - + - - - + + - - - -
accaparer        - - + - - + - + - - - +
accentuer        + - - - - - - + + - - -
accepter         - - - - - - - + - - - -
accomplir        + - - - - + - + - - - +
acquérir         - - + - - - - + - - - +
adorer           - - - - - - - + - - - +
affronter        + - - - - + + + - - - -
aimanter         - - - - - + - + - - - +
alambiquer       + - - - - - - + - - - +
allumer          - - - - - - - + - - - +
 ...
```

	N0 =: Nmr	N0 =: V-n	N0 bénéficiaire	N1 V	N0 en mouvement	N0 V	N1 =: Nhum	N1 =: N-hum	N1 =: le fait Qu P	N1 =: V-n	N0 V N1 avec V-n instr	N1 est V-pp W
abdiquer	-	-	-	-	-	+	-	+	-	-	-	-
abolir	-	-	-	-	-	-	-	+	+	-	-	+
accaparer	-	-	+	-	-	-	+	+	-	-	-	-
accaparer	-	-	+	-	-	+	-	+	-	-	-	+
accentuer	+	-	-	-	-	-	-	+	+	-	-	-
accepter	-	-	-	-	-	-	-	+	-	-	-	-
accomplir	+	-	-	-	-	+	-	+	-	-	-	+
acquérir	-	-	+	-	-	-	-	+	-	-	-	+
adorer	-	-	-	-	-	-	-	+	-	-	-	+
affronter	+	-	-	-	-	+	+	+	+	-	-	-
aimanter	-	-	-	-	-	+	-	+	-	-	-	+
alambiquer	+	-	-	-	-	-	-	+	-	-	-	+
allumer	-	-	-	-	-	-	-	+	-	-	-	+

FIG. 2.7 – *Table 32R2 formatée en ASCII simple ou en LaTeX*

spécifier des grammaires locales ([Sil93]). L'extension des spécifications linguistiques passe par la multiplication de ces automates, qu'il convient de documenter pour éviter les répétitions mais aussi les effets de bord. [Lap94] montre que ceux-ci sont parfois difficilement décelables, et nécessitent une attention particulière.

Pour les tables du lexique grammaire, l'explicitation des informations contenues à divers niveaux constitue une phase d'interprétation indispensable à leur exploitation par des outils d'analyse automatique. [Roc93] propose, par exemple, une formalisation par "automates de référence" pour linéariser les structures recensées. Cette étape manuelle de précompilation nécessite une connaissance conséquente des principes du lexique grammaire, pour permettre ensuite d'associer automatiquement un transducteur d'analyse à chaque entrée.

2.3.4 Points d'entrée

Le problème est maintenant de guider le lecteur vers des parcours cohérents dans un océan de données formelles. Partant du `lexique grammaire`[4], la simplicité de navigation dans un hypertexte W_3 ne demande plus d'explications techniques.

4. URL: http://www-ceril.univ-mlv.fr/LexiqueGrammaire/

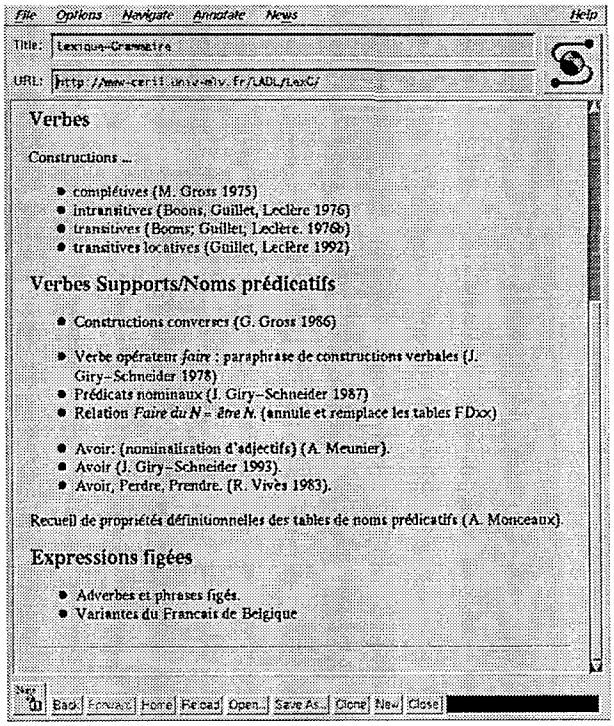


FIG. 2.8 – *Une vue du lexique-grammaire sous Mosaic*

La structure hypertexte la plus simple est l'arborescence, c'est à dire une relation hiérarchique ordonnée, qui formalise assez bien le passage du général au particulier. Les tables syntaxiques peuvent alors être considérées comme unités élémentaires du lexique-grammaire, et former les feuilles des arbres.

Pour documenter des niveaux intermédiaires, les tables syntaxiques peuvent, par exemple, être organisée sous les points de vue suivants:

Catégories majeures Cette décomposition très générale permet évidemment de relier tous les travaux disponibles, soit un arbre à trois niveaux: référence, index du contenu, table. La figure 2.8 montre le document affiché par un des logiciels clients W_3 les plus répandus.

Structures élémentaires La classification par catégories peut être croisée en considérant les structures définitionnelles ([Lec90]) :

$N_0\ V$	$\longrightarrow$ Tables CADV, 31 ...
$N_0\ V\ N_1$	$\longrightarrow$ Tables C1, CAN, CDN, 4, 6, 32A, 32...
$N_0\ V\ Prep\ N_1$	$\longrightarrow$ Tables 1, 2, 7, 8, 33, 34L0 ...
$N_0\ V\ N_1\ Prep\ N_2$	$\longrightarrow$ Tables 9, 11, 12 ...
...	

Cette décomposition peu instanciée est surtout pertinente pour les constructions complétives et les phrases simples, les tables de verbes supports n'y

entrent pas directement. Des structures du type $N\ V_{support}\ N_{predicatif} N_i$ ... ne permettraient pas une répartition très caractéristique. Le verbe support devra donc être mentionné (ex: N_0 *faire Det Vn*).
A titre d'exercice, le lecteur pourra organiser les structures des verbes supports selon un arbre et non une liste.

Codes des tables Ces relations forment un point d'entrée inverse de celui présenté en figure 2.8. En terme documentaire elles ne sont pas forcément redondantes pour autant, puisque le niveau de décomposition des informations peut-être différent.

1-19 : constructions à complétives
33-35 : constructions intransitives
32,36-38 : constructions transitives locatives
F..., A..., E... : noms prédicatifs avec respectivement pour verbe support *faire*, *avoir* et *être* .
P..., C... : adverbes figés ou phrases figées.
$CAN = N_0\ V\ (Ca = deN)$
$CPPN = N_0\ V\ C_1\ Prep\ C_2\ Prep\ C_3$
$PCDC = N_0\ V\ (C\ de\ C)_{Adv}$...

Quelques interdépendances peuvent être intéressantes à souligner comme le déplacement des expressions figées précédemment classées dans la table 5 vers la table C5. Idem pour C7 et C8 ([GL89]).

Classification sémantique Les tables "sémantiquement homogènes" sont peu nombreuses, cette base de description peut être étendue à à des études de phénomènes de moindre étendue: langages spécialisés, ou études multilingues ponctuelles. On reliera par exemple:

Verbes de parole	$\longrightarrow$ Tables 6,9
Verbes psychologiques	$\longrightarrow$ Table 4
Verbes de mouvement ou causatif de mouvement	$\longrightarrow$ Tables 2,3

Etudes multilingues Les lexiques grammaires s'appuie sur les classifications françaises. On trouve par exemple des variantes belges ([LK92]) ou suisses ([Küb95b]) des phrases figées. Existent aussi des adverbes figés italien/français/anglais, des verbes de transfert et de parole français/anglais ([Küb95a]), des noms composés du domaine tennistique grec/français/anglais ([Skl93]), un lexique grammaire comparé avec le Québecois ([Lab88]) ...

Les ambiguïtés entre descriptions comparées adoptant les mêmes conventions d'identification des tables seront levées par préfixation.
Exemple: Pour les expressions figées du français de belgique
A1 $\rightarrow$ B_A1.

La présentation d'études complexes reste aussi à définir, comme par exemple les extensions dérivationnelles de la Table 32RA ([Cle93]), ou l'interprétation sémantique des verbes psychologiques ([YM93]).

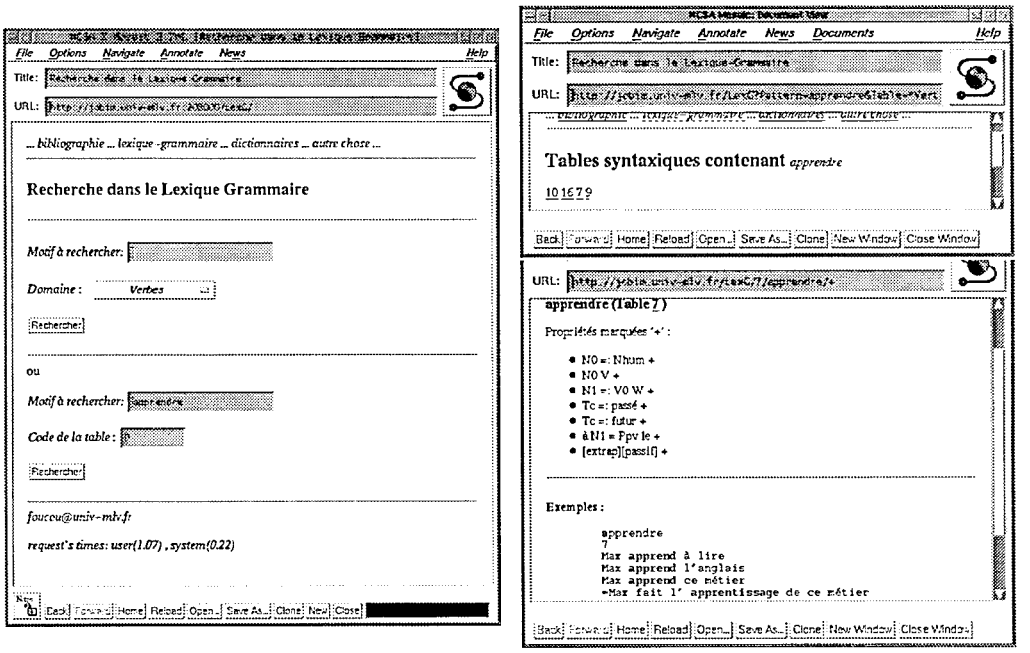


FIG. 2.9 – *Recherche dans les tables*

Ces points d'entrées très généraux ne fournissent certainement pas un niveau de lisibilité suffisant (voir Figure 2.2). Les relations documentées sont hiérarchisées donc simples à parcourir, pourtant les lecteurs peu habitués aux travaux du LADL sentiront immédiatement l'utilité d'une documentation plus précise et les linguistes rompus aux mécanismes du lexique grammaire voudront en extraire des informations pertinentes pour leurs intérêts propres.

Des requêtes simples permettent de croiser la recherche de propriétés ou de formes canoniques dans les tables.

Explicitation des notations

Le découpage en table apparaît donc trop large, et l'adressage de chaque entrée, de chaque symbole devient nécessaire. Chaque notation devant être théoriquement justifiée par des observations syntaxiquement fondées, leur nombre reste limité. L'utilisation d'un nouveau symbole imposerait donc une consultation de l'appareil formel existant. Pourtant les aléas des abréviations font que les codes correspondent à des nomenclatures dont la régularité n'est pas toujours évidente à cerner.

Par exemple on aimerait consulter:

```
<A HREF="/LexiqueGrammaire/Symboles/Nhum.html>subst. humain</A>
```

Il ne s'agit évidemment pas d'imposer au rédacteur le marquage de chaque occurrence des références si leur insertion peut être faite automatiquement.
Par exemple une entrée de la table F9:

```
abasourdissement <A HREF="/LexG/4/abasourdir">abasourdir</A> + + -
```

Les références directes ne sont pas cumulables: dans une entrée adressée `<A HREF="/LexG/4/disposer.Advm">table 4</A>`, on ne pourra pas relier `Advm` à sa définition.

Un ensemble de liens vers des `documents généraux`[5] , du type de ceux proposés ci-dessus, peut être inséré automatiquement, par exemple en bas de chaque page de l'hypertexte.

2.3.5 Propriétés

Selon la représentation matricielle les propriétés sont notées dans les entêtes de colonnes, on en distingue 4 types principaux ([BGL76, p170]):

Distributionnelle : sélection d'une classe de substantifs ou d'une catégorie grammaticale.
Parmi les `300 propriétés`[6] :

```
N0 =: N-hum
    Substantif non humain
N0 =: Nhum
    Substantif humain (répondant à la question Qui?)
N0 =: Ncoll
    Nom collectif
N0 =: Nnc
N0 =: Nnr
    argument non restreint (N, QuP, ...)
N0 =: Npc
    Partie du corps
N0 =: Qu P
    proposition complétive
N0 =: V-n
    nom dérivé d'un verbe
...
```

On trouve bien sûr les équivalents en `N1 =: ...` ou `N2 =: ...`

Structurelle : acceptabilité d'une structure ...
Parmi les `200 propriétés`[7] :

```
N0 V N1 Loc N0pc
N0 V N1 Loc N1pc
N0 V N1 Loc N2
N0 V N1 V0 W
N0 V N1 avec V-n instr
N0 V N1 comme N2
N0 V N1 contre Nhum
```

5. URL: http://www-ceril.univ-mlv.fr/LexiqueGrammaire/Methode/
6. URL: http://www-ceril.univ-mlv.fr/LexiqueGrammaire/Methode/Proprietes/distrib.html
7. URL: http://www-ceril.univ-mlv.fr/LexiqueGrammaire/Methode/Proprietes/structur.html

```
  N0 V N1 dans N2
  N0 V N1 de N2
...
```

Sémantique : précisant une interprétation.

```
  V = convertir en V-n
  V = enlever
  V = mettre
  V = mettre en V-n
  V = rendre Adj
...
```

Complexe : interdépendance de plusieurs colonnes, c'est en fait l'interprétation implicite des marques binaires, au sens où une structure ne sera acceptée qu'avec des arguments sélectionnés, ce qui n'implique d'ailleurs pas que toute structure marquée "+" acceptera n'importe quel argument.

```
Transformations
  [Prép z.]
  [extrap]
  [extrap][passif]
  [passif de]
  [passif par]
  [pc z.]

Formes restructurées
  "P", V N0 à N2
  (Adj) (Qu P)
  (N0 V Loc N1)
  (N1) (Adj)
  (N1) (V-ant W)
  (N1) (V-inf W)
  (N1) (de V-inf W)
  (N1) (être Adj)
  (être Adj) (Qu P)

...
```

Les propriétés doivent bien sûr être remises dans le contexte des tables où elles apparaissent pour essayer d'en comprendre les finesses exactes.

Les discussions sur la portée exacte des propriétés notées dans les colonnes sont souvent laissées de côté, on peut noter par exemple que certaines propriétés définitionnelles peuvent ne pas figurer dans la table parce que redondantes.

Par exemple, la table 38LH , dite à "objet humain" ne contient pas de colonne $N_1 := Nhum$ ([GL92, 409]).

On note aussi que les propriétés structurelles interviennent surtout dans les tables de verbes simples, l'aspect distributionnel ou lexical l'emporte dans les des

études ultérieures. C'est particulièrement évident pour les formes figées.

La liste des `symboles`[8] de base issus de [Gro75, BGL76, GL92], contient par exemple :

N... = argument nominal, Nhum pour humain, N-hum, Nnr, N-a, N-v ...
V... = verbe (Vinf, Vsup, V-ant),
U = verbe opérateur (Table 1)
QuP = complétive (ou QuPsubj)
W = suite quelconque d'arguments, même si dans le DELAF compilé dans INTEX (version 2.0) `<V:W>` est un verbe à l'infinitif
Neg négation (ne pas, ne guère ...)
Adv adverbe (AdvFutur ou AdvManière)
...

Ces symboles ne sont pas toujours automatiquement interprétables. Lorsque le recensement en extension apparaît possible, un lien hypertexte permettra de d'enrichir progressivement l'ensemble des sélections, en les stockant dans le document associé, et ce, sans surcharger la notation matricielle. Cela s'imposerait notamment pour les phrases figées où l'information réside plus souvent au niveau des unités lexicales que des propriétés.

2.3.6 Entrées

La conversion des fichiers informatiques des tables, format ASCII pour les moins récentes ou Excel, est faite quasi automatiquement, mais une certaine normalisation s'impose pour s'assurer de l'utilisation des caractères spéciaux.

Les entrées peuvent être [Gro75, p61-64]:

- des unités lexicales simples identifiées dans DELAS
- des unités composées: `conclure/(35R.33)/en faveur/++++-------+--+-------+-+`
- des unités composites? `(an09) B.A.ba/++----`
- des unités morphologiques : Un exemple issu de la table `AN03` d'Annie Meunier ([Meu81])

```
agressif      f      vité     -++-++++
allègre       ègre   égresse  -++--+++
altruiste     te     me       -++-+-++
ambigu        -      ïté      ++++--++
...
```

Cette dernière notation des relations morphologiques permet d'abréger les spécifications mais ne facilite pas la lecture, et encore moins les recherches automatiques. Il faut en effet conserver la signification différentielle des suffixes.

8. URL: http://www-ceril.univ-mlv.fr/LexiqueGrammaire/Methode/notation.html

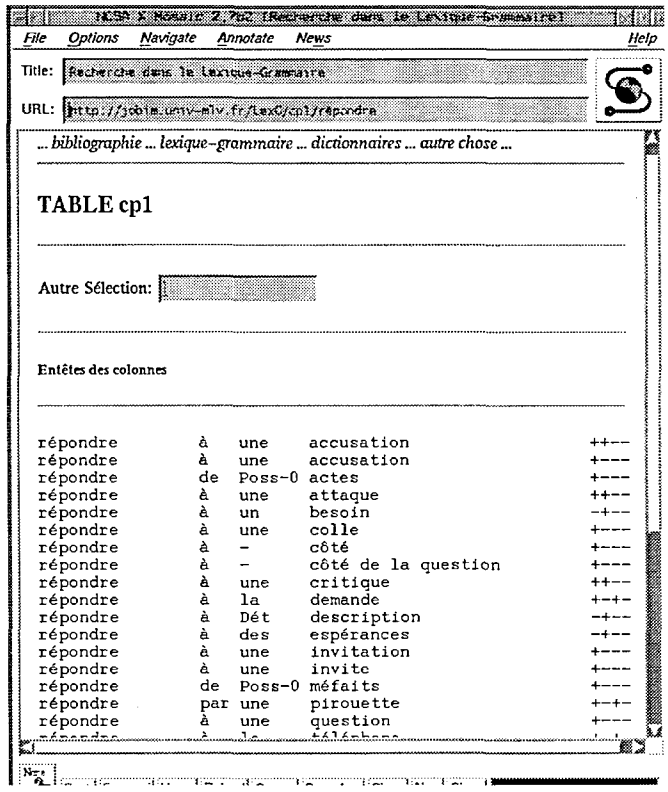


FIG. 2.10 – *Extrait de la table CP1*

Sur le serveur on choisira de rétablir les unités complètes, quitte à définir, à terme, les adressages du type :
`/support/avoir?`*agressif* $\longrightarrow$ agressivité
ou plus généralement:
`/adjectivation?`*nom* $\longrightarrow \Sigma adjectifs$

Le codage des tables normalisé par HTML3.2 permet un marquage non ambigu. Ci dessous, quelques entrées de la Table F9 ([GS78]) :

```
chagrin<TD><A HREF="/LexG/4/chagriner">chagriner</A><TD>+<TD>+<TD>-
...
consolation<TD><A HREF="/LexG/13/consoler">consoler</A><TD>+<TD>+<TD>-
...
ruine<TD><A HREF="/LexG/11/ruiner">ruiner</A><TD>+<TD>+<TD>-
```

Quelques exemples de recherches automatiques seront donnés Section 3.1.4.

2.4 Corpus

La stratégie n'est pas d'accumuler *en local* mais de recenser les liens intéressants. Le but est de favoriser l'accès à différents textes pour expérimenter divers outils d'analyse automatique : étiquetage, levée d'ambiguïté ou d'indexation.

La structuration permettra aussi des analyses sur des domaines spécialisés ou sur des niveaux de langues particuliers.

2.4.1 Stockage local

Il existe quelques sites spécialisés dans les corpus comme `Elsenet`[9] ou `LDC`[10] mais les accès sont souvent tarifés et le français y est ultra minoritaire. Il est donc difficile de trouver un service fournissant des corpus sur lequel on pourrait architecturer notre environnement de traitements linguistiques.

Pour des raisons historiques, opportunités ou acquisitions, quelques textes sont disponibles sur notre serveur. Certains sont bien identifiés, comme le contenu des CD-ROM du journal **Le Monde** , mais beaucoup résultent d'apports ponctuels non répertoriés. Sans aller jusqu'à la base de données documentaire, la première étape est de structurer autant que possible cette foule de choses plus ou moins intéressantes,

Nous obtenons l'arborescence suivante :

	Répertoire	Taille (Ko)
/Corpus/	AFP	1505
	Admin	1681
	Divers	1742
	English	1987
	ExemplesdEncodage	1261
	Large	5851
	LeMonde (gzippé)	125974
	Litter	3373
	Sciences	562
	WWW	1720

Ce dernier répertoire servant de réceptacle aux documents récupérés via W_3. Les textes étant le plus souvent sous *copyright*, il n'est pas question de les rendre disponibles tels que sur le serveur. Ils seront accessibles en local pour les recherches et analyses.

2.4.2 Collecte des adresses

Les textes en français étant aujourd'hui plus rares que ceux en anglais, il faut chercher un peu plus pour trouver des documents présentant un intérêt pour les traitements automatiques. En partant des serveurs quasi-officiels `Classement`

9. URL: http://www.ims.uni-stuttgart.de/info/Newspapers.html
10. URL: http://www.ldc.upenn.edu/

des serveurs français[11] Bottin Internet du Québec[12], mais aussi Elsenet[13], on trouve des documents de tailles significatives, comme comme par exemple *le Grand Secret* du Dr. Gubler (220Ko)[14].

On trouve aussi un certain nombre de documents périodiques. Pour peu qu'elles soient stables et structurées, autant stocker les adresses (URL) de ces sources de corpus plutôt que d'entreprendre un rapatriement si l'on en a pas d'utilisation précise en vue.

Par exemple: les communiqués de presse de l'ONU (Genève ou New York) ouvre l'accès à leur stockage interne:

```
Index of http://www.unog.ch/news/
Name          Last modified     Size  Description

newsen/         25-Sep-96 17:25     -
newseng         12-Dec-95 11:47    2K
newsfr/         25-Sep-96 17:26     -
newsfrny/       25-Sep-96 17:28     -
presrele.htm    13-Sep-96 16:22    47K  Press Releases
rcvr.htm        19-Jan-96 09:15     4K  RECEIVING

      ------------------------------

Index of http://www.unog.ch/news/newsfr
     Name          Last modified     Size  Description

     08011170.fre    01-Aug-96 07:53    40K
     08011171.fre    01-Aug-96 07:53    12K
     08021175.fre    02-Aug-96 14:05    30K
      ...
```

Cependant certains services ne proposent pas d'archives, la même URL menant à un document mis à jour:

Domaine	Taille / Période
bulletin de Météo France	$2K/jour$
bulletin pour Québec	$1K/jour$
Résumé de l'actualité mondiale AFP	$10K/jour$
Résumé de l'actualité mondiale TF1	$20K/jour$
Revue de presse RFI	$7K/jour$
Communiqué de presse + revue de presse du Quay d'Orsay	$30K/jour$
Communiqués du Ministère de la culture	??
...	

On peut ainsi facilement accumuler 100 ko de textes par jour. Nous nous intéressons ici uniquement aux textes directement exploitables, pour des services

11. URL: http://www.urec.fr/France/Classement/Themes/serveurs_par_themes.html
12. URL: http://biq.qc.ca/
13. URL: http://www.ims.uni-stuttgart.de/info/Newspapers.html
14. URL: http://www.cam.org/ pfg/

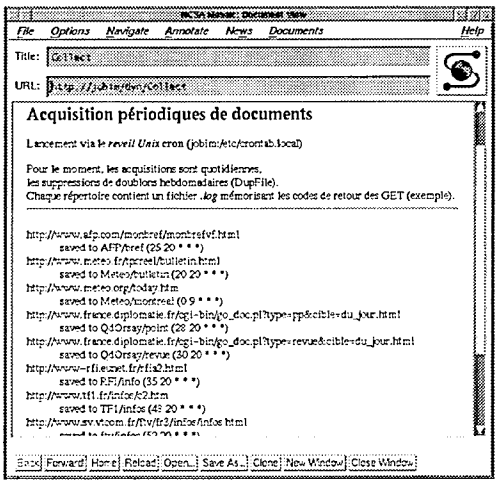


FIG. 2.11 – *Configuration des acquisitions périodiques*

qui remonte à 1994. L'essor de W_3 rend disponibles de plus en plus de sources avec un formatage évolué : HTML comme `Le Soir de Bruxelles`[15], PDF comme `Libération`[16] ou audiovisuelles (`.mov` ou `.wav`) comme `France Info`[17].
On pourrait donc aisément structurer 1Mo par jour rien que pour le français, car ces sources sont tout aussi exploitables, simplement la configuration préalable demande un peu plus de connaissance et d'attention.

Le traitement de ces sources nécessitent des mécanismes de vérification, en effet la mise à jour des serveurs n'est pas toujours ponctuelle, notamment les samedis et dimanches, on retrouve donc deux ou trois fois le même document.

A noter que ces incohérences dépendent de l'investissement des institutions dans leur services. Meteo France et l'AFP dont le métier est la diffusion d'informations, sont le plus souvent irréprochables, ce qui n'est pas le cas chez RFI qui peut laisser 4 jours d'affilée la même revue de presse *du jour*.

Un rapide tri sur la taille des fichiers permettra d'éliminer ces incohérences, avant de s'intéresser au `contenu`[18].

Les problèmes réseaux, plus ou moins ponctuels peuvent aussi altérer ces récupérations aveugles, la commande `GET.daily` pourra être raffinée à l'infini pour obtenir une qualité de service équivalente à celle de la source, mais la nature du corpus d'implique pas forcément de telles attentions.

La Figure 2.11 montre le paramétrage des acquisitions quotidiennes. Les 5 derniers champs spécifient les dates de lancement de `GET.daily` par l'horloge Unix `cron`.
Par exemple `25 20 * * *` signifie tous les jours à 20h25.

15. URL: http://www.lesoir.be/
16. URL: http://www.liberation.fr/
17. URL: http://www.radiofrance.fr/
18. URL: http://www-ceril.univ-mlv.fr/Corpus/Ressources/WWW/Incr/

2.4.3 Filtrages

La qualité linguistique des données est très difficile à évaluer, mais l'utilisation et la documentation de sources répertoriées peut optimiser leur exploitation.

Une condition minimale d'expérimentation est la normalisation du codage, pour l'instant seulement HTML. Les marquages plus riches au niveau de la structure logique, type TEI, permettront peut-être d'appliquer des outils de correction avec plus de discernement ([BD92b]).

Comme pour tout corpus, l'examen attentif de ces documents ouvre un large champ de questions et d'idées d'expérimentation. Il est évident que les opérations pertinentes, pour tirer parti des similitudes et redondances, sont innombrables ([GT94]). Comment traiter l'exemple ci dessous, extrait de la synthèse de l'actualité de TF1 ?

```
< Actualité du 13/08/96
< Une Britannique de 31 ans enceinte de huit enfanys après avoir subi
un traitement contre la stérilité a annoncé samedi qu'elle voulait
garder tous ses bébés.
---
> Actualité du 14/08/96
> Une Britannique de 31 ans enceinte de huit enfants après avoir subi
un traitement contre la stérilité a annoncé samedi qu'elle voulait
garder tous ses bébés.
```

Dans tous les cas, le filtrage des documents peut être défini clairement pour en sortir les informations linguistiquement exploitables. Essentiellement la suppression du formatage HTML de présentation (typographie et liens), environ 10% des données transférées. Les traitements linguistiques seront abordés au chapitre 4.1.2.

Par exemple:

```
<HTML><HEAD>
<TITLE>AFP - Le Monde en bref</TITLE>
</HEAD>
<body bgcolor="FFFFFF">
<A NAME="top"></A><font size=-1>
<a href="/Monbref/MonBrefVA.html">World News Roundup</a><strong> | </strong>
<a href="/Monbref/MonBrefVE.html">El Mundo Breve</a><strong> | </strong>
<a href="/AFP_VF/afpaccueil.html"> Retour page d'accueil </a></font>
<h1> AFP : Le Monde en bref</h1>
<font size=-1>
Résumé de l'actualité mondiale à
05h44 GMT, le 04 juin 1996.
<a href="#copyright">&copy; AFP</a></font> <hr>
</blockquote>
<IMG SRC="/charte/reddot.GIF" WIDTH="14" HEIGHT="14" ALT="*">
<STRONG>OTAN-REFORME</STRONG>
 <blockquote> Les Européens ont arraché à leurs alliés américains de
l'OTAN, lundi à Berlin, ville symbole de la guerre froide, le pouvoir
de mener à l'avenir des opérations militaires de manière autonome,
mais les Etats-Unis ont aussit&ocirc;t considéré peu probable une
telle éventualité.  <br>
 Les ministres des Affaires étrangères de
l'Alliance ont décidé de définir un "pilier européen" au sein de
l'OTAN, en identifiant à l'avance des hommes et des matériels pouvant
&ecirc;tre utilisés à l'horizon 2000 dans des opérations militaires
dirigées par l'Union de l'Europe Occidentale (UEO).
(AFP)</blockquote> </blockquote>
 ...
```

donnera

```
World News Roundup | El Mundo Breve | Retour page d'accueil
AFP : Le Monde en bref
Résumé de l'actualité mondiale à 05h44 GMT, le 04 juin 1996. © AFP
      OTAN-REFORME
    Les Européens ont arraché à leurs alliés américains de l'OTAN, lundi à
    Berlin, ville symbole de la guerre froide, le pouvoir de mener à l'avenir
    des opérations militaires de manière autonome, mais les Etats-Unis ont
    aussitôt considéré peu probable une telle éventualité.
    Les ministres des Affaires étrangères de l'Alliance ont décidé de définir
    un "pilier européen" au sein de l'OTAN, en identifiant à l'avance des
    hommes et des matériels pouvant être utilisés à l'horizon 2000 dans des
    opérations militaires dirigées par l'Union de l'Europe Occidentale
    (UEO). (AFP)
```

Exemple pour ce qui est de la qualité: *Meteo France* fournit ses bulletins sans accents et parfois même tout en majuscules.

```
FPFX42 LFPW 240904 ORIGINE: METEO-FRANCE, TOULOUSE. BULLETIN FRANCE DE
LA MI-JOURNEE DU MARDI 24 SEPTEMBRE 1996.<p>
 CET APRES-MIDI: BEAUCOUP DE NUAGES SUR LE NORD-EST ET LE SUD-OUEST,
VARIABLE AILLEURS.<p>
 SUR LES REGIONS DU NORD-EST JUSQU'AU NORD DES ALPES, LA GRISAILLE
RESTERA DE MISE ET IL FAUDRA S'ACCOMODER DE PETITES PLUIES DE TEMPS A
AUTRE. AU MAXIMUM DE LA JOURNEE, LES TEMPERATURES ATTEINDRONT 14/15
DEGRES SEULEMENT.<p>
 SUR LE SUD-OUEST, CE NE SERA A PEINE MIEUX. LES NUAGES ET PETITES
PLUIES CONCERNERONT ENCORE MIDI-PYRENEES ALORS QUE QUELQUES TROUEES DE
SOLEIL REVIENDRONT SUR L'AQUITAINE.<p>
 SUR LA BRETAGNE ET SUR TOUTE LA FACADE ATLANTIQUE, LES NUAGES
L'EMPORTERONT SOUVENT SUR LES TIMIDES ECLAIRCIES.
```

Les sources étant identifiées on peut aussi filtrer les notations spécifiques. La Figure 2.12 montre une procédure Perl filtrant des bulletins météo.

L'utilisation directe de la librairie LWP (voir annexe B) nuit considérablement à la lisibilité donc à la maintenance du programme. Cela présente tout de même l'avantage de fournir une analyse *robuste* de la structure HTML du document.

```
sub StructMeteo {
    my ($e)  = @_;
    local $text;
    sub meteoback {  # Recherche de la premiere balise <P>
        my ($node)= @_;
        if ($node->starttag() =~ /<p>/i) {
            if (ref($node->content()->[0]) eq '') {
                $text .=  $node->content()->[0] . "\n\n";
                return 0;
            }
        }
        return 1;
    }
   $e->traverse(\&meteoback, 1);
   $text =~ s/voici les temperatures relevees.*$//si;
    print $text;
}
```

FIG. 2.12 – *Filtrage de structure HTML*

Il est évident que la description des *structures documentaires* est un travail à renouveler sans cesse, voir l' `exemple`[19] extrême ci dessous:

```
                                ANDROMAQUE
                                TRAGÉDIE
                          REPRÉSENTÉE POUR LA PREMIERE FOIS
                            DANS L'APPARTEMENT DE LA REINE
                           LE 17e DU MOIS DE NOVEMBRE 1667
                                     PAR
                         LES COMÉDIENS DE L'HOTEL DE BOURGOGNE
                              ACTE PREMIER
                              SCENE PREMIERE. --- ORESTE, PYLADE.
ORESTE.                --- Oui, puisque je retrouve un ami si fidèle,
                           Ma fortune va prendre une face nouvelle;
                           Et déjà son courroux semble s'être adouci,
                           Depuis qu'elle a pris soin de nous rejoindre ici.
                         5 Qui l'eût dit, qu'un rivage à mes v*oeux si funeste
                           Présenterait d'abord Pylade aux yeux d'Oreste?
                           Qu'après plus de six mois que je t'avais perdu,
                           A la cour de Pyrrhus tu me serais rendu?
PYLADE.                --- J'en rends grâces au Ciel, qui, m'arrêtant sans cesse,
                        10 Semblait m'avoir fermé le chemin de la Grèce,
                           Depuis le jour fatal que la fureur des eaux
                           Presque aux yeux de l'Épire écarta nos vaisseaux.
                           Combien dans cet exil ai-je souffert d'alarmes!
                           Combien à vos malheurs ai-je donné de larmes,
                        15 Craignant toujours pour vous quelque nouveau danger
                           Que ma triste amitié ne pouvait partager!
                           Surtout je redoutais cette mélancolie
...
```

19. URL: http://www.brookes.ac.uk/schools/sol/andromaq.html

Chapitre 3

Développement des services

L'essor de W_3 s'est dans un premier temps appuyé sur la rédaction ou la conversion de documents statiques. Aujourd'hui la notion de document, propre aux clients (souvent des logiciels d'affichage), se dilue dans le nombre sans cesse croissant de serveurs (comprendre machines ou sites) et de serveurs (comprendre logiciels diffusants).

Entre rédaction et programmation, la notion de service permet, et/ou contraint, de repenser les habitudes de développement des applications informatiques. L'utilisation des possibilités de W_3 permet d'envisager de nombreuses solutions aux problèmes souvent occultés d'organisation et de dimensionnement des ressources matérielles et logicielles.

La mise en place d'un serveur doit répondre à deux préoccupations :

- Croiser les références pour multiplier les parcours hypertextes.
- Assurer la cohérence des URL.

Ces deux aspects se révèlent contradictoires, car plus les parcours sont entremêlés, plus ils sont difficiles à construire et plus il y aura de sources d'erreur ou d'inconsistance. Nous discuterons ici des principes de développement d'un serveur et de leurs implications quant à la robustesse et la facilité d'utilisation des services. Il est évidemment très difficiles de définir un cas général : les ressources sont par essence hétérogènes et leurs relations difficilement cernables a priori. Dans le chapitre 4, nous nous attacherons à montrer les besoins et les contraintes spécifiques aux traitements des langues naturelles, et plus particulièrement aux opérations sur corpus.

Toute la difficulté sera de maintenir la cohérence de ces ressources pour que ces aspects restent totalement transparents du point de vue du client, qui recevra en tout état de cause un document.

3.1 Définitions des services

Le modèle client/serveur de W_3 est extrêmement souple quant à la définition des services à fournir. En fait, la sémantique de la partie *fichier* des URL (voir Figure 1.6) est laissée à l'entière discrétion du serveur

3.1.1 Typage des ressources

Pour clarifier les concepts, largement masqués par la facilité de rédaction HTML, considérons la typologie suivante :

Ressource stable comme de la documentation.
Le lexique grammaire par exemple n'évolue pas tous les jours. Une structure s'appuyant sur les systèmes de fichiers et répertoires habituels sera tout à fait adaptée.

Ressource dynamique produite par un programme car nécessitant des mises à jour fréquentes et automatisables.
Les processus de mise à jour peuvent être définis d'innombrables manière et correspondent à des qualités de services différentes.
Par exemple un compteur est un document affichant le nombre de clients ayant accédés à une ressource donnée. Selon la fréquence des accès cette information sera plus ou moins *juste*, donc de plus ou moins bonne qualité. La base de temps sera dépendante des clients.
Ce ne sera pas le cas, par exemple, pour un document indiquant la date courante. Sa mise à jour sera uniquement conditionnée par l'horloge.

Les problèmes de *dénombrabilité* surviennent aussi lorsque l'on propose des entrées interactives.
Prenons par un exemple, un service qui demande un nombre pour donner sa transcription en français. Une résolution exhaustive statique ne serait pas des plus pertinentes, c'est donc un programme qui calculera le document résultat.

Ressource temporaire utile à la construction d'autres ressources mais pas publiée telle que.
Si on prend en compte les accès concurrents et les traitements asynchrones une ressource peut avoir un statut **en cours de construction**, **réservée**, **à jour**, ...

Ressource complexe intégrant des ressources existantes. On définit alors des dépendances entre les ressources, ce pourra être une synthèse, un chaînage, une liste ou des structures plus complexes, comme par exemple les résultats des requêtes de traitements de corpus.
Exemple de ressource complexe, la liste des fichiers contenus dans un répertoire, produite par la plupart des logiciels serveurs généralistes

Ressources dupliquées

Certaines ressources extérieures présentent un intérêt certain mais ne fournissent pas un qualité de service suffisante pour l'utilisation que l'on veut en faire. Actuellement, le moyen le plus sain d'améliorer les choses est de répartir les accès sur des sites officiellement déclarés *miroirs* du service original. Il est évident que tout un chacun ne peut prétendre devenir miroir d'`Altavista`, lequel a d'ailleurs des temps de réponse tout à fait satisfaisants.

Nous maintenons par exemple une `copie locale`[1] du `Free On-line Dictionary of Computing`[2] . Ce dictionnaire contient près de 10,000 sigles, abréviations et définitions. Concrètement : un seul fichier de près de 4Mo, éditable directement. Sa maintenance est centralisée par son `éditeur`[3] et quelques éditeurs associés qui valident les définitions que les utilisateurs d'*Internet* enrichissent tous les jours.

Il faut donc définir un protocole de collaboration avec les responsables du site maître pour dupliquer les services et automatiser la mise à jour des informations. Rien n'est actuellement `normalisé`[4] dans ce domaine. Les vieilles techniques de génie logiciel sont donc adaptées à cette gestion documentaire, et chaque nuit un programme calcule, à la source, les différences entre deux versions successives du dictionnaire par la commande standard `diff`. Ces résultats sont accessibles à une URL convenue pour les sites miroirs.

La commande tout aussi standard `patch` permet alors de générer la nouvelle version à partir de la précédente et des *diffs*. Ci dessous, un exemple de résultat de `diff`. Les contextes des différences sont indiqués par leur position dans les fichiers comparés. Ces positions sont des intervalles de numéros de ligne, `diff` et `patch` n'opérant efficacement que sur des fichiers *texte*.

```
*** 49183,49185 ****

!       httpd is designed to be small and fast and to work with most
        HTTP/0.9 and HTTP/1.0 {browser}s.  You can customise your
--- 49194,49196 ----

!       HTTPd is designed to be small and fast and to work with most
        HTTP/0.9 and HTTP/1.0 {browser}s.  You can customise your
***************
*** 108072,108073 ****
--- 108161,108171 ----

+ typeface
+       <text> The style or design of a {font}.  Other independent
+       parameters are size, boldness (thickness of lines), and
+       obliqueness (a sheer transformation applied to the characters,
+       not to be confused with a specifically designed italic font).
+       (02 Aug 1996)
+
  typed lambda-calculus
```

1. URL: http://www-igm.univ-mlv.fr/FolDoC/
2. URL: http://wombat.doc.ic.ac.uk/
3. URL: mailto:dbh@doc.ic.ac.uk
4. URL: http://www.w3.org/pub/WWW/Propagation/

En l'absence de solution plus conviviale, c'est un modèle de collaboration tout à fait efficace dont la maintenance des données linguistiques pourrait s'inspirer.

Ressources coopératives

Offrir des services W_3 c'est, au départ, permettre l'accès à ses propres données. Avec la multiplication des sources, l'organisation des documents devient un service en soi.

Nous avons vu Section 1.4.3 que les indexeurs automatiques permettent des recherches sur de vastes domaines. Si on restreint le domaine d'exploration à notre serveur, on obtient un index de ses propres pages en profitant des ressources d'un service externe.

Par exemple `Altavista` permet avec des requêtes simples de lister :
`host:www-ceril url:Dictionnaire*`

```
Documents 1-7 of 7 matching the query, in no particular order.

DELAF lookup (v5)          [18Jun96] bibliography ... lexicon-gramm
DELAC                      [18Jun96] Dictionnaire électronique des
Codes DELAF                [19Jun96] Codes flexionnels. La flexion
Consultation du DELAF      [19Jun96] bibliographie ... lexique-gram
DELAS                      [20Jun96] bibliographie ... lexique-gram
Marques sémantiques        [21Jun96] Marques sémantiques du DELAS.
CODES DELAS                [27Sep96] Codes DELAS. (version 8, 1996)
```

`host:www-ceril url:LexiqueGrammaire*`

```
Tip: To find a bed-time story: "fairy tale" +frog -dragon

Documents 1-10 of 21 matching the query, in no particular order.

Illustration du lexique-g [21Jun96] bibliographie ... lexique-gram
Glossaire des notions sur [21Jun96] Quelques définitions plus ou m
Symboles pour Proprietes  [26Jun96] Système de notation. Les propr
Description des tables de [26Jun96] Constructions Complétives. (M.
Constructions  Intransiti [26Jun96] Constructions Intransitives. (
Transitives               [26Jun96] Constructions Transitives. (Bo
Adverbes figes            [26Jun96] Tables d'Adverbes figés. Mauri
No Title                  [26Jun96] Phrases figées. Gross, Maurice
Lexique-Grammaire du Fran [27Jun96] Constructions à Verbes Support
Liste des proprietes      [27Jun96] Glossaire des propriétés. Les
```

Il est possible de s'investir un peu plus dans cette coopération. On voit par exemple que beaucoup de descriptions se limitent à
`'' bibliographie ... lexique-gram''`

En effet la majorité des pages de notre serveur ont une entête générique regroupant les principaux points d'entrée. Pour enrichir cette `indexation externe`[5], `Altavista` peut tenir compte de méta-informations balisées dans les documents, par exemple :

```
<META  name="description" content="Liste des dictionnaires électroniques">
<META  name="keywords" content="flexion catégorisation grammaticale">
```

5. URL: http://www.altavista.digital.com/cgi-bin/query?pg=q&what=web&q=host

En affinant ses informations on peut donc affiner la qualité de service externe, et profiter de cette coopération progressive. Les temps de réponse sont de l'ordre de la journée. Bien entendu moyennant finance, ces services externes seront adaptables et personnalisables à l'infini.

3.1.2 Définition du schéma d'adressage

Pour qu'un client puisse accéder aux ressources d'un serveur, il doit en connaître au moins une URL. Le client répète: - *Demande d'une URL*
- *Attente de la réponse*
- *Affichage du résultat.*

Les états stables d'un client sont matérialisés par les documents issus d'un ensemble de ressources distinctes formant un processus global (voir Figure 4.6). L'enchaînement des états est défini par l'utilisateur qui active manuellement les liens vers ce qu'il considère comme la suite. Ces liens pourront être contenus dans le document courant, mais rien n'empêche les clients de revenir en arrière, de trouver des liens dans leurs marque-pages ou sur d'autres serveurs.

Plus les ressources sont factorisables, plus on est tenté de programmer la production des constituants élémentaires, donc moins leurs références seront faciles à déterminer. La définition des adresses des ressources dynamiques se fait en considérant le processus dans sa globalité, et en le décomposant en opérations élémentaires. Cette décomposition doit tenir compte du processus à interfacer mais aussi des opérations inhérentes aux caractéristiques de W_3. Plus le nombre d'états sera grand plus la sémantique des programmes sera difficile à interpréter, donc plus la consistance de l'adressage sera difficile à vérifier.

Pour clarifier les choses nous considérerons les deux cas extrêmes de définition des adresses des documents (schématisés Figure 3.1) :

a) Adresse unique pour toutes les ressources.

b) Adresse propre à chaque ressource.

Session

Pour que la solution (a) permette d'adresser plusieurs ressources, il faut définir les séquences à consulter et mémoriser à quelle étape en est rendu chaque client.

Comme nous l'avons vu Section 1.3.1, la robustesse du modèle client/serveur W_3 s'appuie sur des échanges d'informations en mode **non connecté**. Un serveur reçoit donc une suite de requêtes sans que le protocole lui permette de les ordonner. Par dessus un protocole non connecté, mémoriser l'état atteint par le client, revient à allouer une *clé d'identification* par client, et pour chaque requête ou chaque événement prédéfini, faire évoluer cet état. Plusieurs solutions sont possibles comme considérer l'adresse IP du client ou un mot de passe comme identificateur. A noter qu'en considérant un état pour un groupe de clients, par

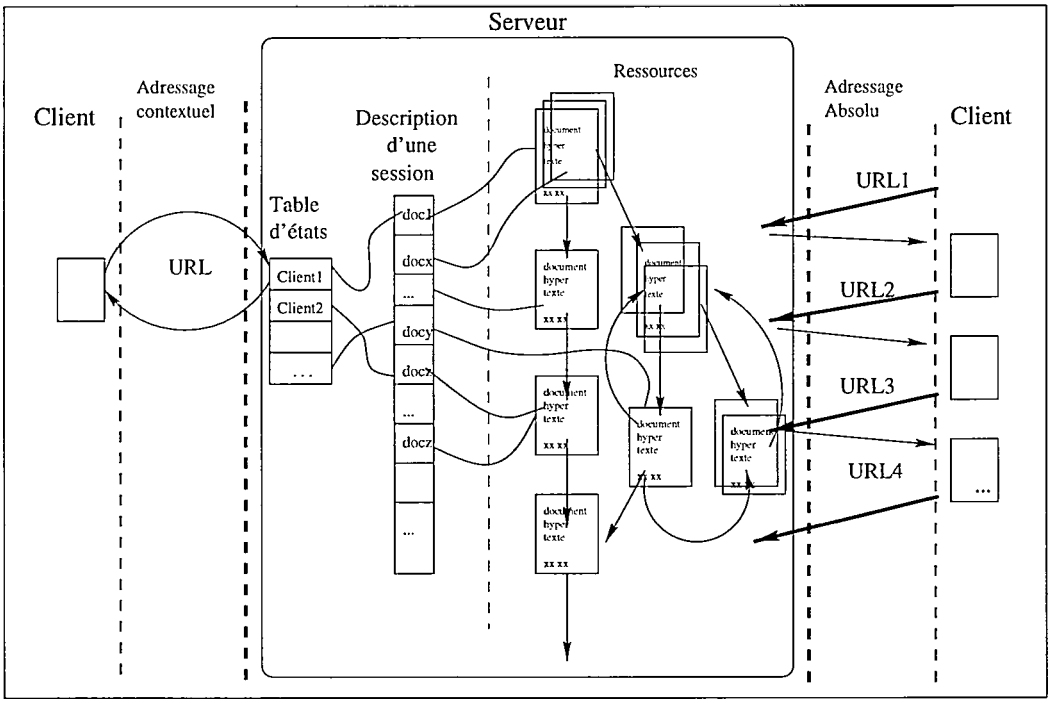


FIG. 3.1 – *Schémas d'adressage des ressources*

exemple tous les utilisateurs d'une machine donnée, ou tous les membres d'un groupe de travail, on ouvre naturellement les possibilités d'interactions.

On parlera de **session**, c'est à dire d'une suite ordonnée d'interactions. La description de ces sessions restant interne au serveur, les mécanismes de vérification seront plus faciles à mettre en oeuvre.

Le problème technique essentiel de la solution (a) réside dans la durée de vie des clés. Dans cette relation entre client et serveur, les connexions sont initialisées, maintenues durant tout le processus, puis libérées.
Combien de temps dure une session? Et quelle stratégie de récupération serait pertinente en cas de perte des informations d'état?

On devie un peu de la philosophie de W_3, puisque l'on fait les choix à la place de l'utilisateur en l'obligeant à passer par un certain nombre d'étapes.

Simuler des sessions pour les clients empêche ceux-ci de relier leurs propres documents aux ressources du serveur, puisqu'une adresse unique ne permet pas de les différencier. Cela bloque complètement toute rédaction hypertexte! Pour les besoins des services de documentation largement accessibles et manipulables, nous ne pourrons nous satisfaire de telles contraintes.

Adressage absolu

A l'opposé, la solution (b) exige un adressage absolu. Une adresse sera une clé d'identification d'un état atteignable du système global. Pour que le client puisse poursuivre ses explorations chaque adresse devra lui être envoyée, donc être reliée à une autre ressource. La relation texte/référence des liens hypertexte impliquera une imbrication très forte entre rédaction et programmation.

Tous les parcours ne sont pas pertinents, loin de là, il ne s'agit donc pas de produire un graphe complet, en reliant chaque page à toutes les autres! La mémorisation des adresses est laissée à l'entière charge du client. S'il juge une ressource intéressante, il devra l'insérer dans ses marque-pages ou dans son historique pour y retourner directement, sinon il devra reparcourir le chemin depuis le dernier point qu'il a mémorisé. Soulignons que dans la solution (a) le client n'avait aucun moyen de choisir ses parcours.

Les parcours étant définis logiquement, cela ne présente pas d'inconvénient, et laisse une grande latitude dans les niveaux de lecture de chacun.

La solution (b) permet un partage naturel de l'accès aux traitements de ressources linguistiques tel qu'il nous intéresse. Elle est particulièrement adaptée lorsque la structure globale est connue, donc pour les ressources stables. Si les ressources sont directement accessibles, comme par exemple des fichiers, la rédaction peut s'appuyer sur les outils usuels de gestion des répertoires.

Remarque: Comme nous l'avons vu Section 1.3.2, l'URL porte des informations constantes: noms du serveur et du client, leur adresse IP, port TCP utilisé (80 par défaut), l'identification de l'utilisateur (nom et mot de passe). La partie variable de l'URL dénote le nom local de la ressource. Dans le protocole HTTP, des informations additionnelles peuvent être associées hors URL. Par exemple le document source (*referer*), des variables saisies ou cachées (méthode POST). Netscape permet aussi de manipuler des valeurs mémorisées par le client (*Cookies*).

Les URL seront autant que possible les adresses des ressources. Le choix des notations pour les informations échangées est donc très vaste, nous privilégierons la lisibilité des URL, pour que les rédacteurs puissent les manipuler simplement en connaissance de cause.

3.1.3 Problèmes d'implémentation

Si l'on s'intéresse au coût du modèle client/serveur W_3, on s'aperçoit rapidement que la robustesse des communications conduit à des méthodes spécifiques de développement des services pour traiter les problèmes de protocole sans état.

Considérons l'exemple d'un processus local écrivant sur sa sortie standard, attachée à un terminal. Pour la consultation d'un document volumineux si le processus s'arrête à chaque page, il a un comportement homogène à deux états: affichage et attente. La même action, une entrée au clavier ou un signal d'horloge,

permettra de passer d'un état à l'autre.

La solution (a) permet de conserver la notion habituelle de contexte, qui est implicite dans tout programme informatique. En demandant la même URL on obtiendra les pages dans l'ordre souhaité.

⇒ `http://server/htmore/fichier`

Dans la solution (b), tous les comportements possibles doivent être dénotés, il y aura donc autant d'états que de pages.

⇒ `http://server/htmore/fichier/page`

Dans le cas d'un service W_3, on est à chaque requête dans la situation du programme qui doit trouver à quelle page il doit commencer.

Si l'on ajoute une variable définissant la taille des pages, notre service W_3, à chaque lancement, doit trouver la taille d'une page et calculer où est la énième à envoyer. Il y aura alors autant d'états que le nombre de pages multiplié par le nombre de taille de pages possibles!

⇒ `http://server/htmore/fichier/page/nlignes`

Avec une méthode naïve, l'envoi de la page 1 ne présente pas de particularité, il faut lire n lignes pour les envoyer au client. La deuxième page oblige à relire la première, puis à lire les n lignes à renvoyer ...

Cette méthode gloutonne est une adaptation sommaire des habitudes de programmation impérative: on retrouve le contexte du programme en répétant son exécution, donc en refaisant en interne le chemin menant à la ressource demandée.

De nombreuses optimisations sont possibles. Il peut s'agir de solutions particulières.

Pour notre exemple, on peut imaginer d'envoyer en référence l'offset dans le fichier plutôt que le numéro de page.

⇒ `http://server/htmore/fichier/offset/nlignes`

Il est évident que ce sera au détriment de la lisibilité des URL, alors que les URL doivent, autant que possible, rester synthétiques et claires pour permettre aux utilisateurs de faire référence aux ressources de manière simple lors de la rédaction de leurs documents. L'emploi d'alias permet de réduire la longueur des URL. Ici, le numéro de page étant un alias de la position correspondante dans le fichier, on pourra maintenir en local une table d'offsets.

Cette adaptation est en fait la solution générale: On s'appuie sur la mémorisation d'informations sur le serveur.

Le programme étant lancé pour chaque requête, la mémorisation des états doit être une référence externe et persistante permettant, si nécessaire, de reconstruire les ressources utiles à l'envoi du document correspondant. Les implémentations de ces mécanismes sont donc multiples mais pas encore normalisés, si tant est qu'ils soient normalisables. Pour des raisons de robustesse, un serveur généraliste est sans état, c'est à dire qu'il n'a pas de fonctionnalité permettant de modifier son comportement en fonction des requêtes. Il envoie un document et oublie aussitôt son histoire. Il se concentre sur les paramètres liés au protocole, en respectant sa

configuration statique. Il existe des serveurs prototypes fournissant des variables persistantes ([LR95]), mais il s'agit de langages de commande ou de programmation avec lesquels, en tout état de cause, il faudra définir l'implémentation du contexte pour chaque type de service W_3.

La production de références peut être généralisée au niveau du processus global. Une fonction du type: $(r1, r2, r3...) = f(a1, a2, a3...)$
peut être mémorisée par $ref_i = Archive(f, r1, r2, r3..., a1, a2, a3...)$

Pour rester dans la philosophie documentaire, la fonction $Archive()$ peut fournir des clés d'identification persistantes basées sur le système de fichiers, dans une association simple entre URL et disque.
$href_i = ArchiveURL(f, r1, r2, r3..., a1, a2, a3...)$

Dans le cadre d'un développement réel, les problèmes de composition et de réutilisabilité sont difficiles à résoudre de façon générale.

Si on produit *directement* tous les *états* possibles, dans notre exemple on se retrouverait avec les N pages de notre fichier original, dans N fichiers. Cette méthode serait éminemment gloutonne en disque, et les résultats difficiles à mettre à jour. Il faudra donc définir de façon plus pertinente les résultats méritant d'être archivés et ceux pouvant être calculés.

Par exemple:

```
sub f(a1, a2, a3 ...) {
    if ( !(o1, ...) = archiv( f, a1, a2, a3 ...) ) {
          # real process
        o1 = g(a1, a2 .. );
        o2 = h(a1, ... );
        o3 = i(a2 ... );
    }
    return (o1, o2, o3 ...);
}
```

On crée alors des dépendances entre les différentes ressources. Si le domaine de définition des symboles s'appuie sur le système de fichiers, on peut utiliser `make` comme moteur de résolution. L'enchaînement des états sera conditionnée par la comparaison temporelle entre une cible (ici $f()$) et ses dépendances ($g()$, $h()$ et $i()$). Le problème est que `make` reste très limité dans son jeu d'instructions et de fonctions, ce qui rend fastidieuse l'analyse des paramètres ~~un peu~~ complexe? De plus la comparaison temporelle n'est pas l'unique critère de dépendance possible.

Il faut garder à l'esprit, que les références sont publiques. Pour améliorer la consistance du processus global, on doit donc fournir un mécanisme d'exclusion mutuelle pour éviter les incohérences dues aux accès concurrents, pendant la mise à jour des archives. Le test sur les ressources archivées ne sera plus booléen mais prendra des valeurs sur `(a-jour, en-cours, inconnue, ...)`.

On pourra alors utiliser des lancements de procédures désynchronisées des requêtes client. La Section 2.4 donne un exemple de configuration d'événements périodiques avec `cron`, qui peut ainsi être vu comme l'anticipation des demandes de consultation de textes distants.

3.1.4 Exemple Perl détaillé

Pour illustrer le processus de décomposition des notations, nous énumérons ici quelques étapes nécessaires à la mise en place d'un service, en prenant pour exemple la consultation des tables syntaxiques. Nous mêlerons à dessein les fonctionnalités désirées avec les considérations techniques.

- Définition du point d'entrée via un serveur généraliste :

  `ScriptAlias /LexG /WWW/modules/lexg.pl`
- Syntaxe et notation :

  On veut pouvoir rechercher dans le lexique-grammaire.
  Notre fonction *recherche*(*table*, *motif*) fera référence aux différentes ressources par des URL de la forme : `/LexG/ table / motif / info`

  La fonction de base est d'afficher la table dont le code est passé en paramètre. Pour limiter la présentation on permet de sélectionner les entrées par un motif.
- si le motif contient une ou plusieurs ',', la recherche se fait par colonne, sinon on opère une recherche globale.

  Par exemple : `/LexG/4/,-,,,,+` présente toutes les entrées de la table 4 n'acceptant pas de sujet humain ($N_o = Nhum$) et ayant un emploi *concret*, soit 82 entrées.

  `/LexG/10/,avec` dénote les entrées de la table 10 ayant comme un complément prépositionnel en *avec*, soit 15 entrées :
  arranger, combiner, comploter, concerter, conspirer, fomenter, goupiller, machiner, magouiller, manigancer, organiser, ourdir, parier, toper, tramer

  Pour permettre les recherches globales, nous choisirons par exemple les abréviations suivantes :
- si `Table` commence par '*' recherche sur une partie du lexique-grammaire. sinon recherche sur la table précise.
  `*Verbes` dénotera les 60 tables de constructions complétives et de phrase simples.

  '*' sera donc la liste complète des tables présentes sur le serveur !

  1 10 11 12 13 14 15 16 17 18 19 2 3 31H 31I 31R 32A 32C 32CL 32CV 32H 32L 32NM 32PL 32R1 32R2 32R3 32RA 33 34L0 35L 35R 35S 35ST 36DT 36R 36S 36SL 37E 37M1 37M2 37M3 37M4 37M5 37M6 38L 38L0 38L1 38LD 38LH 38LR 38LS 38PL 38R 39 4 5 6 7 8 9 TCA1 TCA2 TCA3 TCA4R TCA5 VPA1 VPA2 VPA3 TCA6 a1 a12 a1p2 a1pn c1pnr anp2 c0 c0e c0q c1d c1dpn c1g c1gpn c1i c1ipn c1p c1p2 c1pnp c7 c1r c5 cdn c6 cnp2 cp1 cpn cpp cppn cppq cpq cv c8 cadj cadv can pc pca pcdc pcdn pconj pcpc pcpn pdetc peco pf pjc ppco psy pvco pv padv ad fr2 fs2 es fs3 fr3 is2 dr2 drc dr1 fr1 is1 fs1 dr3 aa f1a f1b f1c f1d f1r f21 f2a f2b f2c f2r f3 f5 f6 f7 f8 f9 f91 faire-de=etre an07 an08 an09 an10 an01 an03 an04 an06 ansy an02 an05 fca fcana fcann fcn fcpn fcpna fcpnn fn fna fnan fnana fnann fndn fnn fnpn fnpna fnpnn fc fcan B_ANP2 B_A1 B_A1PN B_AADV B_C0 B_C1 B_CADV B_CNP2 B_CP1 B_CPP B_CPPN B_E1 B_E1P2 B_EAPC B_ECO B_EPC B_EPCPC B_F0 B_F1 B_F1PN B_FP1 B_FP1N B_YA

  ':' dénotera les recherches sur les propriétés des tables, plutôt que sur les entrées.
- La liste est suffisamment imposante pour comprendre que l'affichage du contenu de n tables les unes à la suite des autres ne présente pas un grand

intérêt pour une lecture raisonnée. Le premier document retourné sera donc limité à la liste des codes des tables où `Motif` apparaît.

- Les tables seront formatées comme le montre la Figure 2.7.

  Pour détailler une entrée on choisira de spécifier un paramètre supplémentaire permettant de séctionner les propriétés.

  `/LexG/4/abasourdir/+` donnera donc les propriétes marquées '+' et les exemples associées à *abasourdir*:
  Cette nouvelle a causé mon abasourdissement.

La Figure 3.2 montre la partie du CGI testant les notations souhaitées. Les recherches effectives sur les ressources internes s'appuieront sur les modules `Ling::LexG`, `Ling::Table`, ... soit environ 1000 lignes de code Perl (voir Annexe).

```
$z = newGetArgs z(Table, Pattern, Entry, State);
print  $z->titre('Recherche dans le Lexique-Grammaire');
if (defined $Table) {
    if ($Table =~ '^\*') {                          # global search
        if (defined $Pattern) { ListeMatchingTables($Table, $Pattern); }
        else {   ListeTables($Table);        }
    }
    else {                                          # specific search
        my $T = LexG::TableByCode($Table);
        if (ref($T) ne 'Table') {  What("Sorry, no file found"); }
        else {
            if ($#Pattern > 0) {
                $T->search(\@Pattern);              # list of pattern
            } else {
                if (! defined($Pattern)) {
                    $T->search();                   # load all
                }
                elsif ($Pattern =~ /,/) {
                    $T->SearchEntry($Pattern);  # list of pattern
                }
                else {
                    $T->Grep($Pattern); # could project $pattern , $pos
                }
            }
            if (defined $Entry) {       # detailed first entry
                pre( $T->sampleEntry($n) ) ,
            }
            else {
                if ($T->nEntries > 0) {
                    print $T->FormatHTML;
                }
                else {
                    print h1("[$Pattern] not found in Table ", $T->Anchor);
                }
            }
        }
    }
}
elsif (defined $Pattern) { globalSearch; } # only 1 pattern
else {    print &Form::OneTable;  }
```

FIG. 3.2 – *Décomposition des requêtes de consultation des tables*

3.2 Sélection des paramètres

Les URL dénotent les états accessibles par les clients. Avec la multiplication des états, l'énumération devient impossible, ou plutôt illisible dans un environnement documentaire où, rappelons le, les résultats produits sont destinés à des utilisateurs humains.

En plus des points d'ancrage, HTML définit des mécanismes de construction d'URL plus synthétiques.

3.2.1 Points d'ancrage

URL relative/absolue

Dans le schéma d'adressage `http:` ou `ftp:` la norme spécifie qu'une référence ne commençant pas par un "/" est considérée comme relative au document courant et sera substituée à la partie *fichier* de l'URL de base.

Par exemple dans le document `http://serv/d1/doc.hml`
`<A HREF="relative.html">`
donne `http://serv/d1/relative.html`
et pas `http://serv/d1/doc.html/relative.html`

La balise `<BASE HREF= ...>` permet de *délocaliser* le document pour la construction des URL complètes. C'est utile pour les copies partielles de structures hypertexte très larges, les liens référant toujours à la source.

"/", "." et ".." ont la sémantique usuelle des systèmes de fichiers, respectivement de séparateur de répertoire, répertoire courant et répertoire père. Cette interprétation est faite au niveau des serveurs.

`http://serv/n'importequoi/../d1/doc.html`
sera équivalent à `http://serv/d1/doc.html`

Fragment

Des références internes aux documents sont aussi possibles. Elles sont notées `#internalref` en fin de l'URL associée au document.
Ce document doit contenir un balisage `<A NAME="internalref">`, qui sera invisible au rendu du document.

Encodage Propre

Pour favoriser la lisibilité des URL, dans le codage des paramètres nous pourrons choisir de considérer '/' uniquement comme séparateur, et de définir '{' et '}' comme marqueurs de groupes.

Ce qui signifie que l'on s'interdit les transformations des URL relatives, pour '/', '..' et '.'.
Exemple: l'URL `http://server/dico/Delaf/{/Corpus/t1}/..`

est analysée de façon standard comme
`http://server/dico/Delaf/{/Corpus/`
qui ne signifie plus rien dans notre notation.

3.2.2 Masque de saisie

Les demandes d'URL par les clients peuvent être construites à partir d'informations supplémentaires contenues dans les documents HTML, définissant les masques de saisie (`<FORM>`).

Les balises `<INPUT>` peuvent être typées `radio, select, checkbox, textarea`, pour présenter différents widgets de saisie ou de sélection : texte, boutons, liste d'options multiples ... Un type `hidden` permet aussi de mémoriser une information sans encombrer l'affichage.

```
<FORM ACTION="URL" METHOD= ...>
<INPUT NAME=champ1 VALUE="something">
<INPUT NAME=champ2 VALUE=" else">
...
<INPUT NAME=any VALUE=" Send to server" TYPE=SUBMIT>
</FORM>
```

Les différents champs sont nommés et les valeurs choisies par l'utilisateur sont transmises au serveur selon la syntaxe suivante:

```
ActionURL?champ1=val&champ2=val& ...
```

Si plusieurs `<FORM>` cohabitent sur une même page, seul sera envoyé celui qui contient le champ de type `SUBMIT` activé.

Le protocole HTTP définit deux méthodes pour l'envoi des valeurs saisies par le client (voir Annexe D) :

POST encapsule les valeurs dans la requête d'appel.
L'URL n'est pas affectée : `http://server/ActionURL`.

GET ajoute à l'URL de base les informations saisies
Exemple: `http://server/ActionURL?champ1=val&champ2=val& ...`

L'URL résultant de l'emploi de la méthode POST ne permet donc pas de faire référence à l'état du client, puisque les paramètres sont encodés dans les données du protocoles. Cette ressource ne pourra être atteinte que si le client mémorise lui même les paramètres, fonctionnalité qui n'est fournie par les logiciels usuels que pendant la durée de leur exécution. Il est alors possible de "re-poster" les valeurs saisies. Si l'utilisateur quitte son logiciel, il devra resélectionner manuellement tout les champs saisis.

L'utilisation de POST sera donc limitée aux envois volumineux, pour conserver aux URL une taille manipulable, par exemple pour l'envoi d'un texte du client à analyser.

Les Figures 2.1, 2.5 et 2.9, illustrant les diverses recherches dans les données linguistiques, donnent quelques exemples de widgets, tout comme on en retrouvera au chapitre suivant pour la sélection des paramètres de traitement de corpus.

L'exemple ci-dessous donne une possibilité d'appel du service de recherche dans les tables (Voir Figure 3.2), spécifiant le nom des paramètres leur domaine de définition. La Figure 2.9 en montre le rendu .

```
  <H1>Recherche dans le Lexique Grammaire</H1>
<FORM ACTION="/LexG" METHOD=GET>
<I>Motif à rechercher: </I><INPUT NAME="Pattern" SIZE=25 VALUE="">
<I>Domaine</I> : <SELECT NAME="Table">
                <OPTION Value="*Verbes" selected> Verbes
                <OPTION Value="*Noms"> Noms prédicatifs
                <OPTION Value="*Adverbes"> Adverbes figés
                <OPTION Value="*Ph"> Phrases figées
                <OPTION Value="*Engl"> Anglais/Français
                <OPTION Value="*Belg"> FrançaisDeBelgique
                </SELECT>
 <INPUT TYPE="submit" VALUE="Rechercher">
</FORM>
ou
<FORM ACTION="/LexG" METHOD=GET>
 <I>Motif à rechercher: </I>
              <INPUT NAME="Pattern" SIZE=25 VALUE="apprendre">
 <I>Code de la table </I>:
                <INPUT NAME="Table" SIZE=10 VALUE="7">
 <INPUT TYPE="submit" VALUE="Rechercher">
</FORM>
```

Comme l'avait noté T. Berners Lee dans ses `Guidelines`[6] des débuts de W_3, pour les clients *en mode texte* (`emacs` ou `lynx`), une interface plus rudimentaire à base de points d'ancrage simples peut fournir in fine les mêmes fonctionnalités, par énumération de toutes les URL possibles. La consultation est alors rendue bien plus fastidieuse.

3.2.3 Image cliquable

Ces images sont balisées de la manière suivante:

```
<A HREF="http://server/MapInterpreter"><IMG SRC="anIMAGE.jpeg" ISMAP></A>
```

Lorsque l'utilisateur clique sur cette image du document courant, les coordonnées sont transmises par le client selon la syntaxe suivante:

```
http://server/MapInterpreter?coordX,coordY
```

à charge pour le programme serveur (`imagemap`[7]) d'associer une URL à ces coordonnées.

Une extension des capacités du client, proposée par Netscape, est de résoudre l'association coordonnées/URL au niveau client. Une image cliquable fera donc

6. URL: http://www.w3.org/pub/WWW/Provider/
7. URL: http://hoohoo.ncsa.uiuc.edu/docs/tutorials/imagemapping.html

référence à une partie non visible du document qui contiendra la définition des zones cliquables.

```
<IMG SRC="blobby.gif" USEMAP="#nom-du-map">
```

```
<MAP NAME="nom-du-map" >
     <AREA SHAPE="rect" COORDS="x1, y1, x2, y2" HREF="url-region1">
     <AREA SHAPE="circle" COORDS="x1, y1, r" HREF="url2">
     <AREA SHAPE="polygon" COORDS="x1, y1, x2, y2 ... xn, yn" HREF="url3">
     ..... more shapes ...
</MAP>
```

3.2.4 Equivalences et limites

Mise à part la mise en valeur typographique,
`<A HREF="/dir/file.html">look at file</A>`
est équivalent à

```
<FORM ACTION="/dir/file.html" METHOD=POST>
<INPUT TYPE=SUBMIT VALUE="look at file">
</FORM>
```

ou encore `<A HREF="/dir/file.html"><IMG SRC="look-at-file.gif"></A>`

L'enchainement des opérations doit être régi par une information de statut, aussi bien sous forme de paramètre supplémentaire
`http://.../cgi/arg1/arg2/.../argi/status`
que de marqueur
`<A HREF="http://.../cgi/arg1/arg2,new_value/.../argi/2">new value</A>`
que de paramètre caché (voir Section 3.2).
`<INPUT NAME=status VALUE="something" TYPE=HIDDEN>`

Notation et déspécialisation

Pour des raisons de robustesse et de sécurité, il faut veiller à tous les niveaux d'interprétation possibles des notations, depuis le logiciel client jusqu'au système (voir `Perl FMTEYEWTK`[8])

Les chaînes de caractères peuvent dénoter :

Balises ou entités HTML il n'y a pas que les caractères spéciaux à prendre en compte, c'est toute la syntaxe qu'il faut considérer. Par exemple une catégorie grammaticale `<N>` doit être encodée dans du texte mais pas dans une URL.

8. URL: ftp://ftp.ibp.fr/pub/perl/CPAN/doc/FMTEYEWTK/safe_shellings

URL avec problème de notations déspécialisées par "%" :

On a vu que les caractères `'?'`, `'&'`, `'+'`, `'/'` étaient réservés. Ces caractères spéciaux pour un schéma d'adressage particulier devront être encodés sous la forme `%xx`, où `xx` est le code hexadécimal du caractère.
Par exemple, une ressource `/Prêt/à/0%` sera adressée `/Prêt/à/0%25`.
Les caractères 8bits, n'étant pas bien supportés par certaines configurations, pourront subir le même traitement.

A ne pas confondre avec les entités HTML, qui seront décodées dans les documents par le client.
`/Pr%EAt/%E0/0%25` n'est pas équivalent à `/Pr&eacute;t/&agrave/0%25`

Les caractères ne sont pas spéciaux uniquement dans les parties *fichier* ou *paramètre* des URL. Certain site FTP demande un mot de passe de la forme "`user@`" alors que "`@`" est déjà réservé pour l'identification de l'utilisateur du client.

Une URL valide sera par exemple :

```
ftp://anonymous:user%40@ftp.ibp.fr/pub/
```

Commandes externes lancés par un appel au shell, donnant lieu à une nouvelle interprétation. Les caractères spéciaux à surveiller sont ceux susceptibles de lancer d'autres commandes : "`‘`" ou "||".
Ceux précisant des redirections affectant les fichiers : "`>`" ou "`<`"

Données internes aux programmes, par exemple en Perl les interprétations sont innombrables : expressions régulières, fichiers, répertoires ...

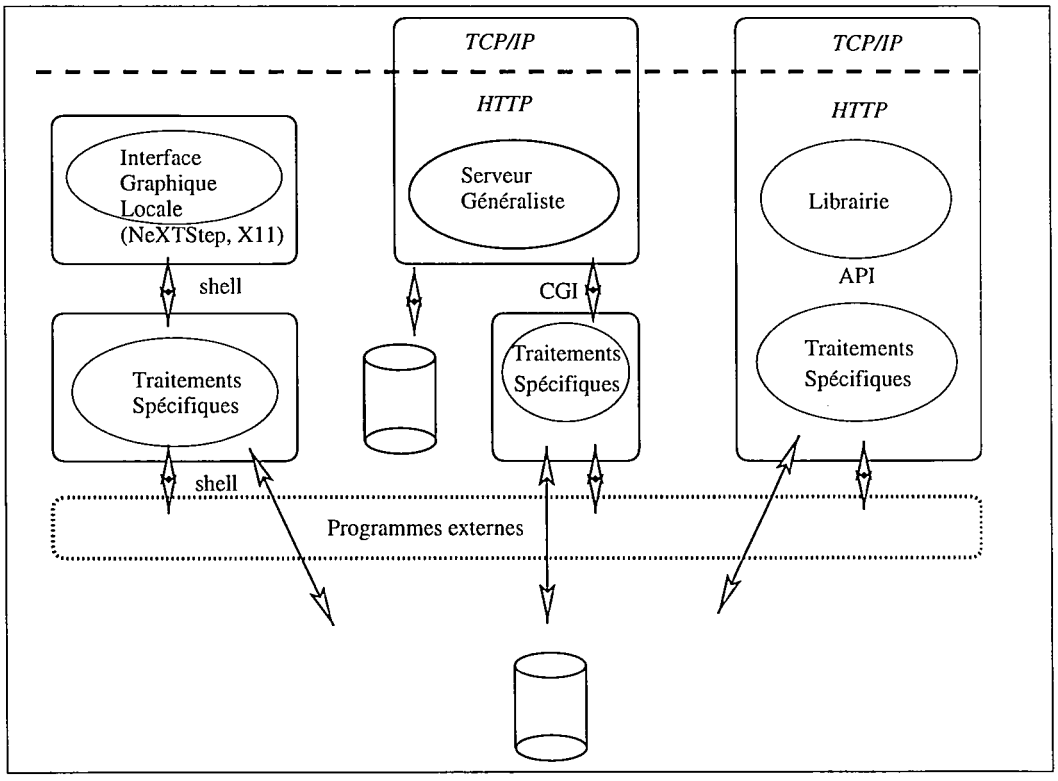


FIG. 3.3 – *Architectures logicielles d'un service*

3.3 Gestion d'un serveur généraliste W_3

Pour supporter les différents types de ressources, la meilleure configuration semble être un serveur *généraliste* pour point d'entrée et un ensemble de programmes externes pour les traitements plus spécifiques que le simple envoi de fichier.

Outre la robustesse, cette solution permet de suivre l'évolution des développements et de la normalisation concernant HTTP.

3.3.1 Quel serveur

Un serveur prend en charge la partie protocole des communications, essentiellement HTTP, et offre de facilités pour la mise en place, la personnalisation et la maintenance de ressources.

Il doit être fiable pour assurer une qualité de service suffisante aux clients, mais aussi performant au niveau système pour ne pas effondrer la ou les machines qui l'hébergent.

Il constitue un point de convergence pour produire des documents et des services à partir de ressources hétérogènes ou disparates. De nombreux raffinements permettent d'homogénéiser les structures sous jacentes aux documents servis.

La mise en place d'un serveur recouvre les étapes suivantes:

1. Téléchargement depuis un site *officiel* ou un `site miroir`[9].
   De nombreux *démons* généralistes ou évolutifs sont disponibles dans le domaine public (`W3C`[10] , `NCSA`[11] , `wn`[12] , `Apache`[13] ...).

   Selon la notoriété de son système d'exploitation on trouvera des versions prêtes à l'emploi ou seulement les fichiers sources à compiler soi même.

2. On choisit en règle générale de faire tourner le programme en permanence[14], dès le lancement de la machine.
   La procédure de lancement du serveur dépendra du système d'exploitation, par exemple sous Unix on modifiera `/etc/rc` ou `/etc/rc.local`.

3. Définir le port de service, le port standard normalisé étant 80 (voir Table 1.3). D'autres ports peuvent être utilisés à des fins de tests ou de services spécifiques connexes.

   On doit aussi définir le nom (ou les noms) DNS sous lequel le serveur opère (voir Figure 1.7).

4. Identifier le propriétaire des processus. Un serveur travaille pour ses clients, encore faut-il limiter leurs droits! C'est particulièrement important dans le développement des CGI où les services encore instables pourraient endommager le système hôte.

   Des caractéristiques comme le nombre maximal de processus concurrents ont une grande importance pour la bonne marche des services mais ne sont optimisables qu'à l'usage.

5. Des clauses de confidentialité peuvent être spécifiées dans les limites des protocoles. A l'heure actuelle, on peut identifier avec certitude les adresses IP des clients. Les mécanismes de mots de passe relèvent encore de la bonne volonté. La meilleure sécurité venant de toutes façons de la pléthore d'informations qui noie ce qui pourrait être considérer comme "sensible".

6. Configuration générale des services.

Les serveurs généralistes offrent d'innombrables `fonctionnalités`[15] plus ou moins compatibles, plus ou moins étendues, avec des `performances`[16] diverses, paramétrables déclarativement ou accessibles via API[17].

9. URL: ftp://ftp.ibp.fr/pub/www
10. URL: http://www.w3.org/pub/WWW/Daemon/
11. URL: http://hoohoo.ncsa.uiuc.edu/docs/Overview.html
12. URL: http://www.gnu.org/
13. URL: http://www.apache.org/
14. d'où le terme *démons*.
15. URL: http://webcompare.iworld.com/
16. URL: http://www.ncsa.uiuc.edu/InformationServers/Performance/V1.4/report.html
17. Application Programming Interface

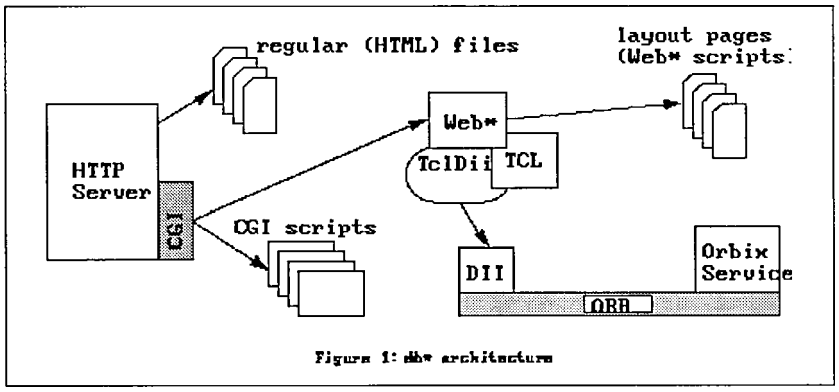


FIG. 3.4 – *Interprétation de méta-HTML (source: Web*)*

3.3.2 Fonctionnalités

Nous ne nous intéresserons qu'aux fonctionnalités essentielles pour assurer un service cohérent et robuste. Les exemples seront tirés de la configuration locale de notre serveur *Apache*, mais sont adaptables à d'autres logiciels.

Liens avec le système de fichier pour l'accès direct aux documents stockés tels quels sur le disque local. La production des listes de fichiers contenu dans un répertoire est une première structure hypertexte très utile. Elle est configurable pour masquer certaines données ou ajouter des descriptions supplémentaires (`FancyIndexing`, `DirectoryIndex`, `IndexIgnore` ...).

Les services de partage de fichiers via d'autres protocoles, type NFS ou NetWare, sont alors superposables au protocole `HTTP`.

Liens avec d'autres serveurs La redirection des requêtes permet d'indiquer aux clients de façon transparente, qu'ils doivent consulter d'autres sources. Cela peut être utiles si une ressource a changé de localisation, cela permet aussi des raccourcis dans la rédaction des liens (Voir par exemple `/Vlib` Figure 3.6). La distribution des services sur plusieurs machines peut donc se faire par la multiplication des serveurs HTTP ou FTP.

Des serveurs autres que HTTP peuvent être utiles. Par exemple les *proxy*, c'est à dire des serveurs intermédiaires se chargeant d'optimiser les accès réseaux pour plusieurs utilisateurs, en fournissant un mécanisme de cache.

Le logiciel client *Mosaic* permet aussi d'utiliser un serveur d'annotations, pour partager des remarques sur les documents parcourus.

Pour mettre en commun des ressources on pourra aussi passer par des services usuels de partage de disques distants type NFS.

Des applications plus complexes permettent encore l'exécution de commandes à distance, du type `rexec` voire même `pvm` pour rechercher des traitements parallèles.

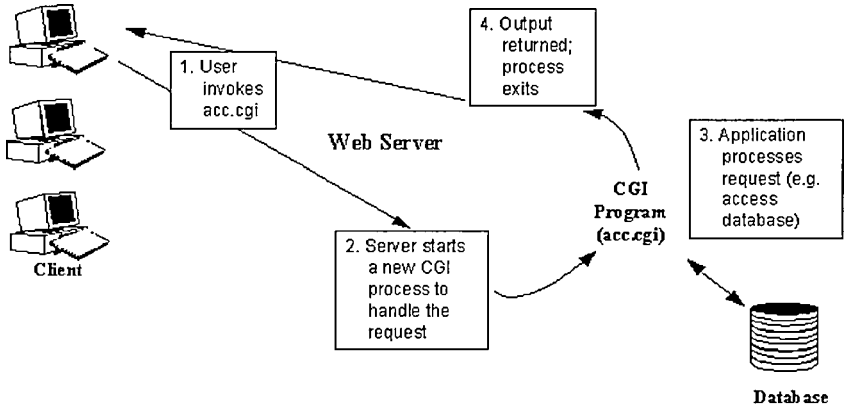


FIG. 3.5 – *CGI (source: www.openmarket.com)*

Liens avec processus externes selon les conventions `CGI`[18] Le standard *CGI* définit le passage des informations au programme externe par variables d'environnement (et entrée standard pour la méthode POST).
Exemple: `http://serveur/run/test.cgi/dir/file?var1=test&var2=idem`

```
REQUEST_METHOD = GET
HTTP_ACCEPT    = image/gif, image/jpeg, */*
PATH_INFO      = /dir/file
PATH_TRANSLATED= /WWW/dir/file
SCRIPT_NAME    = /run/test.cgi
QUERY_STRING   = var1=test&var2=idem
REMOTE_HOST = monge.univ-mlv.fr
REMOTE_ADDR = 193.55.44.175
REMOTE_USER =
AUTH_TYPE =
```

Remarque: le nouveau standard `Fast-CGI`[19] est plus proche d'un serveur indépendant que d'un programme externe.

Pour le client, les documents générés dynamiquement ne diffèrent en rien des documents statiques, cependant les URL peuvent être traitées de manière sensiblement différentes selon les logiciels clients ou serveurs. Mentionnons par exemple que les paramètres suivant "?" ne sont pas cumulables, ce qui revient à dire qu'il n'y a pas de composition des appels de `<FORM>`.
Les redirections opérées par les serveurs ne propagent pas les informations hors URL et pas toujours les paramètres ou les références internes.

18. URL: http://hoohoo.ncsa.uiuc.edu/cgi/interface.html
19. URL: http://www.fastcgi.com/

Métalangage interne Extensions de l'interprétation des directives incluses dans les documents. Le principal intérêt de tout métalangage HTML est d'introduire des abréviations et des variables dynamiques de façon standard et concise. Trouver une notation lisible économique, élégante, modifiable simplement reste un problème ouvert.

Pour introduire des informations dynamiquement, les serveurs *NCSA* ou *Apache* utilisent les commentaires HTML du type: `<!--directive -->` pour insérer le résultat de commandes système (`#exec cmd=...`), ou de CGI locaux (`#exec cgi=...`), des fichiers (`#include file=...`), des variables internes (`#echo var=...`).

Ce métalangage reste limité, et n'offre notamment pas de structures de contrôle.

Des serveurs plus spécialisés offre un langage propre beaucoup plus puissant, comme `Web*`[20] basé sur TCL (voir Figure 3.4).

De même, nos CGI utiliseront un module Perl d'abréviation des notations HTML (voir section 3.3.2).

```
    <IMG SRC="/images/i.gif" ALT="une image">
ou  img({src='/images/i.gif', alt=>'une image'})

    <H1> ... </H1>
ou  h1(' ... ');
ou  h1(' ... ')->as_HTML;
```

Que ces notations raccourcissent la rédaction n'est pas si clair, cela permet néanmoins d'insérer variables et appels de fonctions du langage:

```
h1("[$Pattern] not found in Table ", $T->Anchor)
```

Statistiques sur les accès Traces des transactions par clients, par documents, par logiciels utilisés, par points d'ancrage source ... (voir Table 3.1)

3.3.3 Interprétation des URL

La sémantique des URL n'étant pas prédéfinie, plus les URL seront *lisibles* ou explicites plus la navigation et la rédaction seront aisées, donc plus le service sera accessible. La concision limitera aussi les erreurs de transcription possibles. C'est pour cela que la partie *serveur* de l'URL engendre une explosion des DNS (voir Section 1.3.3).
On préférera avoir : `http://www-ceril.univ-mlv.fr/`
plutôt que `http://193.55.44.154/Serveur/LADL/CERIL/`

20. URL: http://webstar.cerc.wvu.edu/webstardoc/WebStar11Doc.html

Autre avantage de ce nommage symbolique, il autorise une certaine souplesse dans la configuration des services sans que les *marque-pages* des clients soient altérés, car en plus d'être lisibles les références se doivent bien sur d'être stables.

La fonction de base d'un serveur est d'associer des documents aux URL demandées par les clients. Les décompositions de la partie *fichier* des URL concernent:

Reconnaissance de préfixes C'est la convention usuelle de définition d'arborescence virtuelle, s'appuyant sur les chemin d'accès des disques du serveur. Les mécanismes d'alias permettent de désolidariser l'arborescence hypertexte de l'arborescence système sous-jacente.

Pour les systèmes multi-utilisateurs, la convention des *shells* Unix a été reprise, à savoir dénoter le répertoire personnel d'un utilisateur par `~Nom-d-utilisateur`

Les mécanismes de filtrage des préfixes des serveurs ne sont pas toujours équivalents. Par exemple le serveur NCSA permet d'associer n'importe quel préfixe à un programme *CGI*
Ex: `ScriptAlias /service /cgi-bin/programme_de_service`
alors que celui du *W3C* ne peut définir que des sous-répertoires contenant les exécutables: `Exec /service/ /LocalDir/`

Reconnaissance de suffixes La directive `AddEncoding` (ou `AddType`) permet aussi de typer les documents selon l'extension du nom de fichier, pour indiquer au client quels traitements il peut mettre en oeuvre (voir section 1.2.3).

Les traitements locaux au serveur sont spécifiées par les deux directives:
```
AddHandler  action-name   extension
Action      handler-name  /cgi-script/location
```

Il est à noter que que la partie *paramètres* de l'URL (Voir Figure 1.6) n'est pas interprétée directement par le serveur, un fichier statique ne peut être adressé `/xxx?params`.

Récupération sur erreur En cas de problème pour fournir un document correspondant à l'URL demandée, on peut définir quel document de remplacement sera envoyé au client (Voir Annexe D).

Ex: (`ErrorDocument 500 /server.error`).

A noter que la complexité des réécritures est limitée pour des raisons évidentes de clarté et d'efficacité. Les directives sont appliquées dans un ordre prédéfinis (`Redirect, Alias, ScriptAlias`). Les préfixes ne sont reconnus que suivis d'un séparateur ("`/`").
Par exemple: `Alias /biblio /LocalDisk/ortho`
transformera `http://server/biblio/graphe` en `/LocalDisk/ortho/graphe`
mais laissera intacte `http://server/bibliophile/`

```
DocumentRoot /WWW/CERIL/Root/ # Default alias
# The format is : Alias fakename realname
Alias /Corpus/ /LocalDisk/Corpus/WWW/
# ScriptAlias: This controls which directories contain server scripts.
# Format: ScriptAlias fakename realname
ScriptAlias /Corpus/ /WWW/CERIL/run/Next2Isolatin/
ScriptAlias /dyn/ /WWW/CERIL/Modules/
ScriptAlias /LexG /WWW/CERIL/Modules/LexG
ScriptAlias /bib  /WWW/CERIL/Modules/bib
#
Redirect /img/   http://www-igm.univ-mlv.fr/images/
Redirect /VLib   http://www.emich.edu/~linguist/www-vl.html
#
ErrorDocument 500 /dyn/server.error500
ErrorDocument 404 /errors/
```

FIG. 3.6 – *Extrait de Server Resource Map (*`srm.conf`*)*

Les suffixes ne sont pas traités de façon cumulative.
Par exemple : `/file.ext1.ext2` sera traitée comme `/file.ext2`

Les interprétations plus complexes, syntaxiques ou sémantiques, seront définies par les programmes externes, ou comme le propose [BMMM94], par des serveurs de transductions!

Application au transcodage automatique des accents

Sur système `NeXTStep`, le jeu de caractères étant différent de l'*ISO-Latin-1* usuel sous HTML, on peut définir un traitement spécial :

1. Fonction opérant le transcodage vers le jeu de caractères ou les entités HTML.
2. Un programme CGI (`Next2Isolatin`) associant lui même les arguments reçus aux fichiers du serveur à renvoyer convertis.
3. Configuration du serveur (voir Figure 3.6)

L'interprétation des URL se fait donc en deux étapes
Pour les fichiers *texte*, contenus dans le répertoire `/Corpus`, on adopte un suffixe spécial :

```
AddType    text/x-nextplain  .nxt
AddHandler text/x-nextplain  next2isolatin1
Action     next2isolatin1  /cgi/Next2Isolatin
```

Chaque rédacteur voulant éditer ses documents HTML "à la main", directement avec des accents `NeXT`, peut ajouter dans le fichier `.htaccess` de ses propres répertoires la directive :

```
AddHandler text/html next2isolatin1
```

3.3.4 Maintenance et suivi

La cohérence et l'intégrité du serveur sont souvent affectées par les effets de bords de modifications que l'on pensait anodines.

La prise en compte de problèmes diffus sera souvent possible grâce au feedback des utilisateurs, par courrier électronique ou interface d'interrogation spécifique.

On peut guetter les indices de dysfonctionnement par l'analyse des fichiers *trace*. En plus de la détection des erreurs, cela permet d'évaluer l'activité.

On doit aussi lancer des robots d'exploration automatique: `MOMspider`[21] , `checkbot`[22] ... On peut définir les parties du serveur que l'on veut préserver des `explorations automatiques`[23] sauvages. Par convention cette description doit être accessible à l'URL `/robots.txt` ([Fie94]).

```
User-agent: *
Disallow: /LexG
Disallow: /dyn
Disallow: /run
Disallow: /bib
```

La convention ci-dessus relève de la bonne volonté des deux parties. Les données présentées sur le serveur n'ont pas toutes le même statut de confidentialité, on peut vouloir en interdire complètement l'accès. L'administration de listes d'autorisation permet, dans les limites des possibilités d'Internet et plus particulièrement d'HTTP, de contrôler les informations fournies selon le site ou le client demandeur.

Exemple de `.htaccess` pour l'accès aux informations du DELAS:

```
<Limit GET>
  Order deny,allow
  deny from all
  allow from monge.univ-mlv.fr blandine.ladl.jussieu.fr
</Limit>
AuthType Basic
AuthName somedomain
AuthUserFile /web/users
AuthGroupFile /web/groups
<Limit GET POST>
require group admin
</Limit>
```

Le contrôle de ces droits d'accès demande une attention particulière aucun utilitaire interne ne permet de tester de façon globale le contenu et la cohérence de tous ces fichiers `.htaccess`.

21. URL: http://www.ics.uci.edu/WebSoft/MOMspider/
22. URL: http://dutifp.twi.tudelft.nl:8000/checkbot
23. URL: http://info.webcrawler.com/mak/projects/robots/robots.html

Les accès :

```
scooter.pa-x.dec.com - - [01/Jul/1996:02:29:47 +0100] "GET /LADL/LexG/adverbesfiges.html
scooter.pa-x.dec.com - - [01/Jul/1996:02:34:27 +0100] "GET /LADL/LexG/phrasesfigees.html HTTP/1.0" 200 2026
i15.inktomi.com - - [01/Jul/1996:02:44:40 +0100] "GET / HTTP/1.0" 200 1755
world-f.std.com - - [01/Jul/1996:05:19:33 +0100] "GET /Dictionnaires/ HTTP/1.0" 200 1198
liberty.uc.wlu.edu - - [01/Jul/1996:05:59:21 +0100] "GET / HTTP/1.0" 200 1755
pool011.max12.newark.nj.dynip.alter.net - - [01/Jul/1996:06:15:11 +0100] "GET /Dictionnaires/Delaf5_en.html
...
```

Les documents sources:

```
http://www.fask.uni-mainz.de/user/wschmidt/onldict.html#F
                -> /Dictionnaires/Delaf5_en.html
http://www-ceril.univ-mlv.fr/Dictionnaires/Delaf5_en.html -> /images/fr.gif
http://www-ceril.univ-mlv.fr/Dictionnaires/Public/
                -> /Dictionnaires/Public/ClassesFlexionnelles.html
http://www-ceril.univ-mlv.fr/Dictionnaires/ -> /
http://www.dmi.ens.fr/~cousot/dictionnaires.html -> /Dictionnaires/
http://chianti.philosophie.uni-stuttgart.de/romanist.html -> /
http://www.urec.fr/France/webindex.cgi?Dictionnaire -> /
http://www.altavista.digital.com/cgi-bin/query?pg=q&q=francais+%2Bgrammaire
-> /LexiqueGrammaire/Francais/ ...
```

Les erreurs:

```
[Mon Jul 1 14:01:42 1996] HTTPd: access to /WWW/CERIL/Root/Actualit
failed for i15.inktomi.com, reason: No file matching URL: /Actualit
from - [Tue Jul 2 14:25:22 1996] child error: child connection
closed [Tue Jul 2 18:52:26 1996] HTTPd: access to /WWW/CERIL/Corpus/
failed for france.ecila.com, reason: client denied by server
configuration from - [Wed Jul 3 04:09:31 1996] timed out waiting for
- ...
```

Les types de clients, ici triés sur le nombre d'accès. On voit la prédominance de *Netscape Navigator* (que personne sauf ses créateurs n'appelle *Mozilla*!), et l'importance des robots explorateurs de toutes origines.

```
15393 Mozilla
1583 InfoSeek Sidewinder
 856 Slurp/1.0 (http://www.inktomi.com/slurp.html)
 630 ia_archiver/1.2
 512 Freecrawl via Harvest Cache version 1.4pl3
 402 NCSA_Mosaic
 346 IBM-WebExplorer-DLL/v1.1b
 257 Lynx 2.5  libwww-FM/2.14
 234 Wobot/1.00
 196 weblayers/weblayers/2.0 libwww-perl/0.40
 182 ArchitextSpider
 177 MetaCrawler/1.2b libwww/4.0D
  99 Lynx/2-4-2  libwww/2.14
  96 Microsoft Internet Explorer/4.40.308 (Windows 95)
  92 OmniWeb/2.0.1 OWF/1.0
  89 Lycos_Spider_(T-Rex)/1.0
  81 Lynx/2.3 BETA  libwww/2.14
  78 IWENG/1.2.003  via proxy gateway  CERN-HTTPD/3.0 libwww/2.17
  77 Scooter/1.0 scooter@pa.dec.com
  73 Checkbot/1.41 LWP/5.02
...
```

TAB. 3.1 – *Exemples de fichers traces (logs)*

Chapitre 4

Interface de traitements automatiques

Le nombre de textes disponibles, essentiellement en anglais ou en américain sous forme électronique est virtuellement illimité. Des `corpus`[1] de taille supérieure au million de mots sont maintenant courants. L'exploitation d'une telle quantité de données appelle des solutions spécifiques alliant architecture des systèmes, base de données, algorithmique, et linguistique.

4.1 Configuration générale

Ce chapitre présente un modèle d'intégration sous W_3 d'outils de traitement des langues naturelles. Du point de vue utilisateur, l'accès se fait toujours via une unique interface homogène et *universelle.*

Depuis une dizaine d'années, l'augmentation de la puissance des ordinateurs, tant au niveau espace de stockage que temps de calcul, a favorisé les travaux linguistiques basés sur l'utilisation de corpus. Quelques architectures distribuées ([BS94a, Chr94, CGW95, Com94]) ont été proposées pour améliorer l'efficacité des traitements de grands corpus. Cependant la portabilité et la compatibilité restent problématiques pour un utilisateur qui ne serait pas ingénieur système, alors que l'affinement et la validation des résultats passerait par un examen manuel ([Ma93]).

La diffusion de ces outils passe aujourd'hui par l'abstraction des ressources matérielles (disques, mémoires, réseau, ...) et l'organisation des ressources logiques, donc par une documentation des données et des résultats produits.

Du point de vue interne, une structuration hypertexte, basée sur W_3, répond assez bien aux exigences de modularité des traitements du langage naturel. Cela favorise aussi les travaux coopératifs, en évitant par là même les redondances de traitements, puisque tous les utilisateurs partageant un même espace d'adresses.

1. URL: http://www.comp.lancs.ac.uk/computing/users/paul/ucrel/corpora.html

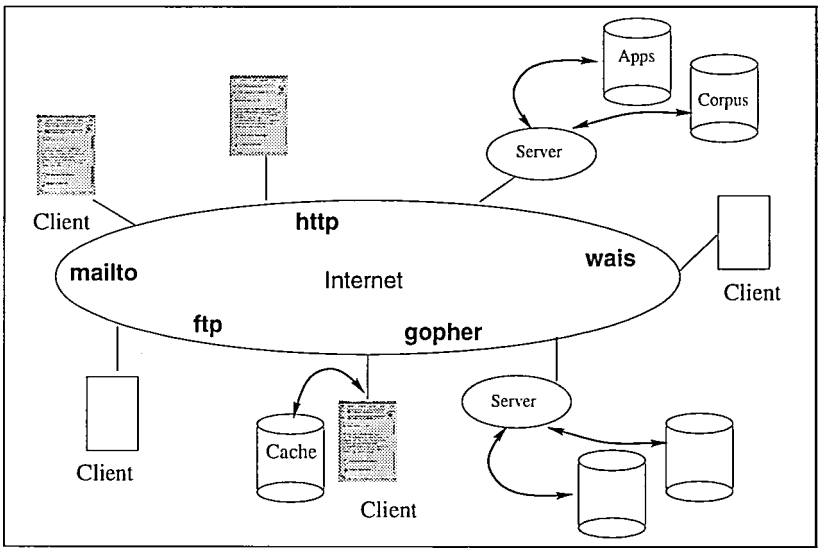


FIG. 4.1 – *Configuration du système*

De plus, la charge des machines (CPU, disque, réseau) pourra être répartie dynamiquement de façon transparente pour l'utilisateur.

La Figure 4.1 schématise la configuration générale du système :

1 serveur HTTP (ou plus) hébergeant les programmes de traitements, les données linguistiques, les corpus ...

Internet et ses protocoles associés par lesquels passent les requêtes, et les réponses.

n **clients** les utilisateurs λ manipulant les données depuis leur visualiseurs standard W_3 (*browsers* de Netscape à emacs, voir Figure 1.4).

4.1.1 Des outils

La plupart des systèmes de traitement automatique des langues naturelles ont évolué vers des architectures modulaires, décomposant les opérations linguistiques en une succession de filtres ([Cle93, Roc93, Sil93, GT94, TRG95]). On obtient en plusieurs passes des suites de caractères, des mots associés aux catégories du discours, aux structures syntaxiques ... Mais les connaissances requises pour l'utilisation de ces outils et l'interprétation des analyses produites sont souvent rédhibitoires (dictionnaires, codes linguistiques, règles d'applications).

Les opérations possibles sur un texte sont nombreuses et souvent interdépendantes:

- calcul du nombres de mots, de leur nombre d'occurrences.
- découpage des phrases
- identification des *lexèmes.*
- recherche de séquences

- recherche de co-occurrences
- levée d'ambiguïtés
- analyse syntaxique

On pourrait ajouter de nombreuses applications spécifiques comme l'indexation, le rétablissement des accents [Yar94], la correction orthographique [LS89], l'interprétation sémantique [YM93], la traduction ...

De nombreux utilitaires standard de manipulations de "textes", avec un support de plus en plus répandu des encodages de langages natifs (NLS), permettent des recherches rapides et déjà pertinentes.

L'histoire fait que ces commandes sont associées au système Unix, comme `grep`, `ptx` , `diff`. Aujourd'hui elles sont portables sur tous les systèmes, notamment grâce au projet `GNU`[2] qui fournit les sources de ces programmes dans le domaine public.

Exemple: `ptx RevueDePresseRFI.txt | grep " plus"`

```
 /apparaît dans les titres, mais   plus encore dans l'illustration/
/de NICE-MATIN, qui donne un peu   plus encore dans la louange. Marc/
    Pour LE FIGARO, il s'agit ni   plus ni moins de la "dernière/
POUR/     /sortant se montre bien  plus brillant que son challenger.
partagés ce/        D'une manière  plus générale, vos journaux sont
.0/    /, mais à une échelle bien  plus réduite. Comment vivre avec 5
   /. Interrogation qui concerne   plus de 2 millions de Français,/
/de nombreux villageois trouvent   plus joli le y, l'instituteur/
référer aux panneaux/          Le  plus simple est peut-être de se
...
```

Exemple: Utilisation de `sort` fourni avec le système HP-UX. Ces commandes utilisent les variables d'environnement pour déterminer les paramètres spécifiques aux différentes langues. Cela comprend les touches composées pour les diacritiques, les formats de dates, et surtout le ou les ordres sur l'alphabet.

`sort extrait.delaf`	`setenv LC_COLLATE french.iso88591 ;` `sort extrait.delaf`
en	en
enamourer	éna
enarbrer	enamourer
énamine	enarmonia
énamourer	encabanage
éna	énanthème
énamine	énantiomérie
énamourer	énantiose
énanthème	enarbrer
énantiomérie	énargite
énantiose	enarmonia
énargite	énarque
énarque	énarthrose
énarthrose	énaser
énaser	encabanage

2. URL: http://www.gnu.org/

De plus en plus d'initiatives d' `aide en ligne`[3] pour ces traitements des textes électroniques au sens large, vont en répandre l'usage hors des habituels utilisateurs d'Unix.

La consultation du `Natural Language Software Registry`[4] permet de constater l'importante activité en matière d'outils spécialisés. Peu de ces outils du domaine public sont aussi complet qu'INTEX [Sil94]. Nous nous intéresserons essentiellement à INTEX dans sa version 2.0 sous NextStep.

Nous essayerons de dégager les solutions d'interface les plus intéressantes pour le traitement du langage naturel. Les points que nous aborderons restent valables pour les versions ultérieures. Ils concernent essentiellement les capacités d'adressage des opérations et l'organisation des résultats, pour optimiser les traitements et favoriser leur examen et leur validation "manuels". Cette organisation sera aussi valable pour d'autres outils du domaine.

Les résultats produits par les logiciels peuvent être convertis présentés en HTML par un convertisseur *statique*, lancé localement. On peut ainsi produire quelques jeux d'exécution avec des paramètres prédéfinis.

Exemple: Une commande `Intex2HTML` fournit un ensemble de fichiers formatés.

Il est évident que pour fournir un accès utile via W_3, il faut pouvoir multiplier les interactions, les incontournables conversions vers HTML ne peuvent donc suffire.

4.1.2 Interfaces sélectives

La normalisation des formats et des protocoles W_3 fait que les clients disposant d'un système graphique accèdent à un premier document composé de widgets (voir Figure 2.5) permettant de sélectionner les opérations et les arguments qui les intéressent.

Le seul présupposé pour que les traitements deviennent possibles est de disposer d'un logiciel client W_3 et de connaître l'URL du serveur. L'interface est donc normalisée, on peut donc se concentrer sur la mise en place de services. La description de documents d'accès en HTML permet aisément de spécialiser les points d'entrée (voir Section 3.2).

Exemple: Pour une correction orthographique, si seuls les mots composés sont recherchés, la sélection des dictionnaires de mots simples devient inutile. L'ensemble de dictionnaires de composés peut par ailleurs être prédéfini et le seul paramètre pertinent reste le texte. La Figure 4.2 montre la page d'accès à INTEX uniquement pour des recherches de mots composés.

Il existe de plus en plus de versions de démonstration accessibles via W_3 :

3. URL: http://etext.lib.virginia.edu/helpsheets/software.html
4. URL: http://cl-www.dfki.uni-sb.de/cl/registry/draft.html

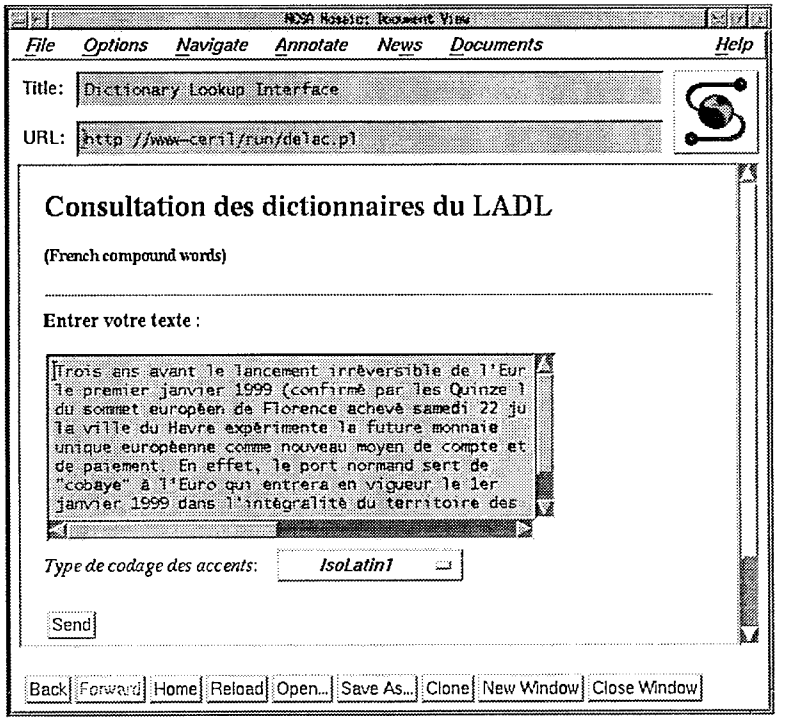


FIG. 4.2 – *Requête simple de recherche de mots composés*

LDC[5] le plus sérieux avec une réelle approche de la problématique des services W_3[6] .

COBUILD[7] fournit des concordances sur un corpus anglais de 50 millions de mots. Le résultat de la démonstration est limité de 40 occurrences maximum !

AMALGAM[8] pas directement accessible mais déjà disponible via e-mail, utilise l'étiqueteur [Bri94].

Citons aussi Xerox[9] qui collabore avec le *Trésor de Langue Française*, ou encore Lingsoft[10] ([Vou94]).

La difficulté d'extension des services (voir Chapitre 3) fait que ces sites sont qualifiés, par leurs auteurs même, d'expérimentaux, Nous ne dérogeons d'ailleurs pas encore à cette règle. Ils ne sont pas encore intéressants à des fins d'expérimentation, mais déjà le croisement des références bibliographiques, permet de se faire une idée des données réellement utilisées. A terme, cela permettra aussi d'envisager une normalisation de codes utilisés, de la syntaxe des requêtes et des formats produits.

5. URL: http://www.ldc.upenn.edu/ldc/online/index.html
6. URL: http://www.ldc.upenn.edu/ldc/online/batch/index.html
7. URL: http://titania.cobuild.collins.co.uk/form.html
8. URL: http://www.scs.leeds.ac.uk/amalgam/amalgam/amalgtag4.html
9. URL: http://www.xerox.fr/grenoble/mltt/Mos/Tools.html
10. URL: http://www.lingsoft.fi/demos.html

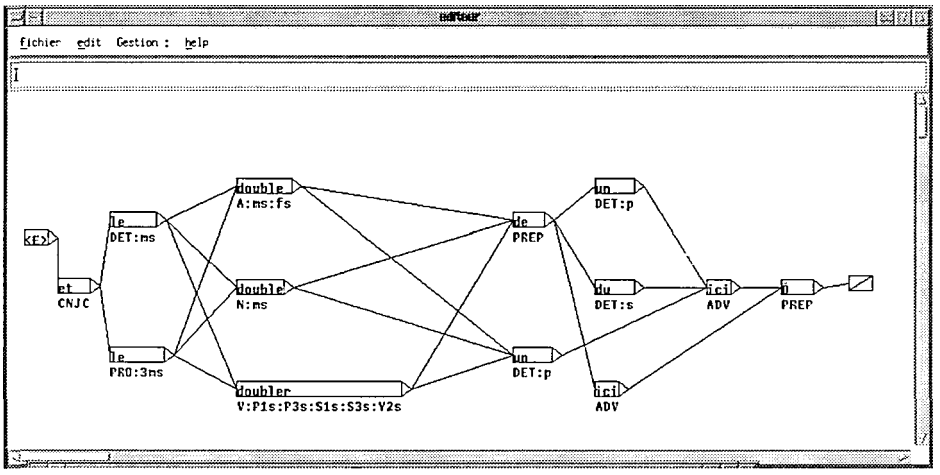


FIG. 4.3 – *Clone X11 de l'éditeur de graphes pour INTEX*

4.1.3 Contraintes de fonctionnement

Cette approche a bien sûr ses limites dues aux spécificités de W_3: modèle centralisé, orienté vers la consultation, car encore peu d'applications permettent directement la maintenance et la correction des données. L'utilisateur navigue dans les données qu'on lui propose mais n'interagit que très peu.

De plus, a tous les niveaux des systèmes d'information, on est tenté d'utiliser un peu plus de fonctionnalités que l'interface standard. Il semble peut probable que la normalisation aille jusqu'à l'édition de graphes du type de celui présenté Figure 4.3. Les applications attenantes (*helper appli.*) qui permettraient de gérer certaines données ne sont pas toujours portables. Généralement dans le domaine public ou sous licence gracieuse pour des utilisations non commerciales, elles sont souvent en version *beta-test* donc instables. Il convient d'évaluer leurs apports réels avant de les utiliser pour enrichir les services.

Par exemple, l'éditeur de graphe Editor.app, utile pour certaines opérations d'INTEX, n'est disponible que sur système NeXTStep. et sa version clonée pour X11 reste à consolider.

C'est le domaine où essaye de s'engouffrer `Java`[11] pour promouvoir des fonctions directement exécutables par les clients W_3, qui ne seraient donc plus limités à la visualisation des documents. Cependant la vitesse de développement en `Java` est sans commune mesure avec l'intégration de programmes externes sur des serveurs. De plus les capacités demandées aux clients alourdissent les vraies manipulations hypertexte. L'utilisation de `Java` n'est vraiment pertinente que pour les applications nécessitant de nombreuses interactions locales aux clients et des préoccupations de temps réel ou de réalité virtuelle. Pour le traitement des langues naturelles qu'il nous intéresse, le modèle documentaire sera parfaitement suffisant.

11. URL: http://www.javasoft.com/

4.2 Interface *complète*

Comme nous l'avons vu Section 4.2.1, le protocole liant le client au serveur ne conserve pas la trace des opérations précédentes. Les accès aux commandes de base doivent être décrits précisément afin de définir formellement l'ensemble des états du processus, puisque

Il est à noter ces questions sont pertinentes quel que soit l'environnement d'exécution. Elles sont incontournables si l'on veut interfacer ces traitements via des appels CGI sous un serveur HTTP, mais elles restent pertinentes sous les interfaces graphiques usuelles (voir Figure 3.3).

Les informations sont passées suivant le standard *CGI*, défini par les serveurs les plus répandus (voir Section 3.3). La fonction *getArgs*() qui permet indifféremment d'obtenir les valeurs de paramètres passés de façon standard sur le système local ou selon les conventions des serveurs généralistes n'est pas la plus compliquée à concevoir (Figure 4.4).

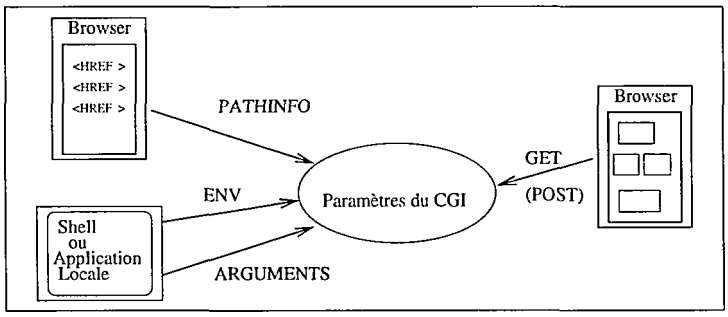


FIG. 4.4 – *Passage des paramètres*

4.2.1 Déroulement des opérations

L'ordre d'application des outils d'INTEX peut être reproduit à l'identique: Découpage du texte choisi en chaînes de caractères, puis confrontation aux dictionnaires de mots simples, de mots composés, recherche sur motifs ou catégories grammaticales ... (Figure 4.5)

Le niveau de décomposition des requêtes en opérations élémentaires tout comme leur ordre d'application peut varier selon les besoins.

Le serveur `http://www-ceril.univ-mlv.fr/` procède aux interprétations suivantes :

`/intex/` Page de sélection des paramètres.

`/intex/Freq/unTexte` demande la liste des chaînes de caractères avec leur nombre d'occurrences dans `unTexte`

`/intex/Dict/unText/(Delaf,sd1,sd2)/cd1` dénote `unText` étiqueté avec 3 dictionnaires de mots simples (DELAF, sd_1 et sd_2) et un dictionnaire de mots composés (cd_1).

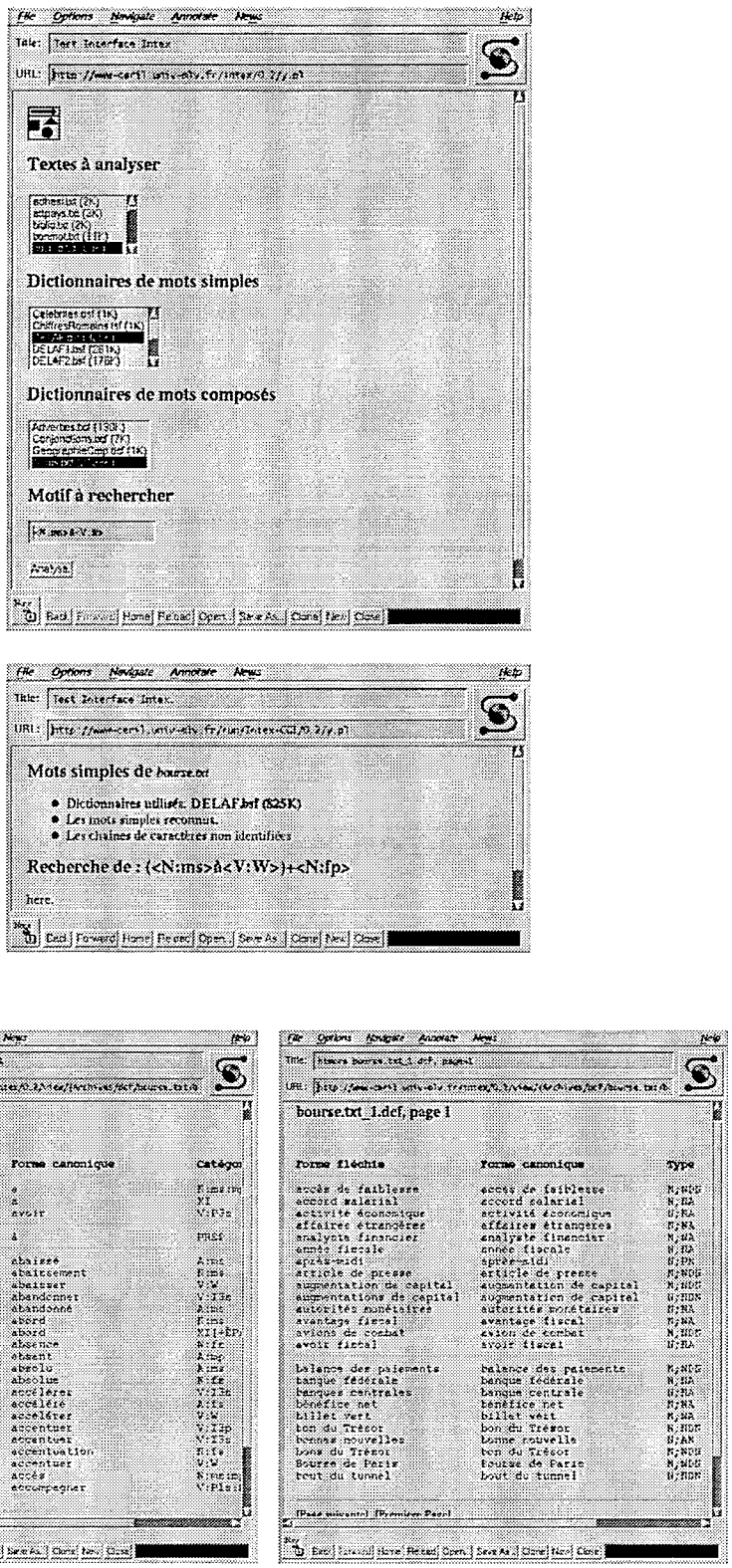


FIG. 4.5 – *Une requête complète*

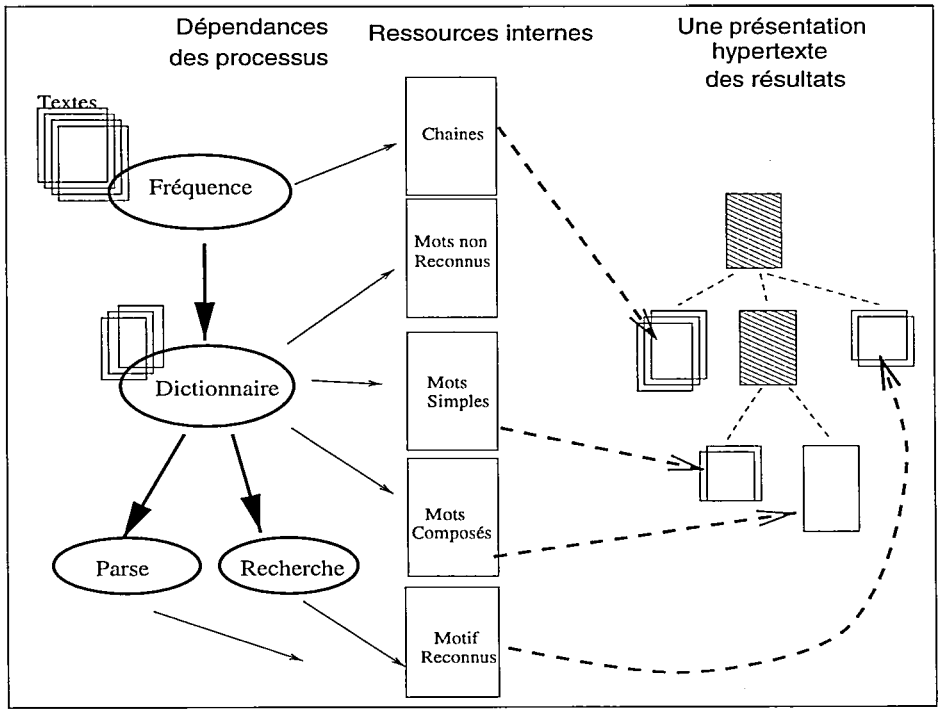


FIG. 4.6 – *"Hypertextage" d'une requête*

`/intex/Search/aText/(Delaf,sd1,sd2)/cd1/<N:ms>` renvoie le résultat de la recherche d'un nom masculin singulier dans le texte étiqueté défini précédemment.

Cette syntaxe d'appel est équivalente à celle normalisée, décrite Section 3.2. Il est important de définir une forme normale des paramètres, en effet il semble clair que

`/intex/Dict/unText/(Delaf,sd2,sd1)/(DELAC,cdi)`

et `/intex/Dict/unText/(sd1,Delaf,sd2)/(DELAC,cdi)`

sont équivalents. Un simple tri sur les arguments suffit à résoudre cette ambiguïté.

Il pourrait être intéressant d'enrichir cette normalisation en introduisant un peu de la sémantique des traitements.
Par exemple:

`/intex/Search/aText/Delaf/-/entournure`

est équivalent à `/intex/Search/aText/-/-/entournure`

puisqu'aucune information linguistique n'apparaît dans le motif de recherche.

Cette première étape de normalisation peut être raffinée à loisir, mais permettrait de limiter les validations manuelles que nous aborderons Section 4.3.

Dans les interfaces habituelles, la stratégie de reprise sur erreur conditionne la robustesse du système. Ce point n'est pas une priorité dans le développement de services W_3. Du fait des échanges en mode non connecté, l'utilisateur ne sera pas

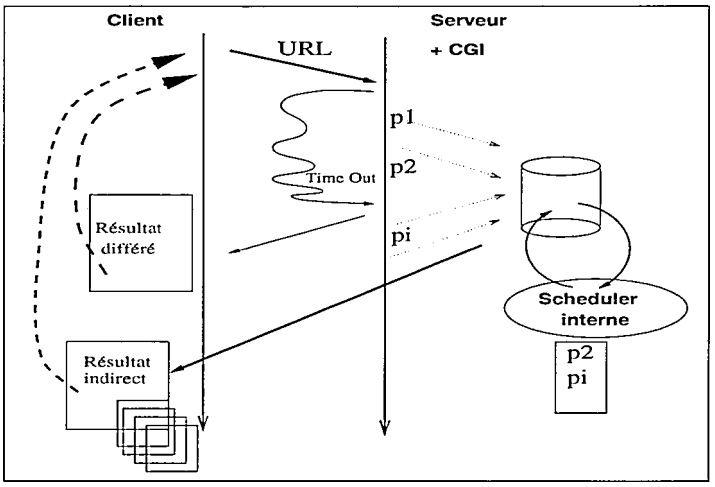


FIG. 4.7 – *Modèles de synchronisation*

bloqué si une erreur survient. Il pourra revenir en arrière, réessayer un parcours annexe et se faire sa propre idée du problème rencontré.

Au niveau du serveur, il sera difficile de présumer de la cause de l'erreur: problème de transmission, inconsistance des requêtes, indétermination des fonctions internes ... Tout au plus, on définira des comportements par défaut, des paramètres implicites comme par exemple un document demandant à l'utilisateur de compléter les arguments que l'on a identifiés comme erronés.

4.2.2 Synchronisation

Le temps de réponse du système doit rester raisonnable pour que l'utilisateur puisse juger du bon déroulement des opérations.

- Le temps de calcul pouvant être très long (supérieur à la minute) il faut pouvoir désynchroniser les calculs et l'envoi d'une réponse, celle-ci étant bien sûr partielle tant que les opérations demandées ne sont pas terminées sur le serveur.
- Le temps de transmission, dépendant essentiellement de la taille des données, les documents résultats seront découpés à la source, pour conserver une qualité de service acceptable.

La Figure 4.7 montre différentes possibilités de synchronisation des réponses. Elles découlent naturellement de l'environnement hypertexte W_3. En effet si on lance une requête produisant une ressource complexe dépendant de n ressources, plusieurs structures de présentation sont envisageables.

L'organisation des liens hypertexte peut privilégier une structure directe, indirecte, différée, complète selon la taille des données, la complexité des interdépendances et la finalité des opérations.

Si l'on peut concaténer les résultats en une seule page, donc en attendant la fin des processus $p_1..p_n$, le problème ne se pose pas.

Pour les résultats volumineux, le client est donc appelé à consulter les unes après les autres ces ressources, les stratégies d'ordonnancement des processus $p_1..p_n$ varieront en fonction des dépendances internes et de la présentation souhaitée (voir Figure 4.6).

Le temps de calcul peut être rédhibitoire pour l'utilisateur, il serait nécessaire de l'informer de l'état d'avancement des opérations. Ce point est cependant très difficile à mettre en oeuvre car il faudrait évaluer la charge du ou des serveurs qui dépend bien sûr de l'évaluation de la complexité des requêtes en cours. Ce travail reste à mener, peut-être en le reliant à un système indépendant de parallélisation des tâches.

4.2.3 Archivage des résultats

Les services produisent donc des documents synthétisant les opérations effectuées, avec des liens sur les résultats produits. L'expérimentation sur corpus revient donc à parcourir l'hypertexte des opérations, qu'elles soient calculées pour la première fois ou consultées dans une archive des opérations passées. L'URL devenant une notation de la séquence d'opérations nécessaires à son obtention.

Pour rester dans la philosophie documentaire, la structure des archives sera un hypertexte HTML. Cet aspect rejoint les systèmes documentaires classiques. Pour tirer le meilleur parti des fonctionnalités des clients W_3, il faut générer les informations spécifiques à chaque type de données présent sur le serveur.

Un archivage des sorties permet d'éviter les redondances de traitements, et d'accélérer considérablement l'obtention des résultats. Il peut faire double emploi avec un cache configuré au niveau du client mais restera utile en cas d'accès par plusieurs clients. La diffusion des documents devra donc être datée par la dernière modification opérée (Paramètre `Content-time:` des réponses HTTP).

Selon le client utilisé, les stratégies de cache peuvent varier, on pourra choisir d'inhiber cette fonctionnalité indésirable en joignant aux réponses la directive `Pragma: No-Cache`.

Le mécanisme de cache sur le serveur permet d'automatiser la compression des ressources, en effet sur du texte on gagne aisément un facteur 2 avec l'emploi des compresseurs usuels, type `gzip`.

Exemple:
```
$ gunzip -l /Corpus/Lemonde/1993.gz
compressed uncompr. ratio uncompressed name
55964730 151503018 63.0% 1993.ASCII-7b
```

Les paramètres de maintenance de ces archives devraient être fonction de la pertinence des requêtes et du coût du résultat:

→ Sur des textes de taille limitée, on pourra indifféremment privilégier l'espace disque ou le temps de calcul.

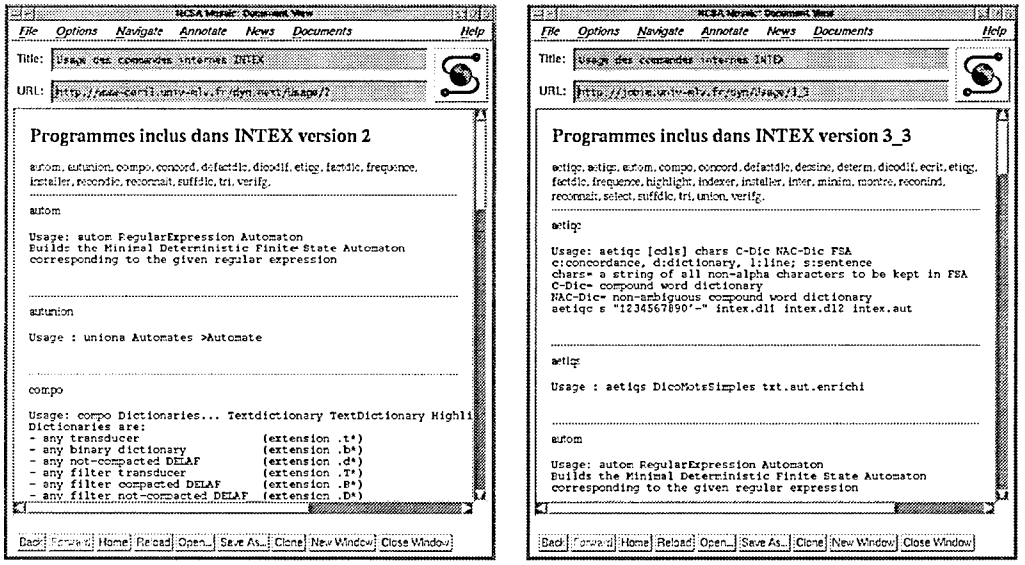


FIG. 4.8 – *Accès comparés aux logiciels et aux machines*

→ Sur des textes de taille importante, on conservera les recherches sélectives voire vaines, puisque leur taille sera limitée.

Si une recherche n'est pas sélective, elle sera de toutes façons difficilement exploitable manuellement. Il faudra lui appliquer d'autres opérations, soit en affinant les paramètres de la requête, soit en définissant une opération supplémentaire dans le schéma global des traitements.

Pour distinguer un petit texte d'un gros, la cohérence du contenu pourra aussi bien rentrer en ligne de compte que le nombre de caractères. En effet, les résultats auront plus de chances d'être cohérents que sur un ensemble disparate exploré à l'aveuglette.

4.2.4 Répartition

L'organisation des services W_3 permet de répartir l'exécution des tâches, dans un souci de répartition de la charge mais aussi dans le but d'homogénéiser l'environnement d'expérimentation.

Par exemple, la version 2 d'INTEX dont nous disposons ne tourne que sur des machines NeXT, alors que sa version 3.3 à été compilée aussi bien pour NeXT que pour HP (sous système NeXTStep).

La Figure 4.8 montre un accès aux deux versions configurées sur deux machines.

`http://www-ceril.univ-mlv.fr/dyn.next/...` sera exécuté sur notre serveur officiel,
alors que `http://www-ceril.univ-mlv.fr/dyn.hp/...`
sera redirigé sur `http://jobim.univ-mlv.fr/dyn.hp/`

Ne serait-ce que pour produire ces documents qui regroupent les informations d'usage de toutes les commandes internes d'INTEX, on est confronté à des problèmes divers de non-terminaison ou de violation des protections syatèmes ! Les comportements n'étant pas identiques d'une machine à l'autre, cette configuration pourra donc aussi aider à la détection des inévitables *bugs*.

4.3 Annotation et validation

Il est évident que les annotations manuelles des requêtes amélioreront considérablement la qualité du corpus. L'expérimentateur est souvent le mieux placé pour interpréter les résultats

Rechercher *Palestine* ou *Noël* dans la Bible n'est pas forcément très utile. De même *cyclone* est quasi-impossible dans les bulletins météo pour la France métropolitaine. Mais ces remarques sont difficilement généralisables, et la détection de la pertinence ou de l'inconsistance des requêtes difficilement automatisable. Là encore, l'enrichissement progressif des données semble être la seule issue.

A l'usage on pourra remplacer les résultats non pertinents par un document expliquant en clair pourquoi cette requête présente ou pas un intérêt. Cela peut tenir à l'origine du texte, aux paramètres choisis ou aux opérations spécifiques que l'on aurait déjà définies. Les occurrences seront plus ou moins prévisibles ou exploitables mais difficilement détectables automatiquement si l'on ne s'appuie pas sur des connaissances très diffuses.

Par exemple, pour le discours direct : dans notre corpus récupéré automatiquement (voir Section 2.4), *tu* n'apparaît quasiment jamais, alors que *je* est largement présent (> 450)

```
QdOrsay/revue.168.html: bras de fer
 'tu lèves l'embargo sur ma viande ou je bloque l'Union'
QdOrsay/revue.170.html:
 fait du chemin avec ARAFAT. Toi aussi tu lui donneras la main si tu veux
QdOrsay/revue.276.html:
  n'as-tu pas honte ? Des balles ont été tirées des fusils que tu leur as donnés,
Meteo/montreal.242.html:
 L'aperçu pour le long week-end... Soleil et plus chaud!!!...
 Non, mais... ça s'peut-tu???!!!
```

Les expérimentations sur les `formes figées`[12] fournissent de nombreux exemples d'annotations ponctuelles à relier avec les descriptions syntaxiques préexistantes ([Gro88, Lap88, Dan88] ...) :

Par exemple, dans l'archive de 1993 du journal **Le Monde** (140Mo) on ne trouve que 4 occurrences d'*entournure* qui correspondent parfaitement à la Table

12. URL: http://jobim.univ-mlv.fr/LexG/*Ph/différend

CNP2 :
gêner à les entournures + + + + - + -

```
national serait-il gêné aux entournures? Dimanche 27 juin, ...
fournisseurs, gênés aux entournures , restent discrets. Quand ce ...
même du PSOE. En plus, le pouvoir, gêné aux entournures ...
Carignon sera peut-être plus gêné aux entournures dans les ...
```

Le degré de figement varie d'une expression à l'autre, les informations du corpus peuvent aider à l'enrichissement des descriptions linguistiques, mais seulement après un examen minutieux, notamment pour l'exemple 2! :

```
RFI/info.245.html:
   ... lui pour régler par les armes leurs différends. Ses ennemis
TF1/infos.184.html:
   ... et aux démonstrations de force des différends courants nationalistes.
ftv/infos.276.html:
 ... Proche-Orient n'a pas permis d'aplanir les différends ni
ftv/infos.276.html:
 ...  Mais les différends sont toujours là...
```

A mettre en rapport avec les tables du lexique-grammaire :
Table C1I : trancher un différend + + - - +
vider un différend + - - - +

et Table C1IPN : aplanir des différends entre + - - - + - - - - +

Des études à portée plus générale utilisent de plus en plus l'expérimentation sur corpus pour corroborer les observations théoriques ([GT95]). Les recherches par approximation ne se limitent plus aux seuls calculs de probabilités, et peuvent prendre la forme de filtrages successifs, d'applications des grammaires locales, qu'il sera important de pouvoir consulter pour d'éventuelles réutilisations ou au moins pour information.

Des opérations peuvent aussi être automatisées pour détecter les occurrences de formes figées, ou encore suivre l'évolution du vocabulaire sur un corpus spécialisé, au fur et à mesure de l'acquisition du corpus.

Il est clair que certaines opérations simples sont incrémentales :

$$grep(entournure, Text_{n+1}) + \sum_{i=1}^{n} grep(entournure, Text_i) = \sum_{i=1}^{n+1} grep(entournure, Text_i)$$

mais on est rapidement tenté de mettre en place des processus plus complexes pour obtenir des synthèses, nécessitant des comparaisons spécifiques ou dépendant de validations manuelles.

Les spécifications restent à faire mais on voit que la structuration de l'exploration de corpus par des traitements automatiques permet non seulement d'éviter les redondances des calculs mais conduit naturellement à automatiser les requêtes.

Conclusion

Face à la complexité des données et des traitements de la langue naturelle, nous avons montré qu'il était possible de définir un environnement homogène intégrant en grandeur réelle les informations linguistiques disponibles.

En nous basant sur les travaux du LADL, nous avons mis en place une approche cumulative, synthétique et coopérative pour faire en sorte que les informaticiens puissent percevoir le nombre et l'ampleur réelle des problèmes et que les linguistes puissent évaluer les outils de traitement automatique, donc contribuent en retour à l'enrichissement des descriptions linguistiques.

Pour cela, la lisibilité et la souplesse d'utilisation de l'hypertexte, et de W_3 en particulier, font l'unanimité. Même si la lecture requiert des mécanismes un peu inhabituels, l'investissement de départ est minime en regard des possibilités offertes.

Nous proposons une certaine normalisation des données linguistiques basée sur une approche documentaire, donc cumulative, par opposition aux approches théoriques qui obligent souvent à occulter tel ou tel aspect des données disponibles. Nous développons en paralèlle des procédures de manipulations adaptées pour fournir différentes vues, plus ou moins directes ou croisées, afin d'en faciliter la maintenance et la consultation.

Cette première étape permet de fournir un environnement d'expérimentation homogène pour le traitement des langues naturelles. Nous définissons un adressage universel des nombreuses opérations possibles à partir des outils de traitements automatiques existants, ce qui nous permet de clarifier et d'optimiser les requêtes en ouvrant des perspectives pour la validation manuelle et coopérative des résultats.

La politique éditoriale de notre serveur W_3 était essentiellement tournée vers l'expérimentation des services, pas vers leur diffusion la plus large. La robustesse des fonctions présentées reste encore à tester en grandeur réelle avec des utilisateurs motivés par l'attente des résultats, plutôt que par les tests.

Une application pédagogique et didactique sera un prolongement naturel de ces résultats. La facilité d'accès aux dictionnaires comme au lexique-grammaire rendent leur utilisation moins abstraite.

Dans ce contexte, on doit malgré tout constater les limites des travaux en collaboration par média électronique, qui restent encore à très courte vue. Chacun s'empresse encore de recopier dans sans réelle volonté d'enrichissement. La

rédaction coopérative et répartie reste encore trop abstraite. Les manipulations automatiques étant encore réservées aux développeurs et la répartition des ressources encore trop exigeante en termes matériels, les habitudes sont difficiles à ancrer. L'avantage décisif d'HTML, qui est d'être retraitable n'est pas toujours perçu. Peut-être aussi parce que pour susciter des collaborations, il faut posséder des arguments et qu'un travail de thèse n'est pas forcément suffisant pour motiver les rédacteurs potentiels.

Ces limites vont s'estomper naturellement avec le développement des normes et logiciels liés à W_3, et la généralisation de leur emploi. Les retombées des synergies engendrées par la communauté W_3 vont contribuer à la recherche sur les langues naturelles, au moment où les utilisateurs constatent de plus en plus l'utilité et les carences de ces traitements, dans leur recherche sur le *Web*.

Outre l'intégration de données linguistiques complémentaires, le prolongement de notre travail s'inscrira donc dans le développement de services spécifiques. Cela passe par la formalisation de l'adressage de processus complexes par des services sans état. Ce point sous-tend de toute mise en place de services cohérents, mais reste peut abordé hors des utilisations minimales de CGI interface de bases de données. Ces questions générales qui ne sont finalement pas propres à un environnement W_3, permettront de décrire plus aisément les opérations complexes sur des donnés peu structurés ainsi que leurs interdépendances avec les connaissances théoriques existantes.

Bibliographie

[Aa93] P. Alcouffe et al. Azote: des tables du LADL au format Genelex. In *Actes du Colloque ILN*, Nantes, dec. 1993.

[BD92a] Emily Berk et Joseph Delvin. *Hypertext et hypermedia handbook.* McGrawHill, 1992.

[BD92b] G. Burnage et D. Dunlop. Encoding the british national corpus. In *English language corpora: Design, analysis et exploitation.* 13th international conference on English Language research on computerized corpora, 1992.

[BGL76] Jean-Paul Boons, Alain Guillet, et Christian Leclère. *La structure des phrases simples en français: constructions intransitives.* Genève: Droz, 378 p., 1976.

[BLa93] T. Berners-Lee et al. the World Wide Web initiative. In *Proceedings of INET 93*, 1993.

[BMMM94] Charles Brooks, Murray S. Mazer, Scott Meeks, et Jim Miller. Application-specific proxy servers as HTTP stream transducers. In *4th International World Wide Web Conference: "The Web Revolution"*, 1994. http://www.w3.org/pub/Conferences/WWW4/Papers/56/.

[Bri94] E. Brill. Some advances in tranformation based part of speech tagging. In *Proceedings of AAAI94*, 1994.

[BS94a] C. Boitet et M. Seligman. The "whiteboard" architecture: a way to integrate heterogeneous components of nlp systems. In *proceedings of COLING 94: Kyoto*, 1994.

[Bès94b] Gabriel Bès. Les tables de Méthodes en Syntaxe: Introduction à un mode d'emploi. *Cahiers de Praxématique*, 22, 1994.

[CGW95] Hamish Cunningham, Robert J. Gaizauskas, et Yorick Wilks. A general architecture for text engineering (gate) — a new approach to language engineering r&d. Technical report, Department of Computer Science University of Sheffield, UK, December 1995.

[Chr94] O. Christ. A modular et flexible architecture for an integrated corpus query system. In *Papers in computational lexicography: COMPLEX 94*, Budapest, 1994.

[Clé93] David Clémenceau. *Structuration du lexique et Reconnaissance de mots dérivés.* Thèse de 3ème Cycle, LADL, Université Paris 7, 1993.

[Com94] TIPSTER Architecture Committee. Tipster text phase ii architecture concept. Technical report, Department of Computer Science, New York University, November 1994.

[CS90] Blandine Courtois et Max Silberztein. Les dictionnaires électroniques du français. *Langue Française 87, pp. 11-22, Paris: Larousse*, 1990.

[Dan88] Laurence Danlos. Les expressions figées. *Langages*, 90, 1988.

[DS96] D. Decouchant et M. Romero Salcedo. Alliance: a structured cooperative editor on the web. In *CSCW et the Web, proceedings of the 5th ERCIM/W4G Workshop*, pages 7–12, 1996.

[Fie94] Roy Fielding. Maintaining distributed hypertext inforstructures. In *proceedings of 1st International Conference on the World-Wide Web*, May 1994. [http://www.ics.uci.edu/WebSoft/MOMspider/WWW94/paper.html].

[Fou95] Pierre-Yves Foucou. Représentation hypertexte de données linguistiques. In *Lexique-Grammaire Comparés et traitements automatiques*, 1995.

[GL92] Alain Guillet et Christian Leclère. *La structure des phrases simples en français - 2 : les constructions transitives locatives.* Genève: Droz., 1992.

[GL89] Maurice Gross et Christian Leclère. Modification de la définition des tables syntaxiques. Technical report, ASSTRIL, 89.

[Gro75] Maurice Gross. *Méthodes en syntaxe.* Hermann, 1975.

[Gro88] Maurice Gross. Les limites de la phrase figée. *Langages*, 90:7–22, 1988.

[Gro94] Gaston Gross. Classes d'objets et description de verbes. *Langages*, 115, 94.

[GS78] Jacqueline Giry-Schneider. *Les nominalisations en français, L'opérateur faire dans le lexique.* Genève: Droz., 1978.

[GT94] G. Greffenstete et P. Tapainen. What is a word, what is a sentence? problems of tokenization. In *Papers in computational lexicography: COMPLEX 94*, 1994.

[GT95] G. Greffenstete et S. Teufel. Corpus-based method for automatic identification of support verbs for nominalizations. In *proceedings of EACL95: Dublin*, 1995.

[Küb95a] Natalie Kübler. *L'automatisation de la correction d'erreurs syntaxiques.* Thèse de 3ème Cycle, LADL, Université Paris 7, 1995.

[Küb95b] Natalie Kübler. "parle voir suisse!". In *Lexique-Grammaire Comparés et traitements automatiques*, 1995.

[Lab88] Jacques Labelle. Lexiques-grammaires comparés: formes verbales figées en français du québec. *Langages, Paris: Larousse*, 90:pp. 73–97, 1988.

[Lap88] Eric Laporte. La reconnaissance des expressions figées lors de l'analyse automatique". *Langages*, 90, 1988.

[Lap94] Eric Laporte. Experiences in lexical disambiguation using local grammars. In *COMPLEX 94. Papers in Computational Lexicography*, 1994.

[Lec76] Christian Leclère. Datifs syntaxiques et datif éthique. In Jean-Claude Chevalier et Maurice Gross, editors, *Méthodes en grammaire française*, pages 73–96. Paris: Klincksieck, 1976.

[Lec90] Christian Leclère. Organisation du lexique-grammaire des verbes français. *Langue Française 87, Paris: Larousse.*, 1990.

[LK92] Béatrice Lamiroy et Jean-René Klein. Expressions figées du francais de belgique. In *Actes du colloque "Lexiques-Grammaires Comparés"*, 1992.

[LR95] David A. Ladd et J.C. Ramming. Programming the web: An application for hypermeida service programming. In *Third International World-Wide Web Conference : Technology, Tools et Applications*, April 10-14 1995.

[LS89] Eric Laporte et Max Silberztein. Vérification et correction orthographiques assistées par ordinateur. In *Actes de la Convention "Intelligence artificielle 1989"*, volume 1, pages 283–298. Paris: Hermès, 1989.

[Ma93] M. Marcus et al. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2), June 1993.

[Meu81] Annie Meunier. *Nominalisations d'adjectifs par verbes supports.* Thèse de 3ème Cycle, Université Paris 7, 1981.

[Nam94] Jee-Sun Nam. *Classification syntaxique des constructions adjectivales en coréen.* Thèse de 3ème Cycle, LADL, Université Paris 7, 1994.

[Rev91] Dominique Revuz. *Dictionnaires et lexiques: méthodes et algorithmes.* Thèse de 3ème Cycle, Université Paris 7, 1991.

[Ric94] Hélène Richy. A hypertext electronic index based on the grif structured document editor. *Electronic Publishing – Origination, Dissemination et Design*, 7(1):21–34, 1994.

[Rif90] Jean-Marie Rifflet. *La communication sous Unix.* Mc Graw-Hill, 1990.

[Roc93] Emmanuel Roche. *Analyse syntaxique transformationnelle par transducteurs et lexique-grammaire.* Thèse de 3ème Cycle, LADL, Université Paris 7, 1993.

[San96] Erik Sandewall. Towards a world-wide data base. In *WWW5 Conference*, May 6-10 1996.

[Sil93] Max Silberztein. *Le système INTEX.* Masson, 1993.

[Sil94] Max Silberztein. Intex: a corpus processing system. In *proceedings of COLING 94: Kyoto*, 1994.

[Skl93] Elsa Sklavounou. Un lexique-grammaire trilingue de noms composés (grec-français-anglais). Application au vocabulaire spécialisé du tennis. Mémoire de DEA, 1993. 124 p. + 368 p. annexes.

[TRG95] E. Tzoukermann, D. Radev, et W. Gale. Combining linguistic knowledge et statistical learning in french part-of-speech tagging. In *proceedings of SIGDAT Workshop: EACL Dublin*, 1995.

[Vas82] Philippe Vasseux. Le système lexsyn de classification des données du ladl. Technical report, LADL, 1982.

[Vou94] Atro Voutilainen. *Designing a Parsing Grammar.* Thèse de 3ème Cycle, Univ. of Helsinki, 1994.

[Yar94] D. Yarowsky. Decision lists for lexical ambiguity resolution: Application to accent restoration in spanish et french. In *proceedings of 32th annual meeting of ACL*, 1994.

[YM93] Y. Yannick-Matthieu. *Interprétation par prédicats sémantiques de structures d'arguments.* Thèse de 3ème Cycle, LADL, Université Paris 7, 1993.

Annexe A

Glossaire

A.1 Lexique-grammaire

Quelques définitions plus ou moins générales et succinctes pour se situer dans les descriptions syntaxiques.

Lexique-grammaire recensement des propriétés distributionnelles et transformationnelles des phrases simples.

Phrase simple unité lexico-sémantique minimale.

Structure Phrase dont certains éléments linguistiques peuvent être représentés par des variables.

Transformation relation d'équivalence sur les phrases simples, conservant le sens modulo variations aspectuelles (contrastive, causative, inchoative ...) "Il s'agit Plus un révélateur qu'un phénomène de la langue" ([Gro75]).

Table syntaxique Description des propriétés des phrases simples sous forme de matrice. Un "+", à l'intersection d'une ligne et d'une colonne dénote l'acceptabilité de la structure associée. Un "-" dénote l'impossibilité.

Verbe support de nom prédicatif Verbe "sémantiquement vide" portant les marques de temps et d'aspect et accompagné d'un nom prédicatif. Propriétés: contraintes sur le déterminant du nom prédicatif (Npred), double analyse de la séquence *Npred Prep N1*, formation d'un groupe nominal par effacement du verbe support.

Verbe opérateur dont l'application sur une phrase simple introduit un nouvel argument.
Ex: *Max met (Luc est en rage) = Max met Luc en rage*

- opW = sur phrase (ex: *je sais QueP*)
- opY = de temps

- op à lien = le sujet est en relation de co-référence avec un argument de la phrase
- opU = sur verbe

Substantif opérateur Introduit un argument phrastique
Ex: *l'impression que P, le fait que P*

Complétive "proposition" de la forme *que P* ou *ce que P* ou *si P ou P*, et servant d'argument à un.

Figement restriction sur la variabilité d'une structure.

Extension maximale description de tous les arguments essentiels.

Complément d'objet premier strictement approprié par rapport au sens premier du verbe
ex: *Paul nage la brasse*

Déverbal dérivé d'un verbe (V_n ou V_a).

Nominalisation paraphrase nominale d'une forme verbale ou adjectivale

Topicalisation mise en exergue d'un élément de la phrase.

Neutralité Relation entre les structures suivantes

- *N0 V N1 W* (ex: *Paul rentre la voiture dans le garage*)
  synonyme de *N0 faire que N1 V W*
- *N1 V W* (ex: *la voiture rentre dans le garage*)

Construction factitive Relation très générale. Ex: *Luc mange / ceci fait manger Luc*

Datif étendu ([Lec76]) ex: *Paul lui claque tout son argent, à Marie*

Et bien d'autres encore ... Merci à Anne Monceaux et Éric Laporte pour leurs remarques sur ces quelques approximations.

Entre autres sources: NCSA[1], Consortium W_3[2], UREC[3], FAQ W_3[4], Free On-line Dictionary of Computing[5].

A.2 W_3

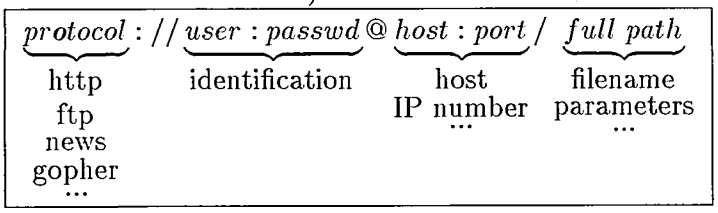
Voir [BLa93] pour une introduction générale à W_3 et *http://www.w3.org/* pour plus d'informations.

W_3 = World Wide Web = WWW = web = The Net = ...

client application du côté utilisateur qui établit la connexion vers un serveur pour obtenir un document.

serveur programme qui calcule puis envoie les documents demandés par ses clients.

URL (Uniform Ressource Locator) identificateur des "documents". (RFC 1738, 1808).



HTTP (HyperText Tranfer Protocol) protocole natif W_3 pour l'échange des documents hypertexte (RFC 1945).

CGI (Common Gateway Interface) standard défini par les principaux logiciels serveurs HTTP, pour le lancement de sous-programmes externes.

Fast-CGI Standard pour l'optimisation des appels fréquents de sous-programmes.

MIME (Multipurpose Internet Message Extension) spécification des formats de messages of Internet (RFC 1521).

1. URL: http://www.ncsa.uiuc.edu/SDG/Software/Mosaic/Glossary/GlossaryDL.html
2. URL: http://www.urec.fr
3. URL: http://www.urec.fr
4. URL: http://www.boutell.com/faq
5. URL: http://www-igm.univ-mlv.fr/foldoc

A.3 Internet

Internet voir Section 1.3.3

TCP/IP Protocoles *Réseau* et *Transport* formant les bases d'Internet.

ISOC (`Internet Society`[6]) organisation responsable des divers aspects du réseau.

IETF (`Internet Engineering Task Force`[7]) coordonne les recherches et développements de tout ce qui touche à Internet. Des groupes de travail étudient les applications, le routage, l'administration ...

IANA (Internet Assigned Numbers Authority) Service de l'*Internet Society* chargé d'enregistrer les valeurs des paramètres utilisés dans les différentes normes de protocoles Internet.

```
E-Mail
To: IANA@isi.edu
Subject: Registration of new MIME Access-type
for Message/external-body content-type
```

ISO (International Standardization Organisation) dépendant de l'ONU.

NIC (Network Information Center) Responsable de la gestion d'un sous-domaine Internet.

OSI (Open Systems Interconnection) Fameux modèle d'architecture des fonctions *réseau*, découpées en 7 couches.

Opérateur organisme qui fait fonctionner les équipements de communication d'un réseau.

Protocole de communication description d'un ensemble de règles que doivent suivre les équipements de communication et/ou les stations pour échanger des informations

Renater Réseau National de Télécommunications pour l'Enseignement et la Recherche (en France).

RFC (Request For Comment) Proposition de standard pour Internet.

Service tout ce que peut apporter un réseau. Les applications utilisables sur un réseau sont des services.

6. URL: http://www.isoc.org/whatis/
7. URL: http://www.ietf.org/

A.4 Standards documentaires

ASCII (American Standard Code for Information Interchange) Un des premiers encodage du jeu de caractères, limité à 7 bits.

ISO-8859 Norme de jeux de caractères 8bits. ISO-8859-1 (ou latin-1) pour le français et les langues "occidentales".

Unicode Norme de jeux de caractères 16bits. Sous-ensemble de la norme ISO-10646.

SGML (Standardized Generalised Markup Language) Norme de balisage de documents, ISO 8879:86.

DTD (Document Type Definition) Définition de la syntaxe d'un document SGML.

HTML (HyperText Markup Language) standard basé sur SGML.

TEX and LATEX Logiciels de mise en page de documents.

RTF (Rich Text Format) Format pivot de documents.

RTFD (Rich Text Format Directory?) Extension hypertexte de RTF, utilisée par le système NeXTStep.

JPEG, GIF, MPEG, MHEG, HyTime ... quelques normes média, multimédia, hypermédia

Documents Actifs

VRML (`Virtual Reality Modeling Language`[8]) Tentative de normalisation des interactions de réalité virtuelle.

Java Langage de programmation orienté objet (OO).

applet Programme `Java` "compilé" pour être exécuté quelle que soit la machine.

JavaScript Utilisation simplifiée de `Java` permettant de manipuler des fonctionnalités de *Netscape Navigator*.

Sous-culture

FAQ (Frequently Asked Questions) compilation des questions des utilisateurs d'un logiciel ou d'une liste de diffusion.

FYI (For Your Information) Note de lecture informelle.

8. URL: http://vag.vrml.org/www-vrml/

WYSIYG (What You See Is What You Get)

FMTEYEWTK ("Far More than Everything You've Ever Wanted to Know") FAQ plus fouillée pour les points délicats de Perl.

:-) The original smiley **:-)**

GNU (Gnu Not Unix) Projet lancé en 1983, visant à fournir un système d'exploitation complet pour le domaine public.

GPL (Gnu Publice License) Droits d'utiliser, de modifier et de redistribuer les sources du GNU (see also *CopyLeft*).

Linux Le plus répandu des noyaux Unix du domaine public.

Perl (Practical Extraction and Report Language) Langage de programmation plus particulièrement dédié aux manipulation de chaînes de caractères. Généralisation des ancêtres Unix dans le domaine comme `awk`, `sed`, `grep` ...

Annexe B

Programmes et Modules PERL

La mode des langages de programmation comme `lisp` ou `Prolog` étant passée, les intérêts convergent vers des langages évolués du type `Perl`[1] ou `Python`[2]
.

Malgré une syntaxe parfois absconse, `Perl` se révèle très puissant dans la manipulation de chaînes de caractères et disposant de beaucoup de librairies.

Les différentes fonctionnalités nécessaires à l'enrichissement du serveur ont été développées en Perl (version 5). Autant que possible nous avons privilégié une organisation du code orientée objet, sans prétendre y voir d'autre intérêt que la relative facilité de maintenance, Très peu d'héritages relèvent des problèmes linguistiques rencontrés.

Un grand nombre de taches, inhérentes à la normalisation W_3 sont gérées par des modules Perl disponibles dans le domaine public. L'évolution de ces librairies oblige à suivre les versions successives de Perl et réciproquement.

libwww-perl (5b6, 5.1b ...5.03) API client W3.

CGI.pm (2.00 .. 2.21) Fonctions de base des CGI.

GD-1.00 Manipulation d'images au format GIF.

dbm Unix key/content pairs database manager.

Perl version 5.0, 5.0001, ..., 5.0003.

Gestion des données linguistiques

Les manipulations de données présentées sont définies par les modules suivants :

1. URL: http://www.perl.com/
2. URL: http://www.python.org/

	Module	description
Ling	Delas.pm	Normalisation des formes simples
	Flexion.pm	Définition des classes flexionnelles et opérateurs de flexion
	Delaf.pm	Recherche de formes fléchies
	Delac.pm	Manipulation des mots composés
	LexG.pm	Gestion de l'arborescence
	Ex.pm	Traitement des exemples
	Prop.pm	... des entêtes de colonnes
	Table.pm	Recherche et formatage
	French.pm	Manipulation de formes canoniques complexes (ex: faire ne Neg s'en)
Doc	Collect.pm	Acquisition périodique pour corpus
	Extract.pm	Suppression de la structure du document
Intex	Format.pm	Conversion vers HTML
	Intex.pm	Fonctions génériques
	Intex2.pm	Appel des utilitaires (version 2)
	Intex3.3.pm	
	Intex3.4.pm	... à suivre

L'exemple suivant montre l'utilisation des modules `LexG` et `Table` pour l'affichage des entrées commençant par 'a' dans la table 4.

```
use LexG;
my $T = LexG::TableByCode( '4');
$T->Grep( "^a");
print $T->FormatHTML;
```

W_3

	Module	description
zCGI	z.pm	Hérite de CGI.pm
	Error.pm	Notification des problèmes
	Cmd.pm	Ordonnanceur de tâches sommaire
	Ressource.pm	Description interne des paramètres
	Archive.pm	Gestion de l'adressage absolu des ressources
	Sesson.pm	Enchaînement obligatoire des accès aux ressources
	asHTML.pm	abréviation des notations HTML
	Recode.pm	Conversions d'encodage des caractères

Points d'entrée principaux

Pour la consultation des dictionnaires :

CGI	description
`/Conjugaison/Verbe/mode - temps`	Consultations du DELAS (version 1996)
`/Flexion/Mot/`	
`/Delas.pl/`	
`/Grep/Perl regexp/num.max.`	Recherches diverses
`/Delaf5.pl/`	Consultation du DELAF (version compactée 1993)
`/Dict/dictname/forme fléchie`	Consultation des dictionnaires français, anglais, italien via Intex2.0

Pour les traitements automatiques sur corpus:

CGI	description
`/sort/texte du client`	Tri simple
`/ptx/texte du client`	Concordances de chaînes de caractères
`/intex/...`	voir Section 4.2.1
`/Collect/source`	statut des acquisitions automatiques de corpus
`/Extract/texte/struct`	Suppression de la structure du document

Les notations choisies pour l'accès au lexique-grammaire sont les suivantes :

CGI	description
/Biblio / ref	Recherche dans la bibliographie
/LexG/ table / motif	Recherche sur les entrées du lexique-grammaire,

Annexe C

Intégrité des liens cités

Hypertexte et Notation

Le document présent est rédigé avec LaTeX. Les liens hypertextes sont dénotés en gras et une note de bas de page indiquant l'URL de référence.

Les macros utilisées sont définies par le style `html.sty` légèrement modifié:

```
\htmlref{ }{ }
\htmllink{ }{ }
```

Vérification avec `Momspider`

```
#
# verification des liens
#
# Lancement:  momspider -i these.mom
#AvoidFile       NLP-U.avoid
MaxDepth 1
<Tree
    Name          Verification des citations
    TopURL        http://www-igm.univ-mlv.fr/~foucou/bookmarks.th.html
    IndexURL      http://www-igm.univ-mlv.fr/~foucou/Check/these.html
    IndexFile     TheseCite.html
    IndexTitle  Vérification de citations
    Exclude     http://www-ceril.univ-mlv.fr/
>
```

Les résultats appellent trois remarques:

- Une liste de liens est toujours à remettre en question. Les URL identifiés comme non valides (*Not Found*) sont laissés à titre illustratif mais ont été mis à jour dans le texte. Ces corrections sont difficilement automatisables.
- Les services non accessibles sont de plus en plus rares, mais au nombre des *Time-out* on peut constater qu'Internet n'est pas (encore?) totalement fiable.
- les liens redirigés sont une facilité pour la maintenance des serveurs, ils ne constituent pas forcément des alertes.

Index started: Wed, 30 Oct 1996 4:20:26 by foucou@monge.univ-mlv.fr

Summary of Results

	References		Unique URLs		Local URLs		Remote URLs	
	number	pct	number	pct	number	pct	number	pct
Traversed	0	0.00	1	0.96	0	0.00	1	0.97
Tested	95	88.79	96	92.31	1	100.00	95	92.23
Reused	3	2.80	0	0.00	0	0.00	0	0.00
Avoided	2	1.87	1	0.96	0	0.00	1	0.97
Untestable	7	6.54	7	6.73	0	0.00	7	6.80
Broken	11	10.28	11	10.58	0	0.00	11	10.68
Redirected	10	9.35	10	9.62	1	100.00	9	8.74
Changed 7	10	9.35	10	9.62	0	0.00	10	9.71
Expired 0	0	0.00	0	0.00	0	0.00	0	0.00
Local	1	0.93	1	0.96	1	100.00	0	0.00
Remote	106	99.07	103	99.04	0	0.00	103	100.00
Totals	107	100.00	104	97.20	1	0.93	103	96.26

Broken Links

Link 404 Not Found http://crl.nmsu.edu/clr/CLR.html
Link 404 Not Found http://webstar.cerc.wvu.edu/lpi/WebStarDoc.html
Link 603 Timed Out
http://www.utoronto.ca/webdocs/HTMLdocs/NewHTML/htmlindex.html
Link 404 Not Found http://info.isoc.org/home.html
Link 602 Connection Failed http://www.texas.org/
Link 603 Timed Out http://hoohoo.ncsa.uiuc.edu/cgi/env.html
Link 603 Timed Out http://www.webcrawler.com/
Link 603 Timed Out http://humanities.uchicago.edu/ARTFL/

Redirected Links

Link 302 Found http://www.omnigroup.com/Software/OmniWeb
Link 302 Found http://www.w3.org/
Link 302 Found
http://www.w3.org/hypertext/WWW/Protocols/HTTP/HTRESP.html
Link 302 Found http://www.w3.org/hypertext/Standards/Overview.html
Link 302 Found http://harvest.cs.colorado.edu/
Link 302 Found http://www.ics.uci.edu/WebSoft/MOMspider/
Link 301 Moved http://dutifp.twi.tudelft.nl:8000/checkbot
Link 302 Found http://xxx.lanl.gov/cmp-lg

```
    Link 302 Found
    http://www.w3.org/hypertext/DataSources/bySubject/Overview.html
    Link 302 Found
    http://www.w3.org/hypertext/WWW/Protocols/HTTP/References.html

Changed Link Destinations

    Link 200 OK http://wombat.doc.ic.ac.uk/
    Last-modified: Saturday, 26-Oct-96 23:28:12 GMT
    Trav 200 OK http://www-igm.univ-mlv.fr/~foucou/bookmarks.th.html
    Last-modified: Wed, 30 Oct 1996 18:19:56 GMT
    Link 200 OK http://www.sil.org/sgml/sgml.html
    Last-modified: Tuesday, 30-Oct-96 01:06:41 GMT
    Link 200 OK http://www.adobe.com/
    Last-modified: Tuesday, 29-Oct-96 02:56:54 GMT
    Link 200 OK http://www.yahoo.com/
    Last-modified: Wed, 30 Oct 1996 08:04:37 GMT
    Link 200 OK http://www.next.com/
    Last-modified: Tue, 29 Oct 1996 02:54:30 GMT
    Link 200 OK http://www.gnu.org/
    Last-modified: Sat, 26 Oct 1996 23:29:03 +0000
    Link 200 OK http://www.python.org/
    Last-modified: Fri, 25 Oct 1996 19:07:04 GMT
    Link 200 OK http://www.submit-it.com/
    Last-modified: Friday, 25-Oct-96 22:46:28 GMT
    Link 200 OK http://www-ceril.univ-mlv.fr/ln/
    Last-modified: Sat, 26 Oct 1996 ? GMT


This index was generated by MOMspider/1.00

Done Traversing http://www-igm.univ-mlv.fr/~foucou/bookmarks.th.html
at Wed, 30 Oct 1996 4:30:51 -- 0 remaining on queue
```

Annexe D

HTTP

(sources: proposition de standard `HTTP/1.0`[1] *)*

D.1 Méthodes d'accès aux ressources

GET Demande d'une ressource avec paramètres encodés dans l'URL.

POST Demande d'une ressource avec envoi des paramètres par message interne au protocole. Exemple:
```
POST /LexG HTTP/1.0
Content-lenght: 20
champ1=val&champ2=val&
...
```

PUT Envoi du document courant au serveur, peu implémenté mais voir par exemple `Symposia`[2] .

HEAD Demande du statut de la ressource (type, date)

CHECKIN Dépôt d'une version d'une ressource (pas implémenté).

CHECKOUT Demande d'une version ... (pas implémenté).

Ces requêtes peuvent préciser au serveur les informations suivantes : `Referer`, `User-Agent`, `From`, `Authorization`, `If-Modified-Since`.

D.2 Codes de retour

Le protocole HTTP normalise des codes de retour pour informer le client du déroulement de sa requête.

Ces codes respectent la nomenclature suivante:

- *2xx* requête complétée,

1. URL: http://www.w3.org/hypertext/WWW/Protocols/HTTP/HTRESP.html
2. URL: http://symposia.inria.fr/symposia/

- *3xx* redirection automatique,
- *4xx* problème au niveau du client,
- *5xx* problème local au serveur

Il n'est pas toujours possible de distinguer clairement les causes d'erreurs, ces codes sont donc données à titre informatif. En plus du code, la réponse peut contenir un document, donc lisible pour l'utilisateur, décrivant le problème rencontré. Ce document ne peut être d'un type MIME autre que `text/plain, text/html`, sauf acceptation explicite du client.

Les codes les plus usités sont les suivants :

OK 200 La requête a pu être satisfaite.

No Response 204 idem mais il n'y a pas de document associé. Utile si le document courant doit rester affiché.

Bad request 400 La syntaxe de la requête est erronée donc ne peut être satisfaite.

Unauthorized 401 Indique au client qu'il doit utiliser un mécanisme d'identification pour que sa requête soit traitée. Le client devra donc réessayer avec un paramètre **Authorization**.

PaymentRequired 402 Indique au client qu'il doit remplir un paramètre de facturation.

Forbidden 403 L'accès à la ressource interdit, indépendamment du paramètre **Authorization**.

Not found 404 Le serveur ne peut associer une ressource à l'URL demandée.

Internal Error 500 Le serveur a rencontré une problème inhabituel qui l'empêche de satisfaire la requête.

Not implemented 501 Le serveur n'offre pas la fonctionnalité demandée.

Moved 301 Renvoie la nouvelle URL où doit être demandée la ressource initiale.

Found 302 La ressource est accessible sur le serveur, mais est susceptible d'être déplacée.

Not Modified 304 Ressource non modifiée depuis la date transmise en paramètre **If-Modified-Since** par le client.

Table des figures