Ajout d'une recherche libre à une base de donnée pilotée par thésaurus

Jean-Pierre Tosoni

CONSERVATOIRE NATIONAL DES ARTS ET METIERS

CENTRE RÉGIONAL ASSOCIÉ DE VERSAILLES

MÉMOIRE

présenté en vue d'obtenir

le diplôme d'Ingénieur C.N.A.M.

Spécialité : Informatique, Systèmes d'Information
option : architecture et ingénierie des systèmes et des logiciels (AISL)

par

Jean-Pierre TOSONI

Ajout d'une recherche libre à une base de donnée pilotée par thésaurus

Soutenu le 24/06/2013

JURY

PRESIDENT :

MEMBRES :

Objet du document

Ce mémoire présente comment, dans un contexte Web, un système de classement par thésaurus paramétrable, basé sur des fichiers et des algorithmes « maison », a été étendu en lui adjoignant un mécanisme de recherche en texte libre, accroissant les possibilités de consultation sans perdre les fonctionnalités existantes.

Résumé

La société TRANSWORK fournit des services Web d'indexation de données basés sur des thésaurus au lieu d'une base de donnée multicritère classique. Elle souhaitait étendre les possibilités du système en lui ajoutant une recherche en texte libre. Cet ajout n'était pas trivial car il devait se greffer sur des structures de données inhabituelles.

La réalisation a porté sur deux fonctions majeures : l'indexation des données et un système d'acquisition / recherche / restitution des résultats, et une fonction secondaire : la conversion des données existantes pour le démarrage de l'exploitation. Les mots à reconnaître sont importés de diverses structures : données utilisateur, thésaurus.

En l'absence de méthodologie d'entreprise, l'étude a suivi la démarche du cycle en V : spécification du besoin, conception, réalisation, tests unitaires, d'intégration et fonctionnels. La conception a nécessité la recherche de composants logiciels spécialisés existants et leur greffe en langage C, SQL et scripts shell dans l'architecture « maison ». L'intégration a pris en compte la structure originale des fichiers et l'adaptation d'une base de données relationnelle existante.

L'aboutissement du projet a eu pour effet immédiat l'amélioration de l'ergonomie de recherche, la création d'une offre commerciale nouvelle et permet d'envisager à long terme une ouverture vers la reconnaissance vocale.

Mots clés : thésaurus, recherche libre, base de données, services Web, Internet

Abstract

The TRANSWORK company provides data indexing services on the web. Indexing is thesaurus-based rather than using classical multi-criteria databases. The company has wished to extend the scope of the system by adding a full text search capability. This enhancement was not straightforward due to the unusual data structures it must fit with.

The implementation focused on two main features: data indexing and a input/search/results rendition system, and a secondary feature: conversion of existing data to start full-scale utilization. The recognized words are taken from various data structures: thesauruses, user provided data.

Missing a company-wide methodology to rely on, the study chose to follow the V-model as its development process: requirements, design, implementation, unit/integration/system tests. The design required investigation in existing specialized software blocks and their insertion in the proprietary software architecture, using C, SQL and shell scripting languages. The integration handled the uncommon files structure and the adaptation of an existing relational database.

The completion of the project immediately resulted in user interface improvements in the search process and the expansion of the product line; in the long run it allows planning for a voice recognition user interface.

Key words : thesaurus, full text search, database, web services, Internet

Remerciements

Je remercie messieurs Edmond CHABOCHE et Joseph TOLEDANO pour m'avoir confié, au sein de la société TRANSWORK, un projet touchant directement à son cœur de métier, tout en me laissant toute latitude pour le réaliser.

Au C.N.A.M, je remercie tant l'encadrement administratif toujours serviable et disponible avec Monsieur BOURDAIRE, Monsieur EGNER et Madame MAST, que les professeurs qui m'ont suivi. J'ai tout particulièrement pu profiter de l'étendue des compétences et de la culture de Monsieur ALEXANDRE, de Monsieur SCHOUHMANN et de mon tuteur Monsieur KERYVEL.

Enfin, rien n'aurait été possible sans la patience de mon épouse Béatrix, qui a sacrifié tant de week-ends et de congés pour dégager le temps nécessaire à mes études. Je remercie aussi à cette occasion toutes les personnes de ma famille qui m'ont encouragées, et auprès desquelles je m'excuse de ne pouvoir les citer exhaustivement.

Sommaire

Table des illustrations

1. Aperçu général du projet

1.1. Le besoin et son contexte

La société TRANSWORK a été créée en 1999 pour développer l'ergonomie des logiciels de gestion de fichiers. A l'époque, l'expérience des fondateurs en matière de serveurs Vidéotex (principalement les services Minitel) leur avait permis d'identifier un problème récurrent dans les services d'accès « en ligne » à des fichiers de données. Tant dans les services Minitel que dans les sites Internet naissants, les systèmes de recherche dits « multicritères » ne guidaient pas suffisamment l'utilisateur ordinaire pour qu'il trouve ce qu'il cherche dans une base de données : les formulaires de sélection, trop proches du concept informatique de clés d'accès, permettaient des croisements de critères aboutissant à une sélection vide sans expliquer comment élargir la recherche à des réponses approchantes. L'usager avait alors tendance à réduire ses critères et aboutissait à des listes longues, sans indication des critères qui permettraient de mieux cibler ses choix.

S'appuyant sur une expérience et un acquis technique en matière de services à distance multi-utilisateurs, TRANSWORK a créé le produit Métaguide. Celui-ci permet de définir des fichiers dont l'exploitation est guidée par un thésaurus qui lui est associé : comme le montre la figure 1.1 ci-dessous, les données du fichier sont classées par une structure extérieure, le thésaurus, dont la conception est confiée à un spécialiste du « métier » auquel s'applique le fichier. Ainsi, le système de guidage des recherches ne s'appuie pas aveuglément sur des clés d'accès mais sur des concepts proches des préoccupations de l'usager.

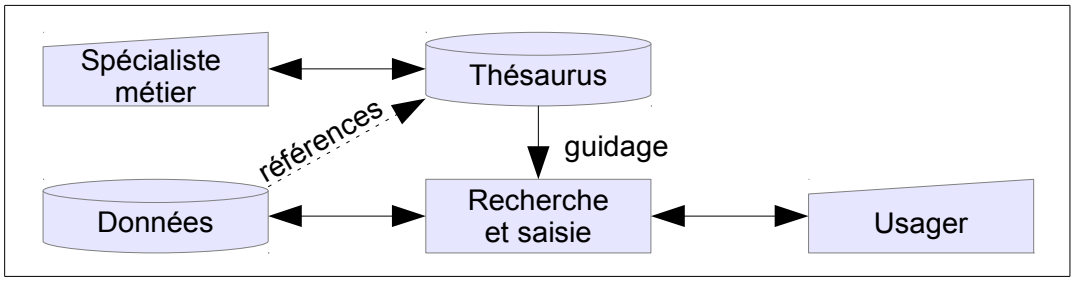


Figure 1.1: Accès aux données piloté par thésaurus

Cependant les systèmes de recherche, comme les moyens techniques, ont beaucoup évolué. L'entreprise, prenant conscience de l'importance croissante des tablettes et autres smartphones chez les usagers d'Internet, a dans un premier temps constaté les difficultés ergonomiques posées par la saisie de textes et la petitesse des écrans, puis imaginé que le moyen d'acquisition idéal serait la voix. Pour adapter Métaguide à cette évolution majeure, il fallait pouvoir analyser des sons, reconnaître des mots, les retrouver dans des fichiers ou des thésaurus.

En parallèle lors de la présentation de Métaguide, TRANSWORK s'est toujours heurtée à une confusion des clients entre la recherche dans un fichier et la recherche de pages dans les moteurs classiques comme Google™, Lycos™, Yahoo™ et bien d'autres. Le point d'entrée de tous ces moteurs de recherche du web est une zone de saisie de texte libre, et les prospects de TRANSWORK s'attendent à priori à une telle interface.

Faisant la synthèse de ces deux besoins, la direction de TRANSWORK a conclu que la résolution du second serait un premier pas vers la solution du premier, et a formulé le besoin d'ajouter, à la recherche guidée par thésaurus, un système de recherche par texte libre.

La direction de TRANSWORK, maître d'ouvrage (MOA), m'a confié la recherche et l'implémentation d'une solution à son problème ainsi défini. Après avoir précisé le cahier des charges, je devrai définir une solution, l'implémenter et la mettre en exploitation sur le serveur de l'entreprise.

Du point de vue du logiciel les principaux axes de développement sont connus :

- Il faut d'abord lier chaque fiche enregistrée, avec les mots qui la concernent. Mais ces mots, contrairement à des fichiers classiques, ne sont pas nécessairement dans les fiches : certains sont dans le thésaurus associé, d'autres dans les informations de « contact » c'est-à-dire en principe les coordonnées du propriétaire de la fiche, d'autres peuvent apparaître dans des pages Web liées à la fiche.
- Il faut aussi ajouter une saisie de texte, l'analyser et comparer les mots (quelconques) avec la masse des mots liés aux fiches, puis pouvoir faire des croisements entre les listes de fiches déterminées par chaque mot saisi.
- Finalement il faut transférer la sélection effectuée à la recherche guidée pour profiter de celle-ci si la liste de réponses est encore trop longue.

Cependant, sans s'arrêter à cette vision très technique, il faudra aussi étudier l'impact sur l'administration du serveur, les sauvegardes, les différentes catégories d'utilisateurs de Métaguide. De plus, bien que la direction de TRANSWORK ne formule pas de demandes à ce sujet, il me faudra vérifier l'existence d'éventuels besoins non fonctionnels.

Pour matérialiser un peu mieux l'ampleur du projet, nous nous sommes servis d'une solution hypothétique, alors même que le cahier des charges était incomplet, pour évaluer un délai de réalisation vraisemblable. La durée du projet a été évaluée dans un premier temps à 31 semaines-homme. L'entreprise s'appuie clairement sur les logiciels libres, garants des possibilités de portage de son produit ; le projet ne devrait pas utiliser des logiciels non libres, sauf à les développer nous-même. On espère un coût nul en achat de produits externes ou de redevances.

1.2. La réalisation

L'aboutissement du projet offre de nouveaux services à plusieurs catégories d'utilisateurs de Métaguide. Les usagers « ordinaires », ceux qui utilisent la recherche guidée par thésaurus, obtiennent un moyen supplémentaire de sélection par mots-clé, qui se manifeste par un nouveau champ de saisie sur la page d'accueil de la recherche. Si les résultats sont nombreux, ils sont ensuite soumis à la recherche guidée : la fonctionnalité essentielle qui fait l'intérêt de Métaguide est préservée.

Une nouvelle catégorie d'usagers apparaît : la zone de recherche par mots-clés peut en effet être isolée et placée sur un site partenaire, qui aurait conclu un contrat de délégation de recherche avec TRANSWORK. Ce site externe, nommé « site appelant », transmet les mots-clés et un contexte de recherche à Métaguide qui procède à la recherche libre, enchaîne si c'est utile une recherche guidée, et renvoie au site d'origine avec l'information finale sélectionnée.

Les gestionnaires, producteurs spécialistes « métier » et documentalistes qui ont en charge la création des thésaurus, obtiennent aussi des informations du système. Les combinaisons de mots-clés proposées par les usagers, mais qui ne sélectionnent pas de fiches, sont des indices importants pour décider une réorganisation ou une amélioration ponctuelle des thésaurus, car ils mettent en évidence les besoins non satisfaits des usagers. Ces combinaisons sont enregistrées dans un journal pour analyse ultérieure.

Pour obtenir ce résultat la complexité réside dans une bonne préparation des données à traiter, en particulier des index de mots-clés. Or la gestion de fichiers dans Métaguide repose à la fois sur une base de données relationnelle classique, et sur GLADIA, un gestionnaire de fichiers maison, peu puissant mais conservé pour des raisons historiques. D'une part, les informations qui contiennent des mots-clés intéressants sont enregistrés dans GLADIA, d'autre part le SGBD contient intrinsèquement les outils capables d'indexer les mots-clé pour une recherche libre. Au delà du projet actuel, l'évolution souhaitable de Métaguide est de privilégier l'utilisation de la base de données par rapport à GLADIA pour rendre le produit plus évolutif et maintenable. En conséquence le projet se concentre sur l'ajout de tables dans la base, pour concentrer les mots-clés, les indexer et utiliser les outils disponibles.

Les mots-clés traités sont issus de trois sources (Figure 1.2: Données indexées pour la recherche libre). Chaque fiche contient un descriptif libre (1ère source), elle est associée à un propriétaire avec des informations de contact (2ème source), et elle fait référence à des attributs dont le texte est dans un thésaurus (3ème source). D'autres sources de mots-clés ont été écartées de cette première version lors de la rédaction des spécifications. Les trois types sont disponibles dans différents fichiers GLADIA, lequel n'offre pas de possibilités de jointure. A cela vient s'ajouter la possibilité future d'opérer les recherches en plusieurs langues. On obtient dans la base une table supplémentaire qui représente les fiches dans chaque langue, une table pour représenter les attributs offerts par le thésaurus associé, une troisième pour les différentes langues des attributs, et une quatrième pour définir les attributs utilisés par chaque fiche. Des clés étrangères relient ces différentes tables.

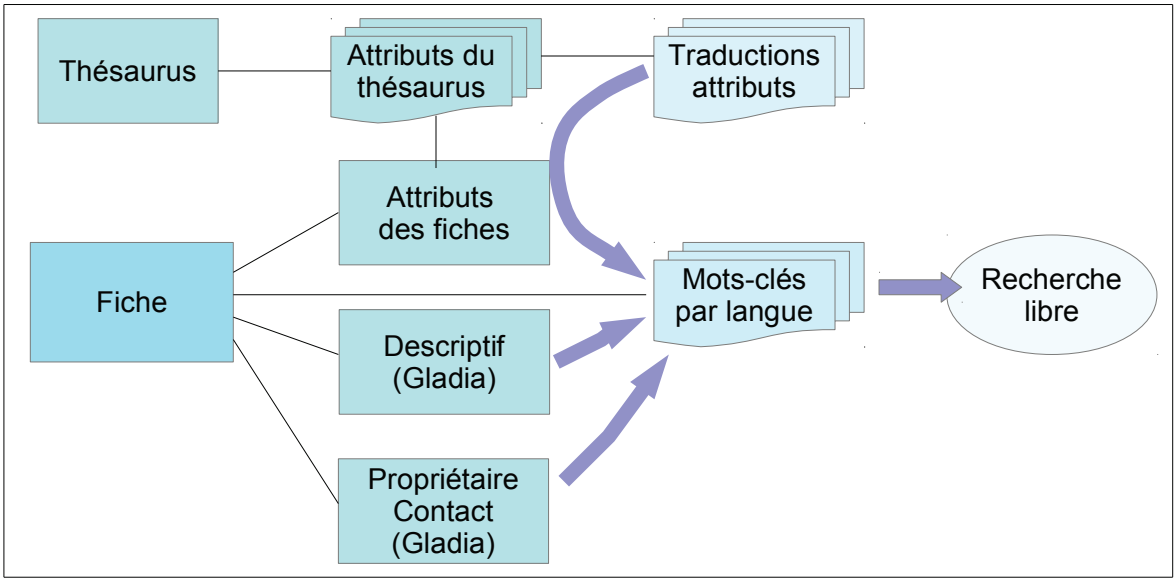


Figure 1.2: Données indexées pour la recherche libre

Une première série de traitements a pour but de créer les informations dans la base de données. Les textes des attributs, issus des thésaurus, sont transférés en une fois lors des opérations de mise en service des thésaurus qui ont lieu occasionnellement. Les autres textes, modifiés à volonté par les usagers et le service de support technique, sont transférés à chaque modification d'une fiche ou de son propriétaire. Le traitement des fiches, éclaté en divers endroits du code de Métaguide, doit être partout revu : ceci est simplifié par la création d'un ensemble de fonctions centrales de gestion des fiches qui remplacent le code dispersé. Le code SQL et des procédures stockées génèrent alors automatiquement les index voulus.

Une seconde série de traitements concerne l'exécution d'une recherche libre à partir des mots-clés. Le code est simple car il utilise des fonctions intégrées au SGBD, mais pour transmettre les résultat à la recherche guidée celle-ci doit subir une petite extension. Les recherches qui n'aboutissent pas sont notées pour analyse ultérieure.

Un troisième groupe de traitements s'attaque spécifiquement à la communication avec les sites appelants. Il s'agit d'intercepter les appels entrants dans le serveur Web, et de renvoyer le navigateur de l'usager vers la page appropriée du site externe à l'issue de la recherche.

L'ensemble de l'étude a lieu dans un contexte de micro-ordinateur à processeur Intel® tournant sous Linux®. Trois systèmes sont utilisés : Un serveur en exploitation, pour y installer les produits de l'étude ; un système de développement installé avec la version de développement des logiciels du serveur (serveur Web Apache, SGBD PostgreSQL, etc.), et un troisième système, dans une machine virtuelle [4], qui reproduit le serveur officiel avec un extrait de ses données, pour procéder aux tests finaux de la réalisation.

Les principaux langages utilisés : shell « bash » de Linux, SQL et langage C, nécessitent à la fois une bonne connaissance des commandes POSIX de Linux et des bibliothèques d'outils maison. Deux d'entre eux sont essentiels dans la réalisation : le composant Osmose est utilisé pour construire des tables (au sens des bases de données) dynamiques en mémoire vive, et on le retrouve dans les interfaces avec les autres composants. Ensuite, le composant de recherche guidée est le maillon final du processus de recherche libre.

L'installation finale sur le serveur de la société a été préparé avec soin pour réduire la durée d'indisponibilité et le risque de défauts de fonctionnement consécutifs à l'installation. Le découpage en phases indépendantes a permis de mieux maîtriser ces aspects.

1.3. La conduite du projet

En l'absence de pratiques recommandées par l'entreprise, le projet de recherche libre a été géré en fixant des méthodes proportionnées à sa taille et à celle de l'entreprise.

Organisé selon le modèle classique du cycle en V, il a pu avancer pas à pas dans les différentes étapes de définition du projet, de conception, de réalisation, test et mise en place, en s'appuyant à chaque fois sur un acquis stable pour aborder l'étape suivante. Ceci a permis en particulier de maîtriser la créativité du MOA, au travers de la rédaction d'une spécification précise, d'un planning prévisionnel, d'une liste de tâches évoluant au jour le jour pour éviter les oublis, et de validations intermédiaires par le MOA.

L'ensemble du projet a été découpé en cinq sous-projets : mise à niveau du SGBD, évolution du schéma de la base de données, création des informations de recherche libre de mots-clés, recherche intégrée aux services existants, recherche fournie comme service à

des sites web externes. Les tâches de ces sous-projets : conception, réalisation et tests, ont été sérialisés en tenant compte des interactions entre les sous-projets.

Les principaux aléas, à savoir un besoin mal défini d'une part, une défaillance du serveur de l'entreprise lors de la mise en service d'autre part, étaient connus et ont été pris en compte bien qu'il n'y ait pas eu de gestion de risques à proprement parler. Une réflexion à postériori a fait apparaître des risques secondaires liés à la disponibilité des personnes ou à l'inadéquation des briques logicielles ou des outils de sauvegarde. Ceci a mené en fin de projet à envisager l'utilisation d'un système de gestion de versions tel que GIT[1] ou Subversion[2], mais cette évolution n'a pas été menée à terme.

Enfin, le planning initial qui s'étendait sur sept mois, a été entièrement revu pour l'adapter à la solution choisie. La quantité de travail est restée en gros la même, mais le délai de livraison s'est allongé à un an, d'octobre 2011 à septembre 2012, pour tenir compte de la disponibilité des personnes.

1.4. Le bilan

Le projet a atteint ses objectifs. Son but a été clarifié, les fonctions hors sujet écartées et les fonctions utiles menées à bien dans les délais attendus. Bien qu'il réponde aux attentes, l'exploitation commerciale du module n'a pas démarré immédiatement. Néanmoins des sites Web de démonstration ont été mis en chantier.

Les aspects linguistiques ont été importants dans la phase de spécification. Un grand nombre de fonctionnalités étaient proposées, et il a fallu faire des choix selon leur cohérence avec les objectifs, leur faisabilité technique et le temps imparti. Ainsi :

- les transformations grammaticales (flexions) ont été prises en charge, mais pas la recherche de mots synonymes ;
- les traitements orthographiques (correction de la saisie) et sémantiques (interprétation de la saisie pour autre chose qu'une recherche) ont été considérées comme des aspects séparés de la recherche libre proprement dite ;
- la possibilité de traiter plusieurs langues a été temporairement écartée. Métaguide permet la traduction des thésaurus et des pages de présentation, mais d'autres informations nécessaires, les descriptifs des fiches, n'existant que dans une langue, il était impossible en l'état de réaliser un support multi-langue complet et cohérent, bien que les modules nécessaires existent dans le SGBD ;
- des extensions sémantiques ont été envisagées mais ne seront réalisées que si le composant s'avère insuffisant à l'usage. Ces extensions pourraient profiter de la structure arborescente du thésaurus pour réaliser des recherches approximatives.

On constate donc qu'il reste de la place pour des améliorations et des évolutions. Des précautions ont été prises pour les favoriser. Tout d'abord, le SGBD PostgreSQL a été modernisé pour profiter à la fois, d'un système de recherche « full text » plus stable, et de mises à jour régulières (qui avaient cessé sur la version installée initialement). Ensuite, les nouvelles tables ont été conçues pour servir dans un contexte multilangue. Enfin, dans l'hypothèse d'un remplacement définitif du composant GLADIA par la base de données, le code de Métaguide a été réorganisé pour isoler les accès GLADIA en un seul point.

[1] http://git-scm.com/
[2] http://subversion.apache.org/

Les performances attendues de la recherche libre de mots sur toute la base, bien que validées par le MOA comme conformes aux attentes, se sont révélées déplorables dans certains cas particuliers négligés lors des tests. Ces défauts, dus principalement à des algorithmes défectueux dans le composant Osmose, ont été corrigés dans un second temps. Le temps de réponse pour l'affichage de la première page (le résultat de la première recherche, celle qui donne le plus de résultats, avant une réduction par la recherche guidée) a été vérifié sur un fichier étalon. Il a pu être ramené à moins de 2 secondes dans tous les cas.

Le serveur dispose de ressources élevées qui ne sont pas très utilisées. Bien que la taille occupée par la base de donnée ait considérablement augmenté (multipliée par trente !), le disque reste inutilisé à 95,5 %. De même la mémoire centrale initialement inoccupée à 90 %, ne sera que peu utilisée par le module (4 % supplémentaires dans le pire des cas). Enfin la charge d'exploitation ne s'accroît pas, toutes les opérations du composant sont automatiquement déclenchées par les actions des utilisateurs, et la sauvegarde de la base de données est inchangée par rapport à l'existant.

TRANSWORK dispose maintenant d'un nouveau module de recherche dans son produit Métaguide, utilisable comme extension de la recherche guidée ou pour explorer commercialement une nouvelle niche : la sous-traitance efficace des systèmes de recherche internes des sites Web.

1.5. Organisation du document

La suite de ce document approfondit les grandes lignes du projet exposé dans cette introduction.

Le chapitre 2 expose le contexte initial du projet et ses contraintes, l'environnement de travail, les motivations du maître d'ouvrage. On y trouvera également une description plus détaillée du produit Métaguide, cadre essentiel de la réalisation.

Le chapitre 3 décrit la réalisation en elle-même, tant du point de vue de son utilisation que de son implémentation.

Le chapitre 4 revient sur l'organisation mise en place pour s'assurer que les objectifs soient atteints.

Enfin le chapitre 5 dresse un bilan du projet ; il examine ses points importants concernant les besoins exprimés tant fonctionnels que qualitatifs, en les plaçant dans la perspective des objectifs initiaux du MOA.

Pour finir, un glossaire définit le vocabulaire propre au projet.

Les renvois à la médiagraphie sont indiqués par un numéro de référence entre crochets, par exemple « [1] » renvoie à la première entrée.

2. La recherche libre, nouveau composant de Métaguide

TRANSWORK développe et commercialise Métaguide™, un outil de définition et d'exploitation de fichiers au travers d'un navigateur Web. L'entreprise a souhaité étendre les possibilités de recherche de ce logiciel, à la fois pour améliorer son offre commerciale et pour préparer de futures évolutions. Nous allons présenter l'adjonction d'un système de recherche par texte libre qui étend et complète le système existant de recherche guidée par thésaurus.

De nombreuses petites structures disposent d'informations qu'elles gagneraient à mettre à la disposition du public par le biais d'Internet, mais elles ne disposent pas de la compétence nécessaire à cette fin. La petite société TRANSWORK a été fondée en 1999 pour offrir à ce marché un outil d'insertion et de recherche de fiches, par exemple des petites annonces ou des produits. Le fondateur de TRANSWORK avait identifié un problème récurrent des systèmes de recherche, qui était (et reste souvent) qu'une requête réalisée par un public non averti est soit trop précise et ne fournit aucun résultat, soit imprécise et en fournit trop... et aucun indice ne permet à l'opérateur d'améliorer sa requête. Pour contourner ce problème, Métaguide guide la recherche en liant les données à un thésaurus arborescent. Cependant ce système ne correspond plus aujourd'hui aux habitudes acquises des usagers du Web.

Se comparant aux moteurs de recherche comme Google™ ou les pages jaunes, l'entreprise souhaite permettre aux usagers de rechercher dans ses fichiers en tapant des mots-clés dans un champ de saisie libre. Cette évolution ouvrirait de plus, la voie vers de nouveaux marchés, comme la mise à disposition du système de recherche à un service informatique extérieur.

Le projet consiste donc à réaliser cette nouvelle fonction, en incluant la définition d'une solution, son implémentation et son installation sur le serveur d'exploitation de l'entreprise.

Une fois réalisé, le composant de recherche libre sera un « frontal de recherche ». Lorsque les usagers des fichiers insèreront des données, celles-ci seront analysées en liaison avec leur thésaurus pour établir des tables d'index propres à une recherche par mots-clés. Les usagers pourront ensuite effectuer des recherches indifféremment guidées par thésaurus ou par texte libre, auquel cas des résultats trop nombreux, seront renvoyés à la recherche guidée. On conservera ainsi les propriétés essentielles de Métaguide.

Ce frontal fournira aussi un point d'accès simple pour l'utilisation automatisée par des services extérieurs qui n'ont pas la connaissance de la structure de thésaurus. Enfin certaines fonctions d'administration et de maintenance seront nécessaires.

Bien que le système doive être pleinement fonctionnel à l'arrivée, au début du projet le maître d'ouvrage n'a pas une vision claire de ce qu'il en attend en matière de performance et de qualité des résultats de recherche. Son idée est de réaliser une maquette grandeur nature, réalisable en 7 mois environ, améliorable avec le temps. Les dirigeants de TRANSWORK, maîtres d'ouvrage, me laissent la responsabilité des choix techniques. Je disposerai donc d'une faculté d'initiative assez large sur l'ensemble du projet, mais des moyens financiers, humains et méthodologiques très réduits, l'entreprise n'ayant ni moyens d'investissement ni organisation de développement logiciel conséquent.

2.1. L'environnement du projet

A partir de 1999, la petite société TRANSWORK a capitalisé sur sa connaissance du marché MINITEL, des serveurs transactionnels et des interfaces utilisateur pour offrir sur Internet des services de création et d'exploitation de fichiers basés sur des thésaurus et non sur des systèmes séquentiel-indexés classiques.

Son logiciel Métaguide™ permet à des utilisateurs privilégiés, les « producteurs », de définir des thésaurus qui à leur tour servent à guider les usagers (utilisateurs finaux) lors des recherches dans les fichiers associés aux thésaurus, ou lors des saisies de fiches.

L'insertion de données peut être gratuite, payante à chaque fiche ou par abonnement. Aussi, suivant le but du fichier, les données peuvent être fournies par des professionnels d'un métier (agences, concessionnaires...) ou par les usagers du Web. La consultation est toujours gratuite.

La concurrence, les partis-pris du grand public et l'optique d'évolution de Métaguide conduisent aujourd'hui à ajouter à la recherche guidée, une fonction de recherche par mots-clés saisis librement.

2.1.1. La société

En France, dans les années 1980-2000, le réseau VIDEOTEX/MINITEL diffusé à faible coût a longtemps été, grâce à un modèle économique rentable, une alternative viable à un Internet alors balbutiant. Un grand nombre de petites sociétés d'informatique se sont lancées sur ce marché. C'est ainsi que M. Edmond CHABOCHE, à l'époque consultant de CAP SOGETI auprès de France Télécom pour l'annuaire électronique, a fondé ENERGIE VIDEOTEX qui a développé, vendu et exploité pour son propre compte des serveurs Videotex sous UNIX. Cette première expérience a permis d'une part de constituer un fonds de logiciels de base (serveur transactionnel MEGABUS, gestion de fichiers multi-index GLADIA, recherche « phonémique ») et d'autre part d'acquérir une grande expérience des applications grand public partagées (messageries, annuaires, communication interactive).

Avec l'arrivée d'Internet cette première activité a peu à peu périclité, et M. CHABOCHE a rebondi en créant TRANSWORK en 1999, dans le but de récupérer avec Internet les marchés qui échappaient alors au Videotex. Une partie des actifs logiciels a été adaptée, et une plateforme Linux/Apache a servi de base à un nouveau produit : Métaguide.

La société a pu placer son produit auprès de divers clients en proposant par exemple à des réseaux de garagistes la « mise en ligne » de fichiers d'annonces pour des automobiles d'occasion, à des intermédiaires en imprimerie des sites d'annonces pour gros matériel, etc. Cependant, insuffisamment pourvue en force commerciale, TRANSWORK doit s'efforcer de maintenir son produit au niveau de la concurrence, dans un marché qui s'accélère dangereusement.

Aujourd'hui l'équipe est constituée de cinq personnes : M. CHABOCHE joue à la fois un rôle commercial et un rôle de moteur dans la spécification de nouvelles fonctionnalités. M. TOLEDANO est Directeur et responsable technique. Un informaticien junior développe pour le compte de la société cliente WhatWhere, et un stagiaire, met en place des démonstrations des extensions développées dans le cadre du présent projet. Enfin, je participe moi-même aux développements, essentiellement dans le cadre du présent projet.

2.1.2. Le marché et les acteurs

Beaucoup d'entreprises veulent apparaître sur le Web sans avoir la compétence nécessaire. Métaguide permet d'attaquer ce marché en s'appuyant sur une structure hiérarchisée d'utilisateurs centrée sur le concept de métier.

L'expérience du Vidéotex a montré que de nombreuses petites structures (sociétés, entreprises, associations...) ont besoin d'utiliser les réseaux publics pour diffuser leurs informations, sans avoir ni les moyens ni les connaissances techniques pour gérer un site Web ou une base de données. Les informations en question sont typiquement de deux espèces : des listes, classiquement organisées en fichiers, et des descriptifs détaillés, classiquement organisés en arborescences de pages liées entre elles. Les fichiers offerts à la consultation publique rentrent en général dans trois catégories : les annuaires, les listes de produits ou de services, les petites annonces. Entre le Vidéotex et le Web, l'ergonomie a fait un bond de géant, mais les besoins sous-jacents et les solutions informatiques ont peu évolué, ce qui a permis à TRANSWORK d'exploiter un important retour d'expérience du marché entre ces deux technologies.

L'approche informatique habituelle consiste à définir précisément une liste d'attributs reflétant une fiche, et de réaliser des pages de saisie, modification et consultation entièrement liées à ces attributs. Bien que des outils généraux existent, telles des interfaces graphiques aux bases SQL, ils sont à la fois trop puissants (comprendre complexes et risqués) pour être mis dans les mains d'utilisateurs non informaticiens, et très restreints quant aux possibilités de présentation et d'ergonomie. De ce fait, il est difficile pour une petite structure de présenter ses données sans une coûteuse assistance spécialisée.

Le produit Métaguide remplit ce rôle en offrant un outil de définition des données générique et éloigné des notions techniques. A l'inverse, les concepteurs de TRANSWORK ont insisté sur la séparation des compétences techniques dans le service offert à ses clients.

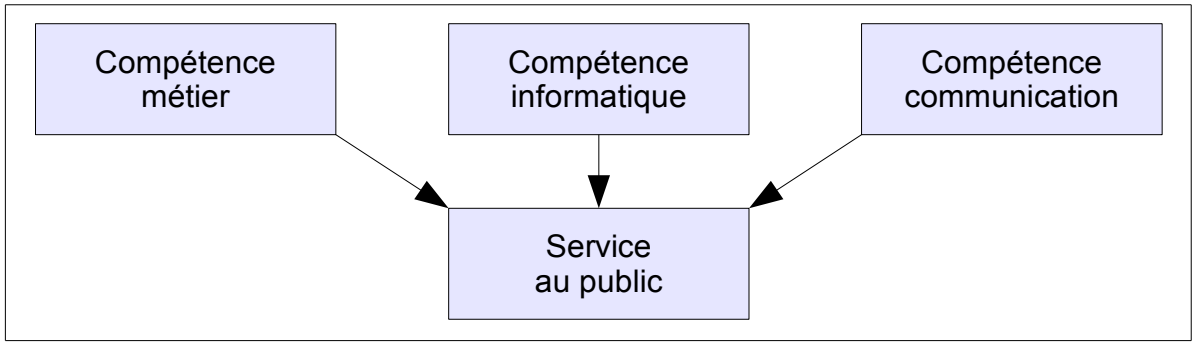


Figure 2.1: Compétences pour offrir un service Web

La compétence « métier » indique comment doivent être structurées les données pour être pertinentes, et Métaguide fournit un outil de définition sous forme de thésaurus. La compétence « éditoriale » définit la présentation, l'aspect des pages, et Métaguide fournit des canevas adaptables de déroulements de saisie/consultation/gestion de fiches. La compétence « Web » définit l'infrastructure système, réseau et outils qui réalisent le service final ; cette compétence est du ressort de TRANSWORK qui la commercialise en louant l'utilisation du service Métaguide à ses clients.

TRANSWORK fournit une assistance dans les différentes compétences. Un documentaliste peut aider le spécialiste « métier » à formaliser ses thésaurus. Les développeurs Web peuvent adapter les canevas de Métaguide mais aussi réaliser des sites simples pour le compte du client et les mettre en place.

La cible client de TRANSWORK est triple et se retrouve dans la structure de Métaguide. Les « producteurs » ont la compétence métier et donc la responsabilité de l'organisation des thésaurus ; il s'agit typiquement d'associations, par exemple une association d'agences immobilières, de clubs de golf... Les « éditeurs » ont la compétence marketing et la responsabilité de valoriser l'information sur leur site Web. Enfin l'exploitant propriétaire des données, désigné dans Métaguide par le terme « client intégré », est l'organisme qui a un intérêt direct à la présentation des informations, par exemple une agence immobilière, un garagiste, un distributeur. Métaguide définit une hiérarchie d'utilisateurs correspondant à ces différents rôles, schématisés dans la Figure 2.2: Hiérarchie des utilisateurs de Métaguide. Le cas échéant, TRANSWORK se substitue aux chaînons manquants de cette organisation.

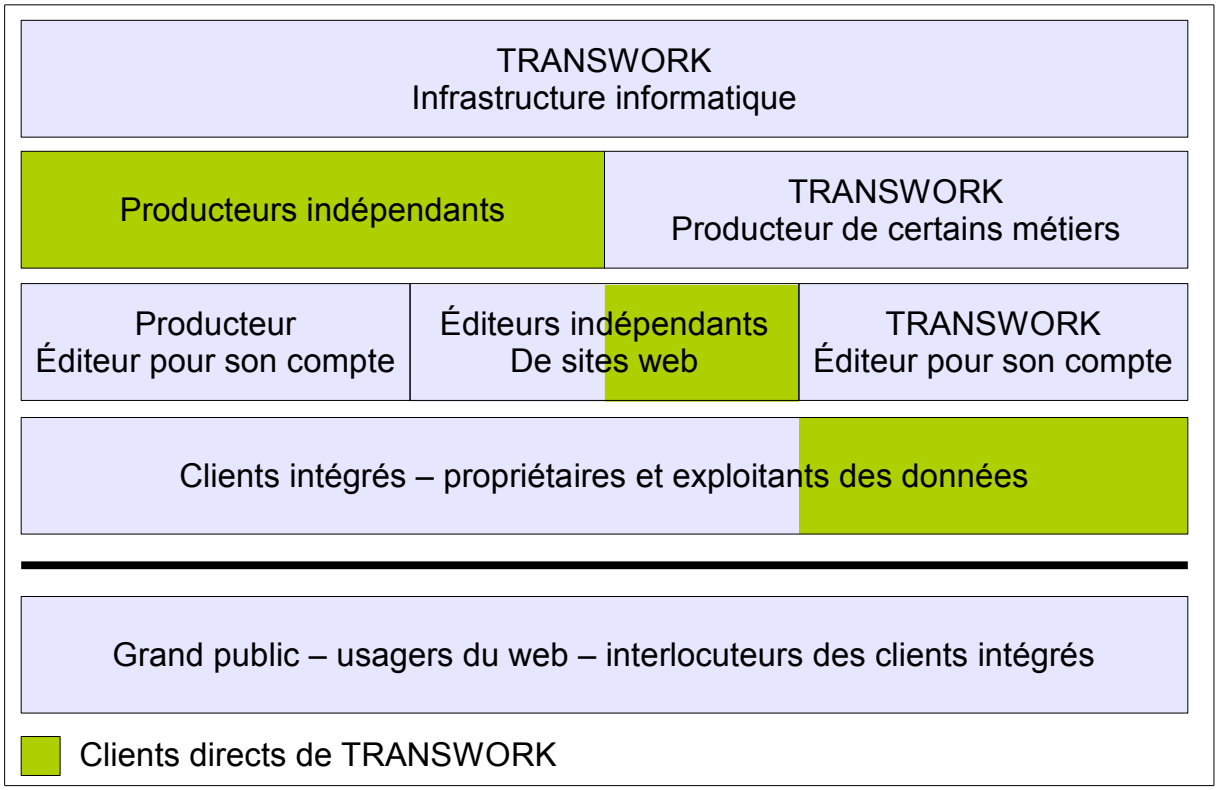


Figure 2.2: Hiérarchie des utilisateurs de Métaguide

Les « clients intégrés » fournissent l'usage de leurs fichiers aux usagers du Web qui utilisent deux fonctions majeures, l'introduction de données dans la base et l'interrogation de la base. L'introduction de données peut être publique ou réservée au client intégré, payante ou gratuite.

2.1.3. Le produit

A partir du concept de thésaurus, la société a développé un système original de définition, de saisie et d'interrogation de fichiers. Le système est organisé autour de « producteurs » spécialistes d'un métier, qui ont la charge de définir les fichiers qui ressortent de leur compétence. Les producteurs mettent à la disposition du public ces fichiers, soit directement soit par abonnement d'intermédiaires professionnels. Dans certains cas l'insertion d'annonces payantes peut être proposée au grand public.

a. L'apport de Métaguide

A la base de la conception de Métaguide, il y a l'observation que les systèmes de recherche multicritère n'assistent pas suffisamment l'usager. Cela était particulièrement flagrant dès la création de l'annuaire téléphonique du MINITEL, et la quasi-totalité des services de recherche dans des bases de données souffrent toujours du problème :

> *Si les critères sont trop précis, ils ne désignent aucun élément dans la base interrogées ; on obtient des messages du style « aucune correspondance trouvée », sans qu'on sache donner un poids aux critères ; et s'ils sont trop vagues ils donnent une liste de réponses sans ordre compréhensible*.

Même si les moteurs de recherche récents classent leurs résultats par pertinence, les critères utilisés sont obscurs et l'usager ne sait pas si ils correspondent à son besoin personnel. Ce n'est que récemment que certains sites comme e-Bay™[3], ayant un intérêt financier majeur dans la qualité de l'outil de sélection d'un produit parmi une multitude, ont développé des systèmes de réductions successives ; encore faut-il remarquer que ces outils sont spécifiques à leur application.

Le but de Métaguide est d'offrir un système paramétrable pour définir des fichiers quelconques, un peu à la manière d'un SGBD, mais évitant intrinsèquement ces écueils par une organisation adéquate des données.

b. La solution thésaurus

Pour réaliser cet objectif de recherche guidée, Métaguide est organisé autour de deux idées ergonomiques principales. La première est que les formulaires de recherche ne permettent de choisir que des critères conduisant à des données utiles. La seconde est que le processus de sélection se déroule par étapes, et se poursuit jusqu'à la sélection d'une liste de taille raisonnable.

Classiquement, un fichier de données (ou table, pour reprendre la terminologie des bases de données), dans lequel s'exerce la sélection, est constitué de fiches (ou tuples), chacune d'elles attribuant des valeurs à des attributs. Mais là où les tables classiques enregistrent séparément pour chaque tuple les valeurs de chaque attribut, qui peuvent prendre des valeurs quelconques sous réserve des contraintes d'intégrité, Métaguide prédéfinit les valeurs possibles de chaque attribut textuel et les liste dans une structure de thésaurus. Ce qui est enregistré pour chaque tuple, c'est non pas la valeur de ses attributs, mais une référence à l'article correspondant du thésaurus.

[3] Http://shop.ebay.fr. Voir annexe I.

Comment les thésaurus résolvent-ils le problème posé ? Lors d'une session de recherche, chaque étape ne permet de choisir qu'un seul critère et aboutit à la sélection d'un sous-ensemble du fichier. Métaguide trie les références de toutes les fiches du sous-ensemble, et ne présente que celles-là comme choix possibles dans le formulaire de sélection de l'étape suivante. Ainsi l'utilisateur ne peut pas indiquer de critères aboutissant à une sélection vide, et il voit en permanence la liste des critères discriminants pour le sous-ensemble courant. Il peut donc choisir en connaissance de cause le critère qui lui paraîtra le plus pertinent, ou « remonter » en annulant la sélection précédente.

Un exemple de session de recherche guidée par thésaurus figure en Annexe B : « La recherche guidée de Métaguide ».

Sur la Figure 2.3: Utilisation des thésaurus, on constate les avantages de cette organisation. En plaçant les règles de validation des attributs dans un fichier, on permet à un utilisateur de redéfinir ses contraintes sans compétences informatiques :

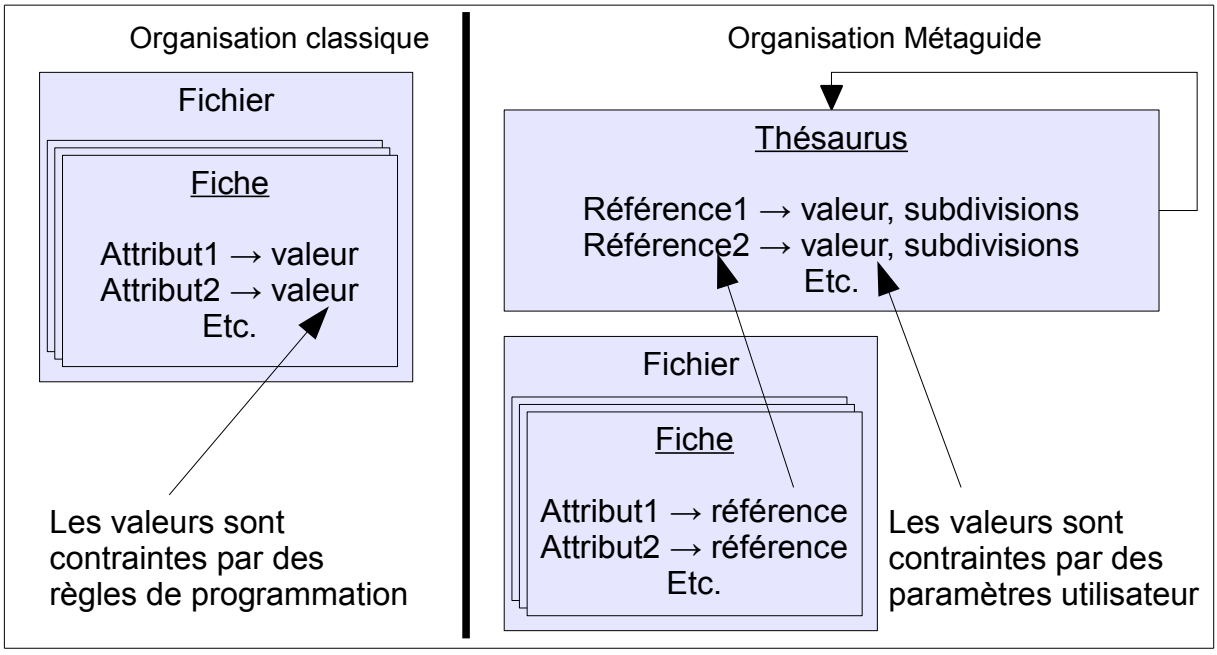


Figure 2.3: Utilisation des thésaurus

De plus Métaguide introduit une hiérarchie des références dans le thésaurus. Ceci permet de structurer l'opération de recherche du général au particulier. Une portion de thésaurus pourrait par exemple indiquer :

Référence valeur	Référence attribut	Attribut dit « contenant »	Valeur dite « contenu »	Subdivision
1	A	pays	france	**B**
2	A	pays	Royaume-uni	C
3	**B**	département	ain	Valeur finale
4	**B**	département	aisne	Valeur finale
5	C	comté	sussex	E
7	C	comté	hampshire	F
Etc.				

Sur cet exemple nous voyons un avantage supplémentaire du thésaurus, à savoir que la liste des attributs n'est pas fixe, elle est déterminée par les attributs plus généraux. Pour des attributs géographiques, on pourra avoir par exemple dans le même fichier un tuple avec des attributs (pays=France, région=Bretagne, département=Morbihan, ville=Vannes) et un autre tuple tel que (pays=Royaume-Uni, comté=Sussex, ville=Hastings) omettant les départements en Angleterre.

La séparation du sens (les références) et des chaînes de caractères qui les représentent permettent de plus de restituer aisément le contenu d'une fiche dans n'importe quelle langue, il suffit d'avoir des versions traduites du même thésaurus.

A l'usage on observe que certaines sections de thésaurus sont semblables d'un fichier à l'autre, par exemple la description hiérarchique de la géographie (pays, régions, départements, villes). Métaguide permet de définir des thésaurus partiels, dits « thésaurus objets », et de les réutiliser dans un thésaurus complet, dit « thésaurus spécifique » parce qu'il est précisément associé à un fichier de données. Dans les thésaurus objets on distingue encore les thésaurus graphiques qui peuvent être visualisés sous forme d'images ou de cartes, au lieu de listes de valeurs.

Malgré ses avantages, cette organisation entraîne un certain nombre de difficultés. Toutes les données ne se prêtent pas commodément à une structuration en thésaurus : les valeurs numériques, les dates et les agendas sont délicats à traiter. Les valeurs non prévues à la construction du thésaurus peuvent être précisées dans une catégorie « divers » de chaque attribut, mais dans ce cas elles ne peuvent pas servir aux recherches ; ce qui rend nécessaire l'amélioration périodique du thésaurus en intégrant les cas manquants. La modification de la structure du thésaurus n'est possible que dans certains cas particuliers, sous peine de perdre des références dans les fiches et donc de l'information. Enfin, bien qu'il soit possible de présenter automatiquement les attributs d'une fiche dans plusieurs langues, les fiches contiennent des textes descriptifs libres, donc hors thésaurus ; ces textes ne peuvent donc pas être traduits automatiquement.

L'Annexe A : L'algorithme de recherche guidée, donne des détails sur la méthode de recherche guidée par thésaurus.

c. Les grandes fonctions

Métaguide est articulé autour de trois grandes fonctions (figure 2.4) composées de nombreuses sous-fonctions. La gestion des thésaurus permet de définir des structures de données, de les modifier et de les mettre en service. Les traitements des fiches permettent la constitution des fiches et la recherche dans un fichier. La gestion des utilisateurs définit les différents types d'utilisateurs et leurs possibilités d'action. Des règles de composition définissent les présentations possibles à destination du public.

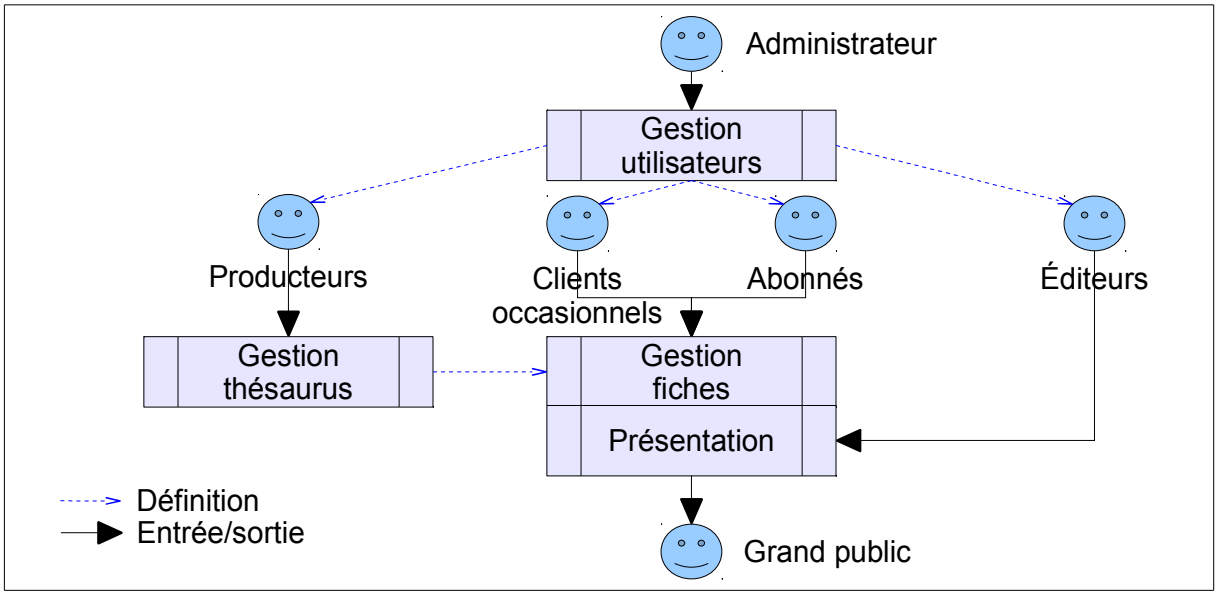


Figure 2.4: Les fonctions de Métaguide et leurs utilisateurs

La gestion des thésaurus est sous la responsabilité du producteur, le spécialiste « métier ». Le producteur nomme un ou plusieurs utilisateurs « documentalistes » qui définissent et composent des thésaurus-objets, en constituant des listes de couples contenant/contenu qu'on peut subdiviser récursivement. Le documentaliste cite ensuite les thésaurus-objets dans un thésaurus-spécifique qui sera associé à un fichier de données. La mise en service est une étape complexe du système (voir Figure 2.5 page suivante).

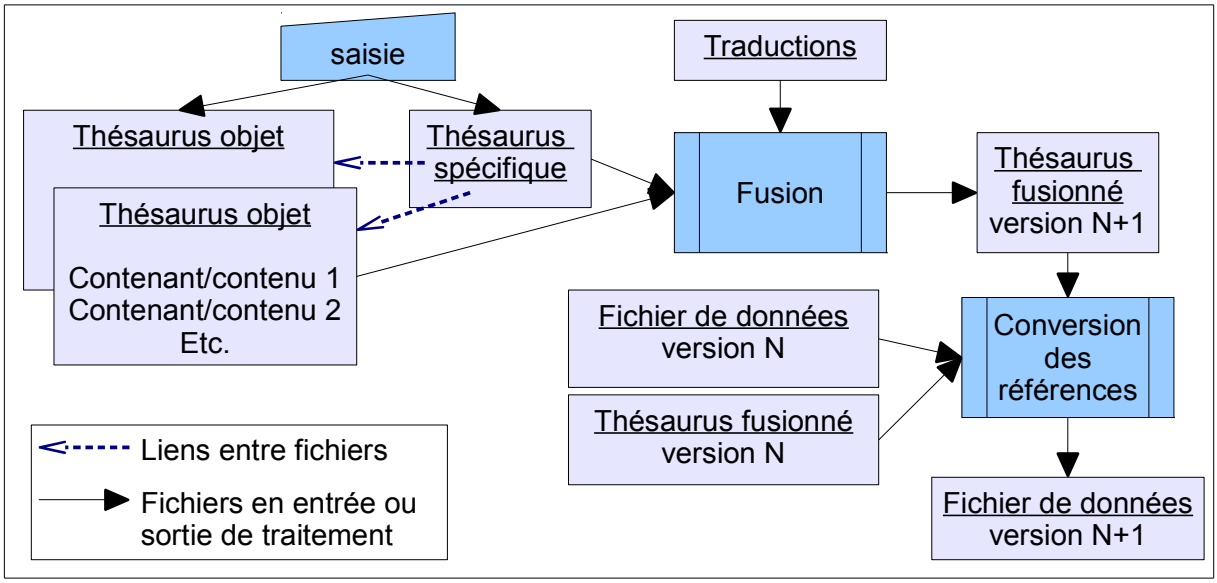


Figure 2.5: Définition et mise en service d'un thésaurus

A cette occasion, les sous-ensembles qui composent un thésaurus-spécifique sont rassemblés en un seul thésaurus dit « fusionné ». L'arbre des couples attribut/valeur a pu changer, et pour gérer les évolutions du thésaurus, deux versions (N et N+1) sont en fait conservées. Les traductions des libellés d'attributs et de valeurs sont aussi vérifiées et intégrées à ce moment. Enfin chaque thésaurus spécifique est assorti d'un tarif de vente du droit de création des fiches et de leurs options.

L'objectif primordial de Métaguide est *in fine* de vendre à l'utilisateur final le droit d'insérer des fiches pour une durée convenue et de les traiter. Le fichier est la propriété du producteur qui définit les conditions de création de fiches (financières et organisationnelles : qui ? Combien ?), mais chaque fiche est la propriété de son créateur qui a le droit de suppression anticipée ou de modification. Pour les clients occasionnels, un module de vente en ligne permet l'achat de fiches et de leurs options (photos, durée de parution). Différents moyens de paiement sont supportés (carte bancaire, chèque avec la gestion des paiements reçus, pay-per-view).Les fonctions de base « CRUD » (création, consultation, mise à jour, suppression) vont de soi, mais l'accent est mis sur la recherche guidée (voir un exemple en La recherche guidée de Métaguide).

Les fiches sont normalement accessibles par des formulaires, ce qui exclut leur repérage par les « bots »[4] des moteurs de recherche généralistes ; aussi, un module optionnel permet de faire apparaître les fiches selon une pseudo-arborescence enrichie de mots-clé prédéfinis pour permettre l'indexation des fiches par ces moteurs.

La présentation des données pour la saisie, la mise à jour, la recherche et l'affichage, est basée sur la définition de modèles de pages (des « templates ») dont la syntaxe est propre à Métaguide. Des règles de construction des URL d'accès permettent d'identifier des sous-ensembles du fichier utilisé, en pré-sélectionnant des valeurs d'attributs par leurs références, mais aussi en identifiant le propriétaire des fiches, ce qui permet de construire, à partir du même fichier, des sites ayant une visibilité plus ou moins large des fiches. L'éditeur

[4] Voir le glossaire.

peut aussi définir des boutons contextuels supplémentaires pour lier les fichiers entre eux (par exemple sur une fiche produit un bouton pourrait mener à une fiche revendeur), ou même définir des abonnés intermédiaires, c'est-à-dire des agents dont les coordonnées apparaissent à la place du vrai propriétaire de la fiche, pour la vente d'objets commissionnée.

Enfin une gestion des utilisateurs assez basique, sous forme de « tableaux de bord », permet à l'administrateur de créer et gérer des producteurs, au producteur de gérer ses éditeurs, à l'éditeur de gérer ses abonnés. On peut remarquer que la gestion des documentalistes est réalisée à part et s'intègre mal dans ce schéma. Les abonnés disposent d'un module « espace abonné » pour exercer leurs droits de création et manipuler leurs fiches.

De nombreuses autres fonctions sont disponibles dans Métaguide : requêtes depuis des serveurs extérieurs, paiement par carte bancaire, journal de ventes, transformation des fiches en arborescence de pages pour les « bots » des moteurs de recherche du Web, exportation, importation, ainsi qu'un fonds de procédures en Javascript qui simplifie la personnalisation des pages présentées aux usagers. Un schéma plus complet figure en Annexe D : Structure interne de Métaguide.

2.1.4. Le projet « recherche libre »

Pour poursuivre et améliorer la commercialisation des services Métaguide, trois phénomènes ont conduit les dirigeants à vouloir y ajouter un mécanisme de recherche par texte libre :

- l'objectif à long terme d'une recherche vocale ;
- la « culture » du Web introduite par les moteurs de recherche majeurs ;
- la conscience que cette fonction a un intérêt en elle-même pour les usagers existants, et ouvre des perspectives pour offrir un nouveau service de « mandataire de recherche » comme expliqué plus bas.

Les dirigeants souhaitent donc mettre en place une solution atteignant ces objectifs.

Le système de recherche de Métaguide peut assez facilement passer sur les nouveaux équipements mobiles, smartphones et tablettes, du fait de sa présentation sous forme de listes courtes tant de critères que de résultats ; en fait on pourrait presque penser que Métaguide a été conçu pour ces appareils d'une surface affichable faible. Pour profiter de ce marché en expansion, la société TRANSWORK a imaginé que les choix de valeurs, au lieu d'apparaître dans des listes déroulantes graphiques, pourraient être sélectionnés vocalement grâce au microphone. Vue de très loin, cette nouvelle capacité devrait utiliser au moins trois nouvelles fonctions : la reconnaissance vocale (extraction des sons), la reconnaissance des mots (homonymies et formes grammaticales), et la recherche de mots quelconques dans les données.

Les dirigeants ont reconnu que cette dernière partie, non seulement ouvrait la route vers les autres fonctions nécessaires, mais aussi apportait plusieurs avantages immédiats. En effet, Métaguide peut être vu comme un SGBD, très simplifié dans le sens où il y manque la fonction essentielle de jointure entre tables. Cependant sa présentation commerciale l'apparente plutôt à un moteur de recherche, et conduit les clients potentiels à des comparaisons inappropriées. L'un des éléments de comparaison est la possibilité de chercher à partir de texte tapé par l'usager et non pas en parcourant un processus par étapes. Le nouveau module de recherche libre répond à cette problématique.

Ce nouveau système de recherche, basé sur la saisie d'un champ textuel unique, ouvre la voie à de nouvelles applications : le champ de saisie pourrait figurer sur une page d'un site tiers et renverrait sur le serveur de TRANSWORK pour y mettre en œuvre une recherche Métaguide – il faut noter qu'il n'entre pas dans la stratégie actuelle de TRANSWORK de commercialiser Métaguide en tant que logiciel sur des serveurs clients ; mais seulement en tant que service. L'intérêt du système est d'enchaîner une recherche libre qui peut donner beaucoup de réponses, avec une recherche guidée si nécessaire.

Il m'est demandé de définir et d'implémenter une solution technique à ce problème dont la définition est au départ, aussi large que vague. Il faudra aussi l'installer sur le serveur public de l'entreprise, en perturbant l'exploitation le moins possible.

2.2. Les fonctions du composant à réaliser

Le composant de recherche libre doit s'insérer dans un produit déjà complet et constitué. Celui-ci est basé sur une infrastructure classique Linux/Apache/base de données PostgreSQL [2], complété par un moniteur transactionnel et un gestionnaire de fichiers « maison ». La recherche libre, qui s'écarte du concept initial du produit, devra s'implanter à tous les niveaux du système existant : interfaces usagers et producteurs, traitement des fiches, traitement des thésaurus, nouveaux points d'accès pour les nouveaux besoins.

L'usager disposera, en première étape avant la recherche existante guidée par thésaurus, d'une présélection par mots-clés. Les index nécessaires seront implantés dans le système et mis à jour de façon appropriée. L'étude déterminera les pré-traitements linguistiques nécessaires sur les mots-clés.

Pour ajouter à Métaguide la capacité de recherche pour le compte de sites clients (« sites appelants »), il sera nécessaire de définir une méthode d'appel du service et une interface de retour des résultats au site client.

Enfin, les fonctions d'administration couvriront la gestion des sites appelants et la surveillance des recherches sans réponses. L'exploitation doit prévoir au minimum une fonction d'indexation des fichiers pré-existants.

2.2.1. L'architecture existante

Métaguide est basé sur une infrastructure classique Linux/Apache/base de données, complétée par une couche de logiciels « maison » : moniteur transactionnel MEGABUS et gestionnaire de fichiers GLADIA.

La partie application proprement dite est principalement constituée de deux modules interactifs, le premier prend en charge toutes les actions des usagers sur les fiches de données et le second permet aux producteurs le développement de thésaurus et leur mise en service. Autour de ces trois grandes fonctions gravitent plusieurs modules complémentaires ou spécifiques à des clients : gestion des utilisateurs, outils de personnalisation, etc. Le schéma suivant montre les relations entre l'application (en jaune) et l'infrastructure (en gris).

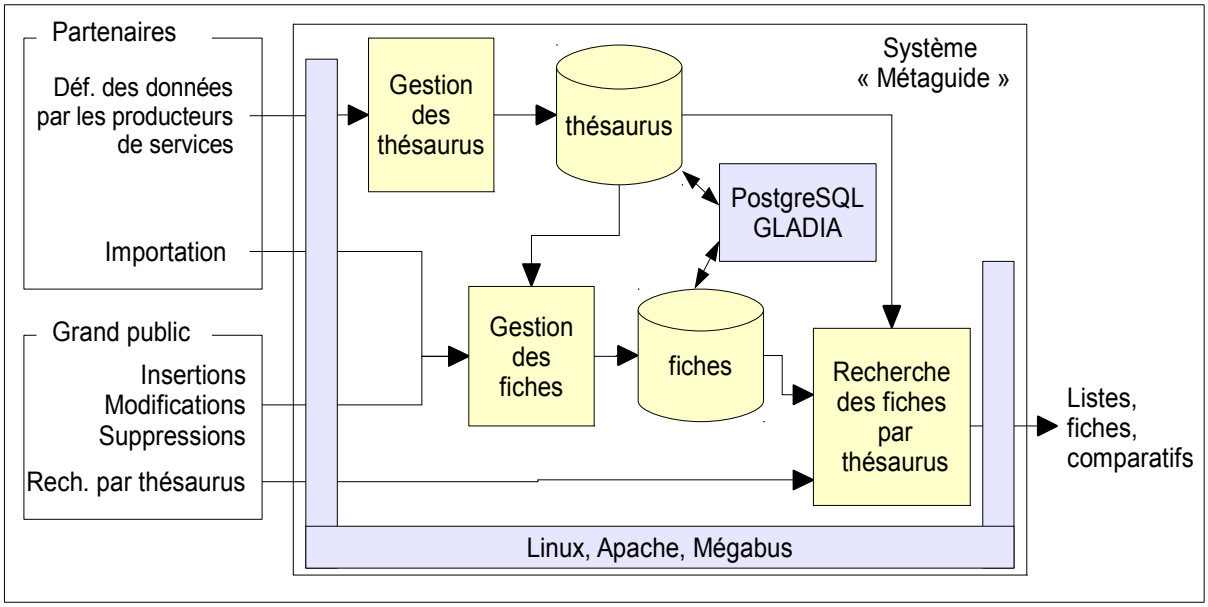


Figure 2.6: Principaux modules de Métaguide

2.2.2. Les nouvelles fonctionnalités destinées à l'usager

Pour effectuer des recherches l'usager dispose actuellement d'un système de sélections successives guidées par thésaurus.

On souhaite lui proposer en plus, une recherche par mots-clés qu'il tapera directement. Les aspects de fautes de frappe, d'orthographe, de grammaire doivent être envisagés. Les mots-clés fournis définiront un sous-ensemble de fiches, sur lequel s'exercera à nouveau la recherche guidée, si le volume de réponses reste important.

Lorsque l'usager saisira, supprimera ou mettra à jour une fiche, les opérations nécessaires à une recherche ultérieure par mots-clés devra se faire de façon transparente, mais pas nécessairement immédiatement.

Les mots-clés ne seront pris en compte que s'ils sélectionnent des fiches : il s'agit de conserver la notion phare de Métaguide, qui est de toujours proposer des réponses pertinentes.

a. Éléments interrogeables par la recherche libre

Le système actuel oblige à passer par plusieurs étapes qui supposent de consulter des listes déroulantes. Le premier besoin que le projet cherche à satisfaire est l'augmentation des possibilités de recherche par une saisie directe de mots-clés. On en profitera pour effectuer la recherche, non seulement sur les valeurs du thésaurus comme c'est actuellement le cas, mais aussi sur les informations complémentaires associées à la fiche : nom et adresse du « contact », description libre. Les fiches peuvent aussi contenir un lien vers une page Web et l'indexation des mots qui s'y trouvent a été envisagé, puis rejeté comme étant trop lourd : Métaguide ne maîtrise pas les pages en question ni les moments de leurs mises à jour ; on serait contraint d'implémenter un moteur de recherche Web complet, ce qui n'est pas l'objectif.

La recherche de plusieurs mots à la fois nécessitera des croisements. Seules les fiches contenant le plus de mots possible seront retenues. Si aucune fiche ne correspond à aucun mot, la totalité du fichier sera sélectionnée et on retombera sur la recherche guidée initiale.

b. Maintien des fonctionnalités existantes

Pour permettre une saisie de texte qui aboutisse à une sélection de fiches sans perte de fonctionnalités, la recherche textuelle sera insérée comme un composant frontal de la recherche guidée. Ainsi, les textes qui sélectionnent peu de fiches aboutiront directement à une liste courte, et les textes qui sélectionnent beaucoup de fiches se comporteront comme une présélection d'un sous-ensemble du fichier, qui sera soumis à une recherche guidée plus rapide ; la recherche guidée sur un sous-ensemble est déjà possible dans le produit. Le passage de la présélection d'un type de recherche à l'autre sera plus ou moins complexe selon que les deux recherches sont peu ou très intégrées. Ces deux approches présentent des avantages et des inconvénients, en premier lieu desquels la performance et la possibilité de déléguer le développement.

Enfin, il sera possible de contourner complètement la recherche libre pour exécuter des recherches uniquement guidées amenées par des liens Web inchangés par rapport à l'existant.

c. Interface utilisateur

La recherche libre sera proposée en tant qu'alternative par un champ de saisie sur la première page de recherche guidée. Il ne sera pas possible de faire une recherche libre sur un sous-ensemble de fiches issu d'une recherche guidée. Les résultats ne seront jamais restitués directement à l'usager mais transmis au module existant de recherche guidée / liste /affichage.

d. Pré-traitement des mots saisis

A la différence de la recherche guidée, l'utilisateur peut taper des mots qui, soit n'existent pas, soit sont mal orthographiés, soit sont conjugués. Le premier cas est traité par les croisements entre mots.

Les deux autres cas ont été longuement pesés par rapport au temps d'étude imparti et aux outils disponibles en open-source. Il a été estimé qu'ils ne sont pas vraiment du ressort de la recherche, mais plutôt d'un pré-traitement indépendant qui aurait avantage à

être réalisé directement dans le navigateur de l'utilisateur, en se basant sur un dictionnaire complet, et en proposant – sans imposer – des corrections en temps réel.

Il faut aussi considérer que la recherche s'effectue par rapport à des mots indexés, qui ont à l'origine été saisis par une personne (le documentaliste en ce qui concerne les thésaurus, l'utilisateur final en ce qui concerne les fiches) qui a elle-même pu faire des erreurs. Après discussion avec le MOA il a été convenu que ces erreurs ne seraient pas traitées, on suppose que la saisie est toujours correcte, l'utilisateur pouvant toujours revenir dessus pour corriger.

La question des synonymes s'est également posée. Les mots tapés par l'utilisateur, même s'ils n'existent pas dans les fiches, peuvent être complétés par des mots synonymes qui augmentent éventuellement les chances de trouver des fiches intéressantes.

Tous ces traitements ont été écartés des spécifications mais il n'est pas exclus d'y revenir dans le futur.

e. Expressions de dépôt

Le MOA demande que, pendant la saisie de mots-clés, certaines combinaisons soient reconnues et proposent à l'utilisateur des liens spécifiques. Par exemple on peut imaginer que le texte tapé puisse être compris non pas comme une recherche mais comme le souhait d'une insertion de fiche, et il est souhaitable pour faire apprécier Métaguide de ne pas décourager l'internaute mais de le guider vers le service approprié.

Cette fonctionnalité peut être implémentée séparément. Elle a été exclue de la délimitation du projet mais fait l'objet d'une spécification séparée.

2.2.3. Les fonctionnalités destinées aux sites intermédiaires

Il a été envisagé de commercialiser Métaguide comme système mandataire de recherche. Un site Web client ayant un fichier propre à une indexation Métaguide, il pourra importer son fichier dans Métaguide, et renverra les recherches libres de ses usagers vers le site de Métaguide. La recherche combinée mots-clés + thésaurus aboutira à la sélection d'une fiche, matérialisée par un lien retournant vers le site client.

L'importation existe déjà mais rien n'est prévu pour le retour vers le site client, et les liens d'appel à Métaguide sont inappropriés. Il faut donc définir et implémenter ces mécanismes.

Prenons par exemple un site Web disposant d'une masse de pages HTML liées entre elles. Pour permettre à l' « internaute[5] » une recherche dans ces pages la solution de facilité souvent retenue est d'utiliser un service Google de recherche limitée au site. La liste qui s'affiche alors est calculée hors du contexte du site, et il est rare qu'elle fournisse des liens significatifs.

Avec Métaguide, moyennant un travail de classement, le site pourra présenter la même zone de recherche libre, mais les résultats seront classés selon le thésaurus par sujets, objets, auteurs, etc ; toutes sortes d'informations utiles pour opérer une sélection complémentaire guidée et obtenir des résultats vraiment utiles. Ce service est facilement extensible à d'autres types de données.

Métaguide répondra à ce type de besoin comme un mandataire de recherches. Dans un premier temps TRANSWORK définira avec le client (le site Web à indexer) un thésaurus

[5] Pourquoi pas après aéronaute, astronaute... Voir glossaire et http://fr.wiktionary.org/wiki/-naute.

adapté à ses données. Puis le client importera dans Métaguide une simple référence à chacune de ses données (un URL par exemple), associée aux éléments de thésaurus et des mots-clés représentatifs, une fonction d'importation existe déjà pour cela. Le site Web étant modifié pour présenter une zone de saisie envoyée au serveur Métaguide, celui-ci exécutera la sélection libre, enchaînée s'il y a lieu à une recherche guidée. Le choix par l'internaute d'une fiche finale renverra au site client en utilisant la référence que celui-ci a fourni pendant l'importation.

2.2.4. Les fonctionnalités destinées aux exploitants

Les fonctionnalités précédentes, décrites du point de vue extérieur, supposent des fonctions de support pour l'initialisation, la surveillance (dans le but d'améliorer les services) et la gestion des sites intermédiaires.

Les nouvelles fonctionnalités nécessitent des tâches annexes qui ne concernent pas les usagers de Métaguide mais l'exploitant du serveur et les producteurs exploitant les thésaurus.

Pour obtenir les temps de réponse attendus (voir section 2.3), les recherches en texte libre vont imposer la création de tables d'index et le maintien de leur cohérence par rapport aux fiches et aux thésaurus. De plus si l'on observe le cycle complet du projet, il y aura une mise en exploitation qui passe par la création initiale de ces index, et éventuellement des conversions de données pour s'adapter aux outils de recherche choisis.

Le MOA demande à pouvoir surveiller les recherches qui n'aboutissent pas de façon à améliorer les services : d'une part en choisissant au mieux le vocabulaire utilisé dans les thésaurus, d'autre part en analysant l'opportunité d'ajouter des modules de synonymie, de correction orthographique... qui seront omis dans un premier temps.

Une table des sites clients, pour lesquels Métaguide est « mandataire de recherche », permettra de filtrer les clients qui sont inscrits à ce service, et de compléter les informations d'entrée (le texte à chercher) par des données générales (le thésaurus, les modèles de pages à utiliser, l'URL de retour au site client). Réduire ainsi les paramètres d'entrée permet de réduire marginalement l'occupation réseau, mais surtout permet d'éviter l'utilisation non autorisée de thésaurus quelconques.

2.3. Les autres objectifs du projet

Le maître d'ouvrage n'exprime au départ aucune exigence non fonctionnelle. En creusant un peu, apparaissent néanmoins des objectifs, quoique peu contraignants : le temps de réponse perceptible pour une recherche libre devrait être peu supérieur au temps de réponse pour la première étape d'une recherche par thésaurus (qui doit être mesuré), et il faut tenir compte d'une volonté à long terme d'éliminer les fichiers maison au profit de l'utilisation généralisée d'une base de données.

Lors de la rédaction du cahier des charges la questions des exigences non fonctionnelles s'est posée. Le MOA dans un premier temps n'a soulevé aucune exigence particulière. Cependant en prenant chaque point un par un : ergonomie et documentation, performances, fiabilité, aptitude à l'exploitation, j'ai pu dégager des réponses sur l'importance relative de ces différents éléments.

Le MOA voit cette étude comme une première étape d'un produit plus évolué, et il la considère comme une maquette utilisable dans un premier temps. Il souhaite avant tout que les performances ne soient pas trop impactées ; il se base sur le délai d'apparition de la première page de recherche, qui est la plus longue (une quantité de classements doit être opérée sur la totalité du fichier). Il estime ce délai à 1,5 secondes sur un fichier existant de 87000 fiches, et ne voudrait pas dépasser 2 secondes dans les mêmes conditions lorsqu'une recherche libre se greffe en amont. Ces estimations concernent un seul utilisateur connecté ; le MOA ne semble pas concerné par la sensibilité au nombre d'utilisateurs simultanés ; il est vrai que l'activité actuelle du serveur est faible.

Quelques mesures montrent que le délai dépend sensiblement de la complexité du thésaurus, du nombre de fiches. Sur le fichier de référence de 87000 fiches le délai se décompose ainsi (en moyenne) :

Comptage des attributs utilisés.....793 ms
Préparation de l'affichage.....76 ms
Allez-retour serveur Web Apache/Moniteur transactionnel/Métaguide.....15 ms
Allez-retour serveur – réseau – navigateur usager (déduit par différence)….496 ms

Total (moyenne de 4 chronométrages manuels).....1380 ms

Les ressources partagées sont le processeur double cœur, la RAM et le réseau de l'hébergeur. La transaction utilise très peu le disque et le réseau (4 ms soit 0,3 % de la transaction). En première approche, le goulet d'étranglement est le processeur, et on peut s'attendre à un temps de réponse qui en dépend proportionnellement lors d'ouvertures simultanées de sessions :

R(n) = (0,44 n + 0,5) s	temps de réponse pour n transactions arrivant en même temps

Le temps de traitement moyen du double cœur étant de (793+76+15)/2 = 442 ms, et le temps passé dans le réseau, 493 ms, est constant si la bande passante est suffisante.

La fiabilité du fonctionnement doit rester pour l'essentiel celle du système actuel : les processus associés à chaque utilisateur sont indépendants, ils sont surveillés, et détruits si nécessaire, par le moniteur Mégabus. Les données sont sauvegardées une fois par jour sur un serveur annexe de l'hébergeur. Il faut prévoir quelques cas marginaux comme l'absence de fiches ou l'absence de résultats de recherche ; mais le rôle du Mégabus est de surveiller les exceptions et on restera dans ce contexte.

Je me suis fixé des objectifs secondaires pour faciliter la maintenance et l'évolution à long terme : dans le domaine du design de la solution, quand un choix est possible, choisir systématiquement l'utilisation de la base de données par rapport aux fichiers maison, et d'une manière générale favoriser les logiciels récents et/ou à l'état de l'art ; dans le domaine de la réalisation éviter les surprises pour mes successeurs, éviter la multiplication des techniques. Métaguide utilise déjà quatre langages de programmation (C, SQL, PHP, Shell) munis chacun de nombreuses API, et des logiciels ayant des mécanismes de configuration très variés (Apache, PHP, cron...) ; il ne s'agirait pas de gaspiller les ressources humaines limitées en dispersant les compétences nécessaires au support de ce logiciel.

2.4. Les contraintes d'intégration dans l'existant

La réalisation doit s'insérer dans un système existant composé, en première approche, de quatre couches : une infrastructure extérieure formée par l'ordinateur, son système d'exploitation et ses services standardisés ; une infrastructure intérieure formée par des outils maison génériques ; une couche de bibliothèques propres à l'application, et enfin l'application elle-même.

Le nouveau composant doit bien sûr s'appuyer sur les outils existants, et les compléter s'il y a lieu.

Métaguide comprend de nombreux sous-ensembles qui, du point de vue du processus de développement peuvent se répartir en trois catégories :

– l'application proprement dite ;

– les fonctions de support développées dans l'entreprise ;

– les fonctions de support extérieures : Linux, PostgreSQL, Apache, langages PHP et shell, et une panoplie d'utilitaires indispensables quoique incomplètement répertoriés.

Ces différents sous-ensembles sont fortement liés et doivent être examinés dans leur ensemble pour apprécier l'ampleur des modifications à leur apporter. Nous allons nous appuyer sur un schéma général des modules en cause pour les décrire plus en détail.

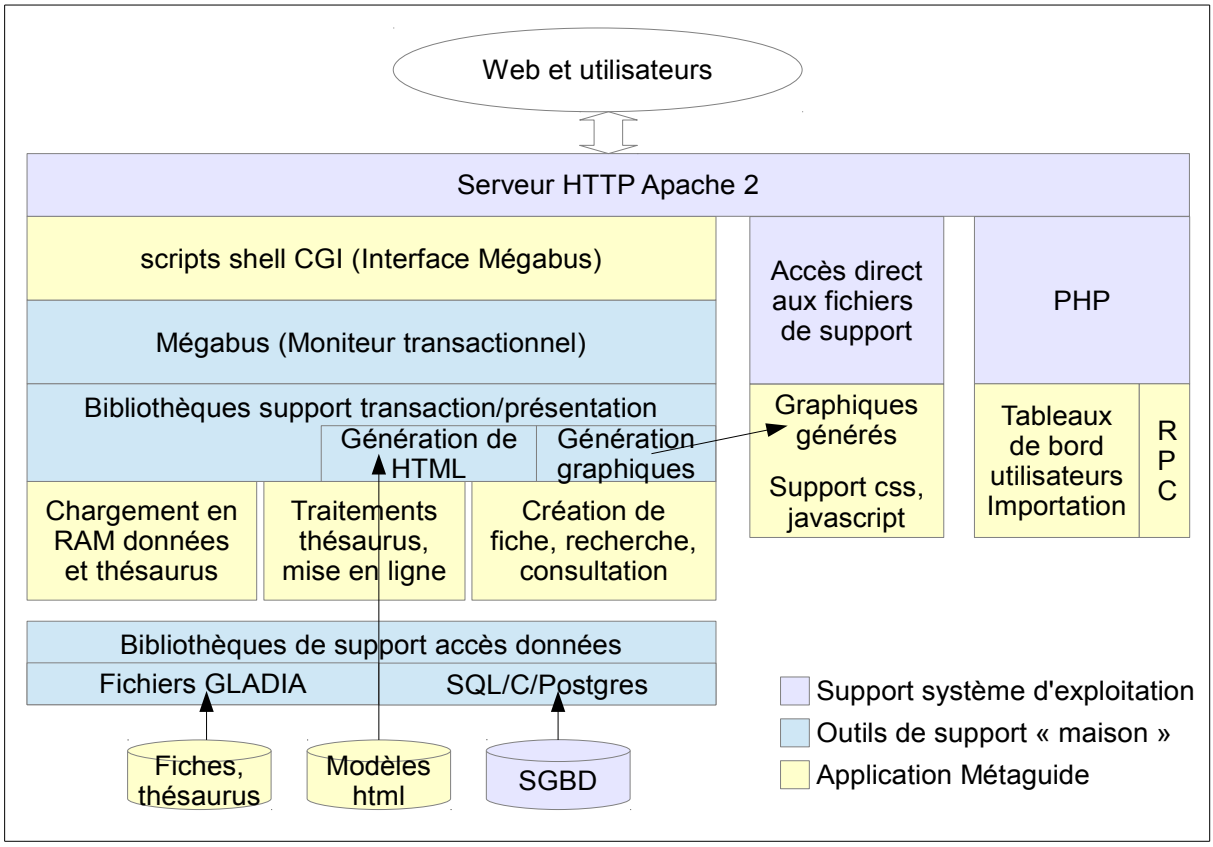


Figure 2.7: Architecture générale de Métaguide

2.4.1. L'infrastructure système

L'environnement système actuel est une version Debian de Linux sur une architecture PC. L'ordinateur est loué à un hébergeur (OVH) qui fournit aussi une solution de sauvegarde. La distribution Debian a été choisie pour sa stabilité (tant en qualité qu'en faible fréquence de mise à jour), sa pérennité et son orientation serveur. Les autres deux principaux composants système sur lesquels s'appuie Métaguide sont le serveur Web Apache et le SGBD PostgreSQL.

Le lien entre Apache et l'application est triple. Les échanges concernant la gestion des utilisateurs sont traités par l'interpréteur PHP intégré dans Apache. Des scripts shell font communiquer Apache et le Mégabus, un moniteur transactionnel maison, qui héberge les programmes compilés traitant les thésaurus et les fiches. Enfin ces traitements « actifs » génèrent des pages qui font référence à des données statiques : feuilles de style, logos, images... qui sont directement extraits du disque dur par Apache.

Issus de deux technologies différentes, le moniteur maison Mégabus et les scripts PHP communiquent difficilement, en particulier pour ce qui est d'utiliser les deux dans la même session utilisateur.

2.4.2. L'environnement transactionnel « maison »

Les usagers des services Métaguide utilisent un navigateur pour communiquer avec le serveur HTTP Apache. Les traitements sur les thésaurus et les fiches de données sont convertis en sessions du moniteur transactionnel Mégabus, conservé pour des raisons historiques. Le Mégabus gère les sessions de travail, distribue les requêtes entrantes vers les applications concernées, détruit les sessions abandonnées, surveille les terminaisons anormales des programmes et prend en charge les mises à jour à chaud des applications.

Le Mégabus associe chaque session avec une instance d'un des deux programmes applicatifs, le traitement des thésaurus ou le traitement des fiches. L'instance de programme conserve dès lors en mémoire le contexte de la session de l'usager : quel thésaurus il utilise, où il en est dans ses étapes de recherche ou de modifications, etc.

Conçu au départ pour l'environnement Videotex, le Mégabus a été adapté pour dialoguer avec un serveur Web HTTP grâce à l'interface CGI. A chaque requête du navigateur, le serveur Web exécute un script « shell » qui formate la requête et la transmet au Mégabus. La requête est routée vers l'application concernée qui la reçoit dans une bibliothèque de communication et la transmet aux fonctions de traitement proprement dites après mise en forme.

2.4.3. Les bibliothèques de support de l'application

Les applications écrites en langage C utilisent plusieurs bibliothèques de fonctions, en particulier pour la communication avec l'utilisateur et part pour l'accès aux données permanentes.

Les données à afficher sont incrustées dans des modèles de pages indépendants. Trois bibliothèques y concourent : l'une gère des listes de tuples en mémoire, la deuxième analyse des fichiers texte – en l'occurrence du HTML – pour y remplacer certaines balises par des champs extraites de ces listes, la troisième utilise ces listes pour colorier des images en fonction de « densités ».

Depuis sa conception initiale, Métaguide utilise un gestionnaire de fichiers ISAM maison nommé « GLADIA » pour y enregistrer les thésaurus et les fiches associées. Cependant l'ajout permanent de fonctionnalités a été supporté en utilisant un SGBDR qui allège considérablement la maintenance et l'évolution, mais les fichiers d'origine sont toujours gérés par GLADIA, ce qui au fil du temps a amené une certaine duplication des données entre les deux gestionnaires. Un schéma de la base et de ses relations avec GLADIA est fourni en Annexe C : Le modèle Entités-relations de Métaguide. Métaguide gagnerait certainement beaucoup à migrer complètement les fichiers GLADIA vers la base de données, il faut favoriser tout choix qui va dans cette direction.

2.4.4. Le module de gestion des fiches

La gestion des fiches est la cible primaire du projet. Ce programme est l'interface principale entre l'usager et le fichier de données. Il inclut à la fois la recherche guidée qu'il faudra étendre, l'affichage de fiches, leur exportation, leur saisie et leur modification.

L'utilisateur final s'intéresse à un fichier de données particulier piloté par un thésaurus particulier. Il dispose de deux points d'entrée : la page de recherche et la page de saisie (des liens sur ces deux pages permettent éventuellement de naviguer de l'une à l'autre). Le thésaurus utilisé est prédéterminé dans le lien qui a amené à ces pages ; par exemple le site d'un fabricant offrira des liens publics vers la consultation du thésaurus de ses produits, vers la recherche dans le thésaurus « annuaire des revendeurs », et des liens d'accès limité vers les pages d'insertion/modification dans ces deux thésaurus.

La page de recherche propose des étapes successives de sélection dans les fiches. Ces étapes guidées par le thésaurus ne présentent que les choix plausibles. La recherche aboutit éventuellement à une liste puis à une « fiche finale ». Des fonctions de comparaison entre fiche sont proposées, et la modification est possible pour un utilisateur identifié comme propriétaire de la fiche finale.

2.5. Les contraintes de gestion de projet

Le Maître d'Ouvrage n'impose aucune méthodologie. Après une étude sommaire, partant de l'hypothèse que tout sera développé « en interne », il a évalué la charge de travail totale à environ 31 semaines/homme (7 à 8 mois). Le composant devra si possible être réalisé à coût zéro. Les objectifs généraux de l'entreprise excluent le recours à des outils soumis à redevance ; il est par contre possible d'utiliser tout ou partie de logiciels libres.

L'entreprise maîtrise la totalité du code source de l'application, et l'infrastructure est constituée de logiciels libres et gratuits. Dans l'esprit du MOA il s'agit de conditions de base à l'existence du produit et de la société, puisqu'elles permettent de maîtriser le coût de revient (pas de redevances à payer pour l'utilisation), l'évolution technologique (le portage des logiciels est réalisable d'une version à l'autre de Linux, et envisageable sur d'autres systèmes d'exploitation), l'offre commerciale (la stratégie de location de services peut sans contraintes devenir une stratégie de vente de produit). Cependant Métaguide est composé de modules réalisés à différentes époques, par différentes équipes, et aucun fil directeur méthodologique n'a été conservé, ce qui se ressent au niveau de la documentation du produit et du peu de cohérence des moyens de configuration, d'initialisation, de compilation, des différents modules.

Pour avoir une première idée de la durée du projet le MOA a imaginé les grandes lignes d'une solution possible (qui n'a pas été retenue au final) et m'a interviewé pour définir et chiffrer les différentes étapes de « sa » solution.

Avant d'établir un planning prévisionnel une solution hypothétique a été définie : la recherche s'appuierait sur des listes inverses de fiches, de contenants/contenus, de pages web associées aux fiches, de coordonnées des propriétaires, etc., en fonction des mots qui s'y trouvent, et de tables de paramètres définissant par exemple le poids de chaque origine du mot reconnu. Les mots saisis par l'usager seraient complétés ou remplacés par des synonymes le cas échéant. Le schéma suivant donne une idée de l'organisation générale de la solution.

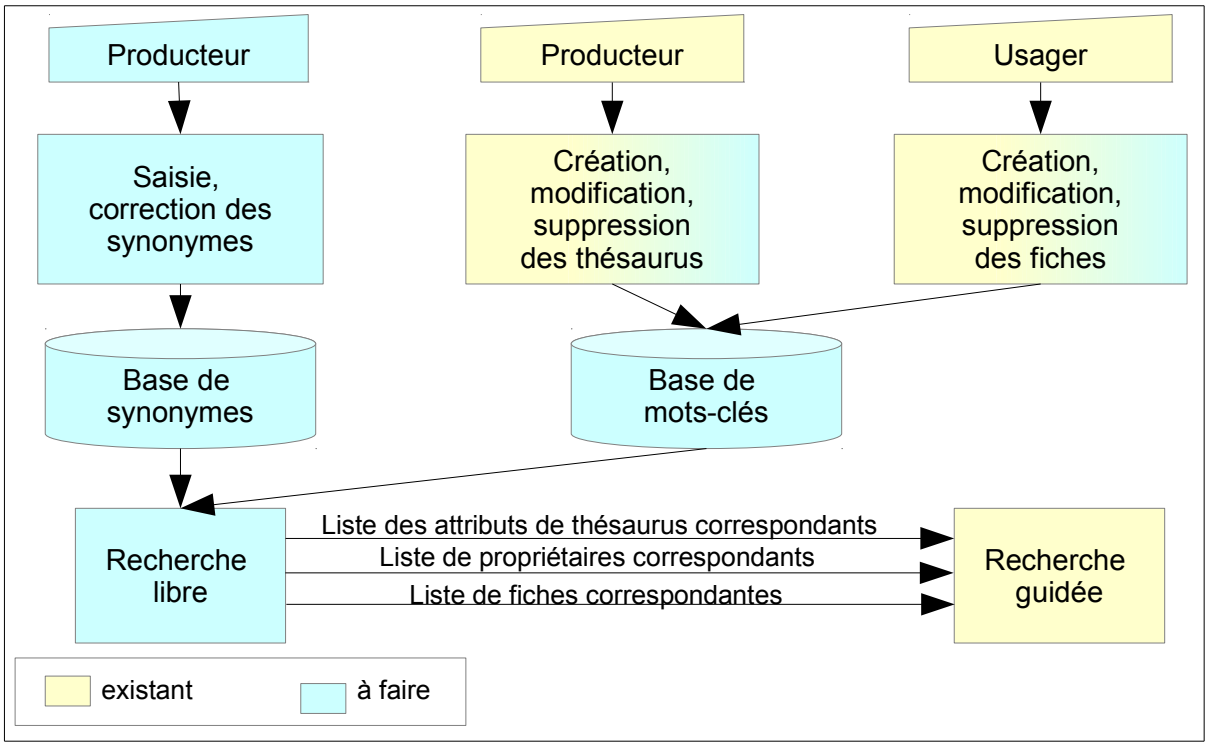


Figure 2.8: Esquisse d'une solution pour l'estimation des délais

A partir de là un planning a été établi sur trois bases :

- Déroulement du projet tel qu'imaginé
- Une seule personne à la réalisation ; peut-être deux
- Maîtrise du code source, même si son origine est extérieure à l'entreprise, et récupération éventuelle de modules existants inexploités (recherche phonétique)

Un découpage grossier a pu être fait sur cette base, comme suit :

Tâche	**durée (semaines/homme)**
Conception générale du composant	2
Conception de la correction orthographique	2
Conception détaillée des fichiers et paramètres à traiter	3
Codage des tables de priorités avec une seule règle	5
Adaptations dans la création/modification des fiches	2
Adaptations dans la suppression des fiches	1
Adaptations dans les mises à jour de thésaurus	5
Création et codage d'un « thésaurus des utilisateurs »	4
Première initialisation du thésaurus des utilisateurs en exploitation	1
Intégration finale des modifications dans le Métaguide	2
Transfert des résultats de recherche au module recherche guidée	1
Navigation (sélection du thésaurus depuis les mots tapés)	1
Mise en exploitation	1 à 2
Total	**30 à 31 semaines**

L'entreprise a très peu de moyens d'investissement, et toute solution gratuite sera préférable. Enfin, l'encadrement technique peu contraignant me laisse libre des méthodes de travail, ce qui me laisse la responsabilité d'en choisir une et de m'y tenir.

2.6. Résumé des objectifs

Métaguide est un système de gestion de fichiers qui sort de l'ordinaire. Son organisation interne permet de procéder à des recherches par étapes sur des sous-ensembles de fiches de plus en plus fins, ce qui permet de guider l'utilisateur sans le « perdre en route », car contrairement à ce qui se passe avec les formulaires de recherche multicritère, il ne risque pas de donner une combinaison de critères trop vaste ou trop restrictive. Ce système est loué aux clients de TRANSWORK sous forme d'un abonnement qui autorise la définition de fichiers et leur mise à disposition pour diverses catégories d'utilisateurs du Web. Pour atteindre ces objectifs, chaque fichier de données est associé à une complexe structure de thésaurus arborescent qui peut être définie par le producteur, la personne qui possède la compétence métier pour l'application finale cible.

Cependant, l'évolution des technologie et la comparaison, quoique inappropriée, avec les moteurs de recherche du web, pousse TRANSWORK à compléter son offre par un mécanisme de recherche libre.

L'usager pourra alors taper ce qu'il veut dans une zone de texte. La saisie devra ensuite être recherchée dans tous les éléments qui définissent les fiches : leurs informations internes, mais aussi les éléments liés et accessibles indirectement : les attributs qu'on peut retrouver dans le thésaurus associé, les coordonnées du propriétaire de la fiche, et peut-être dans le futur les pages Web mentionnées dans la fiche. Les mots reconnus permettront de présélectionner des fiches concernées, et de terminer par une recherche guidée sur le sous-ensemble restant. Enfin, un nouveau type d'usagers doit être supporté, provenant d'un autre site qui paie TRANSWORK pour se servir de la recherche guidée. Ces usagers ouvriront une session de recherche en fournissant directement les mots à chercher, et leurs droits d'utilisation devront être vérifiés en fonction du site d'où ils viennent.

Le projet couvre donc la définition précise, la conception, la réalisation et la mise en exploitation de cette nouvelle fonction. Une première ébauche de solution a servi de guide pour estimer la durée du projet à environ sept mois, avec un coût extrêmement réduit. J'aurai une grande liberté de choix d'organisation, mais sauf exception je travaillerai seul ce qui représente une vaste et lourde responsabilité.

A l'issue de la mise en service du nouveau composant de recherche, l'entreprise aura non seulement une offre technique améliorée mais aussi un discours commercial plus aisé dans les comparaisons avec la concurrence. La possibilité de rechercher à partir d'une seule zone de formulaire permet d'imaginer de nouvelles niches d'utilisations. Enfin l'espoir à plus long terme du maître d'ouvrage est de remplacer la saisie de texte par l'interface vocale sur les produits du type « smartphone ».

Nous allons maintenant préciser comment ces services ont été obtenus au terme du projet, qui a nécessité la modification de nombreuses parties de Métaguide.

3. La réalisation du composant recherche libre

Grâce à la recherche libre, l'usager peut maintenant taper ses propres mots-clés. Ils sont recherchés à la fois dans les valeurs des thésaurus, dans les fiches enregistrées et dans les coordonnées de contact de leurs propriétaires. Les résultats les plus pertinents, c'est-à-dire contenant le plus de mots communs avec la requête, sont isolés et transférés au système de recherche guidée existant. L'essence de Métaguide est préservée : si les résultats sont trop nombreux l'usager dispose de filtres supplémentaires ; s'ils sont peu nombreux ils sont directement listés ; et s'il n'y a aucun résultat, l'ensemble des fiches est re-soumis à la recherche guidée.

Deux types d'utilisateurs sont supportés. Les usagers « historiques » voient apparaître un nouveau champ de recherche sur la page initiale de recherche ; pour eux il s'agit d'une facilité supplémentaire offerte. La saisie de fiches quant à elle, crée de façon transparente les index textuels nécessaires aux futures recherches.

D'autre part, un site Web client peut utiliser Métaguide pour offrir des recherches guidées dans ses bases en y important les critères de recherche de ses données, et exposer à ses visiteurs un champ de recherche textuelle transmis alors à Métaguide, qui se charge des transactions suivantes jusqu'à la sélection d'une fiche. L'identifiant de celle-ci, renvoyée au site Web client, lui permet d'enchaîner des traitements spécifiques. Un tel site client est nommé « site appelant ».

Enfin, deux fonctions d'administration permettent de définir les sites appelants, leurs droits et caractéristiques, et de surveiller les chaînes de recherche qui n'aboutissent pas, pour affiner la compréhension des attentes des usagers.

Au début du projet, une étude des sources montre que Métaguide s'appuie sur un cocktail de fichiers GLADIA et de tables du SGBD PostgreSQL, et que les accès aux différentes données sont dispersés et dupliqués dans des modules variés.

La recherche de solutions montre que PostgreSQL, mis à jour et reconfiguré, simplifiera l'implémentation de la recherche libre. La ligne directrice est de favoriser l'utilisation de la base au détriment des fichiers GLADIA dont on peut prévoir la disparition future. Après avoir reconstitué un diagramme entités-relations croisé de la base et des fichiers GLADIA pertinents pour le projet, on précise les champs, tables, index et contraintes à ajouter à ce schéma. Cette solution met en œuvre à la fois la base et GLADIA, mais procéder à une conversion immédiate et complète des fichiers GLADIA sort complètement du projet, outre qu'elle demanderait un travail conséquent et minutieux.

Le nettoyage des sources en langage C permet de centraliser dans une bibliothèque ad hoc et dans des procédures stockées, les traitements SQL sur les données qui devront être indexées pour la recherche libre.

Le composant lui-même est articulé autour de trois fonctions majeures : l'indexation, la recherche, l'administration. L'indexation est impactée par la manipulation des fiches de données elles-mêmes, mais aussi lorsque le producteur modifie son thésaurus : il agit sur l'ensemble des données qu'il faut alors réindexer. Des scripts « shelll » servent d'interface entre GLADIA et SQL pour réindexer les thésaurus lors de leur mise en service. Le schéma suivant montre les différents développements qui permettent de supporter la recherche libre.

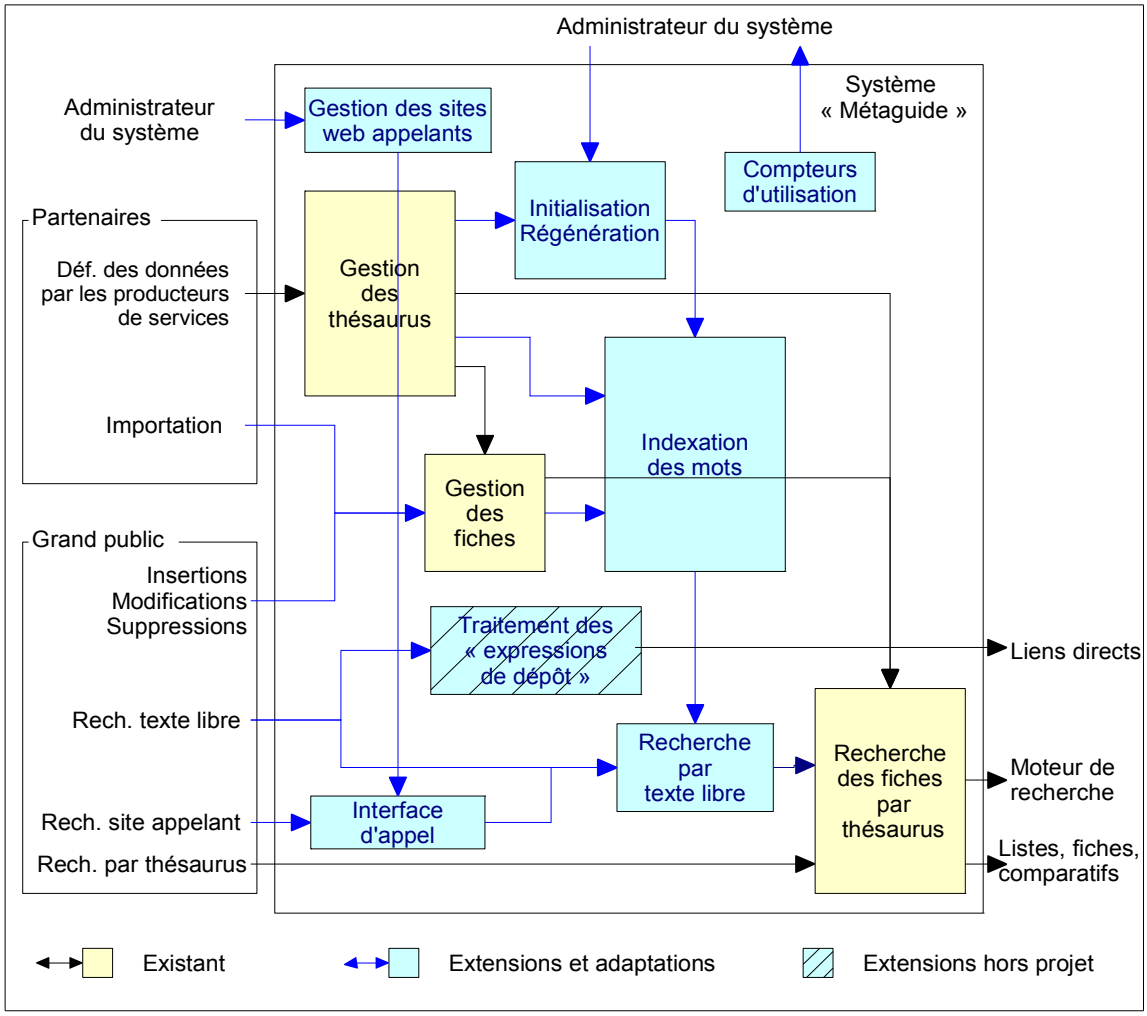


Figure 3.1: Extensions et adaptations réalisées

Une mise en place incrémentale des fonctions et fichiers, soigneusement préparée par des check-lists et des simulations, permet de réduire le risque d'erreurs et la perturbation de l'exploitation. La base de données PostgreSQL est reconfigurée pour gérer les caractères nationaux et leurs modalités d'utilisation (distinction des lettres, ordre de tri). D'autres outils maison sont réutilisés dans la recherche libre à diverses fins : les fichiers GLADIA qui doivent être maintenus en cohérence avec la base de données ; la bibliothèque Osmose qui gère des tables dynamiques en mémoire vive, en particulier pour les résultats de recherches SQL.

Enfin, la documentation a été mise à jour, et quelques améliorations hors projet sont venues s'insérer.

3.1. Les services rendus par le composant

La recherche libre, par mots-clés, opère une présélection des fiches. La liste obtenue est ensuite passée au système existant de recherche par thésaurus. Les mots-clés à rechercher peuvent être acquis via un mécanisme extérieur et injectés lors de l'appel à Métaguide, ou bien Métaguide peut proposer leur saisie interactive comme première étape de la recherche guidée par thésaurus.

Pour conserver l'esprit de Métaguide, les recherches strictes (incluant tous les mots) qui ne donnent pas de résultat, sont élargies progressivement. Ainsi l'étape suivante de recherche guidée opère toujours sur un ensemble de fiches non vide.

L'indexation des fiches par mots-clés est intégrée d'une façon transparente pour les différents acteurs, que ce soit pendant la définition des thésaurus, la saisie interactive de fiches ou l'importation massive de fichiers. L'indexation a lieu lors des créations et modifications, et peut être utilisée immédiatement dans les recherches.

3.1.1. L'opération de recherche

Bien qu'il y ait deux types d'utilisateurs très différents, le mécanisme de recherche est commun : on n'a pas envisagé de comportements paramétrables selon la clientèle concernée. Les mots tapés par l'usager arrivent à Métaguide sous forme d'une requête de formulaire. Cette requête peut inclure une demande d'ouverture de session (cas courant attendu pour les « sites appelants »), ou elle peut faire partie d'une session déjà ouverte (cas courant attendu pour les clients historiques).

Dans les deux cas, les mots sont directement injectés dans une requête SQL de la base de données. Celle-ci se charge de toutes les opérations nécessaires :

- Le texte est nettoyé (élimination de la ponctuation et des mots dits faibles tels les articles, pronoms et prépositions).
- Les mots sont réduits à leur radical pour pouvoir les reconnaître quelle que soit leur inflexion (conjugaison, marque de pluriel, etc.).
- Chaque mot est utilisé pour une recherche dans un index inverse et les listes résultantes sont croisées.

La requête SQL demande un tri des fiches trouvées par nombre de mots différents reconnus, et seul le sous-ensemble contenant le maximum de mots est utilisé, ainsi l'usager ne voit pas les réponses les moins significatives. Si aucun mot n'est trouvé, toutes les fiches sont présentées, ce qui permet à l'usager d'utiliser la recherche guidée. Autrement dit, si l'usager fournit par exemple deux mots significatifs :

- si aucun mot n'est trouvé, alors : recherche guidée sur toutes les fiches ;
- si des fiches contiennent les deux mots et d'autres fiches en contiennent un seul, alors : recherche guidée sur les fiches qui en contiennent deux ;
- si des fiches contiennent un seul des deux mots, mais aucune ne contient les deux à la fois, alors : recherche guidée sur les fiches qui contiennent un mot, quel qu'il soit.

La séparation entre les références d'attributs et leurs valeurs textuelles permet à Métaguide d'afficher l'essentiel des fiches en plusieurs langues, il suffit que le thésaurus,

seul, soit traduit. Mais dans une fiche, les champs descriptifs, examinés dans la recherche libre, ne sont pas traduits. Pour éviter de mélanger les textes de plusieurs langues, ce qui se traduirait par des résultats de recherche surprenants voire incompréhensibles, le service ne sera disponible qu'en français dans un premier temps. Mais les tables de mots sont d'ores et déjà indexées par la langue, pour rendre la recherche possible en plusieurs langues dès qu'une décision sera prise sur le traitement convenable des textes mono-langue.

3.1.2. Le service pour les anciens usagers

Les usagers « historiques » passent en général par un site client de TRANSWORK qui contient des liens vers la recherche guidée hébergée sur le serveur Métaguide. Souvent, pour donner l'illusion de ne pas changer de site, la page Métaguide est incrustée dans le site d'origine par des balises HTML spécifiques[6]. La recherche guidée impose de choisir dans des listes, qui peuvent se présenter sous forme graphique.

La recherche libre se présente comme un simple champ de saisie non contrôlé qui peut être ajouté sur la première page de recherche, à l'initiative du compositeur de la page :

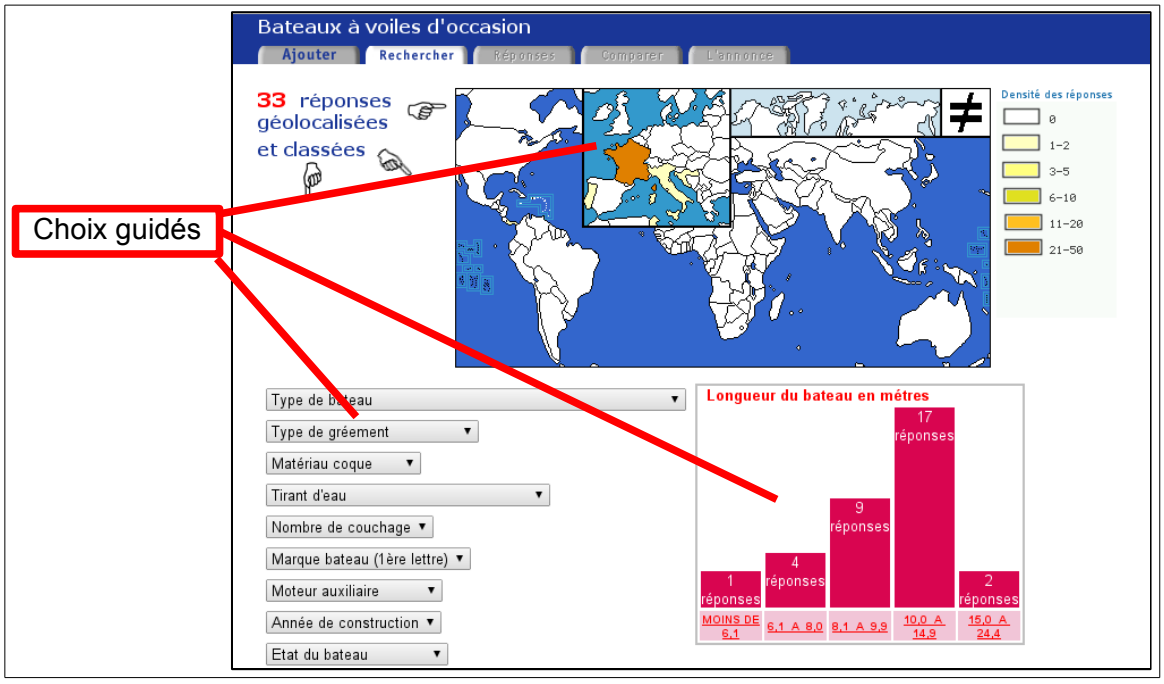


Figure 3.2: Recherche guidée seule (http://www.marinas-yachting.com/voile.html)

[6] Balise <IFRAME> ou anciennement <FRAMESET>...<FRAME>

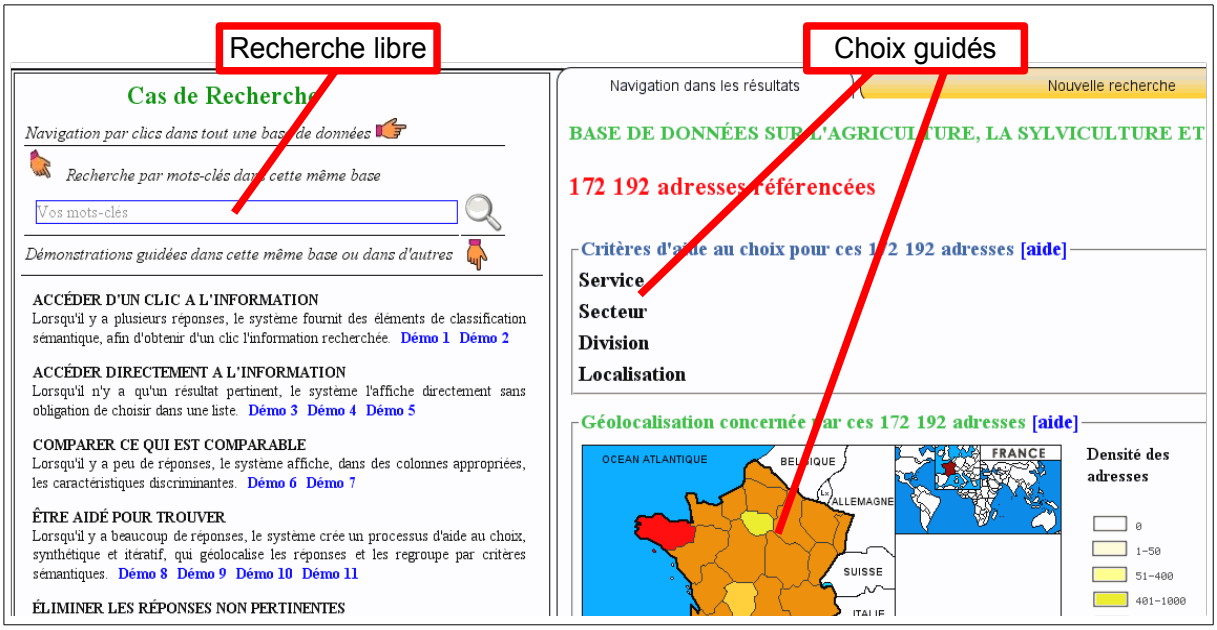


Figure 3.3: Recherche combinée libre + guidée (http://www.whatwhere.com/tablette.php)

L'usager peut soit sélectionner un choix dans une des listes proposées, soit taper des mots à rechercher. Dans les deux cas la page suivante est semblable, mais réduite aux choix encore possibles. A partir de là le champ de recherche libre n'est plus proposé.

Les pages de saisie et de modification de fiches sont inchangées, car la mise à jour des index de recherche libre est automatique (et obligatoire). A la saisie, l'indexation est immédiate et on peut immédiatement trouver la fiche en cherchant les nouveaux mots.

3.1.3. Le service pour l'exploitant extérieur et ses usagers

Il s'agit ici de proposer au client la sous-traitance des recherches dans ses fichiers publics. Typiquement il pourrait s'agir de catalogues de produits avec un thésaurus décrivant les familles, les tailles, les couleurs... ou de créer une recherche thématique sur les pages du site client. Ce service, rendu possible par le nouveau composant, n'est pas encore commercialisé lors de la rédaction de ce mémoire.

Pour mieux visualiser l'intérêt de ce système, prenons l'exemple du site commercial BBBB.com (le site existe, le nom a été changé). Sur sa page d'accueil il y a une zone de recherche. Si on recherche les mots « accessoire noir » **on n'obtient rien, et on ne sait pas pourquoi** (alors qu'il y a effectivement des réponses possibles). Si on recherche simplement « accessoire » **on obtient une liste très longue** par ordre alphabétique (qui n'est pas un ordre très intéressant dans ce cas) :

Figure 3.4: Recherche d'origine avec liste longue sur BBBB.fr

Chaque résultat de recherche est une courte description et un lien vers une page. Maintenant imaginons que le site demande à TRANSWORK de traiter ce genre de recherches. Dans un premier temps un thésaurus serait défini d'un commun accord entre TRANSWORK et BBBB.fr pour classer tous les liens possibles d'une façon plus efficace. Chaque lien serait associé à un produit, une gamme, une couleur, une utilisation...

Le service informatique du BBBB.fr fournirait un fichier des liens avec les éléments (produit concerné, utilisation, genre d'information, couleur...) permettant de les classer par rapport au thésaurus. Chaque lien serait une « fiche » au sens de Métaguide. La « zone de recherche » du site BBBB.fr enverrait la requête vers Métaguide qui afficherait non pas 143 réponses mais des listes de choix permettant d'affiner la recherche et de lui donner un sens :

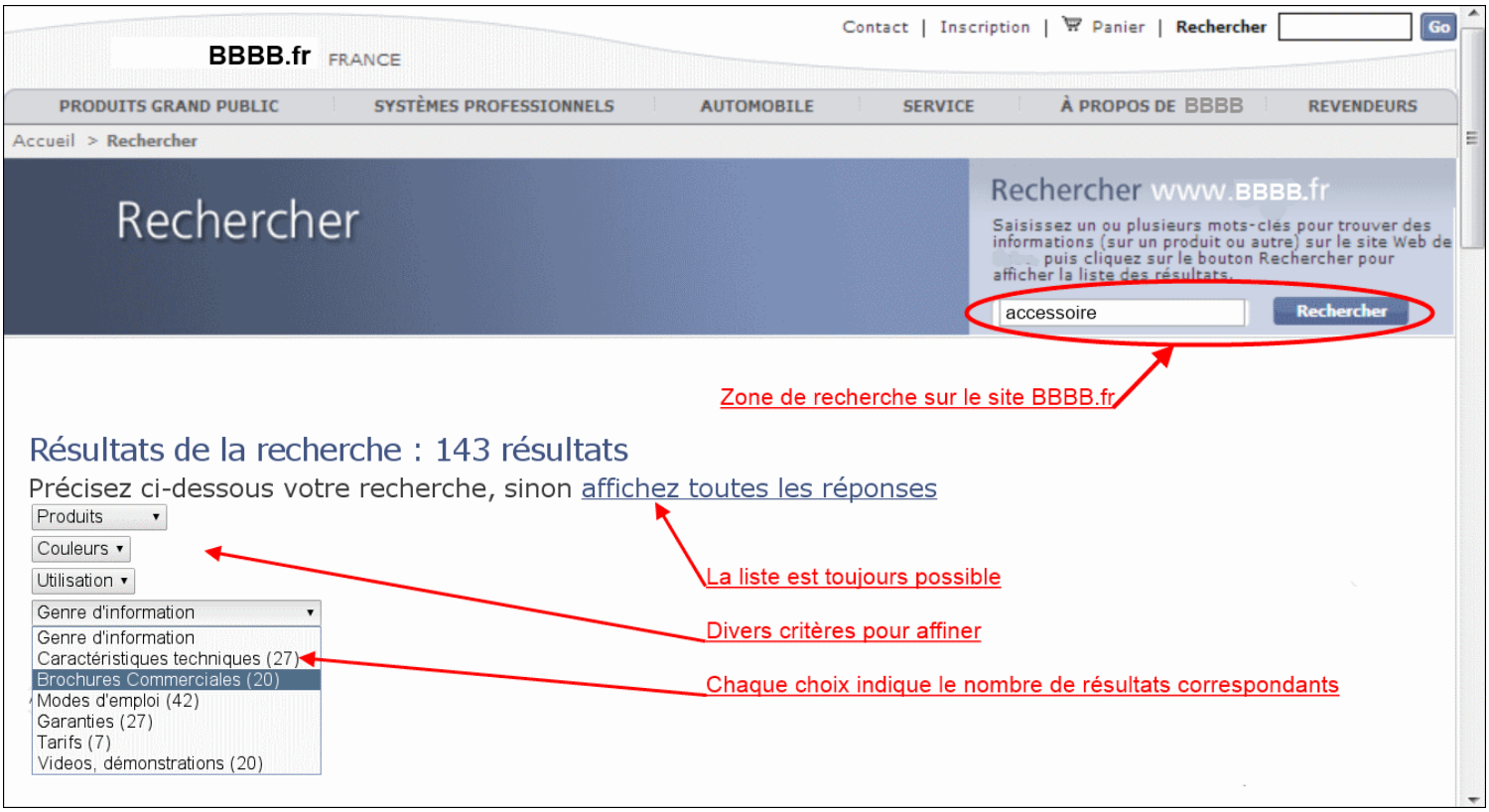


Figure 3.5: Recherche avec liste longue sur BBBB.fr

L'usager disposerait ainsi d'informations utiles pour trier ses 143 réponses. La liste des attributs lui donnerait aussi un point de référence pour appréhender l'organisation des données qu'il examine.

Dans le cas la recherche de « accessoire noir » il obtiendrait uniquement les fiches qui contiennent les deux mots. Si aucune ne les contient, il obtiendrait toutes les fiches contenant l'un ou l'autre de ces mots. Si la liste obtenue est trop longue (la longueur maximum est un paramètre du thésaurus), les critères apparaissent à la place pour affiner le choix.

Ce service est rendu dans trois échelles de durée : l'échelle du contrat, la durée de vie des données indexées, la transaction de recherche. Le composant intervient dans ces trois dimensions du temps.

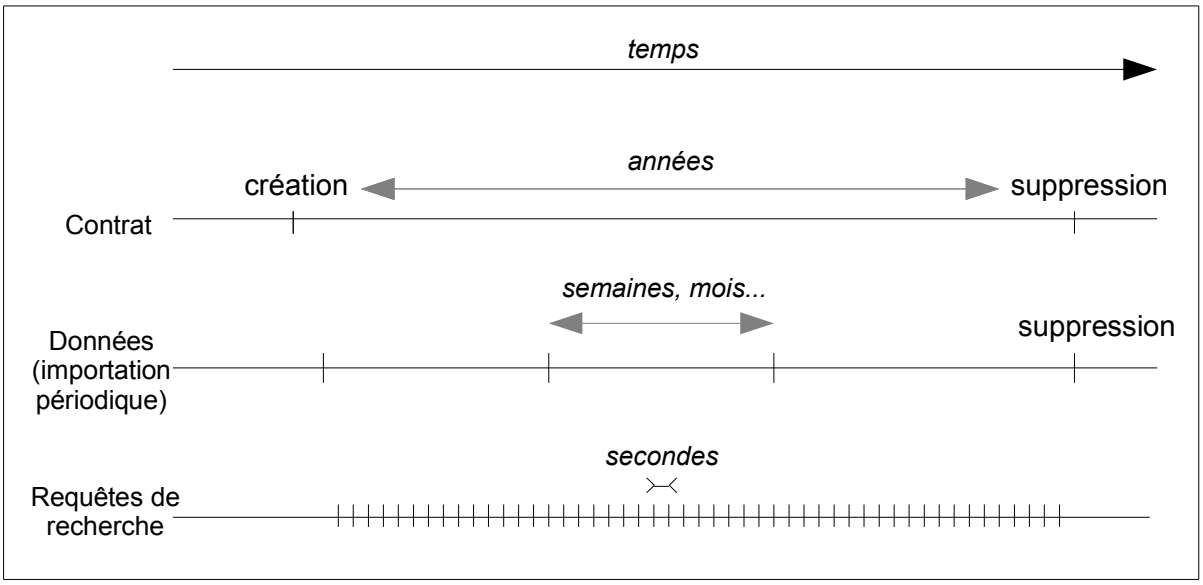


Figure 3.6: Échelles de temps des traitements des sites appelants

A l'échelle du contrat, lors de sa conclusion, un thésaurus décrivant les données du client doit être créé avec les outils existants. De plus, des données permanentes sont introduites dans la base de données, pour lier, lors d'une requête entrante, le contractant, le fichier à utiliser, les modèles de pages à présenter à l'usager, et le mécanisme de retour vers le site client. Aucune interface de saisie n'a été développée pour cette étape. L'administrateur, suffisamment compétent, utilise un outil graphique générique de saisie dans la base de donnée[7]. En fin de contrat, l'administrateur utilise le même outil pour enlever les données du contrat, et supprime le thésaurus concerné par les moyens existants.

Le client peut ensuite importer ses données dans Métaguide et modifier ses pages pour inclure le formulaire de recherche. Celui-ci doit contenir des champs cachés pour identifier le site client (voir Annexe E :Le formulaire HTML de recherche libre).

Le client doit préciser le jeu de caractères qu'il utilise sur son site, pour que Métaguide interprète correctement les mots à rechercher. Les codages supportés sont UTF-8, code universel, et ISO8859-15, qui représente les caractères des langues occidentales.

Les données qui servent à la recherche sont importées périodiquement, à chaque fois que le client le juge nécessaire, ou selon les modalités du contrat. Ces données :

- ne contiennent que les champs utiles aux recherches,
- contiennent une chaîne dite « opaque » qui est renvoyée telle quelle au site client quand l'usager a choisi une fiche à l'issue de sa recherche ; le mot « opaque » rappelle que ce champ n'entre pas dans des calculs des traitements Métaguide.

[7] Il s'agit de phppgadmin, disponible et documenté sur le site http://phppgadmin.sourceforge.net.

Le client ne révèle donc à TRANSWORK que les données utiles aux recherches. Il utilise une fonction d'importation existante, qui permet de mettre à jour un fichier par Internet.

A l'échelle des requêtes, l'usager visite le site client (Figure 3.7). Il souhaite effectuer une recherche sur le fichier cible du client, pour cela il tape des mots dans le formulaire simplifié qui lui est proposé : un champ textuel plus éventuellement un bouton de soumission. Le formulaire est expédié non pas au site client mais directement au serveur Métaguide. Là, le code client permet de retrouver le fichier à traiter, les mots-clés du formulaire sont recherchés dans les champs soumis à recherche libre, et le résultat de recherche est envoyé au navigateur de l'usager.

Ce résultat peut être une page intermédiaire de recherche guidée ou une liste ; l'usager navigue jusqu'à sélectionner une fiche.

Le lien de cette ultime sélection renvoie à un URL composé d'une partie fixe convenue avec le site client, et d'une partie variable qui est le champ « opaque » associé à la fiche et transmis lors de l'importation.

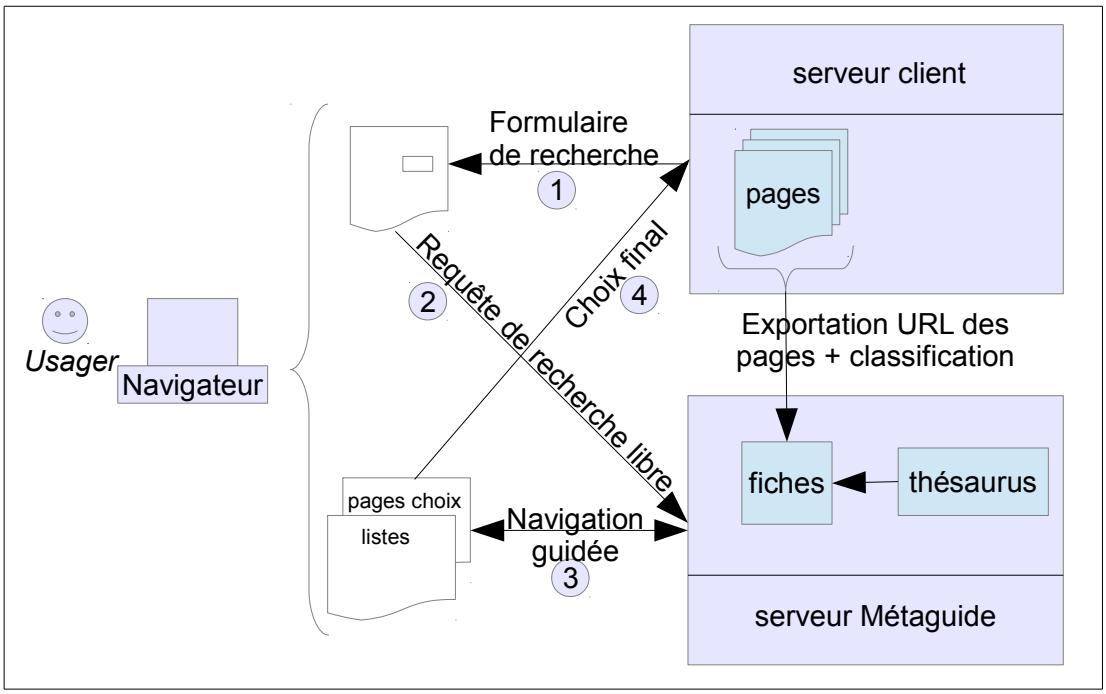


Figure 3.7: Séquence de recherche depuis un "site appelant"

Les systèmes de recherche de pages qu'on trouve couramment sur de nombreux sites Web utilisent des compilateurs de mots-clés et créent des index liant les mots à la liste des pages qui les contiennent. Le « ranking », c'est-à-dire l'évaluation de la qualité des réponses, se fait généralement sur deux critères : le nombre de mots de la requête trouvés dans la réponse, et la proximité de ces mots dans la réponse. Bien souvent le site se contente même de lier son formulaire de recherche aux services de Google™. Autant dire que dans la liste, potentiellement longue, de résultats, aucune classification basée sur le sens ne permet à l'usager de faire un choix éclairé.

En enchaînant la recherche libre à une recherche guidée par un thésaurus spécifiquement conçu, Métaguide offre une bien meilleure ergonomie à l'usager. De plus ce système n'est pas limité aux pages du site client mais peut être appliqué, séparément ou ensemble (il suffit de complexifier le thésaurus), à des fichiers de données.

3.1.4. Le service pour le producteur et l'éditeur

Lorsque aucun mot tapé n'est reconnu, le fichier n'est pas filtré et toutes les fiches sont soumises à la recherche guidée. Cette situation peut se produire si les mots introduits dans le thésaurus par le producteur ne sont pas ceux que l'usager tend à taper naturellement. Il est important de détecter et de corriger ces défauts.

Quand cela arrive, la requête de l'usager est enregistrée dans un journal accessible au producteur, avec quelques informations sur l'origine de la demande. Seul l'historique des trois derniers mois est conservé ; les requêtes plus anciennes sont automatiquement supprimées pour réduire au maximum les responsabilités du producteur.

Celui-ci peut utiliser ce journal pour identifier les mots manquants dans le thésaurus, et le compléter. Il peut aussi choisir de ne rien faire, ou de ne pas consulter le journal, ce n'est pas une obligation. L'interface de consultation est « phppgadmin », l'outil d'accès à la base de données déjà mentionné [1].

On peut imaginer que l'exploitation du journal débouche à l'avenir sur une extension du projet, pour mieux supporter les synonymes des mots.

3.2. L'organisation interne du composant

L'application existante n'utilisait pas le SGBD de façon systématique. Les fichiers GLADIA eux-même étaient modifiés par du code dupliqué en différents endroits. Un objectif annexe était de ne plus créer de nouveaux fichiers GLADIA dans une perspective d'évolutivité.

Le schéma de la base a été complété (annexe C), l'existant a été muni de bibliothèques d'accès pour s'y interfacer aisément. Les données permanentes gérées par le composant, en provenance des fichiers GLADIA, sont converties et transférées dans le SGBD, tandis que les données des utilisateurs sont mises à jour simultanément dans le SGBD et dans GLADIA. Pour cela, lors des modifications de fiches, une nouvelle bibliothèque de fonctions masque les détails de l'enregistrement à l'application.

La plus grosse partie des traitements nouveaux a pour objectif de constituer et de maintenir la cohérence de tables d'index de mots-clés compatibles avec la recherche « full text » de la base de données. La recherche elle-même n'est ensuite qu'une question d'interface avec le requérant, la base de données et la recherche guidée.

L'approche, consistant à n'utiliser que la base de données et des interfaces existantes ou nouvelles, résulte en une organisation des données cohérente, un code plus facile à maintenir et un système globalement plus évolutif.

3.2.1. Les données gérées

D'un thésaurus à l'autre, la quantité de fiches associées varie beaucoup. Certains fichiers comme les annonces de location de bateaux, contiennent une trentaine de fiches. D'autres comme l'annuaire de l'agriculture, en ont plusieurs dizaines ou centaines de milliers. Pour obtenir des durées de recherche raisonnables, il faut préparer des tables d'index qui listent directement les fiches contenant un mot donné. Ces mots doivent être recherchés dans les descriptifs de fiches, les valeurs associées dans le thésaurus et les coordonnées du propriétaire. Pour chaque thésaurus, trois fichiers GLADIA contiennent ces informations :

- les fiches proprement dites,
- le fichier des coordonnées d'annonceur,
- le thésaurus fusionné, qui liste les couples attribut/valeur.

Seul le premier fichier est partiellement dupliqué dans le SGBD. Historiquement, cette duplication a permis d'ajouter des fonctionnalités sans remettre en cause l'existant. Mais les relations entre les tables et les fichiers GLADIA sont complexes. La table SQL des fiches proprement dites ne contient pas les champs descriptifs, qui sont justement une partie des mots à rechercher. De plus il y a une seule table pour toutes les fiches de tous les thésaurus, alors qu'il y a autant de fichiers GLADIA que de thésaurus.

Les informations à indexer sont :

- les **libellés des contenus** associés à la fiche, dont les textes sont dans le thésaurus
- les **coordonnées** du propriétaire, qui sont dans le fichier des annonceurs
- le **descriptif** de la fiche, qui est dans la fiche
- les ~~libellés de contenants~~ ne sont pas indexés, car figurant dans la plupart des fiches
- les ~~pages Web associées~~ aux coordonnées ne sont pas indexées, car il n'y a pas de mécanisme pour découvrir quand elles sont modifiées.

Enfin il faut supporter les fonctions annexes de définition des sites appelants et de journalisation des défauts de recherche.

En fin de compte nous voulons obtenir des listes de fiches à partir de mots, en français dans un premier temps, en plusieurs langues dans le futur. La première étape est donc de créer une représentation des fiches dépendante de la langue. Le diagramme suivant montre la table ajoutée « fiche_lang » en relation N-1 avec la table « fiche » : il y a autant de sous-fiches que de langues supportées. Ensuite nous devons intégrer les textes des contenants/contenus. Leurs références sont dans un tableau fixe dans les fiches GLADIA, et ne sont pas représentées dans la table du SGBD. S'agissant d'une relation 1-N on crée une table « fiche_cnacnu » indexée par le code fiche et la référence contenant/contenu.

Dans la base, la clé primaire des fiches était une concaténation complexe du nom de thésaurus, sa version, et la clé unique de fiche utilisée dans GLADIA. Pour simplifier les liaisons entre les tables, on crée une nouvelle clé unique, qui sert d'alternative à la clé primaire.

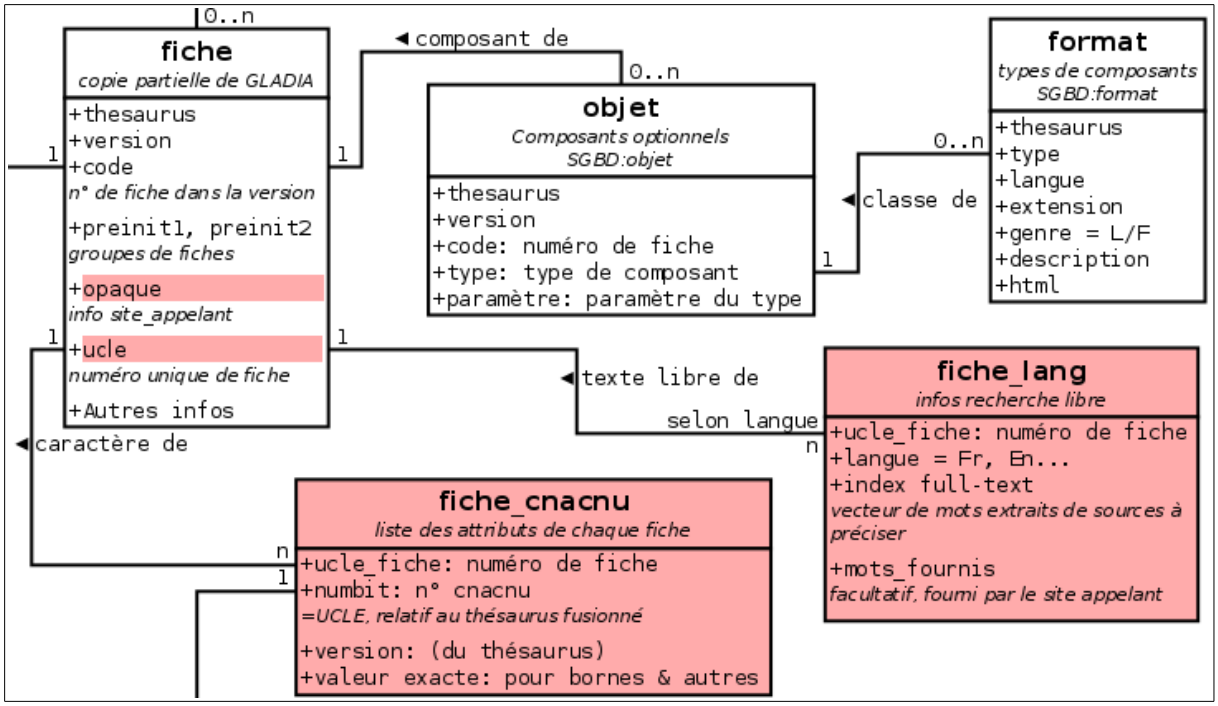


Figure 3.8: Extensions de la table "fiche"

Un élément nommé « opaque » sert à identifier la fiche sélectionnée lors du retour vers un site appelant.

Pour accéder aux libellés contenants/contenus eux-même il est inutile de les lier aux fiches, il y aurait duplication car les mêmes libellés se retrouvent dans de nombreuses fiches. A la place on crée une table « cnacnu_fus » des contenants/contenus du thésaurus et une table « cnacnu_fus_lang » de leurs libellés selon la langue.

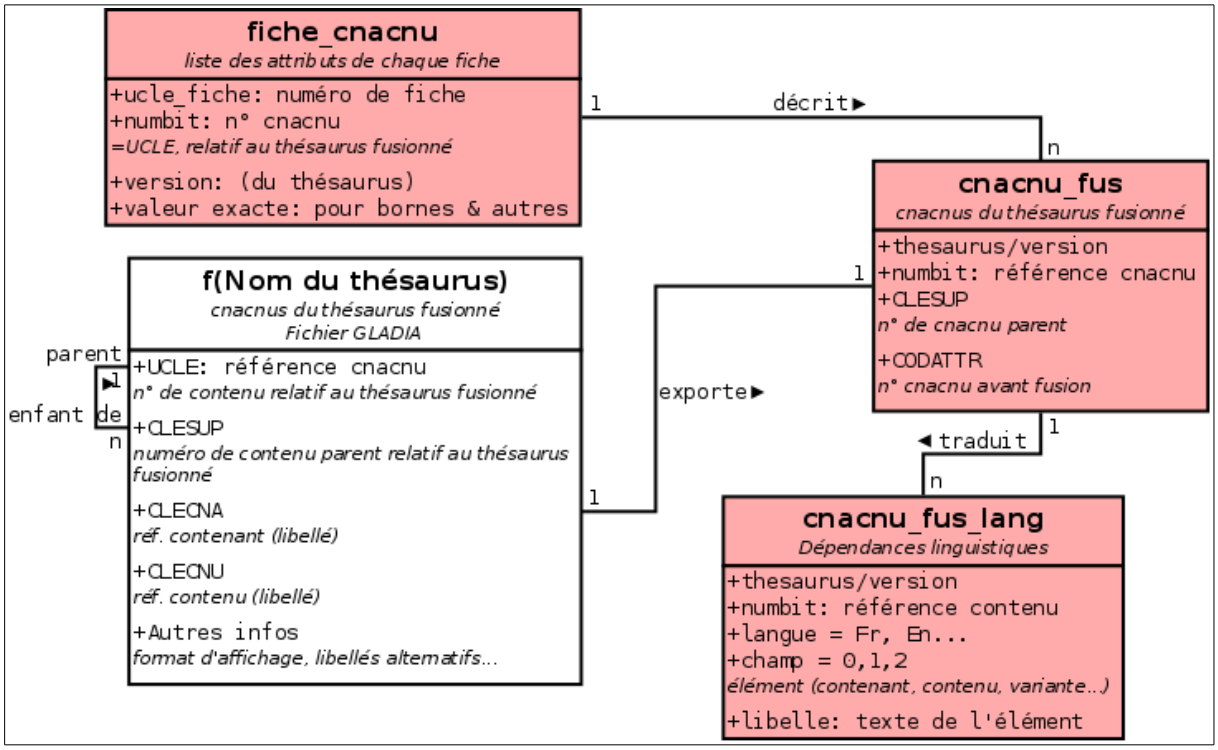


Figure 3.9: Extensions du fichier thésaurus fusionné pour les contenants/contenus

On remarque que dans « cnacnu_fus_lang », chaque contenant/contenu n'est pas associé à un tuple de libellés par langue, mais trois (repérés par des valeurs 0, 1, 2 de l'attribut « champ »). Ceci permet de séparer les différents libellés associés à un contenant/contenu. Dans l'idéal il devrait y avoir quatre tables : contenants, libellés de contenants, contenus, libellés de contenus. Cependant, historiquement les libellés de contenants sont dupliqués dans chaque contenu concerné, ce qui aurait compliqué la mise à jour de quatre tables (il faudrait regrouper les contenants équivalents mais la fusion fait perdre une partie de l'information nécessaire).

Dans l'annexe C on peut noter qu'il pré-existait des tables de contenants/contenus avec leurs libellés traduits. Ces tables ne conviennent pas car elles représentent les thésaurus à un instant de leur établissement, et non pas au moment de leur mise en service : elles ont vocation à être modifiées par le documentaliste pour préparer la prochaine version. A l'inverse les nouvelles tables représentent la version figée, référencée par les fiches.

Lorsqu'une fiche est créée ou modifiée, cela déclenche automatiquement une procédure stockée qui vient insérer dans « fiche_lang » les mots significatifs contenus dans « cnacnu_fus_lang ». Ainsi on peut directement utiliser la recherche libre intégrée au SGBD. Le regroupement des mots-clefs de la fiche est fait lors des mises à jour, supposées beaucoup plus rares que les transactions de recherche.

Enfin, deux tables presque indépendantes sont ajoutées : le journal des défauts de recherche « stat_textsearch » et la table administrative « site_appelant ». Ces deux tables ont été sommairement discutées dans les sections 3.1.3 et 3.1.4 et ne posent pas de problèmes particuliers.

3.2.2. Les traitements

Des résultats attendus et des données à gérer, nous pouvons déduire les traitements nécessaires. Ils s'insèrent en quatre points dans l'existant : à l'entrée d'une recherche, à la sortie des résultats, lors de la modification des fiches (création, suppression, mise à jour), et au niveau des fonctions administratives (initialisation du système, paramétrage et surveillance).

L'étude de l'existant a fait ressortir que les traitements sur les fichiers de données étaient dupliqués en plusieurs endroits du code : dans les actions de l'usager, dans la fonction d'importation et dans les actions administratives de correction d'erreurs sur les fiches. Plutôt que modifier trois codes différents pour mettre à jour les tables d'index des mots de recherche libre, ces répétitions ont été regroupées dans une bibliothèque de fonctions pour réduire le risque d'erreur dans les évolutions prévues et la future maintenance du code.

a. L'appel de la recherche libre

L'interface d'appel de la recherche libre est double. Dans le cas de l'usager classique, s'il utilise le formulaire de recherche libre qui figure sur la première page de recherche guidée, un message est reçu dans sa session et provoque l'appel de la fonction de recherche avec la chaîne tapée en paramètre. Dans le cas du site appelant, la chaîne tapée est passée dans l'URL qui ouvre la session, et la même fonction de recherche est appelée immédiatement.

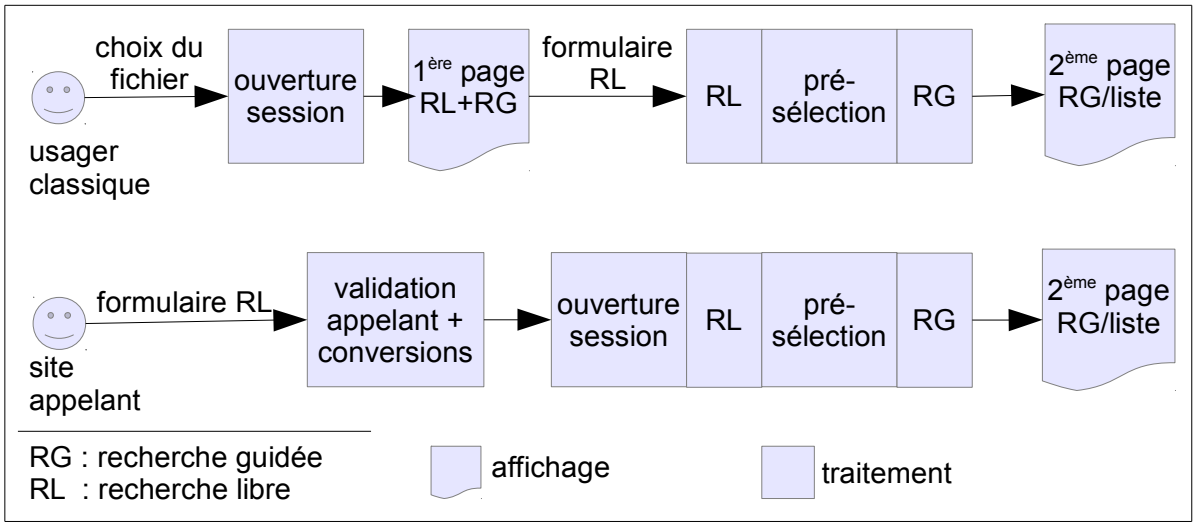


Figure 3.10: Les deux séquences d'appel d'une recherche

La requête initiale du site appelant arrive dans Apache qui exécute un script « shell » d'interface avec Métaguide. Ce script a aussi pour fonction de vérifier le client entrant et de compléter les paramètres d'appel de Métaguide avec des informations contractuelles comme le nom du thésaurus. Si le jeu de caractères convenu est UTF-8, la requête est convertie en ISO8859 qui est utilisé partout dans Métaguide. Tout ceci est réalisé par des requêtes SQL et des commandes Unix classiques (sed, expr, iconv...). L'extrait de code suivant montre la coopération entre le shell, les commandes Linux et l'interpréteur SQL de PostgreSQL.

```
01| #
02| # Initialement TS contient la chaîne à chercher, CI est le code du site appelant
03| # PGSCHEMA et PGUSER sont des paramètres d'accès à la base de données
04| #
05| f_sql() {
06| psql "$PGSCHEMA" "$PGUSER" -F"       " -XAqtc "$1" 2>&-
07| }
(partie supprimée)
08| set -- `f_sql "select enabled,thesaurus,encoding from site_appelant where ci='$CI'"`
09| sa_enabled=$1
10| sa_thesaurus=$2
11| sa_encoding=$3
12| case "$sa_encoding" in
13|        ''|iso8859*)
14|                ;;
15|        *)
16|                TS=`echo "$TS" | url2utf8.x | iconv -f "$sa_encoding" -t iso8859-
1`;;
17| esac
```

Figure 3.11: Extrait de code montrant la coopération Shell/Linux/SQL

Dans ce code, « psql » est un interpréteur SQL en ligne de commande, fourni dans PostgreSQL. Dans cet extrait la fonction shell « f_sql » l'appelle avec la requête SQL complétée par des paramètres invariables. Cette fonction générique est appelée en ligne 08 et le résultat, les trois champs demandés, est transformé par la commande « set » en variables shell $1, $2, $3. Après un renommage pour la lisibilité, la variable « sa_encoding », est « iso8859 » ou un dérivé. Si ce n'est pas le cas, la chaîne du formulaire de recherche est débarrassée de l'encodage URL par la commande Métaguide « url2utf8.x » et convertie en ISO8859 par la commande Linux « iconv ». Le résultat est remis dans la variable shell d'origine « TS ».

b. La recherche proprement dite

La recherche libre elle-même est assez simple, elle est l'aboutissement de tous le processus d'indexation. La chaîne tapée est passée sans préparations particulières[8] à une première requête SQL :

```
SELECT code FROM fiche, fiche_lang
         WHERE fiche.thesaurus='%s'
               AND fiche.version='%s'
               AND fiche.ucle=fiche_lang.ucle_fiche
               AND plainto_tsquery('french','%S')@@fiche_lang.mots
         ORDER BY code;",
```

Dans cette requête la fonction de PostgreSQL « `plainto_tsquery` » réalise l'analyse lexicale, supprime les mots faibles, extrait les radicaux et induit un « et logique » sur la contrainte d'identifier chaque mot. L'opérateur de comparaison« `@@` » est vrai si tous les mots du premier argument existent dans le second argument. A noter que le symbole «%S » est remplacé par la chaîne à rechercher au moment de l'exécution.

Si elle obtient des réponses, cette requête suffit et la sélection est passée à la recherche guidée. Sinon on réexécute la requête en spécifiant un « ou logique » entre les

[8] À l'exception bien sûr des traitements nécessaires pour éviter les attaques par injection SQL !

correspondances de mots, et on trie les fiches par nombre de mots trouvés. La fonction « ts_rank » de PostgreSQL le permet directement :

```
SELECT code,ts_rank(mots,qry) as rank
   FROM  fiche,
         fiche_lang,
         cast(
               regexp_replace(plainto_tsquery('french','%S')::text,
                           '&', '|', 'g')
         as tsquery) as qry
   WHERE thesaurus='%s'
         AND version='%s'
         AND ucle=ucle_fiche
         AND qry@@mots
   ORDER BY rank desc,code;
```

On note deux changements par rapport à la requête précédente : D'abord le résultat de « plainto_tsquery » est transformé pour remplacer les « et » (les symboles '&') par des « ou » (les symboles '|'). Le résultat est nommé « qry » pour pouvoir s'y référer. Comme dans la première requête on retrouve le sélecteur de recherche libre « qry @@ mots » mais cette fois un nombre quelconque de correspondances est accepté. Le résultat est trié par « rank », qui est le nom donné au résultat du calcul des deux premières lignes. La fonction PostgreSQL « ts_rank » analyse la qualité de correspondance du champ « mot » avec le sélecteur « qry ». Par défaut il compte simplement les mots correspondants.

A l'issue de la recherche on ne conserve que les premières réponses, celles qui ont le nombre de correspondances le plus élevé.

Après cette seconde passe, on n'a peut-être toujours rien trouvé : on enregistre le fait dans le journal des défauts « stat_textsearch », et on conserve la totalité des fiches pour la recherche guidée.

c. La modification de la recherche guidée

Le système de recherche guidée est un module séparé qui constitue le cœur de Métaguide. Une liste des fiches est conservée en permanence en mémoire vive partagée avec leurs références de contenants/contenus pour un calcul rapide des listes de contenants et de contenus restant à proposer. Cette liste partagée est complétée, dans chaque session, par une pile des fiches retenues suivant les critères successifs de l'usager. Une bibliothèque de fonctions manipule ces listes en empilant ou en dépilant (pour annuler un choix) les critères choisis.

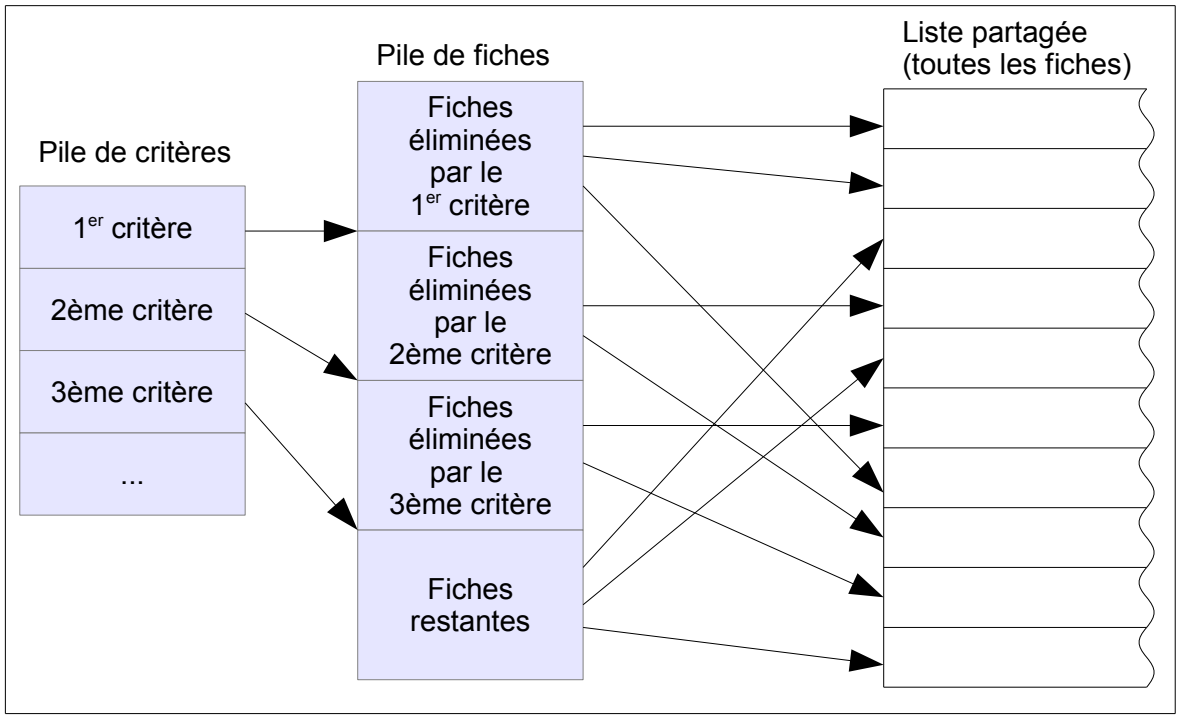


Figure 3.12: Organisation de la pile de critères

La solution retenue consiste à compléter la bibliothèque par une fonction qui, au lieu de sélectionner des fiches par un contenant/contenu, les sélectionne à partir d'une liste fournie. On conserve toute la logique de la bibliothèque, avec un nouveau type de critère. La nouvelle fonction partitionne donc la pile de fiches restantes de la session, en fiches trouvées par la recherche libre et fiches non trouvées. Du coup les structures de données ne changent pas et l'ajout est transparent pour les autres traitements d'affichage et de recherche guidée.

On pourrait étendre ce principe pour permettre la recherche libre à toutes les étapes de la recherche guidée. Mais la sélection SQL continuerait à s'appliquer à toutes les fiches, et on peut craindre une lenteur inattendue alors que l'usager croit en traiter un petit nombre.

d. La sortie vers le site appelant

Dans le cas du site appelant se pose le problème du retour. L'usager interagit avec le site Web de Métaguide. Au moment où il sélectionne une fiche il faut que le lien qu'il utilise force le navigateur à retourner au site appelant.

Il y a deux cas. Lorsqu'on choisit une fiche dans une liste, Métaguide peut fabriquer un lien de retour avec le champ « opaque » de la fiche. Il suffit de modifier le modèle de la page « liste », et de rendre disponible au modèle ce champ supplémentaire. Le traitement est identique aux autres champs de la fiche et ne pose aucun problème particulier.

Par contre quand la sélection d'une fiche résulte d'un choix de critère qui débouche sur une seule fiche, l'information sur la fiche n'est pas disponible au moment du choix. Le traitement de recherche génère une page « fiche » comme cas particulier de la page « liste ». La solution dans ce cas est d'avoir un modèle de page « fiche » qui contient du code javascript ou des balises spéciales qui forcent le navigateur à changer de site sans afficher la page.

Dans tous les cas le problème est résolu sans modification de Métaguide, mais seulement des modèles de page qui sont sous la maîtrise de l'éditeur du service.

e. Le regroupement des fonctions d'accès aux fiches

Pour conserver la base de données dans un état cohérent il est important de traiter de la même manière tous les accès qui modifient des éléments des fiches. On peut identifier dans le code d'origine, huit endroits où les tables de fiches sont modifiées, à savoir :

- la création d'une fiche par l'usager
- la création automatique de fiches de démonstration
- la création par importation
- la modification d'une fiche par l'usager
- la modification d'une fiche par la fonction « service après vente » par exemple pour valider une fiche payée par chèque
- la suppression d'une fiche par l'usager
- la suppression d'une fiche par la fonction « service après vente »
- la suppression de toutes les fiches par le producteur.

Non seulement le code des modifications de fiches était répété à chacun de ces endroits, mais encore les détails du codage étaient différents d'un endroit à l'autre.

Le projet a donc inclus une phase de définition d'une bibliothèque de fonctions centrales incluant la création, la suppression et divers modes de modification d'une fiche (les besoins sont différents selon qu'on modifie les coordonnées, le descriptif ou les contenants/contenus). Il a fallu aussi transformer le code existant pour appeler ces nouvelles fonctions en respectant la nouvelle interface définie. La principale difficulté consistait à gérer, d'une façon cohérente, les erreurs pouvant survenir à la fois du SGBD et de GLADIA : la dualité des fichiers ne permettait pas de considérer les transactions comme atomiques.

Ces fonctions ont ensuite pu être augmentées pour gérer la recherche libre, avec la certitude que tous les cas sont traités. Elles agissent à la fois sur les tables de la base de données, les tables GLADIA et les fichiers d'images associées à chaque fiche.

L'Annexe F :« La bibliothèque d'accès aux fiches », documente la liste des interfaces ; elles-même appellent des fonctions secondaires pour traiter le SQL et GLADIA. Elles sont écrites en langage C pour pouvoir être communiquer avec le code compilé qui les utilise.

La création de cette interface simplifiera grandement une future migration complète de GLADIA vers la base de données.

f. L'indexation des mots à partir des fiches

Au moment de leur insertion dans les fichiers GLADIA, Les textes directement attachés aux fiches sont concaténés et mis dans deux champs (un pour le descriptif, un pour les coordonnées) de la table « fiche_lang » prévue pour cela. Un champ secondaire contient le code de la langue utilisée (toujours forcé à « Fr » dans cette version).

```
"INSERT INTO fiche_lang(ucle_fiche,langue,mots1,mots4)
      SELECT %d, langue, '%S %S %S', '%S %S %S %S %S'
            FROM cnacnu_fus_lang
            WHERE thesaurus = '%s'
                  AND version = '%s'
                  AND numbit = 0
                  AND langue = '%s'
            GROUP BY langue;"
```

Il s'agit d'une chaîne de caractères du langage C où les codes ' %' vont être remplacés par les données courantes pour produire l'ordre SQL final.

Le champ mots1 est chargé avec le descriptif composé de 3 champs GLADIA, le champ mots4 est chargé avec les coordonnées composées de 5 champs GLADIA.

Le passage à plusieurs langues suppose deux phases : d'abord créer les traductions de thésaurus, puis permettre aux utilisateurs de saisir leur descriptif en plusieurs langues. Entre les deux étapes, le SELECT permet de trouver les langues définies pour ce thésaurus. Le critère « … AND langue = '%s' » peut être supprimé par une option de configuration sans recompiler Métaguide. La première phase est donc déjà testable. Les critères du SELECT ne sélectionnent qu'un tuple en passant par une clé primaire, le SELECT n'ajoute donc pas de temps de traitement significatif.

On utilise alors un « TRIGGER » SQL pour déclencher automatiquement une procédure SQL stockée qui va convertir le code langue de Métaguide (un code ISO sur deux caractères) en nom de « configuration full text » compréhensible par PostgreSQL. Ce nom est stocké dans un champ séparé de la table. Chaque ligne de la table est ainsi associée indépendamment à une langue.

Un second « TRIGGER » reprend alors les champs de texte, et utilise la fonction PostgreSQL « tsvector_update_trigger_column » pour les analyser, extraire les radicaux des mots en fonction de la langue, et placer le résultat dans un champ d'index inverse « mots ».

```
CREATE TRIGGER B_mots_update BEFORE INSERT OR UPDATE ON fiche_lang
       FOR EACH ROW EXECUTE PROCEDURE
              tsvector_update_trigger_column(mots, mots_config,
                     mots1, mots3, mots4, mots5, mots_fournis);
CREATE INDEX fiche_lang_gin ON fiche_lang USING gin(mots);
```

L'index de type « GIN » (Generalized Inverted Index) constitue une liste inverse de toutes les fiches contenant un radical donné dans le champ indexé.

La séparation en deux champs permettra de moduler facilement les catégories de mots qu'on souhaite indexer, si les besoins évoluent. De même d'autres sources de mots sont prévus : les libellés de contenants, les libellés de contenus, des mots-clé supplémentaires à définir (« mots_fournis »).

g. L'indexation des mots à partir des thésaurus

L'indexation des libellés de contenus est plus complexe car ils ne sont pas passés directement aux fonctions de création/modification ; il faut les extraire par une jointure. D'autre part il y a un cas particulier : certains contenus sont bi-valués. En effet, quand le thésaurus propose des fourchettes de nombres, l'utilisateur peut fournir une valeur exacte en plus. Il est aussi possible de créer un contenu fourre-tout qui permet à l'usager de fournir sa valeur si elle ne fait pas partie des contenus proposés pour ce contenant.

Mais, contrairement à un fichier classique, ici la modification d'une fiche n'est pas la seule cause de remplacement des mots indexés. A tout moment le producteur peut décider de changer les libellés de ses contenus ou leur traduction. Ces textes sont conservés dans des tables de travail et l'une des fonctions de l'opération de « fusion » est de constituer un corpus intangible de libellés « en exploitation » qui seront utilisés dans tous les traitements de fiches. A l'origine ces libellés sont agglutinés dans un fichier séquentiel, chargé en mémoire vive au démarrage du système.

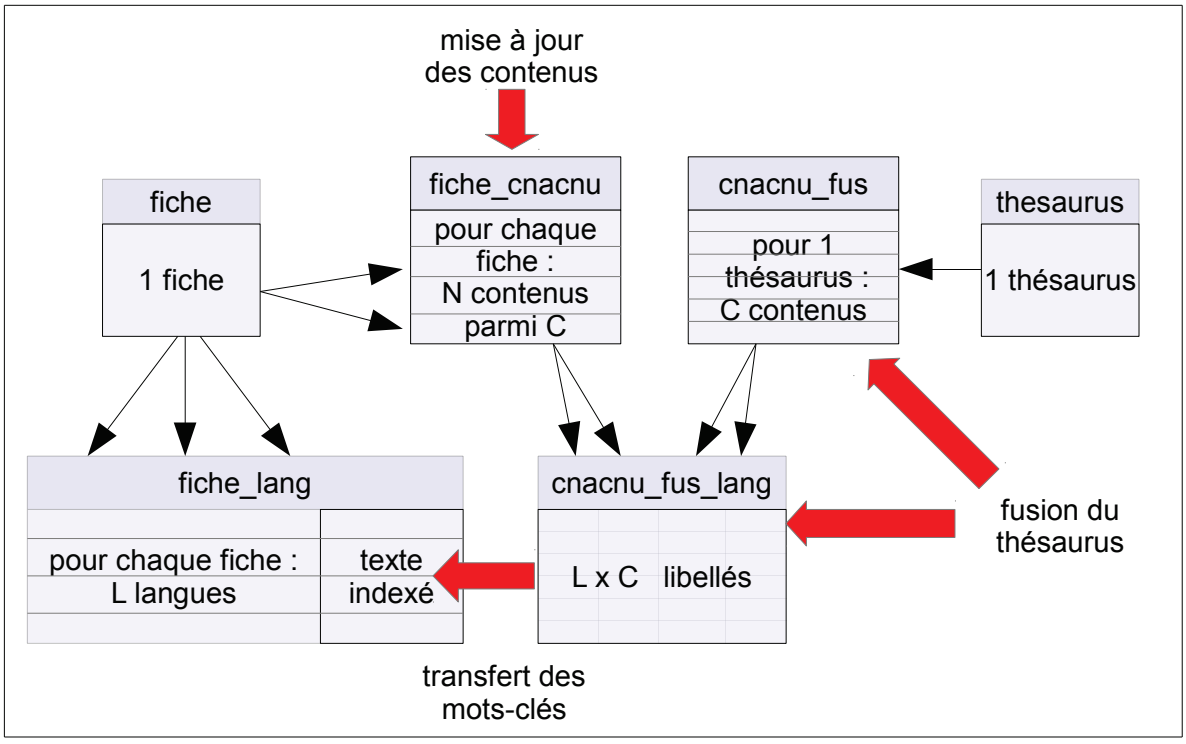


Figure 3.13: Les mises à jour des libellés de contenus

Nous allons maintenant étudier ces différents cas de mise à jour des libellés du thésaurus.

La mise à jour des contenus se fait par suppression et re-création des lignes de la table « fiche_cnacnu ». Lors de la ré-insertion on pourrait avec un trigger, mettre à jour la table de recherche libre « fiche_lang », mais ce trigger, et la ré-indexation qui s'ensuit, seraient appelés pour chaque contenu ; le traitement serait lent. L'approche adoptée est d'insérer d'abord tous les contenus dans « fiche_cnacnu » puis de mettre à jour « fiche_lang » en une seule écriture, et donc une seule ré-indexation.

```
                        -- remplacement total dans fiche_cnacnu
01- DELETE FROM fiche_cnacnu WHERE ucle_fiche = %d AND version='%s';
02- INSERT INTO fiche_cnacnu(ucle_fiche,version,numbit)
03-                    SELECT %d, '%s', unnest(ARRAY[%s]);
(...partie supprimée...) -- traitement des erreurs
                        -- transfert des libellés de contenus à indexer
04- UPDATE fiche_lang as fl SET mots3='%S'||sub.libs FROM
05- (SELECT langue, array_to_string(array_agg(libelle),' ') as libs
06-   FROM cnacnu_fus_lang as cfl, fiche_cnacnu as fc
07-   WHERE -- conditions sur cnacnu_fus_lang
08-       cfl.thesaurus = '%s' AND cfl.version = '%s' AND langue = '%s'
09-       AND cfl.numbit = fc.numbit AND cfl.champ = '1'
10-       -- conditions sur fiche_cnacnu
11-       AND fc.ucle_fiche = %d AND fc.version = '%s'
12-   GROUP BY langue) as sub
13- WHERE fl.ucle_fiche = %d AND fl.langue = sub.langue;
```

Le code SQL utilise plusieurs fonctions spécifique à PostgreSQL. Dans l'étape de remplacement, l'argument du type ARRAY est la liste des références de contenants/contenus ; la fonction « `unnest` *» convertit cette table en liste de lignes pour le SELECT.*

Pour réaliser le transfert des libellés de contenus, on sélectionne les références (champ « `numbit` *») de tous les contenus concernés (ligne 11). On réalise une jointure (lignes 8-9) avec les libellés de contenus pour chaque langue. Les libellés (ligne 5) sont groupés par langue (ligne 12) dans une table par la fonction de groupe «* `array_agg` *» (ligne 5), puis concaténés par la fonction «* `array_to_string` *». La chaîne obtenue est renommée «* `libs` *» et le résultat du SELECT, une ligne par langue, est renommé «* `sub` *» (ligne 12). La dernière étape est un UPDATE (ligne 4) à partir de la pseudo-table «* `sub` *» :* `sub.langue` *est utilisé pour la jointure SELECT/UPDATE (ligne 13) et* `sub.libs` *est utilisé pour modifier les mots à indexer (au passage il est concaténé avec les secondes valeurs des contenus bi-valués).*

Lorsqu'une opération de fusion modifie le thésaurus, les relations de parenté dans l'arbre des contenants/contenus peuvent être affectés. Les fiches doivent donc être retraitées et il serait très difficile de retracer les modifications des libellés : les références ont pu changer et pas les libellés, ou l'inverse, ou les deux, ou dans certaines langues seulement.

Aussi, à la fin d'une fusion réussie, on recrée entièrement les tables « cnacnu_fus » et « cnacnu_fus_lang » à partir du thésaurus de travail ; puis on recrée entièrement « fiche_cnacnu » à partir de son image dans le fichier GLADIA pour lequel il existe déjà une procédure de mise à jour stable. Enfin des commandes SQL similaires à la mise à jour des contenus d'une fiche, mais appliquées à tout le fichier, rétablissent la cohérence des index de recherche libre.

Ces traitements sont insérés dans des scripts « shell » existants, d'une part au moment de la création des fichiers de libellés plats, d'autre part lors de la conversion des fiches GLADIA à l'issue de la fusion.

```
sql="psql metaguide -F$sep -Xaqtc"
thslist=/tmp/gllib.$$
# exportation GLADIA de certains champs du thésaurus vers fichier plat temporaire
exportgl f$thes$vers $exptab $thslist - - '$UCLE$\t$CODATTR$\t$CLESUP$'
# mise a jour de cnacnu_fus
$sql "delete from cnacnu_fus where thesaurus='$thes' and version='$vers'"
awk 'BEGIN{OFS=""} {print "'$thes'","'$vers'",$1}' $thslist |
   $sql "COPY cnacnu_fus(thesaurus,version,numbit) FROM stdin"
# mise a jour de cnacnu_fus_lang
$sql "delete from cnacnu_fus_lang where thesaurus='$thes' and version='$vers'"
$sql "insert into cnacnu_fus_lang(
                              thesaurus,version,numbit,champ,langue,libelle)
         select '$thes','$vers',ucle,champ,langue,libelle from $temptable"
```

Régénération des tables de contenants/contenus d'un thésaurus fusionné : `exportgl` *est la commande d'exportation GLADIA (un peu simplifiée pour cette présentation). Elle crée un fichier texte* `$thslist` *reformaté par la commande UNIX* `awk` *puis inséré par une commande SQL dans la base.*
La table temporaire `$temptable` *dont la préparation à partir du thésaurus de travail n'est pas montrée ici, est importée ensuite dans* `cnacnu_fus_lang`.

```
ww_datacnacnu.x $age "$glbd" export |
   awk -F' ' 'BEGIN{OFS="  "} {print $2,$3,$4,$5}' |
   $sql " TRUNCATE TABLE $temptable;
          COPY $temptable(t_code, t_numbit, t_isnull, t_val) FROM STDIN;"
# move to fiche_cnacnu
$sql "DELETE FROM fiche_cnacnu USING fiche f
          WHERE ucle=ucle_fiche AND thesaurus='$ths' AND f.version='$ver';"
       INSERT INTO fiche_cnacnu(ucle_fiche,numbit,version,valeur)
          SELECT ucle,
                 t_numbit,
                 '$ver',
                 CASE WHEN t_isnull THEN NULL ELSE t_val END
              FROM $temptable,fiche
              WHERE thesaurus='$ths' AND version='$ver' AND code=t_code;"
```

Régénération des tables de libellés contenus des fiches : `ww_datacnacnu.x` *est une exportation spécifique de GLADIA qui extrait les références de contenants/contenus. Elles sont d'abord insérées dans une table temporaire pour réaliser le reste du traitement en SQL. La clé primaire de f*`iche_cnacnu`*, qui ne figure pas telle quelle dans le fichier GLADIA, est retrouvée par jointure avec la table fiche grâce à une clé secondaire unique commune. '*`t_isnull`*' marque les contenus bi-valués.*

Il faut noter que toutes les données (thésaurus et fiches) en exploitation sont versionnées. Ceci autorise la suppression et la recréation des tables à volonté et on n'a pas besoin de se soucier de l'atomicité du processus de fusion : en cas d'erreur il peut être relancé autant de fois que nécessaire.

h. La suppression

La suppression des fiches inclut la suppression dans deux fichiers GLADIA, dans les tables de la base de données et dans les répertoires du disque dur où sont gardées les images. Elle est inchangée sauf en ce qui concerne la recherche libre.

Tous les éléments de recherche libre sont dans la base de données et dépendent des fiches. La déclaration correcte des tables « fiche_lang » et « fiche_cnacnu » avec des clés étrangères et l'option CASCADE permet d'automatiser la suppression tout en garantissant l'intégrité référencielle. Il n'y a donc aucun traitement de suppression.

```
CREATE TABLE fiche_lang (
(...partie supprimée...) -- liste des champs et des clés de la table
   FOREIGN KEY (ucle_fiche) REFERENCES fiche(ucle)
          ON UPDATE CASCADE ON DELETE CASCADE );
```

i. L'indexation initiale

Pour pouvoir utiliser la recherche libre sur les fichiers déjà en exploitation, il est nécessaire de préparer les tables d'index des mots-clefs. Cette opération ne s'exécute qu'une seule fois, à la première mise en service du système de recherche libre, et plus précisément de la phase qui permet l'indexation.

Il suffit d'appeler les deux scripts d'indexation à partir des contenants/contenus, décrits au point g ci dessus, dans une boucle balayant tous les thésaurus. On exécute ainsi la dernière phase nouvellement ajoutée dans la fusion. Un troisième script appelé dans la même boucle exporte, pour tous les thésaurus exploités dans le système, les descriptifs et les coordonnées de contact des fichiers GLADIA pour les insérer dans la base de données.

j. Les fonctions réalisées avec phppgadmin

Dans un premier temps, la table « site_appelant » et le journal « stat_textsearch » ne devraient être utilisées que par une seule personne à TRANSWORK, la première pour y créer des contrats de sites appelants, la seconde pour examiner l'efficacité de la recherche.

La personne concernée ayant déjà un accès complet à la base de données, une interface graphique complète n'a pas été réalisée. En effet des fonctions standard du service Web « phppgadmin » permettent de lister, modifier et exécuter des sélections sur une table PostgreSQL.

3.3. Les moyens techniques

Le composant doit traiter à la fois avec les fichiers GLADIA, la base de données, les requêtes HTTP, la génération de pages HTML et le moteur de recherche guidée. Tous ces éléments peuvent être utilisés au travers d'interfaces adaptées soit au langage C compilé, soit au shell interprété.

Un certain nombre de bibliothèques de fonctions existent pour accéder en langage C à ces ressources. En particulier le composant « Osmose » fournit des listes chaînées bidimensionnelles en mémoire vive très commodes pour toutes sortes d'usages. Dans le cas de GLADIA et de PostgreSQL, au-dessus de la bibliothèque native (fournie par le concepteur de l'outil), Métaguide a créé des sur-couches utilisant Osmose pour uniformiser les structures de données entre les outils et les programmes d'application. Les listes Osmose contiennent aussi les données à insérer dans les modèles de pages HTML ou les cartes graphiques.

Le shell, langage de commande de Linux, est aussi très utilisé pour les traitements dont le temps d'exécution n'est pas critique. Les scripts shell de Métaguide utilisent toutes sortes d'utilitaires POSIX de traitement de texte. Ils extraient les données des fichiers GLADIA avec une commande d'exportation et communiquent avec la base de données grâce à une commande qui exécute des scripts SQL.

L'utilisation efficace des outils de recherche libre de PostgreSQL a imposé une mise à jour de ce SGBD suivie d'une conversion de la base de données pour définir correctement ses caractéristiques liées au langage : codage des caractères et fonctions de comparaison.

3.3.1. Le système de développement

Les phases de réalisation et de test du projet ont eu lieu sur un ordinateur différent du serveur d'exploitation. En effet, celui-ci, installé chez un fournisseur d'accès et ouvert sur Internet, est installé avec le minimum de logiciels nécessaires à l'exploitation, pour réduire les risques d'intrusions malveillantes.

L'ordinateur de développement est un PC[9] de configuration semblable au serveur d'exploitation de TRANSWORK. Tous deux sont installés avec la même version de la distribution Debian de Linux. Un serveur Web complet est installé sur le système de développement et accessible localement. Le contenu de la base de données du serveur est partiellement recopié pour réaliser des tests réalistes.

La mise à jour de parties de Métaguide sur le serveur d'exploitation est possible en utilisant les programmes Linux « `ssh` » et « `scp` ». `ssh` ouvre une session en ligne de commande protégée par cryptage à clés asymétriques. On utilise `scp` pour recopier des fichiers sur le serveur avec les mêmes protections.

3.3.2. Le système de gestion de bases de données PostgreSQL

PostgreSQL [2][3] est un SGBD relationnel « open source » de qualité reconnue et disposant de nombreuses fonctionnalités : langage SQL aux nombreuses extensions, gestion des droits d'accès, langage de procédures stockées, transactions ACID, dérivation de tables considérées comme des classes (relationnel orienté objet), extensibilité par composants enfichables, utilisable dans de nombreux langages de programmation, etc.

[9] C'est-à-dire un ordinateur potentiellement capable de supporter Microsoft® Windows®

PostgreSQL est dérivé de la base de donnée commerciale Ingres dont les premières versions remontent à 1977. Le support correct des transactions est la raison majeure qui a conduit TRANSWORK à le choisir face à MySQL il y a une dizaine d'années.

a. Les interfaces SQL

Le moyen essentiel d'utilisation de PostgreSQL est le langage SQL. Ce SGBD possède des interfaces d'utilisation de SQL adaptées à chaque cas dans Métaguide :

- *psql* est un utilitaire prévu à l'origine pour passer des ordres SQL en ligne de commande. Cependant dans Métaguide il est couramment utilisé dans des scripts « shell » où il est appelé avec des options qui suppriment l'interaction clavier et les entêtes descriptifs des résultats de SELECT.
- *phppgadmin* est une interface graphique à PostgreSQL. Il s'intègre au serveur Web Apache en définissant un mini-site Web. Avec un navigateur, on peut visualiser et modifier la structure des bases de données, lister leur contenu, créer des rapports, et d'une façon générale réaliser toutes les opérations SQL classiques, d'une façon conviviale. Dans le développement de Métaguide il est très utilisé ; mais il l'est aussi dans l'exploitation pour accéder à certaines tables dont le faible taux d'accès et/ou la compétence des utilisateurs n'a pas exigé de créer une interface spécifiquement contrôlée. C'est par exemple le cas des tables du projet « site_appelant » et « stat_textsearch ». L'Annexe G : «Copie d'écran de phppgadmin » montre une page de phppgadmin affichant la structure d'une table de la base.
- *libpq* est une bibliothèque de fonctions fournie dans PostgreSQL. Elle permet à un programme compilé de passer des commandes SQL. Dans Métaguide il est utilisé par les programmes écrits en langage C, au travers d'une couche de fonctions supplémentaires (ci-dessous).
- *cl_sql.o* est un groupe de fonctions qui fait partie de Métaguide. Elle simplifient l'utilisation de *libpq* de plusieurs façons.
  - ✔ En regroupant des fonctions que Métaguide appelle toujours ensemble
  - ✔ En insérant les résultats des SELECT dans des listes dynamiques, les listes Osmose (décrites plus bas)
  - ✔ En indiquant dans chaque ordre SQL s'il fait partie d'une transaction
  - ✔ En définissant une fonction de formatage qui, à la manière de la fonction printf du langage C, insère dans une commande SQL des variables internes au programme appelant
  - ✔ En luttant contre l'injection SQL, en centralisant le contrôle des caractères spéciaux (quotes ou barres obliques) dans les données insérées dans le code SQL.

b. La recherche plein texte

La principale fonctionnalité de PostgreSQL utilisée dans la recherche libre est bien sûr la « recherche plein texte » ou « full text search » en anglais. PostgreSQL la supporte grâce à un nouvel opérateur et deux types de données. Ainsi dans l'expression :

`achercher @@ ouchercher`

par exemple : `'auto & moto' @@ 'auto bato moto et avion'`

`achercher` de type `tsquery`, contient les mots à chercher et des relations ET, OU, SAUF. `ouchercher` de type `tsvector`, contient une liste de mots à examiner, valués par importance. L'opérateur `@@` renvoie VRAI ou FAUX selon que l'expression `achercher` correspond ou pas à la liste `ouchercher`. Typiquement le `tsvector` est un champ d'une table et le `tsquery` provient d'un utilisateur.

Généralement on n'utilise pas directement les mots ; le `tsquery` et le `tsvector` sont retraités avec des fonctions spécifiques à PostgreSQL, pour améliorer la qualité des correspondances. Les fonctions to_`tsvector`() et to_`tsquery`() utilisent pour cela un analyseur syntaxique et des dictionnaires pour éliminer la ponctuation, les mots faibles et extraire les radicaux des mots en fonction de la langue du dictionnaire, et tous ces outils sont des composants remplaçables.

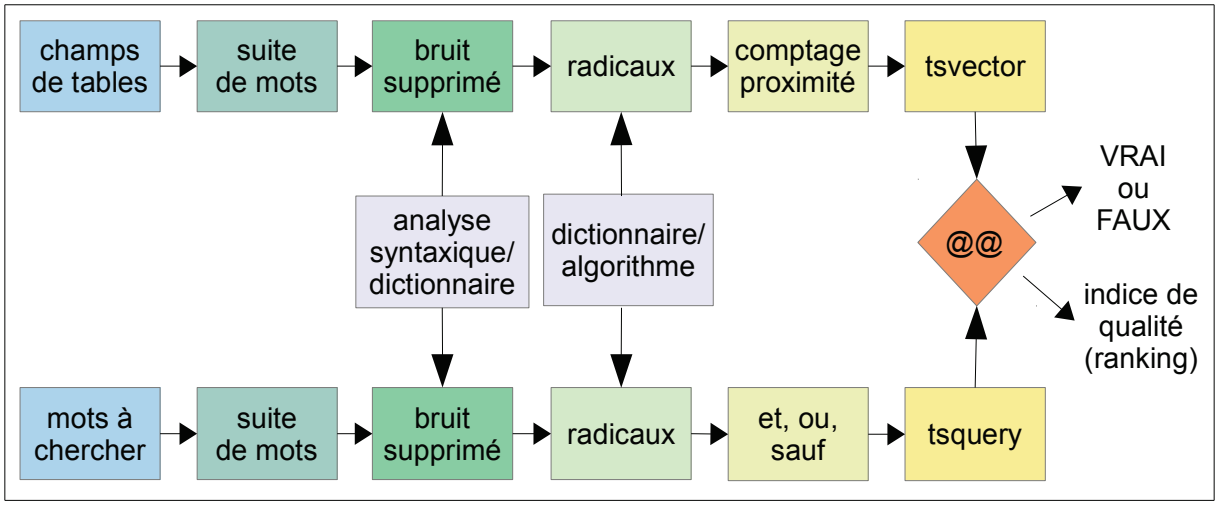


Figure 3.14: Traitements full-text dans PostgreSQL

Enfin on peut créer une table d'index selon le principe de la liste inverse qui permet de retrouver les lignes d'une table dont le champ `tsvector` indexé contient un mot donné.

c. L'évolution du gestionnaire de base de données

Au démarrage du projet, le produit utilisait le SGBD PostgreSQL version 7.4. Après étude des différentes alternatives pour le traitement du texte libre, il s'est avéré que PostgreSQL permettait tous les traitements nécessaires, tout en permettant des évolutions futures grâce à un système ouvert d'extensions enfichables. Cependant la version 7.4 de PostgreSQL souffrait de plusieurs inconvénients :

- la recherche de texte libre n'était supportée que comme extension extérieure
- elle souffrait de quelques restrictions
- elle nécessitait la capacité de prendre en compte des jeux de caractères nationaux
- elle n'était plus maintenue, les versions officiellement supportées dans Debian étaient 8.4 et 9.0, et la version la plus à jour sur le site PostgreSQL était 9.1.

En fait la mise à jour était artificiellement verrouillée sur le serveur Métaguide pour éviter d'introduire d'éventuelles différences de fonctionnement.

L'alternative était de conserver la version 7.4 en ajoutant l'extension appropriée et en composant avec les restrictions, ou de passer à la version 8.4 en contrôlant toutes les incompatibilités possibles, ou de passer à la version 9.0 de la même façon. Quitte à modifier le SGBD dans tous les cas, il aurait été dommage de rester avec une version obsolète. Le choix final entre les versions 8.4 et 9.0 a tenu au risque perçu à sauter de deux versions majeures en une fois.

La version installée 7.4 était la dernière de la série 7. Après une étude approfondie des notes de mise à jour des versions 8.0, 8.1, 8.2, 8.3, 8.4, et de l'impact possible dans Métaguide des dizaines de modifications significatives à chaque version, il est apparu que les versions 8.0 et 8.1 ne présentaient que deux incompatibilités potentielles mineures, les autres évolutions ne touchant que les paramètres d'installation du SGBD. La version 8.1 présentait en outre l'avantage d'être disponible dans les archives Debian, bien que non maintenue. Par contre un test sur l'ordinateur de développement a montré que le remplacement de la version obsolète nécessitait des reprises manuelles non documentées.

De la version 8.1 à 8.4, par contre, des évolutions concernant les types de données, principalement la suppression de certains transtypages automatiques, nécessitait une révision générale du code de Métaguide. Une liste de tous les ordres SQL visibles dans les sources a été dressée, chaque ordre a été examiné et corrigé le cas échéant. Cet effort a permis d'utiliser la version 8.4 tout en ouvrant la possibilité d'évolutions futures.

d. L'évolution de la base de données de Métaguide

Enfin, l'utilisation du SGBD pour la recherche libre suppose qu'il connaisse les langages traités (par exemple pour l'extraction de radicaux) et donc les codages de caractères non-ASCII. Métaguide étant multi-langue, un jeu de caractères international était souhaitable et l'UTF-8 s'est présenté naturellement comme ouvrant le maximum de possibilités. Cependant le codage UTF-8 est multi-octets, c'est-à-dire que les caractères du jeu ASCII occupent 1 octet dans les textes, tandis que les autres caractères en occupent jusqu'à 4. Par exemple le « é » occupe 3 octets. Ceci amenait plusieurs écueils à éviter :

- La base existante, bien que déclarée comme contenant de l'ASCII 7 bits, contenait déjà des caractères internationaux à priori codés en ISO8859, un codage mono-octet

– Tout le code de Métaguide, en particulier le code écrit en langage C, est conçu pour traiter un codage mono-octet

– Toute l'acquisition de données, dont les pages HTML de formulaires et le programme d'importation, sont conçus pour le jeu de caractères ISO8859.

Aussi, plutôt que l'UTF-8, le jeu de caractères utilisé pour les données a été défini en ISO8859-15 pour les alphabets occidentaux. Ceci limite les langues supportables, mais on peut remarquer que cette limite existait déjà. Le support des caractères multi-octets demanderait une révision très conséquente de Métaguide.

Le codage des caractères n'est qu'un aspect du problème. Quand le SGBD ne fonctionne pas en ASCII, on doit spécifier les paramètres régionaux qui ont au moins deux influences majeures. Pour les tris des clés de type caractère, le « collating sequence » spécifie l'ordre de comparaison entre les caractères : le « é » est-il avant le « e » ou après lui, ou après toutes les lettres ? La réponse à cette question, malheureusement, est différente selon la langue employée, même parmi les langues occidentales. D'autre part, le « character type » différencie le code d'une lettre, d'un chiffre, d'une ponctuation, d'un espacement. Cette caractéristique est importante pour l'analyse syntaxique intégrée à la recherche libre. Sur ce point par contre, la table d'identification des types est identique pour toutes les langues utilisant l'ISO8859-15. La base de données a donc été configurée pour opérer un tri dans l'ordre numérique des codes ISO8859 (indépendant de la langue), en remarquant qu'il n'y a pas de clés contenant des caractères nationaux dans la base ; et un typage des caractères pour la langue française, qui après vérification est identique aux autres langues européennes.

3.3.3. Le gestionnaire de fichiers GLADIA

GLADIA est un gestionnaire de fichiers ISAM multi-index. Il a été initialement conçu pour simplifier l'écriture d'applications MINITEL. De ce fait, il contient aussi un générateur de formulaires et de pages liées qui n'est pas utilisé dans Métaguide. D'autres fonctions de statistiques, d'organisation des fichiers… ne présentent pas d'intérêt pour notre propos.

a. Caractéristiques générales

– Compilateur de description textuelle de fichiers

– Possibilité d'indexer plusieurs champs pour les recherches, de composer des critères (champs et types de comparaison) ; optimiseur de recherche intégré

– Possibilité de déclarer des champs de type « recherche libre » qui, comme dans PostgreSQL, généreront des fichiers de listes inverses ; cependant l'algorithme de comparaison est limité aux caractères majuscules non accentués

– Programmes d'importation et d'exportation

– Génération automatique d'une clé primaire (numéro de séquence)

– Chaque fichier GLADIA réside dans son propre répertoire contenant principalement sa description textuelle et compilée, un fichier de données et une liste inverse et leurs index respectifs

– Pas de jointure possible entre les fichiers

– Champs de taille fixe, sans outils pour les redéfinir en taille, en type ni en nombre

– Hautement portable sur les variantes d'UNIX et Linux, et sur divers processeurs.

b. Interface avec le Shell

Les deux commandes « import.x » et « export.x » permettent d'utiliser un fichier GLADIA dans un script « shell » en convertissant un fichier de données texte de et vers GLADIA. Ces commandes utilisent des fichiers texte annexes qui décrivent où sont les données dans le fichier à importer ou exporter. Dans Métaguide on n'utilise que l'exportation, pour convertir un fichier GLADIA en texte et le retraiter avec les utilitaires POSIX classiques.

c. Interface avec les programmes compilés

GLADIA peut s'utiliser en langage C grâce à une bibliothèque présentant les fonctions très classiques open, close, read, write, les fonctions moins classiques car orientées ISAM select, insert, delete, modifie (update), des fonctions pour extraire et modifier un champ d'une fiche : prendre et mettre, et des fonctions pour obtenir des informations sur la structure du fichier.

Ces fonctions ne sont pas documentées mais on dispose d'une bonne base d'exemples d'utilisation.

La logique de fonctionnement peut être surprenante : par exemple la fonction « select » ne fait qu'ajouter un critère de sélection à une liste qui sera utilisée à la prochaine lecture. La lecture s'effectue obligatoirement par blocs de plusieurs fiches, qu'il faut toutes traiter avant de poursuivre la lecture.

3.3.4. Le système de recherche par thésaurus

Pour croiser efficacement les contenants/contenus de toutes les fiches associées à un thésaurus, ces attributs sont stockés en mémoire vive de l'ordinateur, partagée entre tous les processus qui en ont besoin. Chaque fichier est stocké séparément dans un segment de mémoire, avec son thésaurus, la liste des fiches et leurs attributs sélectionnables. Un programme charge les thésaurus en mémoire au démarrage du système et à chaque mise à jour d'un thésaurus.

Une bibliothèque de fonctions sert d'interface avec les programmes utilisateurs, obligatoirement compilés (le système de recherche n'est pas accessible aux langages interprétés comme PHP ou le shell).

Pour résumer le contenu de cette bibliothèque, chaque processus utilisateur crée une instance de l'objet `th_user` qui représente l'utilisation d'un thésaurus par un utilisateur. Il récupère alors une liste de références de fiches. Ensuite il peut réduire la visibilité du fichier en se limitant à un propriétaire de fiches, puis, sous l'action de l'usager, ajouter successivement des filtres (des critères de sélection basés sur des références de contenant/contenu combinés par ET ou OU). Les flèches jaunes de la figure 3.15 montrent cet ajout.

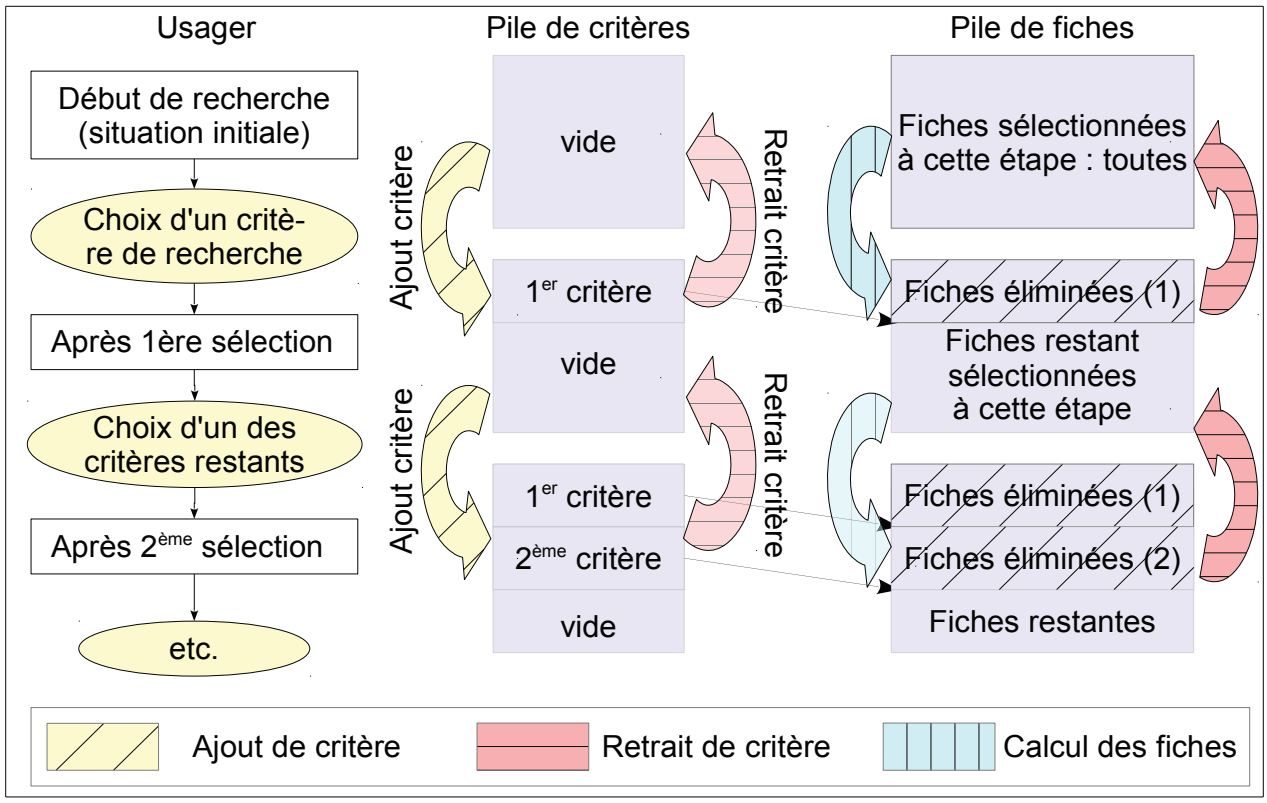


Figure 3.15: Correspondance actions usager / actions Métaguide sur la pile de critères

Le processus peut retirer le dernier filtre ajouté (les flèches roses), ce qui correspond à un retour à la page précédente de recherche pour l'usager. Sinon il doit appliquer le filtre à la liste des fiches restantes (les flèches vertes) pour les partitionner en fiches éliminée par le dernier filtre, et fiches restantes à nouveau. Le point de partitionnement est noté avec le critère, pour pouvoir l'annuler sans calcul supplémentaire.

Le processus utilisateur peut alors demander le calcul des contenants qui restent significatifs à cette étape, et demander pour ces contenants significatifs la liste de leurs contenus correspondant effectivement à des fiches. Ces informations servent à constituer la page Web contenant les listes des prochains choix.

Il est enfin possible d'ajouter ou d'annuler une fiche. Les fiches annulées sont définitivement supprimées par un « batch » quotidien.

3.3.5. Le shell

Dans Métaguide, les processus qui n'exigent pas de performances particulières sont réalisés en shell, le langage de l'invite de commande de Linux. On ne présente plus ce langage interprété — en réalité une famille de langages — omniprésent dans tous les dérivés d'UNIX.

Le shell utilisé est « bash », une version très étendue du shell préconisé par POSIX.

Il intervient principalement dans 4 situations ;

- la liaison entre Apache et Métaguide sous forme de scripts CGI
- les traitements de fusion des thésaurus
- la liaison entre PHP (les tableaux de bord des utilisateurs) et Métaguide
- les processus quotidiens, sauvegardes, nettoyages, statistiques, traitements clients particuliers.

Le projet utilise le shell essentiellement à la fin du processus de fusion de thésaurus, pour transférer les libellés de contenants/contenus dans la base et mettre à jour les index de recherche libre.

Les scripts shell de Métaguide utilisent lourdement les utilitaires POSIX de traitement de fichiers texte. En particulier, les traitements qui impliquent à la fois la base de données et les fichiers GLADIA commencent en général par exporter le fichier GLADIA dans un format texte, une fiche par ligne, champs séparés par un caractère ASCII TAB. Ce format est particulièrement commode pour les utilitaires POSIX comme sed , join, comm, cut, grep, awk... qui, réunis, offrent les fonctionnalités d'une petite base de données.

L'interaction avec la base de données elle-même utilise « psql », déjà décrit.

La connaissance de bash et de ces utilitaires est essentielle pour comprendre les scripts de Métaguide, qui ne sont pas triviaux.

3.3.6. L'affichage « Osmose » et les tables dynamiques

L'un des problèmes traités dans Métaguide est la séparation des traitements et de l'interface utilisateur. Dans les interfaces utilisateur réalisées en langage C, trois composants réalisent cette fonction : l'analyse des requêtes HTTP en entrée, la génération de pages HTML paramétrées, la génération de cartes graphiques réactives. Tous s'appuient sur un quatrième composant de gestion de listes dynamiques. L'ensemble se nomme « Osmose ».

Une « liste Osmose » est une structure en mémoire vive équivalente à une table de base de données. Elle possède un nombre de lignes numérotées (les tuples) qui peut croître dynamiquement, un nombre de colonnes nommées (les champs) dont la liste est aussi dynamique, et chaque couple (ligne, colonne) donne accès à une valeur de type chaîne de caractères. Il est possible d'ajouter et supprimer des lignes, des colonnes ou des valeurs, et réaliser quelques autres opérations comme le tri. Une liste porte un nom et peut être créée ou supprimée dynamiquement. En sous-produit, ces listes fournissent au langage C le type « chaîne de caractères dynamique » qui lui manque cruellement, et pour cette raison les listes Osmose sont omniprésentes dans le code de Métaguide.

La fonction d'analyse des requêtes HTTP découpe l'URL en couples (variable, valeur) et place les valeurs dans la ligne zéro d'une liste Osmose, dans le champ du nom de la variable.

La génération de cartes réactives s'appuie sur des listes de polygones définis dans les contenus des thésaurus graphiques. Ces listes sont passées par des listes Osmose à une fonction qui calcule les couleurs les plus appropriées en fonction de la densité de fiches concernant chaque zone de la carte.

Les pages HTML de Métaguide sont sous forme de modèles (« templates ») qui contiennent des balises particulières. Des balises de la forme

```
{$nom_liste.nom_champ$}
```

indique au générateur de pages qu'à cet endroit il faut insérer un champ de la ligne zéro de la liste Osmose indiquée. Une balise

```
<GROUP LIST=nom_liste> … {$nom_champ$} … </GROUP>
```

demande la duplication du code dans le GROUP, pour chaque ligne de la liste Osmose, en remplaçant nom_champ par la valeur du champs sur la nième ligne. Il est possible d'avoir des boucles imbriquées.

Au final, les listes Osmose sont l'outil principal de stockage intermédiaire en mémoire vive dans Métaguide. Son emploi a été généralisé pour échanger des tables de données entre les fonctions C ou comme couche intermédiaire pour l'appel des commandes SQL ou GLADIA. Ainsi la fonction de *cl_sql.o* qui exécute un SELECT depuis le langage C, renvoie les lignes obtenues sous forme d'une liste Osmose.

3.3.7. Les tests en PHP

L'étude initiale des possibilités de recherche libre de PostgreSQL, et la validation de la fonctionnalité « sites appelants » sont réalisées avec une simple page HTML.

Le serveur Web Apache inclut un interpréteur PHP qui peut accéder au SGBD. Les pages HTML générées peuvent inclure des listes de résultats extraits de la base de données Métaguide, ce qui simplifie l'écriture et l'exécution des tests.

L'utilisation du PHP permet aussi de changer dynamiquement le codage de caractères utilisé en le changeant à chaque rafraîchissement de la page HTML. Ainsi on peut réaliser tous les tests en plusieurs codages avec une seule page.

3.4. Le déploiement

La mise en service du composant a eu pour principale contrainte de ne pas perturber les usagers plus que nécessaire.

Pour réduire le risque d'erreurs, l'installation sur l'ordinateur d'exploitation a été découpée en étapes successives de mise en place des données et des programmes. Chaque étape a donné lieu à la rédaction d'une check-list pour éviter tout oubli ou erreur de manipulation. Une machine virtuelle chargée avec les fichiers et programmes en exploitation, a permis de vérifier la check-list et de tester l'absence d'anomalies de fonctionnement à l'issue des opérations.

Enfin, la documentation a été mise à jour, et quelques améliorations hors projet sont venues s'insérer.

3.4.1. Les étapes successives

La mise en exploitation de la solution imposait de mettre à jour le SGBD, de convertir la base de données, puis de la compléter et installer les modules modifiés ou ajoutés. Chaque évolution de la base pouvait nécessiter une modification de Métaguide à mener en parallèle pour ne pas perturber l'exploitation.

Plutôt que de réaliser tous les travaux d'installation en une seule fois à la fin du projet, nous avons choisi de les découper en petites étapes successives, de façon à mieux maîtriser chacune d'elles. Les quatre évolutions considérées se sont déroulées en six étapes.

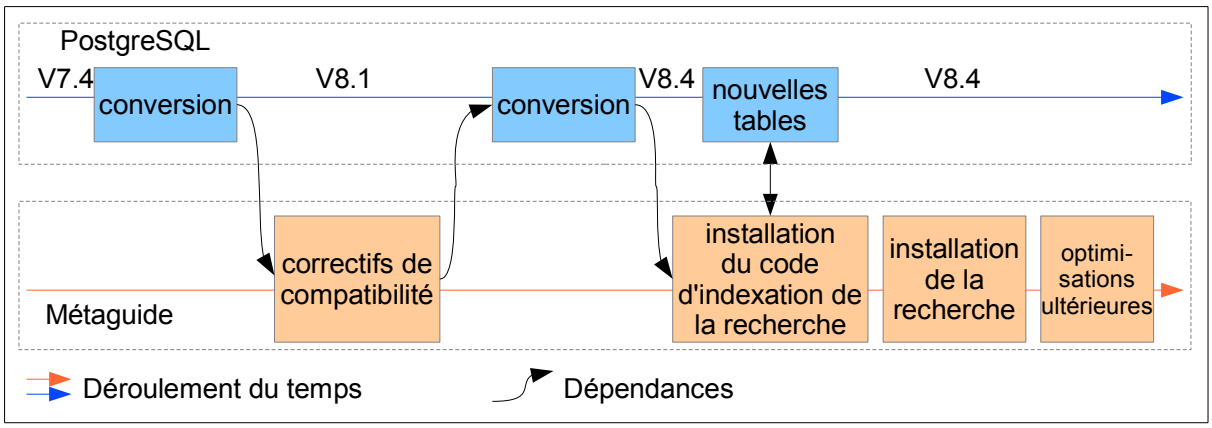


Figure 3.16: Étapes de la mise en service

1) La mise à niveau du SGBD de la version obsolète 7.4 à la version obsolète 8.1. L'intérêt de cette étape était double. D'une part cette évolution n'apportait pas d'incompatibilités dans le code de Métaguide, d'autre part les outils fournis dans la distribution Debian permettaient cette migration, alors qu'ils ne fonctionnaient pas pour migrer de 7.4 à 8.4 directement. Cette étape a nécessité un arrêt du service.
2) Amélioration de Métaguide. PostgreSQL 8.4 apporte quelques incompatibilités signalées dans les notes de version, à cause de l'abandon de quelques syntaxes SQL non standard. Une revue complète des ordres SQL utilisés dans Métaguide a corrigé le petit nombre de cas où ces notes s'appliquaient. Les corrections étaient compatibles avec les versions antérieures du SGBD et ont pu être installées entre les deux mises à jour de celui-ci.
3) La mise à jour du SGBD de la version obsolète 8.1 à la version supportée 8.4. Dans cette étape on en a profité pour spécifier les paramètres régionaux, ainsi que l'encodage des caractères LATIN9 (alias pour ISO8859-15) après avoir vérifié que toutes les données dans la base utilisaient déjà ce code. Cette étape a nécessité un arrêt du service.
4) La création des tables et des procédures stockées nécessaires au composant de recherche libre. Simultanément on a installé les programmes nécessaires pour remplir et maintenir ces tables, et on a exécuté le programme de chargement initial. Cette étape a nécessité un arrêt du service.

5) L'installation de la fonction de recherche. C'est la seule fonction qui a un effet visible sur l'usager, et on a retardé son installation pour laisser « vivre » quelques temps les nouvelles tables et vérifier l'absence d'anomalies dans les traitements d'indexation avant de les rendre visibles. Dans un premier temps la fonction n'était accessible que par des pages « cachées » pour des besoins de validation et de démonstration.

6) Quelques semaines plus tard l'utilisation sur des fichiers réels a montré quelques lenteurs dans certains cas, et des optimisations ont été apportées et installées.

A l'issue de ces étapes le serveur offrait la nouvelle fonctionnalité et l'ensemble des arrêts de service n'avait pas excédé quelques heures.

3.4.2. L'organisation de la mise en service

Chacune des étapes de l'installation du nouveau système de recherche libre sur le serveur de la société est soumise à deux contraintes importantes. La première est de ne pas introduire de dysfonctionnements, la seconde de réduire au maximum les interruptions de service. On remarque que, à part la quatrième, chaque étape concerne soit les évolutions des données seulement, soit celles des programmes seulement.

De façon à éviter les dysfonctionnements, les modifications des programmes ou de la structure des données sur le serveur d'exploitation sont précédées de tests sur un ordinateur témoin simulant le serveur. Il s'agit en fait d'une machine virtuelle tournant dans un environnement Oracle® Virtualbox[10] sous Windows, et qui exécute la même distribution Linux Debian et les mêmes programmes de Métaguide que le serveur, et ses données sont un extrait réaliste des données réelles.

La mise à jour des programmes est assez simple. Les programmes dont l'exécution est manuelle, comme les scripts d'initialisation, sont simplement transférés dans le répertoire des exécutables. Quant aux programmes dont l'exécution est déclenchée par les actions de l'usager, le moniteur MEGABUS possède un mécanisme de basculement automatique de versions au moment de l'ouverture d'une session. En résumé, la mise à jour des programmes n'implique pas d'interruption de service.

La mise à jour de la base de données, elle, suppose par sécurité de la sauvegarder entièrement et de cesser toute activité pendant une conversion pour ne pas perdre les transactions en cours au cas où il faudrait restaurer la sauvegarde. De plus le changement de l'encodage ou de paramètres régionaux nécessitent la recopie intégrale de la base. En conséquence la modification de la structure de la base de données impose en général un arrêt du service. Le volume total de la sauvegarde, environ 11 méga-octets, est assez petit pour être créé, recopié et même édité le cas échéant, en quelques secondes.

La mise à jour du SGBD et les changements de structure enchaînent beaucoup d'opérations qu'il est préférable d'exécuter manuellement pour surveiller les éventuelles erreurs inattendues. Pour ce genre d'opérations une check-list précise est rédigée en parallèle avec le codage du composant, puis mise au point et validée au moment de la validation du code sur le simulateur.

[10] Voir https://www.virtualbox.org/

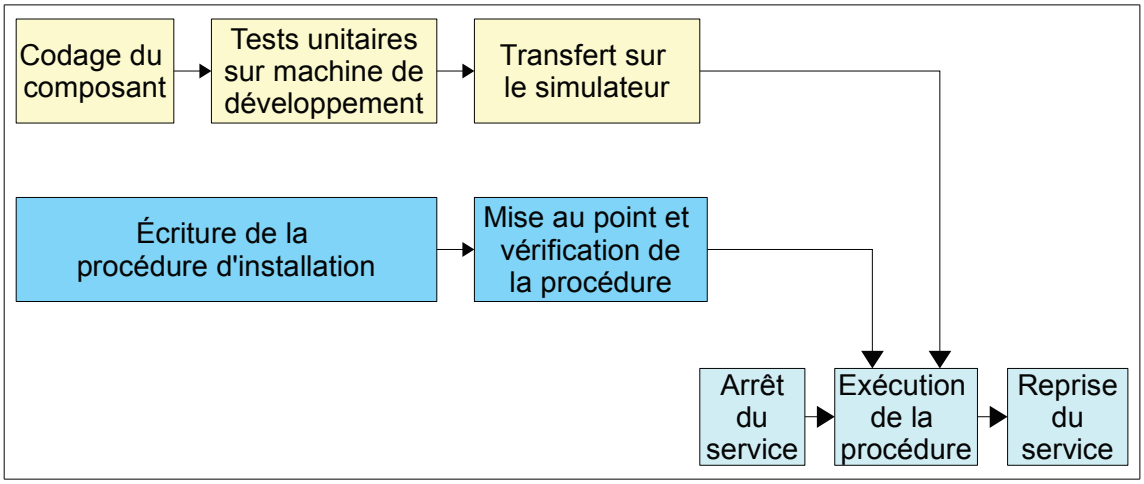


Figure 3.17: Étapes d'une installation partielle

La check-list indique les commandes à exécuter, pour réduire les erreurs on utilise des copier-coller de la check-list dans l'invite de commande. En général ou y trouve :

- des préparations « non intrusives » : téléchargements, préparation de fichiers de configuration... qui n'ont pas d'effet sur le service
- un arrêt du service : la création de session renvoie sur une page d'avertissement de maintenance, et les sessions en cours sont maintenues pendant quinze minutes
- un arrêt total du service et du SGBD
- des manipulations qui dépendent de ce qu'on installe
- un redémarrage du service
- une sauvegarde de la nouvelle configuration
- une liste de vérifications qui s'étalent sur plusieurs jours, pour s'assurer de la possibilité de redémarrer l'ordinateur, de la rotation des journaux, des sauvegardes, du bon fonctionnement général.

L'Annexe H : Exemple de checklist d'installation montre les opérations prévues pour la migration de PostgreSQL 8.1 à la version 8.4. Selon leur complexité ces installations prennent de 30 minutes à deux heures. Elles sont faites à une heure de faible utilisation pour réduire encore la gêne occasionnée.

3.4.3. La documentation

La documentation de Métaguide se répartit en quatre documents.

- installation et maintenance de l'infrastructure : « Serveur transwork »
- conception technique : « Documentation du programmeur Métaguide »
- mode d'emploi de la gestion des thésaurus : « Documentation de l'administration »
- mode d'emploi des tableaux de bord utilisateurs : « Documentation de la gestion »

Trois de ces documents sont mis à jour du fait de la recherche libre. Les notes d'installation mentionnent la mise à jour de PostgreSQL et la migration due au changement des paramètres régionaux. Le mode d'emploi des utilisateurs, qui expliquait la structure d'un lien vers Métaguide, est complété pour mentionner les nouvelles possibilités. La documentation technique donne une description succincte des nouvelles tables.

Issu de ce projet, un nouveau document important qui manquait cruellement apparaît. L'étude de l'existant a permis de reconstituer d'une part un diagramme fonctionnel grossier, d'autre part le schéma entités-relations conjoint des fichiers GLADIA et des tables de la base de données. Elle permettra aux développeurs futurs d'appréhender beaucoup plus vite la structure des données. Enfin le cahier des charges et de conception détaillée est conservé pour référence avec les autres manuels.

3.5. Les autres travaux

En marge de cette étude, le maître d'ouvrage a demandé plusieurs travaux annexes de maintenance ou des évolutions mineures. D'autres m'ont paru indispensables quoique hors projet stricto sensu.

Les extensions de fonctionnalités ont concerné d'une part, la possibilité de prédéfinir les contenus à la saisie d'une fiche, lorsqu'on connaît le contexte de l'usager par le lien qu'il a utilisé. Il s'agissait de définir un complément de syntaxe du lien d'appel à Métaguide, de l'analyser et de simuler la saisie des contenus appropriés. D'autre part, on souhaitait pouvoir supprimer un contenant dans un thésaurus alors que des fiches font référence à un contenu de ce contenant. Le script de fusion de thésaurus, déjà complexe par tous les traitements réalisés pour adapter les fiches aux modifications de leur thésaurus, a dû être complété pour cela.

Les corrections ont touché principalement le code PHP de Métaguide, qui implémente les tableaux de bord et l'insertion de photos dans les fiches ; une récente migration de PHP version 4 vers la version 5 avait laissé quelques incompatibilités dues à des changements subtils de sémantique des variables globales de PHP.

Par ailleurs j'ai proposé et réalisé la révision du processus de sauvegarde du SGBD, dont j'ai observé qu'il sauvegardait directement les fichiers de la base sans tenir compte de possibles effets de cache dans la mémoire centrale du processus SGBD ; il y avait donc un risque important d'erreur à la restauration, bien que la sauvegarde soit faite tard dans la nuit. Le script de sauvegarde a été repensé en suivant les recommandations des concepteurs de PostgreSQL, grâce à l'utilitaire fourni. Enfin, le processus de compilation de Métaguide a été simplifié pour éviter des imperfections qui obligeaient à des reprises manuelles.

Ces modifications participent à une meilleure fiabilité du système et augmentent ses possibilités, mais ont retardé d'autant le projet, en général de quelques heures seulement, mais parfois plus longtemps quand il s'agissait de traquer des défauts de portage de PHP ou des extensions de fonctionnalités.

3.6. Conclusion

Le composant de recherche libre concerne toutes les catégories d'utilisateurs de Métaguide. A l'usager ordinaire, il permet de fournir des mots-clés pour orienter sa recherche sans fouiller dans les listes d'attributs proposés, et ceci sans perdre la fonction de guidage. L'éditeur du site utilisateur peut choisir d'ajouter cette fonction ou pas selon les caractéristiques de son service. Le producteur et son documentaliste disposent d'un retour d'information pour juger de la qualité des libellés de contenus dans les thésaurus qu'ils gèrent. L'administrateur peut créer un nouveau type de client, le site appelant, pour offrir un service de délégation de recherche.

Pour supporter ces besoins multiples, d'importantes extensions apparaissent dans le schéma de la base de donnée de Métaguide. Cinq tables ont été ajoutées, elles sont liées aux tables existantes et entre elles par plusieurs clés étrangères. Des procédures stockées et de nouveaux triggers SQL aident à maintenir la cohérence de l'ensemble. L'alternative qui consistait à faire les recherches directement dans les fichiers GLADIA n'a pas été retenue dans une optique d'évolution à long terme. Mais ce choix entraîne une certaine duplication des données entre les deux systèmes de gestion de fichiers. Actuellement les données dupliquées ne servent qu'à la recherche libre, mais on peut concevoir le jour où la version GLADIA sera abandonnée pour simplifier l'ensemble du système.

Une fois les données correctement structurées, leur utilisation pour la recherche libre repose essentiellement sur les capacités du gestionnaire de bases de données PostgreSQL. Quelques interfaces simples sont ajoutées pour intégrer la recherche libre et la recherche guidée, et gérer les nouveaux URL de requêtes venant du Web.

Par contre, l'indexation des mots-clés doit prendre en compte leurs multiples origines. Les textes issus des thésaurus sont récupérés lors de leur fusion. Pour les textes des fiches, il a fallu réorganiser les multiples cas où on les manipule pour centraliser le code avant de le faire évoluer ; cette stratégie simplifiant les tests et la maintenance à plus long terme. Finalement, au moment de la mise en service du composant, un programme d'installation à usage unique analyse l'ensemble de toutes les fiches dans tous les thésaurus du système, pour charger les nouvelles tables.

La boîte à outils de Métaguide qui a servi de socle à cette réalisation comporte en particulier un gestionnaire de listes dynamiques, les listes Osmose. Elles sont le pilier de nombreux autres composants, de la communication Web avec la manipulation de données HTML et HTTP, à la gestion de fichiers avec des sur-couches d'accès au SGBD et aux fichiers GLADIA, en passant par la colorisation de cartes et l'interface au moteur de recherche guidée.

Pour supporter la fonction de « full text search », équivalent PostgreSQL de la recherche libre, le SGBD doit être mis à jour et la base Métaguide doit migrer pour tenir compte des paramètres régionaux. La version utilisée précédemment était bloquée à un niveau périmé depuis longtemps pour éviter les incompatibilités provoquées par les mises à jour. Ce projet a fourni une bonne occasion de passer à une version officiellement supportée, au prix d'un travail minutieux d'examen des sources de Métaguide à la lumière des notes de version de toutes les versions intermédiaires de PostgreSQL.

L'installation sur le serveur Web de la recherche libre et les évolutions du SGBD ont été découpés en tâches plus simples pour diminuer la prise de risque à chaque étape. Ces tâches ont été réalisées au fur et à mesure de l'avancée des études, sans attendre l'aboutissement du projet quand c'était possible, pour éviter une longue coupure du service. A l'issue des installations la recherche libre est exploitable sans incidence sur le fonctionnement de l'existant.

Nous allons maintenant voir l'organisation qui a permis de parvenir sans heurts à ce résultat.

4. La conduite du projet

Le projet s'insère dans le cycle de vie du produit Métaguide comme une phase d'évolution d'un système déjà utilisé. Cette évolution couvre toutes les phases d'un projet logiciel, du cahier des charges à la mise en exploitation. Aucune méthodologie n'étant définie ou exigée, il fallait s'imposer des règles de travail raisonnables, en évitant une rigidité qui eut été sans rapport avec l'envergure du projet comme de l'entreprise.

Pour la méthodologie générale il a été décidé d'utiliser le modèle du cycle en V, bien connu des participants. Une liste des tâches à réaliser, évoluant au jour le jour, a permis de maintenir le cap sur les objectifs. Un point informel avec la maîtrise d'ouvrage a eu lieu à chaque étape saillante de la conception et du déploiement : validation des spécifications et de la solution retenue, mises en place partielles, mise en place finale. Le planning établi initialement a évolué considérablement, cependant le délai global a été respecté.

Après de longues discussions avec le maître d'ouvrage pour bien préciser le cahier des charges, un premier document de conception générale de la solution a été rédigé et approuvé par le MOA, puis chaque fonction a été précisée dans la mesure nécessaire à sa réalisation. Il en a découlé cinq travaux partiellement parallélisables, deux concernaient l'amélioration du système existant (mise à niveau du SGBD et regroupement de fonctions de mise à jour des fichiers), une concernait la création et le maintien des nouveaux index, les deux dernières avaient pour but d'exploiter ces index pour les usagers historiques d'une part, pour les nouveaux « sites appelants » d'autre part.

Les principaux risques liés au projet étaient la mauvaise définition des objectifs et l'introduction d'aléas de fonctionnements du serveur en place. Parmi les risques de moindre importance on a aussi noté la qualité de conservation du code source des programmes. Bien que cela ait été envisagé il n'a pas été possible de mettre en place un système de gestion du code source, et nous sommes restés avec un archivage très basique mais exposé aux erreurs de manipulation.

4.1. L'organisation du projet

La recherche libre s'inscrit comme un complément d'un logiciel déjà en activité. L'organisation du projet doit considérer deux aspects : l'étude de l'extension proprement dite, et sa mise en œuvre sur le serveur Métaguide.

Comme il n'existe pas dans l'entreprise de recommandations particulières pour l'organisation des projets, il était nécessaire de choisir une ligne directrice. Lors des premières discussions sur le contour de l'étude, il s'est avéré que les besoins étaient très flous pour un volume de travail prévisible important. Dans ces conditions, une méthode par maquettes successives, comme dans les méthodes agiles, a paru propre à dériver trop aisément. J'ai proposé de suivre la démarche du cycle en V, pour pousser les participants à définir rigoureusement les objectifs. Le fait que d'avoir déjà pratiqué cette démarche était aussi un « plus » important dans cet environnement peu directif.

Après un travail de définition du besoin et de recherche de solution, le projet s'est organisé en tâches relativement indépendantes et le cycle en V s'est démultiplié en plusieurs V superposés, laissant le choix, soit de mener les étapes similaires de front, soit de terminer une tâche après l'autre, soit de déléguer des tâches.

La figure de la page suivante montre le découpage en sous-projets et en tâches et leurs interdépendances. La conception de la solution globale débouche sur cinq études plus spécialisées :

- Étude de la mise à jour du SGBD et des évolutions nécessaires dans Métaguide pour la supporter (en bleu foncé) ;
- Étude des modifications à apporter au schéma de la base de données pour supporter la recherche libre (en bleu clair) ;
- Étude des modifications à apporter à Métaguide pour enregistrer les informations nécessaires dans les nouvelles tables de la base (en turquoise)
- Étude de la mise en œuvre de la recherche libre dans ses aspects interface d'interrogation, recherche, traitement des résultats (en jaune) ;
- Étude du traitement des sites appelants dans ses aspects interface d'interrogation, restitution des résultats, administration (en orange).

Entre ces sous-études il existe des dépendances fortes. Le schéma de la base ne peut être fixé qu'après avoir complètement défini comment sont créées et utilisées les tables d'index de la recherche libre, et comment les « sites appelants » seront administrés. La définition des données et index utiles à la recherche (Conception R.L.) précède leur entretien (Conception Indexation). Certains tests ne peuvent être menés correctement que si les données correspondantes sont en place, de même pour l'installation définitive.

Bien que la parallélisation des tâche ait été possible, elle n'a pas eu lieu et tout a été réalisé séquentiellement. Pour un travail efficient, chaque sous-étude a été menée tour à tour jusqu'à être arrêtée par une dépendance.

Finalement, chaque étape d'installation a donné lieu à un plan de transition du système en exploitation, comme exposé à la section 3.4.

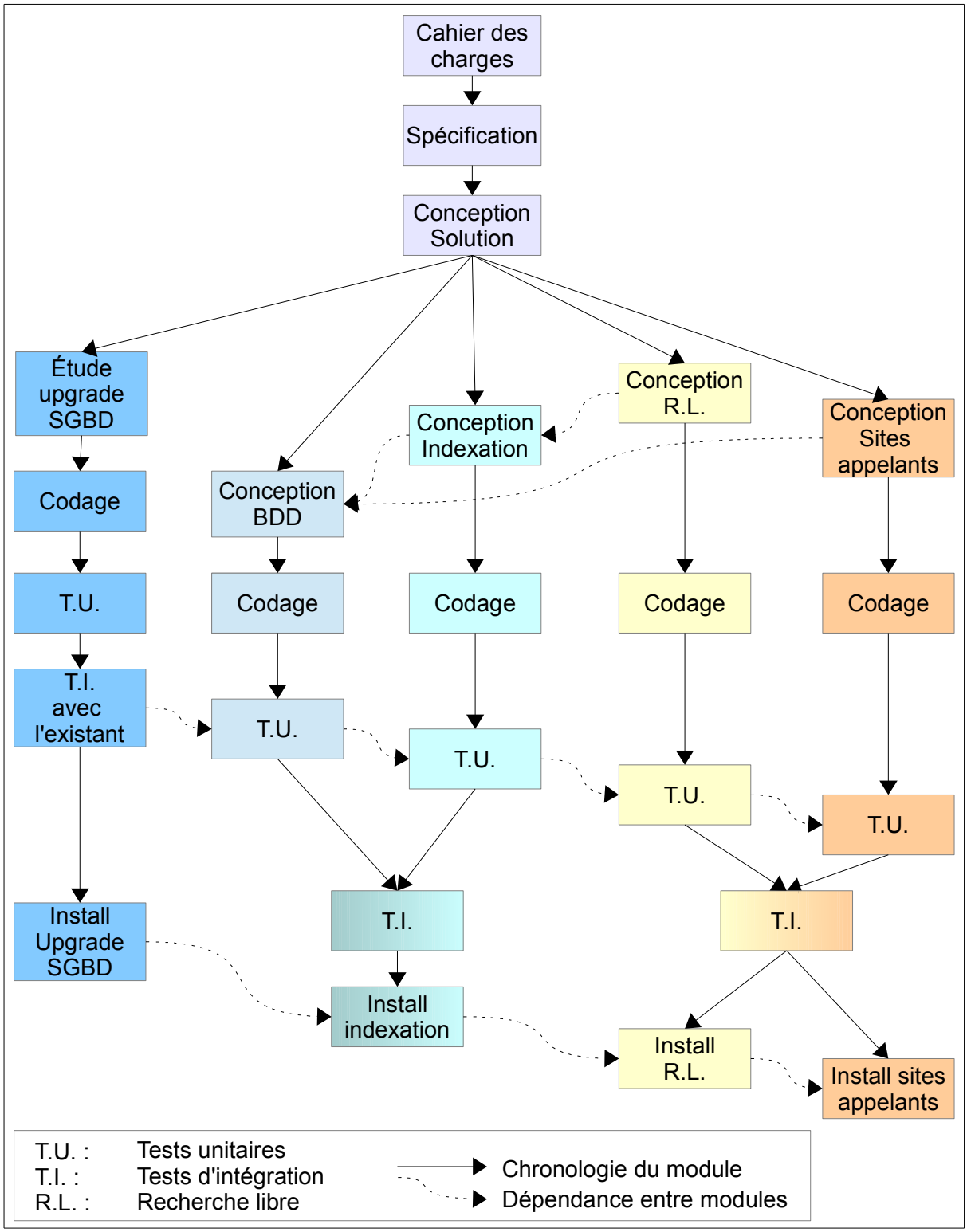


Figure 4.1: Organigramme des tâches du projet

4.2. Le planning

La première estimation de temps, établie à partir de l'ébauche d'une solution qui n'a finalement pas été retenue, prévoit une réalisation linéaire sur 31 semaines, et une mise en service unique sans mise à niveau du SGBD.

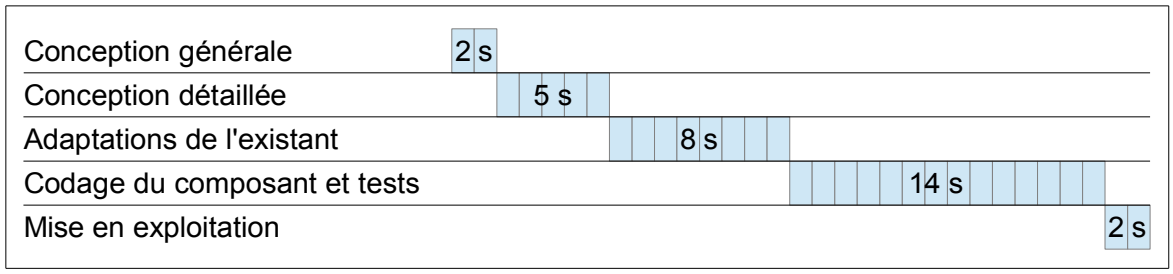


Figure 4.2: Planning initial

Le démarrage du projet au 1er octobre 2011 permettait d'anticiper une disponibilité début juin 2012 pour une personne seule à plein temps.

Après une phase de conception générale beaucoup plus longue que prévu, le planning a été profondément modifié en fonction des tâches effectives à mener. La définition précise des besoin a permis de recadrer les demandes en identifiant ce qui était intrinsèque au projet, ce qui pouvait constituer un projet accessoire séparé et ce qui n'était pas utile par rapport aux objectifs. La révision du périmètre du projet d'une part, l'utilisation d'outils existants d'autre part (la base de données), ont permis de réduire un peu la durée de réalisation prévue.

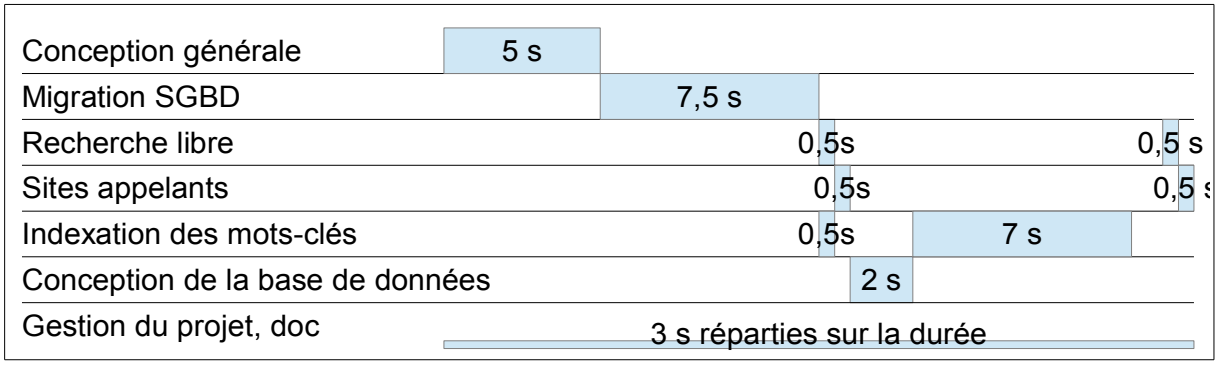


Figure 4.3: Planning réalisé, sur la base d'un plein temps

Le système était complètement opérationnel le 29 septembre 2012, le délai supplémentaire s'expliquant par le travail à temps partiel. Après la mise en service ont suivi quelques travaux mineurs de vérification et d'optimisation.

4.3. La gestion des risques

Nous n'avons pas mis en place de gestion formelle des risques. l'analyse à postériori fait ressortir certains risques non négligeables, comme le montre le tableau suivant.

Aléa	Vraisemblance de réalisation	Gravité (pour le projet)	Gravité (pour l'entreprise)	Note de risque
imprécision du besoin	4	4	3	16
indisponibilité prolongée du personnel	1	4	2	4
destruction du code développé	2	4	2	8
outils logiciels inadaptés	3	3	2	9
performances résultantes insatisfaisantes	3	3	3	9
dépassement significatif des délais prévus	2	3	2	6
indisponibilité prolongée du serveur	2	2	4	8
anomalies de fonctionnement du serveur dues à la mise en service du composant	3	3	4	12

La vraisemblance de réalisation de l'aléa est notée de 1 à 5 selon le barème :

1 : très improbable ; 2 : improbable ; 3 : vraisemblable ; 4 : probable ; 5 : très probable.

Les degrés de gravité sont notés de 1 à 4 selon le barème :

1 : insignifiant ; 2 : gênant ; 3 : action corrective nécessaire ; 4 : action immédiate nécessaire

La note de risque, de 1 à 20, est le maximum des deux produits *(vraisemblance de réalisation x gravité pour le projet)* et *(probabilité de réalisation x gravité pour l'entreprise)*

Le projet était utile mais pas vital pour l'entreprise, d'où une note un peu basse en cas d'échec. Les principaux risques étaient donc l'imprécision du besoin et l'éventualité que le nouveau composant induise des anomalies de fonctionnement accessoires.

Ces différents risques ont été pris en compte par des mesures appropriées ; certaines relèvent d'ailleurs simplement des bonnes pratiques.

Aléa	Mesures de prévention
Imprécision du besoin	Bien préciser chaque besoin, vérifier la cohérence interne de l'ensemble et la conformité aux objectifs
Indisponibilité prolongée du personnel	Le MOA est informé de l'avancement, l'étude procède en plusieurs étapes indépendantes
Destruction du code développé	Faire des sauvegardes régulières sur un site éloigné du lieu de développement
Outils logiciels inadaptés	Tests préalables des capacités de l'outil envisagé
Performances résultantes insatisfaisantes	Tests préalables des capacités de l'outil envisagé
Dépassement significatif des délais prévus	Les fonctionnalités indépendantes, accessoires ou imprécises sont exclues du périmètre du projet
Indisponibilité prolongée du serveur	Préparer les opérations de mise en service
Anomalies de fonctionnement du serveur dues à la mise en service du composant	Des tests complets sont réalisés sur une plateforme de simulation avant la mise en service

4.4. Le suivi et le reporting

Ce projet possède de multiples facettes et il était important à la fois d'avoir une vue d'ensemble et de ne pas oublier les détails. L'équipe étant réduite, cette longue étude présentait le risque de surestimer notre capacité à maîtriser l'ensemble et d'oublier ainsi des détails au fil de l'avancement.

Pour éviter cela, le principal outil utilisé a été une « liste des tâches à réaliser » constamment tenue à jour. Initialement limitée à quelques items concernant la rédaction des spécifications et la recherche de solutions, elle s'est développée au fur et à mesure que les détails étaient abordés. Dans une seconde partie du même document, un journal de bord précisait ce qui était fait pour chaque tâche, quand, et des notes de travail quand elles étaient utiles à la compréhension des choix faits.

Le second document de travail essentiel était la spécification. Par commodité, le même document a été utilisé pour contenir dans différentes sections, l'expression du besoin, la description de la solution retenue, la conception générale et détaillée. Ce document a évolué au fil du projet pour atteindre une quarantaine de pages.

Les documents externes, destinés aux différents acteurs (responsable d'exploitation, administrateur, producteur, éditeur, informaticien) ont été mis à jour à chaque avancée, sans attendre la fin du projet. Il en est résulté des versions de documents plus nombreuses mais cela réduisait le risque d'oublis.

Le MOA a été consulté à chaque étape importante, pour valider la vision technique des spécifications, accepter la solution retenue, et pour le planning de mise en exploitation. Il vérifiait lui-même si le développement était acceptable en procédant à des tests du point de vue utilisateur final. Il n'y a donc pas eu de recette formelle.

4.5. La gestion des sources

Le code source de Métaguide est constitué d'environ 3400 fichiers source. Le développement du module de recherche a modifié de nombreux fichiers source existants.

En l'absence d'un outil global de suivi des versions, le suivi est manuel : les programmeurs doivent annoncer sur quelle partie du code ils travaillent pour éviter d'utiliser des parties en cours de modification. Une sauvegarde quotidienne sur une clé USB est complétée par une sauvegarde avant et après chaque groupe de modifications. Ces moyens ne sont pas très fiables puisqu'elles dépendent de la rigueur des intervenants.

Ces moyens ont été conservés pendant le projet mais une impulsion a été donnée pour mettre en place un système de contrôle de versions. Une personne a commencé à examiner les outils Open Source les plus répandus : GIT, SubVersioN, CVS. Les grandes contraintes dans ce choix sont :

- facilité d'apprentissage
- utilisation décentralisée (les programmeurs travaillent souvent de chez eux)
- accès par Internet suffisamment sécurisé (risque de divulgation de la technologie, risque de destruction ou d'installation d'un cheval de Troie)
- facilité d'administration (l'équipe est trop petite pour se disperser)

Aucun choix n'était encore arrêté à l'issue du projet de recherche libre.

4.6. Conclusion

La gestion du projet s'est appuyée sur le concept du cycle en V. Après une étude poussée du besoin et des différents moyens d'y répondre, la solution retenue était constituée de cinq sous-ensembles relativement indépendants.

Le délai initialement prévu a été tenu, bien que le planning ait été complètement revu après quelques semaines, pour s'adapter à la solution. Bien que la communication avec le MOA ait été informelle, les jalons qui nécessitaient son feu vert, choix fonctionnels et mises en service, ont été respectés.

Le projet a été géré sans outils spécialisés. Un simple traitement de texte a servi à maintenir une liste de tâches et un journal d'avancement. La gestion des risques n'a pas été perçue comme utile par le MOA, et une rigueur dans le travail a suffi pour éviter les deux gros écueils identifiés : se disperser dans un besoin mal cerné, et introduire des dysfonctionnements du serveur en exploitation. La gestion des versions du code source est resté très artisanal quoique un effort soit en cours pour utiliser un outil.

Nous allons maintenant examiner quelles leçons on peut tirer de cette étude et de son déroulement.

5. Les résultats obtenus

On peut évaluer ce projet technique de différents points de vue. Concernant le produit lui-même, on peut s'intéresser non seulement à ses fonctionnalités mais aussi à ses caractéristiques non fonctionnelles. On peut aussi examiner le point de vue des différents acteurs : les utilisateurs finaux bien sûr, mais aussi les administrateurs exploitants, le maître d'ouvrage et enfin, le réalisateur, moi-même.

Les objectifs fonctionnels sont remplis : on dispose de deux modes d'accès à une recherche libre, et de données de surveillance, dont les résultats sont conformes à l'attente du MOA. Son critère de performance, le temps de réponse de 2 secondes pour la première page (entendre, l'ouverture de la session) est respecté, bien qu'il ait fallu un round supplémentaire d'optimisations pour l'atteindre. Ces buts ont été atteints sans qu'il soit nécessaire de compliquer l'administration et l'exploitation du site. Ainsi les utilisateurs, et parmi eux en premier lieu la fonction commerciale et marketing, peut s'appuyer sur un système de recherche par mots-clés semblable aux moteurs de recherche classiques du Web, imbriqué avec une recherche guidée thématique, ce qui en fait un produit unique.

Outre l'accroissement de ses qualités visibles, Métaguide a profité d'une mise à niveau conséquente de son SGBD, d'une impulsion pour la mise en place d'une gestion de versions du code source, et diverses améliorations internes. Ces mises en place permettront par la suite d'améliorer le système de recherche soit intrinsèquement en ajoutant la gestion de mots synonymes ou en jouant sur l'origine des mots (thésaurus, fiche...) ; soit à l'extérieur en facilitant la saisie par un correcteur orthographique ou une reconnaissance d'expressions.

A l'inverse nous regrettons que la fonction multi-langue n'ait pas pu être complètement conservée, bien que des précautions soient prises pour l'implémenter facilement lorsque le besoin aura été éclairci de façon cohérente. Également, les tables de surveillance et de gestion des sites appelants auraient mérité une interface graphique plus conviviale.

L'attente est grande sur l'exploitation commerciale du module de recherche libre car l'entreprise n'a jamais vraiment décollé. Pour cette raison le projet est motivant au-delà de ses aspects techniques, bien que l'absence d'encadrement sérieux et de collègues avec qui confronter les idées, ait réduit l'enrichissement potentiel que j'aurais pu en tirer. Cependant, à titre personnel je peux considérer que le projet est une réussite et que la formation du CNAM m'a aidé à orienter ma pensée dans bien des aspects de sa réalisation.

5.1. Les objectifs fonctionnels et les aspects linguistiques

Le cahier des charges expose deux besoins principaux : la recherche libre comme première étape de la recherche existante, et la recherche libre comme moyen d'accès à Métaguide depuis un autre site partenaire. De plus quelques informations de contrôle sont destinées à l'administrateur.

Ces trois grandes fonctions ont été implémentées avec succès. Les modes d'accès, les résultats de recherche, les croisements de critères donnent les résultats attendus par le MOA.

Dans le détail, l'étude préalable effectuée sur les aspects d'analyse phonétique et sémantique, de contrôle orthographique et grammatical, de support multi-langue a accouché d'une souris : nous avons examiné différents projets open-source impliqués dans l'analyse du langage (traitements de texte LibreOffice et AbiWord, SGBD PostgreSQL, navigateur Mozilla), mais il s'est avéré que d'une part leurs traitements linguistiques sont orientés vers la correction et non la recherche, et que d'autre part ils tendent à utiliser les mêmes moyens, en particulier le correcteur orthographique Hunspell[11] et le « stemmer snowball » qui calcule les radicaux des mots. Ces algorithmes supportent de nombreuses langues. Comme ils sont aussi implémentés dans PostgreSQL, nous avons utilisé directement ses possibilités. Pour diverses raisons à la fois pratiques et conceptuelles, l'étude de faisabilité a écarté les autres fonctions linguistiques. Examinons ces raisons.

a. Le support multi-langue

Les textes à indexer ne sont pas tous disponibles dans toutes les langues souhaitées. Bien que les textes qui constituent les thésaurus soient traduisibles – et traduits – en plusieurs langues, il paraît difficile d'imposer à l'usager qui saisit une fiche, disons par exemple une petite annonce, de saisir en plusieurs langues son descriptif libre. De même, Métaguide ne prévoit rien pour traduire les coordonnées du contact associé à la fiche.

Un système multi-langues complet doit aussi tenir compte d'un jeu de caractères très étendu que seul permet le codage UNICODE, un codage multi-octets. Or, aujourd'hui Métaguide ne supporte que des codages à raison d'un octet par caractère. Même si la recherche libre était étendue à plusieurs langues, cette contrainte continuerait à limiter le système aux langues occidentales.

L'objectif multi-langue initial a donc dû être explicitement écarté comme demandant beaucoup de travaux annexes. C'est regrettable car c'est une fonction remarquable de Métaguide : dans un fichier paramétrable, un système de recherche en plusieurs langues sans besoin de traduire les fiches ! La fonctionnalité est conservée mais uniquement pour l'étape guidée. L'étape de recherche libre ne fonctionne que dans une langue. La recherche sur un fichier peut donc s'effectuer soit guidée en plusieurs langues, soit libre puis guidée en une seule langue.

Cependant les tables ont été conçues en tenant compte de cette fonctionnalité future. Le code langue est un index dans toutes les tables appropriées, et le SGBD PostgreSQL supporte la recherche de texte libre sans modifications pour un grand nombre de langues. Aujourd'hui la recherche libre de Métaguide est, disons, « mutli-langue alternatif » : il peut fonctionner en n'importe quelle langue occidentale, mais le choix est un paramètre de configuration fixé une fois pour toutes.

[11] http://hunspell.sourceforge.net/, http://en.wikipedia.org/wiki/Hunspell

b. La grammaire

La recherche libre intégrée à PostgreSQL se sert de plusieurs mécanismes qui améliorent considérablement les comparaisons. D'une part elle élimine les « mots faibles », ceux qui n'ont qu'une fonction grammaticale ou qui sont très répandus : pronoms, déterminants, prépositions (le , la, je, dans, ou, en...). D'autre part elle extrait les radicaux des mots avant de les comparer.

Le premier mécanisme élimine le « bruit », les correspondances fortuites sans signification pour une recherche. Le second mécanisme permet repérer l'égalité entre des singuliers et des pluriels, entre le même verbe conjugué de différentes façons, etc. Il ne s'agit pas de radicaux au sens ou l'entendrait un grammairien car il existe deux façons d'aborder ce problème : avec un dictionnaire, au risque de ne pas avoir tous les mots nécessaires, ou avec un algorithme de transformation, au risque de générer des radicaux grammaticalement faux. Comme il s'agit de comparer des radicaux entre eux, et non de s'en servir pour créer un texte, la seconde méthode est retenue dans l'algorithme multi-langue « snowball stemmer » de PostgreSQL.

Il en résulte des comparaisons pleinement significatives du point de vue de l'objectif fixé par le MOA.

c. La correction orthographique

La variété d'erreurs de saisie que peut produire l'usager est un problème épineux. L'objectif est de pouvoir trouver des égalités entre ce qu'a saisi le créateur de l'information – documentaliste composant le thésaurus ou usager insérant une fiche – et ce qu'a saisi l'usager qui effectue une recherche. Les erreurs dites orthographiques sont généralement des erreurs de transcription phonétique mais aussi des fautes de frappe (qui changent la lecture du mot). On peut proposer des corrections à l'usager, mais on ne peut pas corriger automatiquement, entre autres parce que plusieurs corrections sont généralement possibles. La correction orthographique s'analyse donc comme une fonction d'aide à la saisie. Il faut noter que les logiciels du marché sont en général incapables de reconnaître un mot dans lequel il y a plusieurs erreurs, ce qui est particulièrement gênant dans les langues à accents, ou l'usager peut par exemple omettre plusieurs accents dans le même mot.

Sans être écartée des objectifs du MOA, la correction orthographique a été reconnue comme un projet séparé, à implémenter dans l'interface graphique (le codage des pages Web) plutôt que dans le moteur de recherche.

d. La synonymie

L'usager peut utiliser des termes qui n'existent pas dans le thésaurus ou les données, mais qui ont pourtant un sens, simplement il n'utilise pas les mots exacts. Une première version du cahier des charges prévoyait de rechercher les mots tapés dans un dictionnaire de synonymes, puis de chercher les synonymes de la même façon que les mots initialement saisis.

La difficulté de ce mécanisme est de définir les poids à attribuer aux réponses selon qu'elles sont obtenues par des mots saisis, par des synonymes ou par un mix des deux. Imaginons que l'utilisateur d'un thésaurus de petits commerces tape quatre mots :

recherche de : resto viande bon marché

On pourrait trouver « viande + marché » → l'étal d'une boucherie en plein air (2 mots tapés) ou bien « restaurant (par synonymie de 'resto') + viande + qualité (par synonymie de 'bon') »

→ un restaurant spécialisé (1 mot tapé + 2 synonymes). Faut-il faire passer en priorité les mots tapés bien qu'ils soient moins nombreux ? Ou se guider sur le nombre de mots reconnus ? Selon les cas la réponse pourrait être différente, et nécessiterait peut être un traitement sémantique [6][7] (quel point commun entre ces quatre mots ? Ou trois par trois ? Certains groupes de mots forment-ils une expression toute faite, un idiotisme ?)

Cette fonction très intéressante a été écartée pour deux raisons. D'une part elle gonflait de façon importante la complexité du projet, et donc son délai de réalisation. D'autre part elle supposait disponible un dictionnaire de synonymes qui n'a pas été fourni par le MOA avant les étapes de conception de la solution. On peut cependant regretter que les dictionnaires de synonymes compatibles avec la recherche de PostgreSQL n'ont pas été examinés.

e. Les « expressions particulières »

Le MOA, dans sa vision d'un service complet, souhaitait que dès la saisie dans la zone de recherche libre, des expressions type soient reconnues et aiguillent l'utilisateur vers d'autres services. Autrement dit si dans un service qui propose un fichier « Location de maisons et appartements » l'utilisateur tape « louer ma maison » on l'aiguille non pas sur un résultat de recherche de maison à louer, mais sur la saisie d'une nouvelle fiche pour insérer son offre de location dans le fichier.

L'intention est louable, mais à notre avis les expressions en question sont extrêmement difficiles à cerner et à exprimer. D'autre part cette fonctionnalité était clairement en dehors du projet de recherche libre, et elle a été sortie du projet bien qu'une spécification générale séparée ait été rédigée.

f. L'indexation des pages Web associées

Le créateur d'une fiche peut y placer l'URL d'une page Web, la page d'accueil de son entreprise par exemple, si le fichier est un annuaire professionnel. Initialement, le MOA souhaitait que les mots de cette page puissent être utilisés pour rechercher la fiche, de la même façon que le descriptif libre de celle-ci. Plusieurs considérations ont abouti à abandonner ce mécanisme :

- La page peut changer sans que Métaguide en soit averti, ce qui demanderait de développer en plus une extraction périodique d'une multitude de pages pour vérifier leur indexation ;
- Les pages HTML contiennent des balises, qu'il conviendrait d'analyser en profondeur pour repérer les textes intéressants à indexer (y compris les balises META et les attributs TITLE et ALT) ;
- En fait, il ne s'agit pas nécessairement de l'URL d'une page HTML, il faudrait développer un navigateur partiel pour reconnaître le type et le contenu du fichier... et on sait à quelle vitesse de nouveaux formats apparaissent ;
- Et toujours, le problème de la langue, puisque la fiche ne peut citer qu'un seul URL et pas les URL équivalents en d'autres langues. La page elle-même présente peut-être des liens vers ses traductions, mais les retrouver est hors de portée d'un système automatisé et général.

g. Les croisements de recherches

Le MOA a imaginé au départ un algorithme complexe pour prendre en compte les résultats de recherches comprenant plusieurs mots-clés, variant en fonction de l'endroit où

ils sont trouvés, des mots composés (tels « pont-levis », « reine-marguerite »), des noms propres ou communs, des expressions toutes faites (telles « syndicat d'initiative »), et de règles paramétrables de priorité. Ces règles étaient censées varier selon les cas, pour appliquer des ET ou des OU (des intersections ou des réunions des ensembles de fiches sélectionnées par chaque mot).

Plusieurs de ces critères posent des problèmes de définition précise, surtout dans un environnement multi-langue. Les tirets des mots composés n'ont pas forcément le même sens dans toutes les langues. Les noms propres sont souvent des noms communs (Monsieur Boulanger...). Les expressions toutes faites sont souvent abrégées quand on les saisit pour une recherche.

Nous avons pu arriver à une solution beaucoup plus simple : d'une part utiliser directement les fonctions de recherche intégrées PostgreSQL, d'autre part ressortir les fiches qui présentent le plus de mots communs avec la requête soumise. Ceci résout le problème des combinaisons : PostgreSQL dispose d'une fonction de « ranking » des résultats, qu'on peut configurer pour compter le nombre de mots trouvés en commun. Il suffit d'un ordre SQL avec un tri sur ce critère, pour récupérer en premier les fiches qui ont le plus de mots communs. C'est équivalent à faire une recherche (mot-clé1 ET mot-clé2 ET...) de tous les mots, puis de n'importe quel groupe de mots sauf un, puis n'importe quel groupe de mots sauf deux, etc.

Au bout du compte le croisement de recherches fonctionne comme espéré par le MOA, sans besoin d'une configuration complexe.

h. Le traitement des non-réponses

Lorsque la recherche directe des mots clés ne donne pas de résultat, on renvoie l'usager à une recherche guidée sur toutes les fiches. En utilisant le fait que les thésaurus sont arborescents, il aurait été possible de réaliser une étape intermédiaire plus sophistiquée. Les mots-clés auraient pu être recherchés uniquement dans les contenus du thésaurus (même s'ils ne correspondent à aucune fiche), puis de remonter progressivement dans l'arbre des contenants/contenus jusqu'à trouver un nœud qui corresponde à des fiches ; on obtient ainsi une recherche qui trouve les fiches ayant un « thème voisin ». Par exemple dans un fichier d'annonces de bateaux à voile :

recherche de : catamaran lorraine

Imaginons qu'il n'y ait pas d'annonces de catamarans en Lorraine ; on chercherait les nœuds supérieurs (France à la place de Lorraine, et Multi-coque à la place de Catamaran) pour trouver les trimarans lorrains et les catamarans Alsaciens.

Ce mécanisme pourra faire l'objet d'une extension assez simple en modifiant la table `cnacnu_fus_lang` pour lui ajouter un index de recherche libre.

i. L'interface d'administration

Deux fonctions ont une interaction avec l'administrateur du système : la configuration des sites appelants et l'affichage du journal des échecs de recherche. Nous nous sommes limités à nous reposer sur les capacités de l'administrateur à utiliser directement la base de données, grâce à « phppgadmin », l'outil général d'accès au SGBD par navigateur.

Bien que cette interface soit suffisante aujourd'hui, il aurait été plus convivial de créer des pages spécialisées dans les menus des « tableaux de bord » de l'administrateur et des producteurs. Cela aurait aussi réduit les risques d'erreur de manipulation de ce puissant outil.

5.2. Le bilan qualitatif

5.2.1. Les temps d'exécution

La seule exigence formulée en matière de performances était le « temps de réponse de la première page ». On souhaitait ne pas augmenter ce temps de réponse en passant d'une recherche guidée à une recherche libre.

Dans ce qui suit, les temps concernent toujours la première page de recherche avec un fichier étalon (des données sur l'agriculture et l'élevage). Ils sont mesurés sur le serveur en exploitation. Les paramètres influents sont le nombre de fiches F, la complexité du thésaurus T (le nombre de contenants/contenus), et la variété V des contenus mentionnés dans les fiches (mesurée en nombre de contenus affichés sur la première page). Le fichier étalon pour cette mesure contient 87984 fiches, le thésaurus contient 9779 contenus, dont 240 doivent être affichés sur la première page.

Le temps de traitement de la recherche guidée se décompose en une initialisation proportionnelle à F, et un classement par contenus proportionnel à (F x V). Le fichier étalon est traité en 800 ms environ, auquel il faut y ajouter le temps de construction de la page Web, sa transmission et son affichage par le navigateur (environ 600 ms).

Pour la recherche guidée, le MOA avait mesuré 1,5 secondes entre l'action de l'usager et l'affichage de la première page. Ce temps ne devait pas excéder 2 secondes dans le cas d'une recherche libre. Des tests préalables avec le SGBD avaient conclu que les requêtes SQL de recherche par mots-clés s'exécutaient en environ 0,5 à 1 seconde, ce qui permettait de rester dans les limites exigées en ajoutant un peu de traitement de mise en forme et le transfert au navigateur.

Dans un premier temps, lors de l'utilisation en réel (le serveur d'exploitation avec le réseau de l'hébergeur), nous avons constaté un temps de réponse plutôt meilleur sur une recherche libre que sur la première étape d'une recherche guidée, et le MOA a accepté le projet en l'état.

Mais dans la recherche libre, ce temps est variable selon le nombre de mots cherchés, leur fréquence dans la base et l'état de la mémoire cache du SGBD ; après quelques jours d'utilisation sur un fichier réel, nous avons isolé des cas particuliers où le temps de réponse pouvait monter à plus d'une minute, ce qui pouvait amener le navigateur à rompre la session avec le serveur. Ainsi dans la figure 5.1, les quatre dernières mesures ont été obtenues en recherchant successivement « professionnel », « vie », « ville », « france ». Le graphique dessine une parabole, croissant en $O(n^2)$.

Figure 5.1: Effet de l'optimisation

Pourquoi un tel délai ? Le code ajouté étant essentiellement du SQL, il était naturel de soupçonner le SGBD. Cependant, la même requête exécutée par des utilitaires fournis dans PostgreSQL ne durait que quelques secondes. La mesure de chaque étape du processus de recherche (figure 5.2) entre l'arrivée de la requête au serveur et la génération de la page Web, a montré que la cause première était la grande quantité de lignes de résultats renvoyés par le SGBD à Métaguide quand on recherchait des mots très courants.

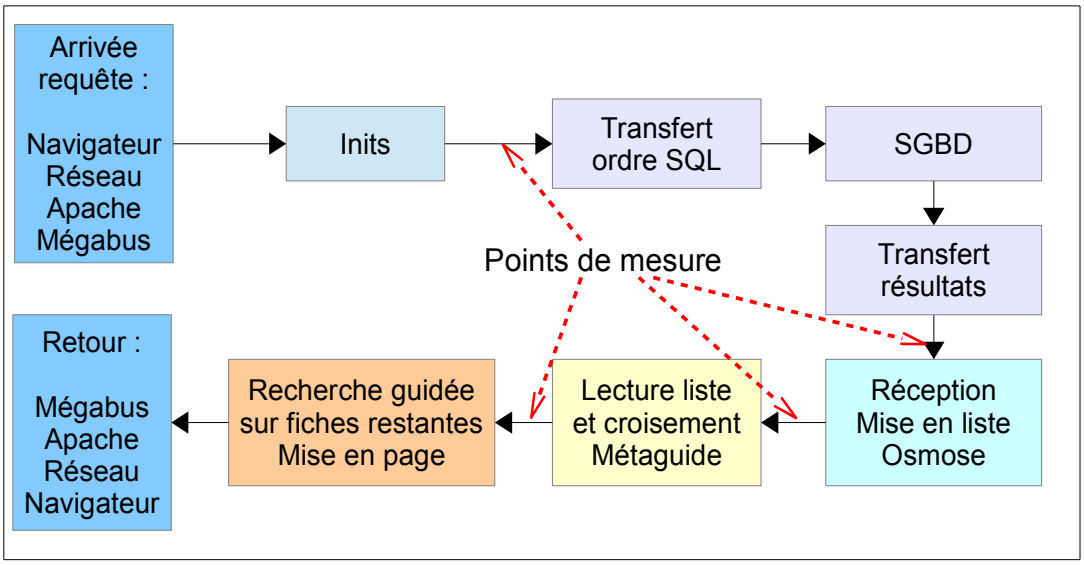


Figure 5.2: Étapes chronométrées pour l'optimisation

La cause n'était pas dans le SGBD, mais dans l'outil Osmose qui consomme l'essentiel du temps. Les données reçues du SGBD sont dans un premier temps transférées dans une liste Osmose, puis relues de cette liste pour croiser les réponses avec la « pile de fiches » que Métaguide maintient en mémoire centrale. Cette liste chaînée est donc parcourue séquentiellement deux fois, d'où le délai important sur les mots qui apparaissent dans beaucoup de fiches. Une première solution était de sauter l'utilisation de la liste, mais l'impact sur la modularité du logiciel aurait été gênant, avec l'implantation de code spécifique à PostgreSQL à plusieurs endroits du système.

Alors, pourquoi ces listes (chaînées, en mémoire) étaient-elles si lentes ? Un petit programme de test a permis de calculer le temps d'insertion de N items dans Osmose (figure 5.3, où l'on retrouve la parabole). Il apparaît que chaque accès à un item d'une liste chaînée nécessite de la parcourir en entier jusqu'à l'item. Lors de la réception des résultats du SGBD, on est dans le pire des cas puisque chaque résultat est inséré en fin de liste.

Figure 5.3: Test d'insertion dans Osmose (ms X items)

Une modification simple d'Osmose pour gérer un « pointeur courant » sur la liste et un pointeur de fin de liste pour les ajouts, a complètement supprimé le problème. Sur la Figure 5.1: Effet de l'optimisation, le second tracé montre que

le temps est désormais linéaire en fonction du nombre de fiches, avec un taux de croissance très faible, de l'ordre de 4 microsecondes par fiche. Sur le fichier étalon, le temps de réponse maximum de 2 secondes demandé n'est jamais dépassé.

La modification a été apportée sous la forme de nouvelles fonctions d'interface, pour ne pas remettre en cause le code existant. Les performances générales de Métaguide pourront probablement être améliorées en utilisant cette nouvelle interface partout où c'est possible, or, Osmose apparaît partout dans le code de Métaguide.

5.2.2. Le nombre d'utilisateurs

Si nous reprenons les mesures effectuées avant l'ajout du composant de recherche libre et les optimisations,

R(n) = (0,44 n + 0,5) s	temps de réponse pour n transactions arrivant en même temps

il semble que le goulet d'étranglement principal, le processeur, ne supporte que 1/0,44 = 2,3 transactions par seconde, ce qui est très peu.

En réalité cette formule prend en compte le temps maximal d'exécution d'une transaction, et non le temps moyen. Ce maximum constitue le critère de satisfaction du MOA, et c'est pourquoi nous nous sommes concentrés sur cette contrainte lors de l'étude. Cependant il est intéressant d'avoir une idée de la durée moyenne de transaction de façon à évaluer plus justement le nombre d'utilisateurs réellement supportés. Nous comparerons les résultats avant et après l'adjonction de la recherche libre optimisée.

a. Le temps de transaction moyen

Nous avons utilisé le fichier étalon pour étudier le temps de calcul de chaque page dans une session. Une session est typiquement constituée d'une page d'ouverture, de pages de recherche successives, d'une liste de réponses et de fiches finales. L'usager fait en général des allez-retours entre la liste et les fiches, plus rarement des allez-retours entre ses critères. Nous avons constaté que le temps de calcul d'une page liste ou fiche est presque constant, et que le temps de calcul des pages de recherche, à une constante près, décroît fortement à mesure que l'on affine les choix. Les temps ont été mesurés sur le serveur en exploitation. Voici par exemple les temps de calcul d'une session typique au début du projet (donc sans recherche libre) :

Page calculée	temps (ms)	fiches traitées
Première page	1 x 884	87984
2ème page de sélection	1 x 52	2732
3ème page de sélection	1 x 41	385
4ème page de sélection	1 x 37	48
5ème page de sélection	1 x 37	20
liste de fiches (4 passages)	4 x 173	6
fiches finales (4 passages)	4 x 12	1

Ce qui donne ici un temps de calcul par transaction de 138 ms en moyenne pondérée. Des tests sur une plus grande variété d'actions de l'utilisateur mènent à une moyenne de 110 ms.

L'optimisation du composant Osmose a réduit ce temps à 76 ms (92 ms pour l'exemple).

b. Le nombre d'utilisateurs au début du projet

Au début du projet, Métaguide travaille presque exclusivement en mémoire centrale. D'autre part le serveur est installé chez un fournisseur d'accès sérieux (OVH) et nous pouvons supposer que la bande passante du réseau n'est limitée que par notre carte réseau, Ethernet 100 Mbps. Enfin il faut tenir compte des délais de transfert à travers Internet et d'affichage, estimés à 500 ms par transaction, quelle que soit la charge.

L'affichage d'une page demande le transfert d'environ 50 ko, le réseau nous limite donc à environ $100x10^6 / 8 / 50^3 = 250$ transactions/s ; le processeur dual core nous limite à $2 / (110x10^{-3}) = 18$ transactions/s. Ce dernier est donc le goulet d'étranglement. On remarque qu'il est bien moins contraignant que ce que laissait supposer la prise en compte de la seule première page.

L'usager a un choix à faire sur chaque page, ce qui suppose qu'il en lise au moins une partie. Nous tablons sur un temps de réflexion de 5 à 15 secondes, avec une moyenne de 7 secondes en tenant compte des utilisateurs avertis. Nous pouvons estimer le nombre d'utilisateurs maximum en définissant deux systèmes dans une boucle d'interactions : un système « N terminaux » et un système « processeur » :

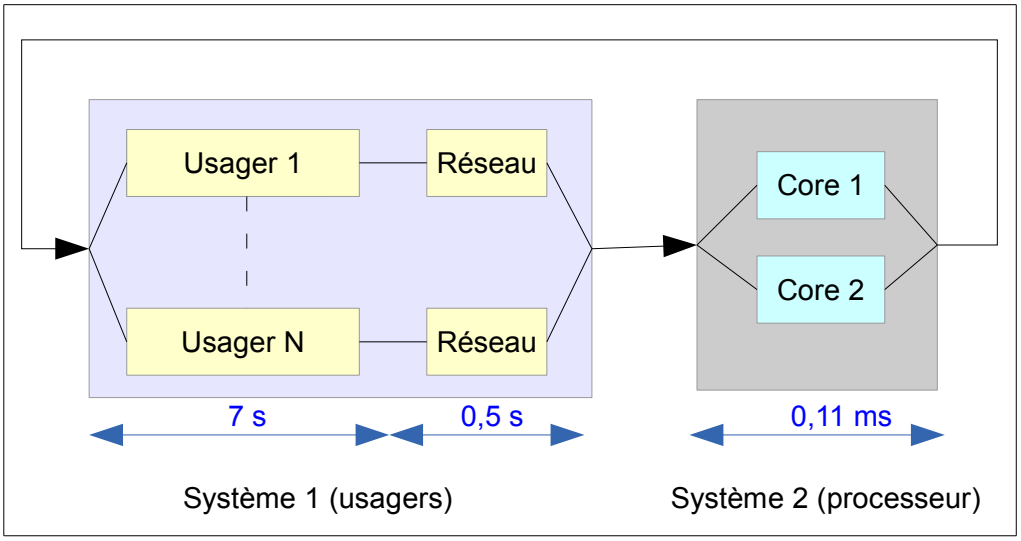


Figure 5.4: Délais dans l'interaction usagers/Métaguide

Pour saturer le processeur sans le surcharger, les débits des deux systèmes doivent être égaux entre eux et égaux au débit limite du processeur, soit $N / 7500 = 2 / 110 \rightarrow N = 136$ usagers au maximum. Mais il existe dans Métaguide une table de sessions de taille fixe, actuellement limitée à 64 sessions, c'est donc elle qui impose la limite actuelle, pratique, du système.

c. Le nombre d'utilisateurs à la fin du projet

En matière de temps de réponse, l'implémentation de la recherche libre a deux impacts sur le système.

– cas « historique » d'une recherche guidée seule : l'optimisation d'Osmose améliore légèrement les performances (l'utilisation extensive des optimisations améliorerait plus encore, en demandant un travail supplémentaire).

– cas « nouveau » d'une recherche libre complétée d'une recherche guidée ; la première page est calculée selon l'algorithme de recherche libre, puis dans les transactions suivantes (guidées) on profite aussi de l'optimisation d'Osmose.

Sur le cas « historique » qui se compare directement avec la situation en début de projet, le calcul donne N / 7500 = 2 / 92 → N = 163 utilisateurs simultanés au lieu de 136.

Sur le cas « nouveau », le problème est plus complexe. Il dépend maintenant des performances du SGBD, de son mode de fonctionnement et du nombre de correspondances trouvées dans la recherche libre. En particulier, l'effet de cache joue un rôle très important. Par exemple la recherche libre des termes très courants « jean bretagne » dure 1,8 secondes juste après le démarrage du SGBD, et 0,2 secondes les fois suivantes.

Utiliser la recherche libre fait sauter le calcul intégral de la 1ère page de recherche guidée, et économise beaucoup le temps moyen d'une transaction. Nous pouvons amorcer la recherche donnée plus haut en exemple, en effectuant une recherche libre en première étape, avec un mot-clé choisi pour donner les mêmes résultats :

Page calculée	temps (ms)	fiches traitées
Zone de recherche libre	0	0
1ème page après recherche libre	714	2732
2ème page de sélection (guidée)	41	385
3ème page de sélection	37	48
4ème page de sélection	37	20
liste de fiches (4 passages)	4x173	6
fiches finales (4 passages)	4x12	1

Nous avons un temps de calcul moyen par transaction de 81,5 ms à comparer avec 92 ms en recherche guidée pure. Cependant, dans le cas général, le temps passé dans le SGBD semble proportionnel au carré du nombre de fiches trouvées, ce qui pourrait devenir un problème avec des fichiers plus volumineux. D'autre part l'effet de cache est hors de portée du moyen de mesure utilisé, qui consiste à dater des parties choisies du code de Métaguide.

d. Les pistes d'amélioration

Dans tous les cas une condition préalable sera de faire sauter la limite interne de 64 sessions dans Métaguide. Ensuite, en remarquant que c'est l'ouverture de session qui contribue le plus à la durée moyenne des transactions, on pourra explorer les solutions suivantes :

– Améliorer l'algorithme d'ouverture de session et de calcul de la première page.

- Ouvrir plus de « guichets » de traitement de cette transaction, soit en dispersant le calcul sur plusieurs ordinateurs, soit en augmentant le nombre de processeurs ou de « cores » ; dans ce dernier cas il faudra tenir compte de la bande passante du bus processeur/mémoire centrale.
- Étudier l'impact de la mémoire cache du SGBD. Aujourd'hui le ratio (accès au cache) / (total des lectures) est de 0,75, ce qui signifie que 3 accès en lecture sur 4 sont cachés. Il est peut-être possible d'aller au-delà.
- Déporter la base de données vers un serveur dédié. Les accès à la base de données générés par la recherche libre seraient alors traités en parallèle avec les autres opérations de Métaguide. Il faudra alors comparer ce qu'on gagne en temps de calcul SGBD, avec ce qu'on perd en calcul de protocoles réseau.
- Réviser les algorithmes du composant Osmose et utiliser les optimisations dans tout le code. Cela aurait un effet sur toutes les transactions, pas seulement sur l'ouverture de session.

5.2.3. L'évolutivité

Métaguide est un progiciel destiné à des clients divers pour des applications variées. A ce titre, encore plus que les logiciels sur mesure, il est amené à intégrer de nouvelles fonctions. Bien que ce ne soit pas un point mis en avant dans le cahier des charges, l'évolutivité était sous-jacente lors de la conception, on pourrait dire qu'elle fait partie de la culture d'entreprise. Le projet de recherche libre en lui-même est une étape d'une vision plus vaste qui aboutira à une recherche guidée vocale.

Les évolutions futures ont été prises en compte à plusieurs niveaux. Lors de l'analyse du problème posé tout d'abord, certaines fonctions ont été écartées de la réalisation : la correction orthographique, la recherche par synonymie, les expressions particulières (voir X). Mais elles n'ont pas été ignorées pour autant : le document de spécification et conception les étudient en détail, les identifient comme des projets séparés, et annexent une spécification préliminaire.

La conception a orienté ses choix dans le sens de l'évolutivité. La version installée de la « recherche plein texte » de PostgreSQL, aurait été suffisante pour la recherche libre, mais pas compatible avec les versions plus récentes du SGBD. Nous avons fait le choix de consacrer beaucoup de temps à sa mise à niveau. Le but principal de l'opération était de faciliter les évolutions futures de son utilisation ; le gain immédiat était d'avoir une recherche libre intégrée (et non sous forme d'ajout) et indépendante du codage de caractères ; en prime nous avons gagné des correctifs de sécurité, des fonctionnalités, une meilleure compatibilité avec les standards, un support des mises à jour dans la distribution Linux.

Ensuite, le système de fichiers GLADIA, soutenu par le MOA, n'a pas été utilisé, au profit d'une conception « tout SGBD ». Le choix d'utiliser le module de « recherche plein texte » PostgreSQL a aussi été largement influencé par la possibilité de l'adapter dans le futur, en le configurant pour la synonymie.

Enfin la réorganisation d'une grande partie du code d'accès aux fiches simplifiera sa maintenance future ; dans l'esprit d'une disparition des fichiers GLADIA, les accès en langage C sont maintenant centralisés pour faciliter la migration complète vers la base de données.

Les futures évolutions ont donc été prises en compte, tant dans le détail pour celles qui étaient prévisibles, que dans l'organisation générale pour les nouveautés à venir.

5.3. Le point de vue des utilisateurs

Les producteurs et les usagers historiques vont utiliser le nouveau composant ; mais ce sont principalement l'exploitant et les services informatiques clients qui ont à le connaître plus en détail. Leurs préoccupations et attentes respectives sont différentes.

a. Les exploitants

Les exploitants de Métaguide, représentés par le MOA, souhaitent ajouter des fonctionnalités « vendeuses » sans augmenter les charges d'exploitation. De ce point de vue, les deux aspects d'exploitation significatifs sont la consommation de ressources du serveur et les opérations de maintenance. L'utilisation de mémoire centrale s'est accrue de façon marginale (+4 % dans les pires conditions) par rapport aux 3 Go disponibles et actuellement sous-exploités.

Les ressources impliquées sont de quatre sortes : débit réseau, débit disque, espace disque et temps de calcul. Le débit réseau est inchangé par rapport à l'ancien système. L'espace disque utilisé a beaucoup augmenté du fait des nouvelles tables et leurs index qui représentent maintenant 93 %du total : cependant la taille totale de la base de données, soit actuellement 2,3 Go, reste insignifiante par rapport à l'espace disque disponible (450 Go). Les accès disque sont très sollicités quand on utilise la recherche libre, mais le SGBD maintient un cache important (28 Mo) en mémoire centrale, et se repose aussi sur le cache de Linux (toute la mémoire inoccupée, soit 2,7 Go actuellement). Ainsi une recherche libre SQL effectuée deux fois de suite, s'exécute deux à quatre fois plus vite la deuxième fois. Enfin le temps de calcul d'une recherche libre est du même ordre de grandeur (1,5 s sur un fichier de 87000 lignes) que la recherche guidée. En conclusion, les ressources système ne sont pas un problème aujourd'hui.

La maintenance est nulle pour le nouveau module. En particulier tout a été réalisé avec le support système existant, seul le SGBD a été mis à jour. Du coup les sauvegardes des nouvelles données créées sont intégrées aux sauvegardes existantes. Toutes les tables sont mises à jour et nettoyées de façon transparente.

En conclusion, l'impact sur les travaux d'exploitation est nul, ce qui est important pour l'entreprise qui a peu de personnel. L'impact sur le matériel ne se fera sentir que si les données gérées augmentent considérablement.

b. Les sites appelants

Les services internes des clients du type « sites appelants » doivent pouvoir fournir des fiches à importer dans Métaguide avec les éléments qui permettent de les classer par rapport à un thésaurus. Apporter une amélioration réelle par rapport à une recherche « à la google », demande un travail préalable d'analyse de la structure des données à classer pour lequel TRANSWORK peut fournir un appui de documentalistes. Dans le pire des cas, par exemple si on cherche à classer des articles ou des pages d'information en vrac, le fichier n'existe même pas chez le client, et le travail de classement doit être complété par un travail informatique pour créer un fichier qui soit correctement mis à jour lors des évolutions du site.

L'aspect visuel du site doit éventuellement être modifié pour intégrer le champ de recherche aux endroits ad hoc, et le retour depuis Métaguide (avec la clé de la fiche choisie) doit déclencher une action qui affiche la page correspondante au choix.

Ces travaux ne doivent pas être éludés pour réussir la mise en place de ce système de « délégation de recherche ».

5.4. L'apport du projet au processus métier

Deux buts motivaient ce projet. Un objectif technique lointain était de munir Métaguide d'une interface vocale, et la nécessité d'une certaine souplesse à l'entrée du système a conduit à envisager la recherche libre comme première étape. Ensuite, la disponibilité prochaine de ce nouveau module a ouvert au MOA la perspective d'exploiter une niche, la sous-traitance d'un service de recherche facile à intégrer dans un site existant.

Dès la disponibilité du module, des sites de démonstration ont été mis en chantier afin de servir de base au développement d'une action commerciale d'offre de sous-traitance de « recherche guidée orientée métier ».

Bien que le produit satisfasse le MOA, il n'y a pas encore de retour d'expérience (présentations, ventes...) permettant d'évaluer la pertinence du projet et son utilité commerciale. De même l'interface vocale est un but encore lointain qui nécessitera d'autres travaux préparatoires.

5.5. Le bilan personnel

En partant seulement des grandes lignes de l'objectif à réaliser, d'une esquisse de solution fournie par le MOA et du planning correspondant, ce projet a couvert toutes les étapes d'un projet informatique, du cahier des charges à la mise en exploitation.

Ma position à la fois de maître d'œuvre et d'exécutant dans un projet complet m'a apporté une vision globale. j'ai pu m'organiser pour maîtriser un projet de longue durée. Les recherches préliminaires m'ont ouvert à la complexité des traitements linguistiques. Le résultat est fonctionnellement satisfaisant, fourni dans les temps et installé sans anicroches. Inversement on peut estimer que les rapports avec le MOA auraient dû être plus formels, au moins dans la régularité du reporting ; la révision du planning initial n'a pas non plus été formellement communiquée une fois la solution définie.

Certains cours du CNAM m'ont plus particulièrement servi. Le cours sur les bases de données m'a permis d'aborder la restructuration de la base Métaguide avec une certaine hauteur de vue et des moyens pour la représenter. Les sessions d'intégration de systèmes et l'analyse des conférences de culture générale m'ont poussé à ne jamais perdre de vue une vision globale : considérer le point de vue non seulement d'un utilisateur (l'utilisateur final) mais aussi de tous les intervenants ; séparer les demandes imaginaires des vrais besoins et confronter les avis ; penser non seulement au « quoi » et au « comment » en technicien, mais aussi au « qui », au « pourquoi », au « quand » ; et enfin envisager les différentes échelles de temps.

5.6. Conclusion

Au delà des techniques informatiques, beaucoup de notions linguistiques tournent autour de la recherche libre. Les multiples fonctionnalités envisagées au démarrage du projet ont été filtrées selon leur faisabilité, leur convergence avec l'objectif principal et les moyens disponibles.

Ainsi la recherche par mots-clés commence par éliminer les mots faibles puis s'effectue à partir des radicaux des mots. Ces deux techniques améliorent notablement la pertinence des résultats de recherche. D'un autre côté, certaines fonctionnalités n'avaient pas vraiment de rapport avec la recherche libre : la correction orthographique et le traitement d'expressions particulières sont plutôt des aides à la saisie, et sans être ignorés, ils ont été traités comme des projets complémentaires à venir ; alors que l'indexation de contenus Web semble infaisable sans y mettre des moyens importants qui dans tous les cas ne sauront résoudre l'aspect multilangue.

Entre ces extrêmes, certaines fonctionnalités peuvent être facilement ajoutées si cela semble utile. En effet, PostgreSQL supporte la recherche par synonymie, et les nouvelles tables sont conçues pour permettre une sophistication future du croisement des mots-clés recherchés. Enfin, le support multi-langue a été intégré dans la conception pour ne pas bloquer son développement futur, mais il se heurte aujourd'hui à la question du contexte d'utilisation de Métaguide : les informations des thésaurus peuvent être traduites par une autorité centrale, le producteur ; mais pour les descriptifs des fiches et les contacts, la bonne méthode reste à trouver.

Préserver ces possibilités d'évolution du produit ont été une exigence forte durant la conception et la réalisation. Pour cela nous avons écarté les solutions qui auraient mis en œuvre des techniques dépassées. L'ancien gestionnaire de fichiers GLADIA n'a pas été utilisé, et le SGBD a été modernisé pour profiter de ses meilleures possibilités. Le code de Métaguide a été réorganisé pour simplifier à la fois son adaptation au projet de recherche libre, et la disparition future de GLADIA. Au passage, on a pu constater que le composant Osmose utilise des algorithmes simplistes qui nuisent aux performances de l'ensemble ; il a dû être amélioré pour atteindre la rapidité d'exécution demandée par le MOA.

Au final TRANSWORK dispose d'un nouvel outil économique en ressources, que ce soit le temps de réponse de l'ouverture de session qui reste sous la barre des deux secondes, l'utilisation du disque et de la mémoire qui restent largement en deçà de l'espace disponible, ou les contraintes d'exploitation qui sont inchangées.

Outre qu'il constitue un pas vers une future interface vocale, le nouveau composant apporte la possibilité d'offrir un service de recherche combinée libre+guidée à un site extérieur. Ledit site doit effectuer un important travail préparatoire mais comme on a pu le voir au chapitre 3.1.3, l'amélioration obtenue vaut probablement l'effort fourni.

6. Conclusion générale

6.1. Les systèmes de recherche

La recherche libre, appelée aussi recherche en texte plein (full text search) ou recherche par mots-clés, autrefois centrée sur les systèmes documentaires comme les bibliothèques, est de nos jours une technique très largement répandue sur le Web. Outre les moteurs de recherche généralistes, de nombreux sites la proposent pour trouver les pages de documents qui peuvent intéresser l'usager. L'une des raisons de ce succès est l'existence de logiciels prêts à l'emploi pour indexer directement tout le contenu d'un site Web, ce qui simplifie le travail de l'administration de ces sites. Cependant cette méthode de recherche présente des défauts. Le principal est le manque d'intelligence dans la comparaison des termes significatifs entre ce que demande l'usager et ce qui existe dans les documents. Malgré les tentatives pour évaluer la concordance entre la requête et les résultats (le ranking), il en résulte souvent des listes longues, sans rapport clair avec l'esprit de la demande.

D'autres systèmes existent. Le plus ancien, encore très utilisé, est la recherche multicritère, qui consiste à identifier des attributs fixes dans les données et à les comparer avec des valeurs saisies par l'usager. Ce procédé, plus approprié pour la recherche en base de données que dans des documents, souffre de défauts similaires, car il aboutit souvent à des listes trop longues ou complètement vides, sans guider l'usager pour améliorer ses critères.

Le produit Métaguide offre depuis une quinzaine d'années une autre approche : la recherche guidée par thésaurus arborescent. Cette technique commence seulement à apparaître dans de grands services Web orientés vers la vente en ligne (Ebay, Radiospares, etc.). Elle présente une excellente ergonomie en montrant en permanence à l'usager les catégories et sous catégories d'articles recherchés avec les populations de fiches correspondantes, ce qui autorise à chaque critère un choix éclairé. Cependant cette technique, en contrepartie, impose une certaine rigidité dans la saisie des données, qui doivent indiquer leur classement dans chaque catégorie utile. Ainsi, dans Métaguide, les informations associées à chaque fiche, ne sont pas saisies directement par l'usager, mais choisies dans des listes extraites d'un thésaurus associé au fichier. Certaines données complémentaires ne rentrent pas dans ce schéma pour des raisons évidentes : le nom d'une personne, un prix exact, des détails spécifiques.

6.2. L'évolution visée pour Métaguide

Métaguide existe dans un contexte culturel et commercial. Bien que ses fonctionnalités en fassent un produit ergonomique et souple, le public est aujourd'hui habitué aux moteurs de recherche fonctionnant par saisie de mots-clés ; les dirigeants de TRANSWORK ont donc formé le projet de doubler la recherche par thésaurus d'une recherche libre. Cette première expérience dans le domaine de l'acquisition de texte en langage naturel devrait à terme déboucher sur des interfaces vocales, appropriées à la nouvelle génération de micro-ordinateurs mobiles, smartphones et tablettes.

Le projet a donc commencé par délimiter les contours d'une solution, certes incomplète par rapport aux nombreuses possibilités envisageables autour du langage naturel, mais à la fois réalisable en un temps limité, offrant une utilité effective, et réalisée dans l'esprit que d'autres évolutions sont à venir. Les sources des textes à indexer étaient à la fois dans de multiples attributs de fiches (les descriptifs et les coordonnées de propriétaires), dans de multiples

fichiers (les données et les thésaurus) et dans de multiples gestionnaires de fichiers (la base de données PostgreSQL et le gestionnaire maison GLADIA).

Nous avons vu que la recherche par thésaurus est ergonomique mais contraignante à la saisie, alors que la recherche libre est peu contraignante mais donne des listes de résultats longues et malaisément exploitables. Nous avons voulu offrir le meilleur des deux mondes en procédant en deux temps : la liste de résultats découverts par la recherche libre est passée à la recherche par thésaurus, qui « sait » comment présenter efficacement les catégories qui restent.

Nous avons remarqué que la recherche libre pouvait être utilisée de deux façons différentes. Elle peut être vue comme une simple extension de Métaguide, et apparaître sur les pages Web qui font partie de son environnement applicatif. Mais puisque la session de recherche peut être dorénavant déclenchée par un formulaire de saisie ordinaire (ce qui n'était pas le cas dans la recherche guidée par thésaurus), on peut placer un tel formulaire sur un site indépendant, ce qui permet à TRANSWORK de se placer comme sous-traitant pour le service de recherche libre + guidé, et ceci pour toutes sortes de données. Une offre commerciale supplémentaire devient possible, visant à améliorer les systèmes de recherche existants sur des sites tiers. Du point de vue technique, les fonctions d'administration nécessaires ont été dûment intégrées au projet.

Hormis la réalisation des fonctions majeures d'indexation et de recherche, il fallait aussi prendre en compte les aspects de maintenance, de performance, d'évolutivité. La mise en exploitation du nouveau système ne devait pas perturber le fonctionnement du serveur qui ne dispose pas de capacités de haute disponibilité.

6.3. Le travail accompli

Dans un premier temps l'étude a examiné de nombreux aspects du langage naturel pour déterminer le contour des fonctionnalités réalisables dans le temps imparti, et pour proposer des possibilités d'extensions futures. Il s'est avéré que la plupart des outils linguistiques pertinents disponibles en open-source traitaient surtout de la correction orthographique (les outils des traitements de texte) ou d'indexation de page Web. Les versions récentes du SGBD PostgreSQL savent traiter la recherche libre d'une façon très complète. Ce n'était pas le cas de la version ancienne utilisée par Métaguide, et la mise à niveau posait des problèmes de compatibilité mettant en jeu la stabilité du serveur de l'entreprise.

Une fois la décision prise d'utiliser un PostgreSQL récent, l'étude s'est scindée en plusieurs parties relativement indépendantes. Une première tâche assez longue consistait à mettre à jour le SGBD, avec des sous-tâches d'analyse de copieuses notes de version, d'identification et de correction des incompatibilités, de conversion de la base de données pour supporter les nouvelles notions de jeu de codes de caractères et d'ordre de classement. Une autre tâche se focalisait sur le support et l'administration des « sites appelants » effectuant des recherches depuis un site Web externe.

Les trois dernières tâches concernaient directement la recherche libre selon trois aspects : définition des structures de données permanentes (l'évolution de la base de données), mise à jour de ces données (l'indexation des nouvelles fiches) et utilisation des données (la recherche proprement dite). Ici l'essentiel était les interconnexions avec les différents composants de Métaguide, qu'il a fallu faire évoluer en parallèle de la création de code.

La rédaction des spécifications a donné l'occasion de faire le tri dans les besoins variés exprimés par le MOA. Ensuite la réalisation a dû faire face à la maîtrise de nombreux composants (GLADIA, Osmose, moteur Métaguide, SGBD, Apache, Mégabus) et langages (C, SQL, Shell, PHP).

La mise en service, scindée en étapes pour simplifier les processus et donc réduire les risques d'erreur, a été chaque fois minutieusement préparée et simulée sur une machine virtuelle pour réduire les temps d'indisponibilité et les risques de mauvaises manipulations.

6.4. Résultats et perspectives

Nous avons réussi à installer dans Métaguide une recherche par mots-clés malgré l'hétérogénéité du système. Les fonctions implémentées conviennent au maître d'ouvrage, et permettent une exploitation commerciale immédiate.

Les techniques utilisées ont visé non seulement à réaliser les fonctionnalités, mais aussi à ouvrir la porte aux futures évolutions. Les différents éléments à indexer sont regroupés et indexés en utilisant les possibilités du SGBD (triggers, index inverses, opérateurs de correspondance textuelle). L'outil s'est révélé pratique et extensible. Il prend notamment en charge l'élimination des mots faibles et les flexions des mots sous une forme simplifiée. Il supporte aussi la recherche et l'indexation multi-langue, cette fonction pourra donc être aisément ajoutée quand tous les textes à indexer pourront être saisis en plusieurs langues. De même, le SGBD devrait simplifier la recherche de termes synonymes.

Finalement, le recours systématique à la base de données prépare la suppression du gestionnaire de fichiers maison pour aller vers un logiciel plus homogène et plus souple à faire évoluer. Inversement, le composant maison Osmose a été conservé et amélioré, ce qui profitera aux prochaines évolutions. Le code actuel pourra aussi être révisé avec profit pour utiliser ces améliorations.

Grâce à une vérification préalable des performances de PostgreSQL et des optimisations apportées à Osmose, la vitesse d'affichage de la « première page », celle qui ouvre la session de recherche, n'a pas varié sensiblement. Nous en avons profité pour étudier les performances générales du serveur. L'établissement d'un temps moyen de traitement par transaction n'est pas simple en l'absence de statistiques d'utilisation. Les calculs effectués sur des sessions type, indiquent que le processeur pourrait supporter plus de cent utilisateurs simultanés, loin de la limite de 64 utilisateurs imposée « en dur » par Métaguide. Par ailleurs les capacités de stockage, tant en disque qu'en mémoire vive, sont actuellement sous-exploitées.

Il reste de la place pour de nombreuses améliorations dans Métaguide : au niveau des capacités linguistiques, outre la recherche multi-langue et par synonymie déjà citées, l'utilisation plus poussée du thésaurus permettrait une forme de recherche par contexte qui trouverait des fiches ayant un rapport avec la demande de l'usager, sans avoir aucun mot commun avec celle-ci.

Un peu à l'écart des problèmes de recherche, des outils interactifs d'aide à la saisie, comme un correcteur ou un glossaire de termes, pourraient réduire les discordances dues aux différences d'orthographe entre la saisie lors de la création des fiches et la saisie des mots-clés à chercher. Des expressions particulières pourraient aussi être reconnues comme n'étant pas une demande de recherche, de la même façon que Google™ reconnaît des demandes simples de calcul arithmétique.

La future interface vocale sera probablement aussi un module indépendant de la recherche proprement dite. Le sujet est complexe : comme le fait remarquer Frédéric Duvert dans l'introduction de sa thèse [8], le langage parlé inclut des parasites, des hésitations, des approximations sonores et grammaticales, qui sont moins proéminents à l'écrit. De plus, il est maintenant prouvé que l'échange oral efficace entre humains ne s'appuie pas exclusivement sur le son, mais aussi sur la vue et peut-être d'autre sens [9] ; faudra-t-il en tenir compte ? L'entreprise devra se reposer sur des modules prêts à l'emploi, ou s'intéresser à la recherche actuelle dans ce domaine, comme le projet LUNA [10].

Enfin, la reconnaissance vocale n'est qu'un aspect du problème ; il faudra en parallèle s'intéresser à la présentation des listes de guidage et des résultats, qui pourraient eux aussi être restitués, au moins partiellement, sous forme vocale. Ceci pourrait amener à revoir aussi la cinématique des interactions entre l'usager et les composants de recherche.

> *We choose to go to the moon, and do the other things, not because they are easy, but because they are hard.*
>
> *John Fitzgerald Kennedy*

Annexe A : L'algorithme de recherche guidée

La recherche libre développée dans ce projet est un frontal de la recherche guidée existante. La recherche guidée utilise un fichier annexe, le thésaurus, qui liste les valeurs licites de chaque attribut d'une fiche. Le procédé utilisé est couvert par un brevet [5].

La recherche guidée par thésaurus utilise cinq structures de données principales. Deux sont partagées entre tous les utilisateurs, le **thésaurus** et la **liste complète des fiches**, Trois autres sont propres à chaque utilisateur, la **pile des critères** successifs qu'il a choisis, la **pile de fiches** sélectionnées et la **synthèse des attributs** :

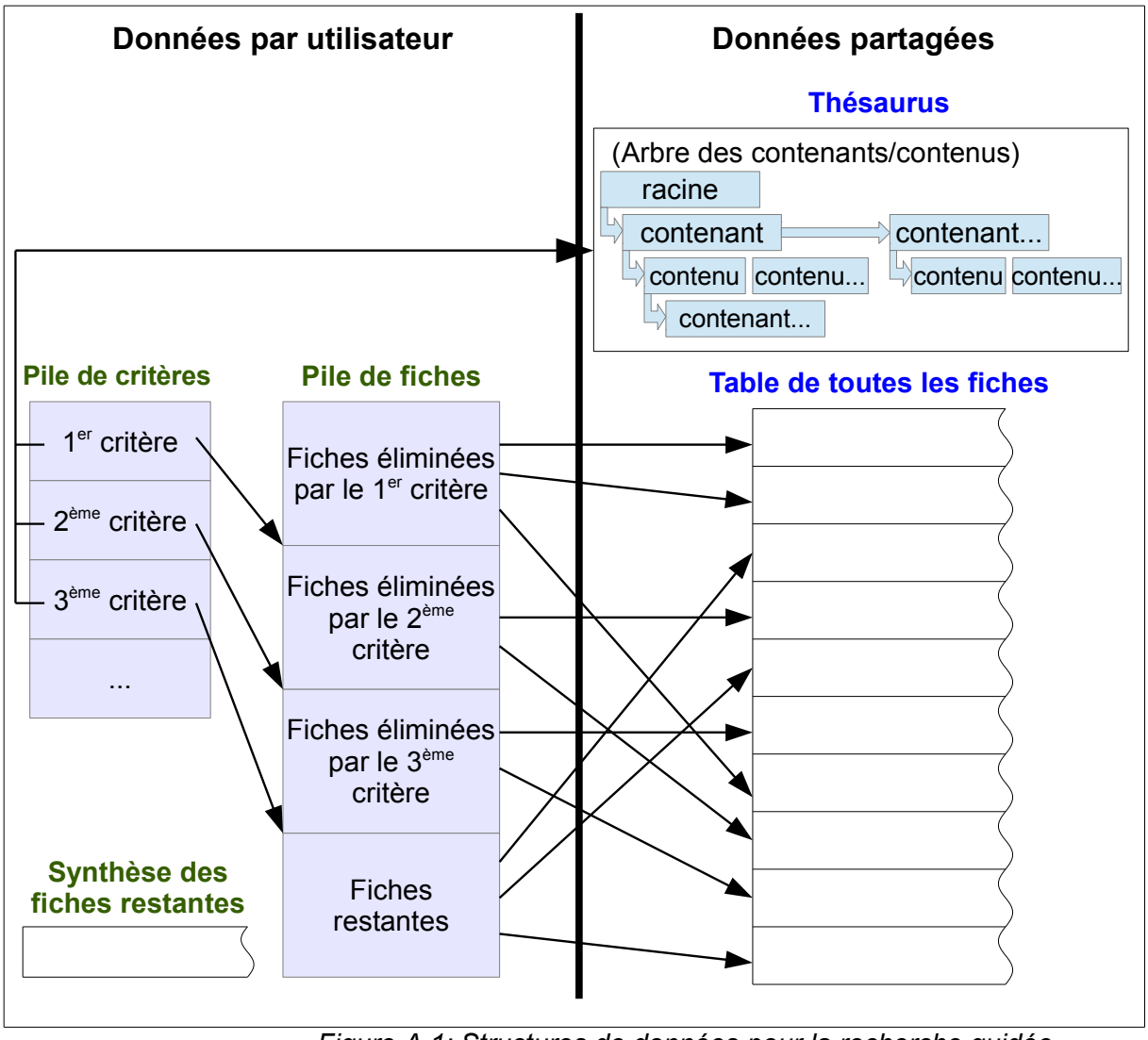


Figure A.1: Structures de données pour la recherche guidée

Nous allons d'abord décrire sommairement les deux structures partagées, puis nous verrons comment un programme les utilise pour opérer des réductions successives grâce à trois structures propres.

a. L'arbre des contenants/contenus du thésaurus

Cet arbre représenté en haut à droite sur la figure A.1 est constitué de listes chaînées dans un segment de mémoire centrale partagée. Il représente le thésaurus.

Le thésaurus lui-même définit les attributs possibles d'une fiche, avec toutes les valeurs possibles. La définition est récursive : chaque valeur peut elle-même être décomposée en attributs avec leurs propres valeurs. Cette organisation permet d'avoir des décompositions différentes selon les valeurs utilisées. Prenons l'exemple d'un thésaurus sur les vêtements.

Des vêtements ont tous une matière, une couleur, un type (pantalon, jupe, chandail) mais une fois qu'on a choisi le type « pantalon » on n'a pas à décrire « col rond, col en V » qui ne s'applique qu'à un chandail. On aboutit au thésaurus suivant, à deux niveaux de contenants et trois niveaux de contenus (en comptant la racine) :

		N° contenu
Racine		0
→ Couleur	Définition de contenant (attribut)	
Couleur = noir	Définition de contenu (valeur)	1
Couleur = blanc		2
Couleur = bleu		3
→ Matière		
Matière = laine		4
Matière = coton		5
Matière = lin		6
→ Type		
Type = pantalon		7
Type = chandail		8
→ Forme		
Forme = col rond		9
Forme = col en V		10
→ Manches		
Manches = longues		11
Manches = courtes		12
Type = jupe		13

A chaque contenu est affecté un numéro d'ordre (le code contenu).

b. La table de toutes les fiches

Les fiches qu'on veut rechercher sont par exemple les différents produits d'une marque de vêtements. Chaque produit est associé à diverses informations comme l'adresse du magasin où on peut le trouver, ou un descriptif commercial. Par ailleurs chaque produit contient les codes des contenus pour chaque contenant utile du thésaurus. Par exemple :

Jupe bleue coton :	0, 13, 3, 5
Chandail bleu lin col rond, manches longues :	0, 3, 6, 8, 9, 11

Deux remarques : toutes les fiches incluent (théoriquement) le contenu 0, et toutes les fiches ne sont pas décrites par le même nombre de contenus.

Enfin, chaque fiche est associée à un propriétaire qui, seul, peut la modifier ou la supprimer.

Les « diverses informations » dont il est question plus haut ne servent pas à la recherche guidée, et elles restent sur un fichier externe GLADIA. Par contre les codes contenus sont en mémoire centrale, où chaque fiche est constituée de :

– un numéro de fiche qui sert de code d'accès au fichier GLADIA

– un « vecteur de contenus », table de bits dont chaque bit vaut 0 ou 1 selon que la valeur est attribuée ou pas à la fiche.

Toutes les fiches ont le même nombre de bits, 14 dans notre exemple de thésaurus vêtements, et donc une structure de taille fixe. Voici une représentation des deux fiches « Jupe bleue coton » et « Chandail bleu lin col rond, manches longues, telles qu'elles sont en mémoire :

Description	**fiche**	**propriétaire**	**vecteur des contenus**
Jupe bleue coton	1234	21	1 0 0 1 0 1 0 0 0 0 0 0 0 1
Chandail bleu lin col rond, manches longues	5678	22	1 0 0 1 0 0 1 0 1 1 0 1 0 0

c. Le processus itératif de recherche

Le processus qui effectue une recherche initialise ses structures propres, opère éventuellement une présélection sur le propriétaire, et remplit la pile de fiches (au centre de la figure A.1) avec un pointeur par fiche dans la mémoire partagée. Ainsi on commence à travailler avec la totalité des fiches. Ensuite le processus va itérer les actions suivantes :

1) Calcul de la synthèse des fiches restantes à cette étape
2) Affichage et choix d'un contenu (un critère de sélection) par l'usager
3) Application du critère pour extraire des fiches restantes, celles qui contiennent le contenu choisi.

d. La synthèse des fiches

Pour lister les choix possibles, le processus de recherche crée une synthèse des fiches. Elle se fait par balayage séquentiel de toutes les fiches, en faisant un « OU inclusif » de tous les vecteurs de contenus. Le résultat est un vecteur de même nature, il identifie quels sont les contenus qui sont instanciés dans les fiches restantes.

A la première itération, dans notre exemple la synthèse serait :

Synthèse	-	-	1 0 0 1 0 1 1 0 1 1 0 1 0 1

En parcourant ce vecteur séquentiellement, on retrouve les contenants qui ont plusieurs contenus instanciés, ceux-ci sont candidats pour être proposés comme choix à l'usager. Quand un tel contenant est identifié on saute directement au contenant suivant, pour éviter de proposer des sous-choix (longueur des manches) en même temps que le choix principal (type de vêtement). Cette analyse est faite en se référant à l'arbre des contenants/contenus.

Dans l'exemple on obtient un choix sur le Type (jupe, chandail) et sur la Matière (coton, lin). Le choix de couleur est sauté car un seul contenu est instancié pour ce contenant, et les choix du col et des manches (bits 9 à 12) sont automatiquement sautés car il font partie du

contenant significatif Type qui s'étend du bit 7 au bit 13. Ainsi Métaguide a « vu » que les choix couleur, col, manches sont soit inutiles soit pas encore significatifs dans le contexte.

e. L'application d'un critère

Tous les choix proposés à l'usager correspondent au final à un bit à « 1 » dans le vecteur de synthèse. L'étape de sélection consiste donc à choisir l'un de ces bits (bien entendu l'usager voit les textes correspondant à ces bits). Cette fonction s'appelle la « réduction » car elle utilise le critère choisi pour réduire la liste des fiches et la synthèse.

Son application successive réalise le « ET » de tous les critères. Métaguide dispose aussi du « OU » que nous n'avons pas détaillé ici.

La liste des fiches restantes est balayée séquentiellement pour repérer celles qui contiennent le bit choisi. Cette boucle est extrêmement rapide, même sur de gros fichiers, car elle ne fait que masquer un bit. Chaque fiche qui ne contient pas le bit est déplacée vers une extrémité de la pile de fiches, à la fin de la boucle les fiches restantes sont partitionnées en « fiches qui répondent au critère » et « fiches qui ne répondent pas », comme il est montré dans l'étape 2 de la figure Erreur : source de la référence non trouvée. Les itérations suivantes seront menées sur la liste nouvellement restante.

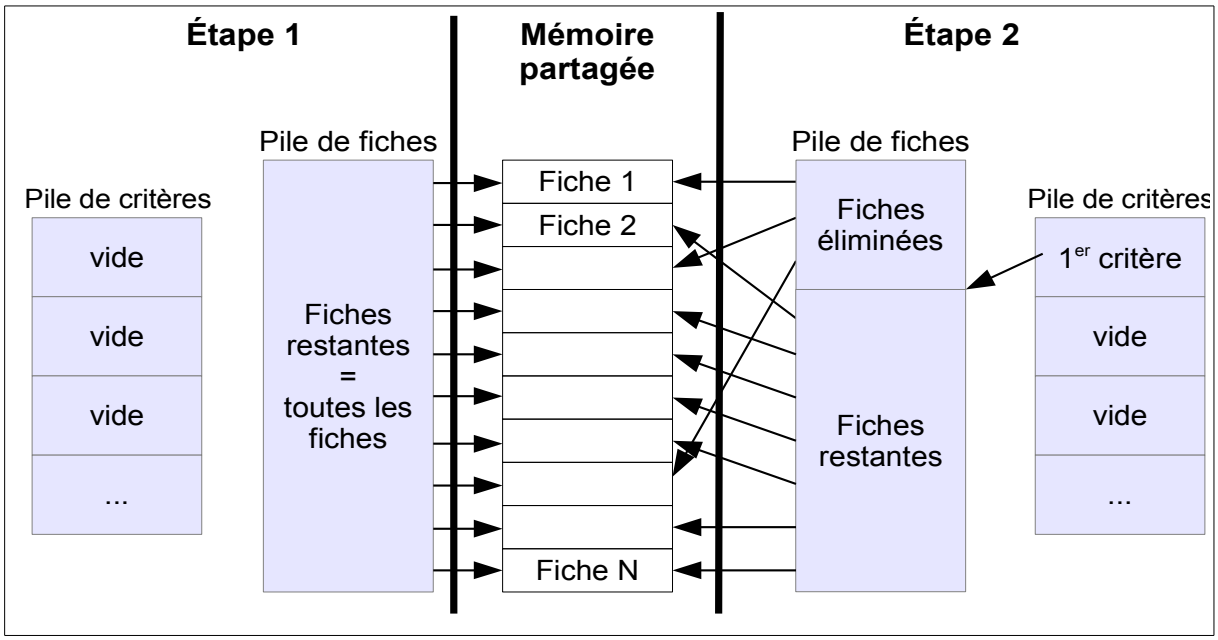


Figure A.2: Avant et après l'application d'un critère en recherche guidée

La limite de la nouvelle partition est notée dans la description du critère, ce qui permet de revenir en arrière très rapidement en repassant successivement les fiches éliminées dans les fiches restantes sans déplacer de données.

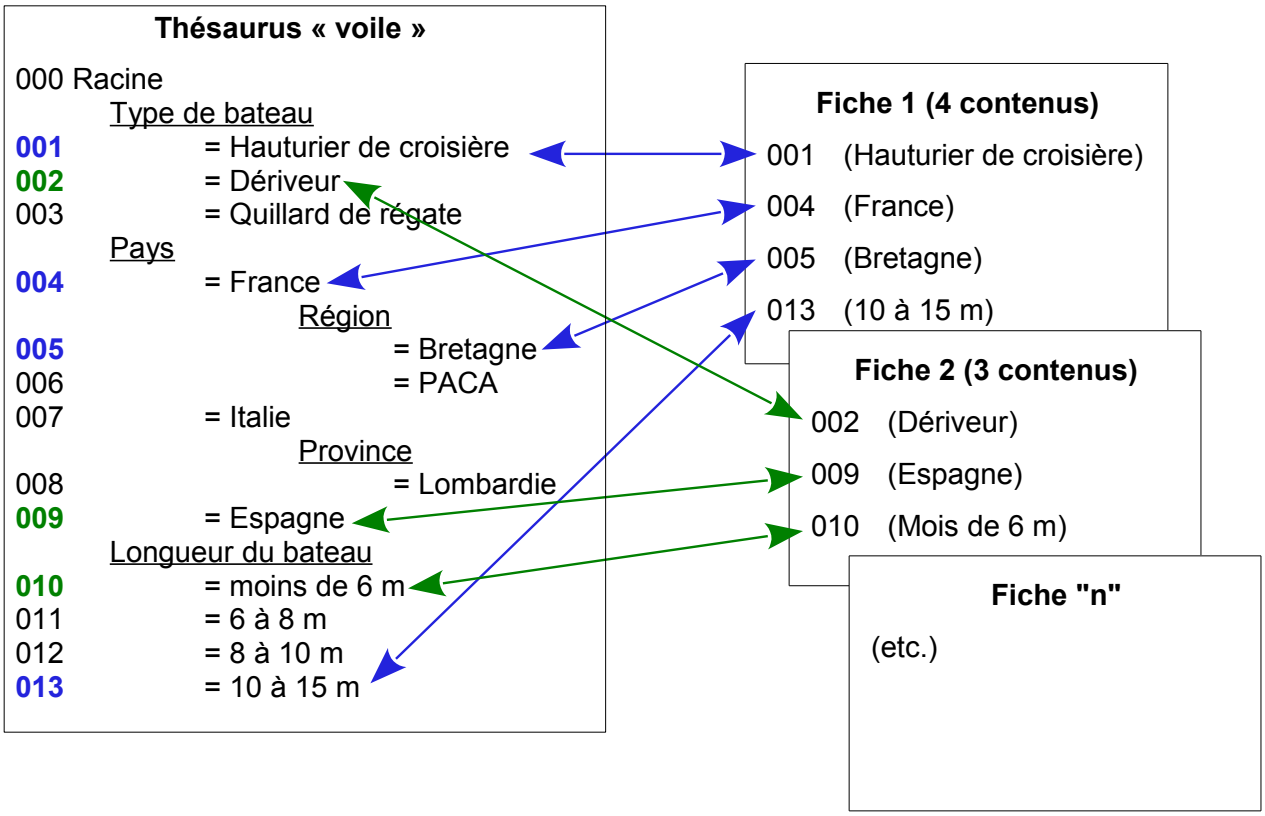
Lorsqu'il ne reste qu'un petit nombre de fiches, le processus de sélection se termine et une liste de taille raisonnable peut s'afficher sur le navigateur de l'usager, réalisant ainsi l'objectif de Métaguide :

Pas de liste trop longue, pas de liste vide, guider l'usager.

Annexe B : La recherche guidée de Métaguide

Supposons que nous cherchions à acheter un bateau de croisière. Le site http://www.marinas-yachting.com/voile.html propose des petites annonces concernant la navigation de plaisance, la recherche est pilotée par Métaguide.

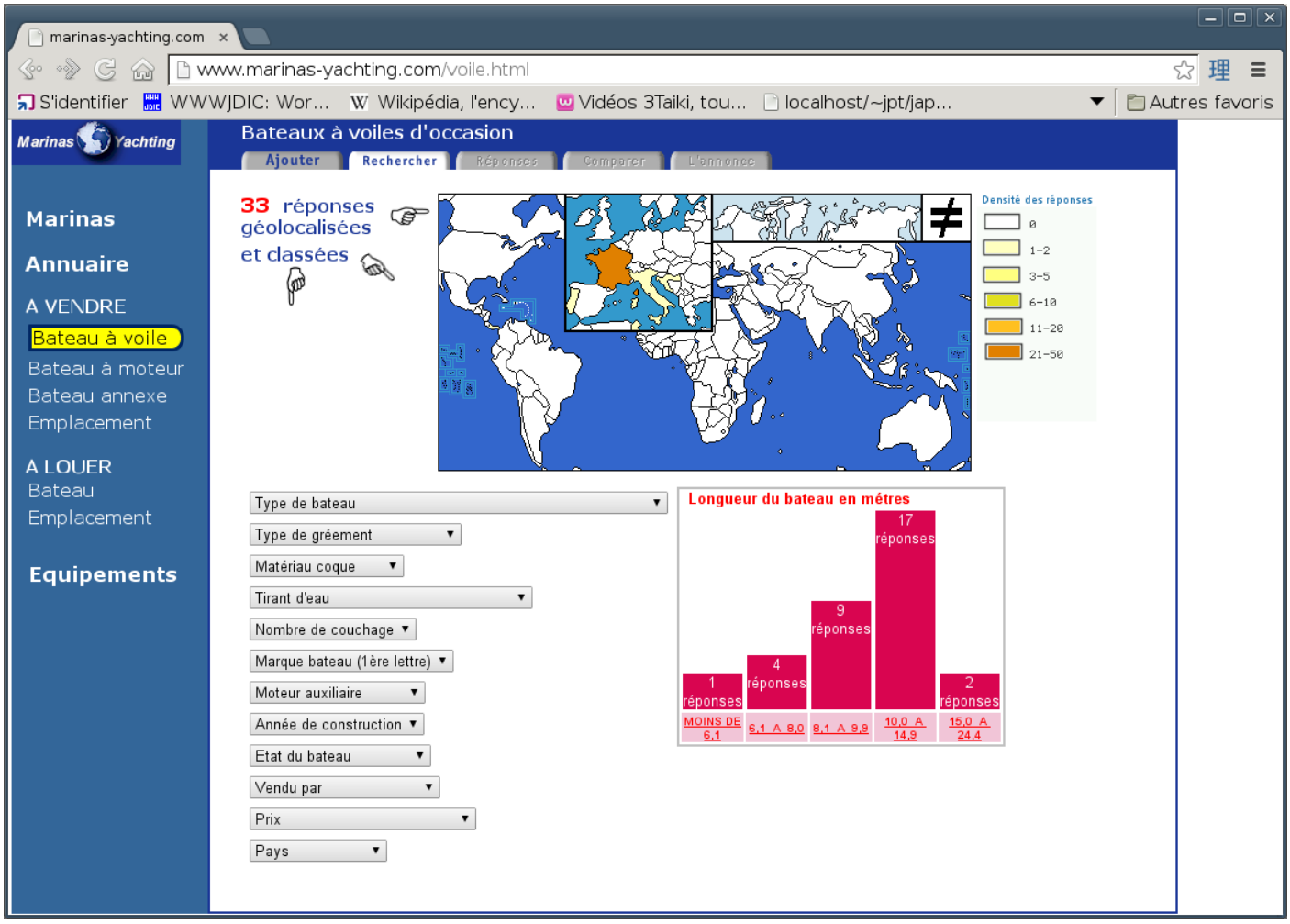
Voici d'abord pour l'exemple, un extrait du thésaurus associé à ce fichier :



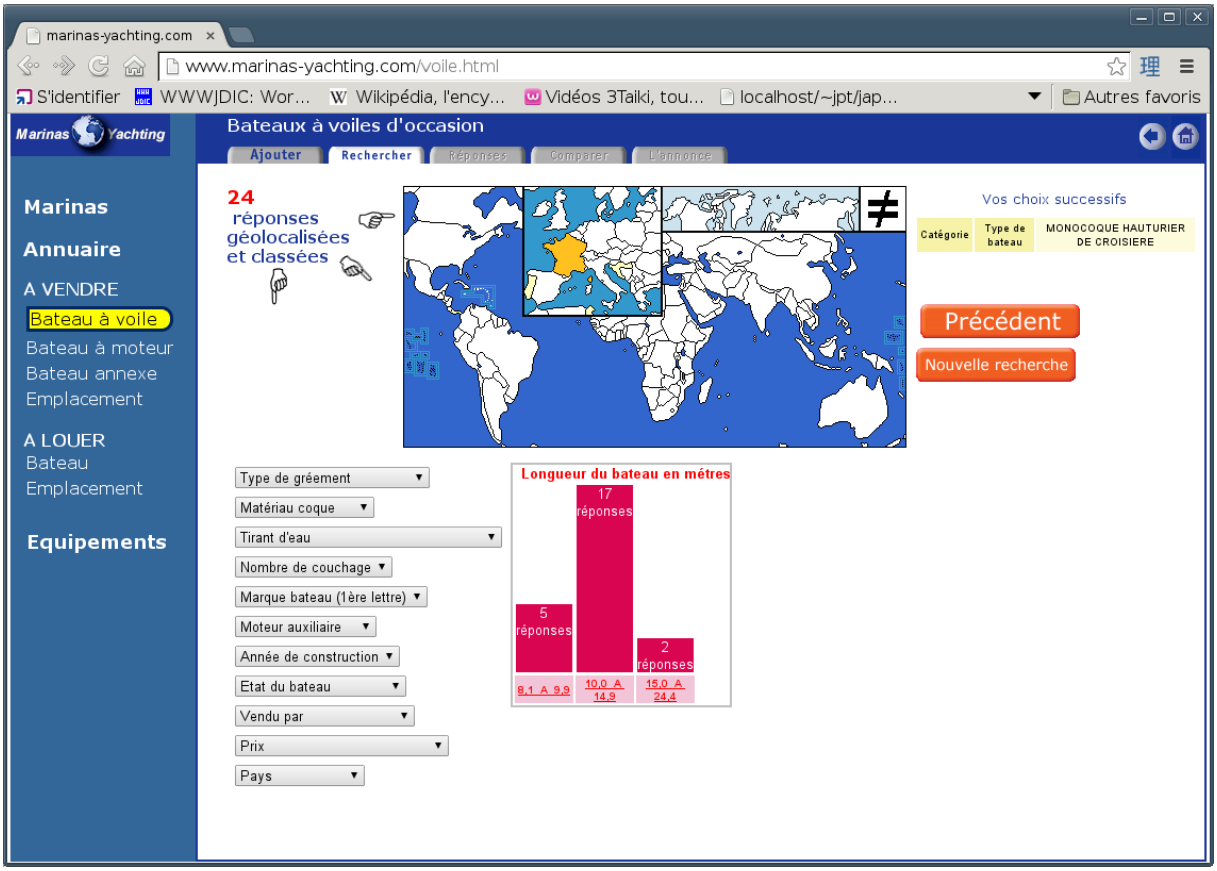
Nous allons maintenant utiliser ce thésaurus pour faire une recherche guidée.

La page d'accueil pour les annonces « voile » propose une liste de caractéristiques : type de bateau, type de gréement, etc. Dans chaque menu déroulant on peut choisir une valeur de la caractéristique. Seules les valeurs qui sont instanciées dans des fiches sont présentées.

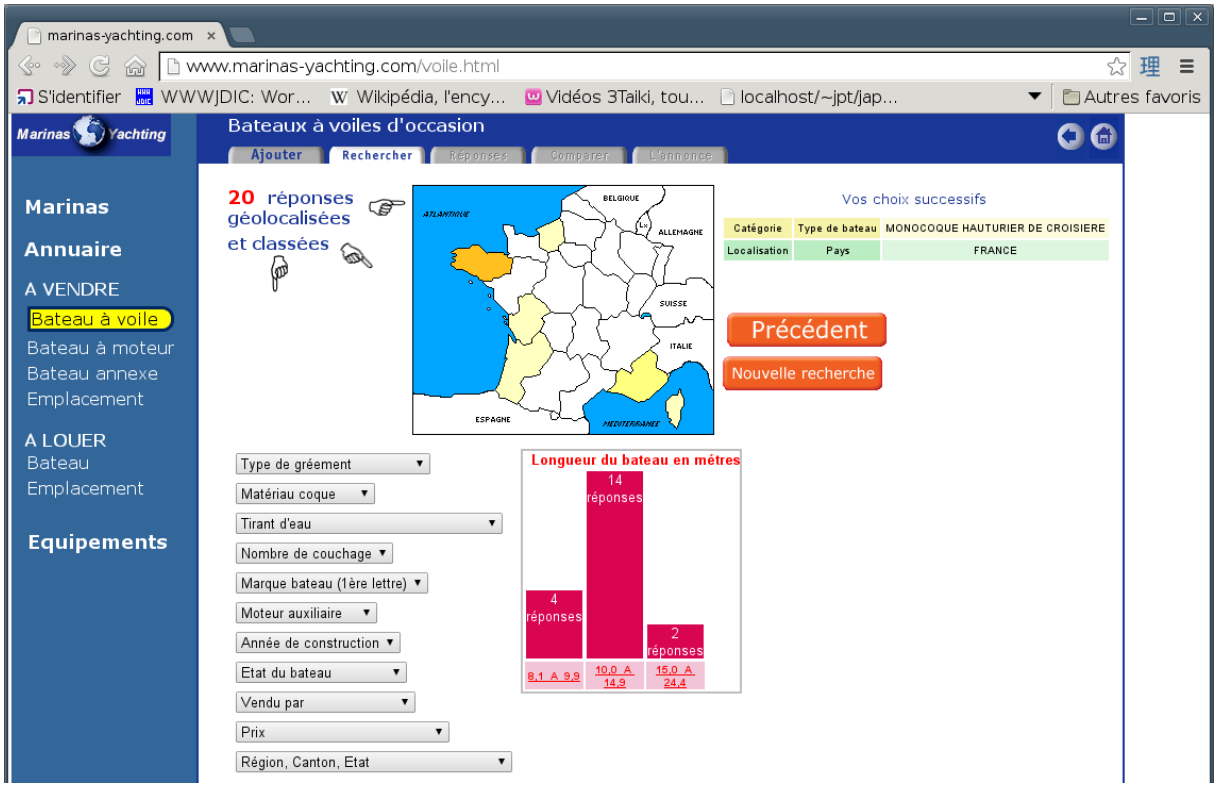
A part la liste déroulante, deux autres présentations sont possibles. La carte, par exemple pour choisir un pays, est réactive, c'est-à-dire qu'on peut cliquer directement sur la zone choisie. La légende, composée des couleurs et les intervalles de valeurs est calculée dynamiquement. L'histogramme est aussi réactif. Quand le type de données s'y prête, on peut y représenter les quantités de fiches associées à chaque valeur. Dans ce cas l'ordre des valeurs dans le thésaurus donne l'ordre d'affichage.



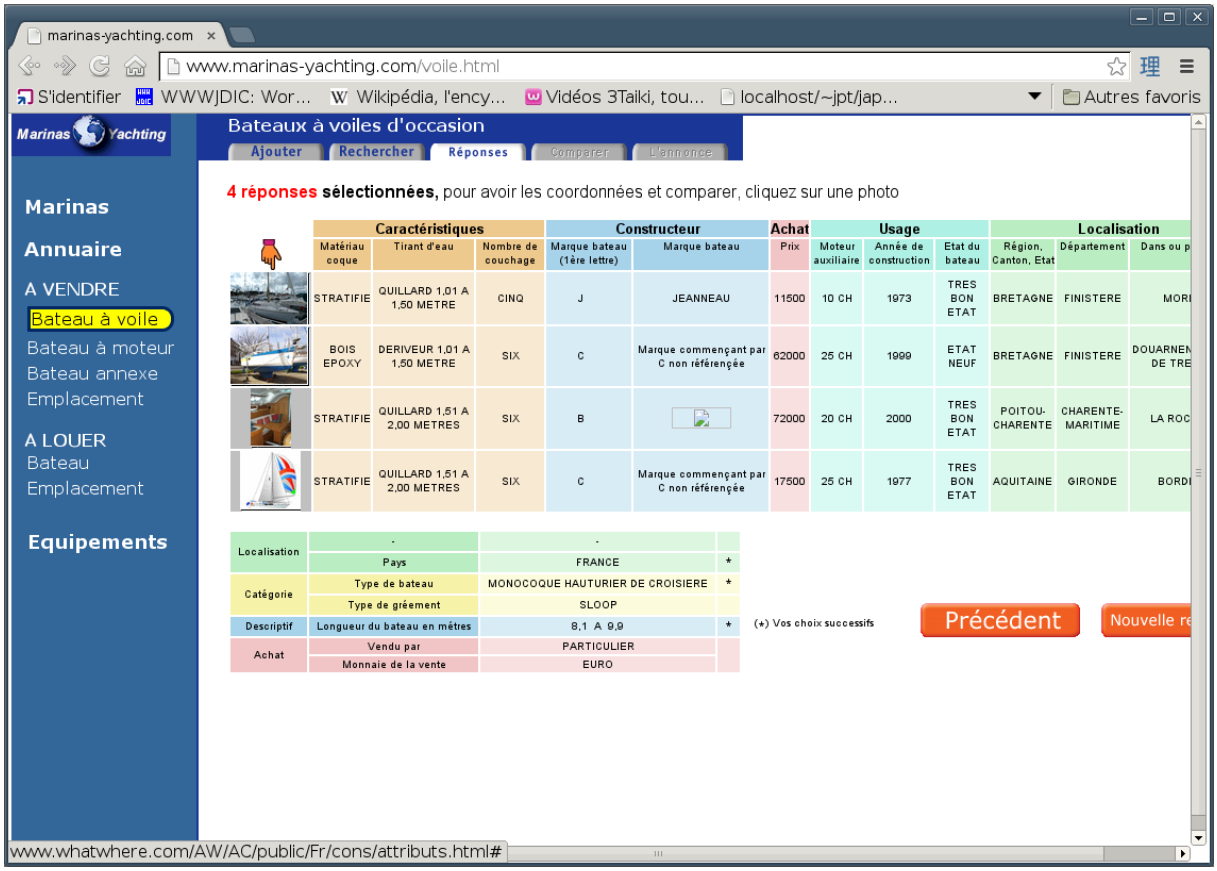
On sélectionne « Type de bateau → Hauturier de croisière » **dans les listes déroulantes**. La page suivante apparaît, les choix possibles ont été réduits en fonction des fiches restantes :



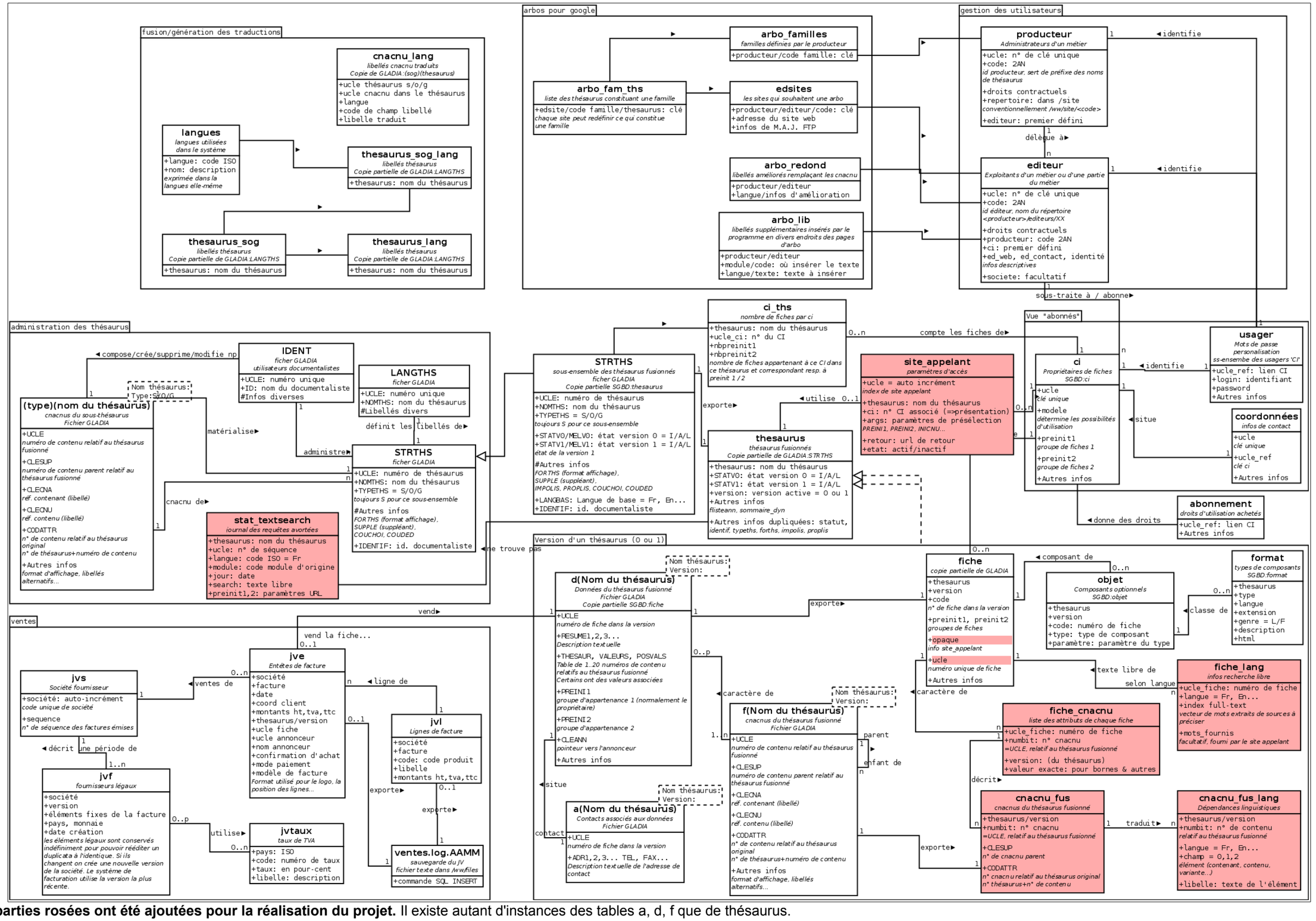
Dans la carte on clique sur la France pour réduire encore le champ de recherche :



Enfin on choisit les bateaux de moins de 10 mètres dans l'histogramme, et on obtient une **« liste courte »** de 4 annonces. On peut ensuite afficher les détails et comparer les photos d'annonces entre elles.

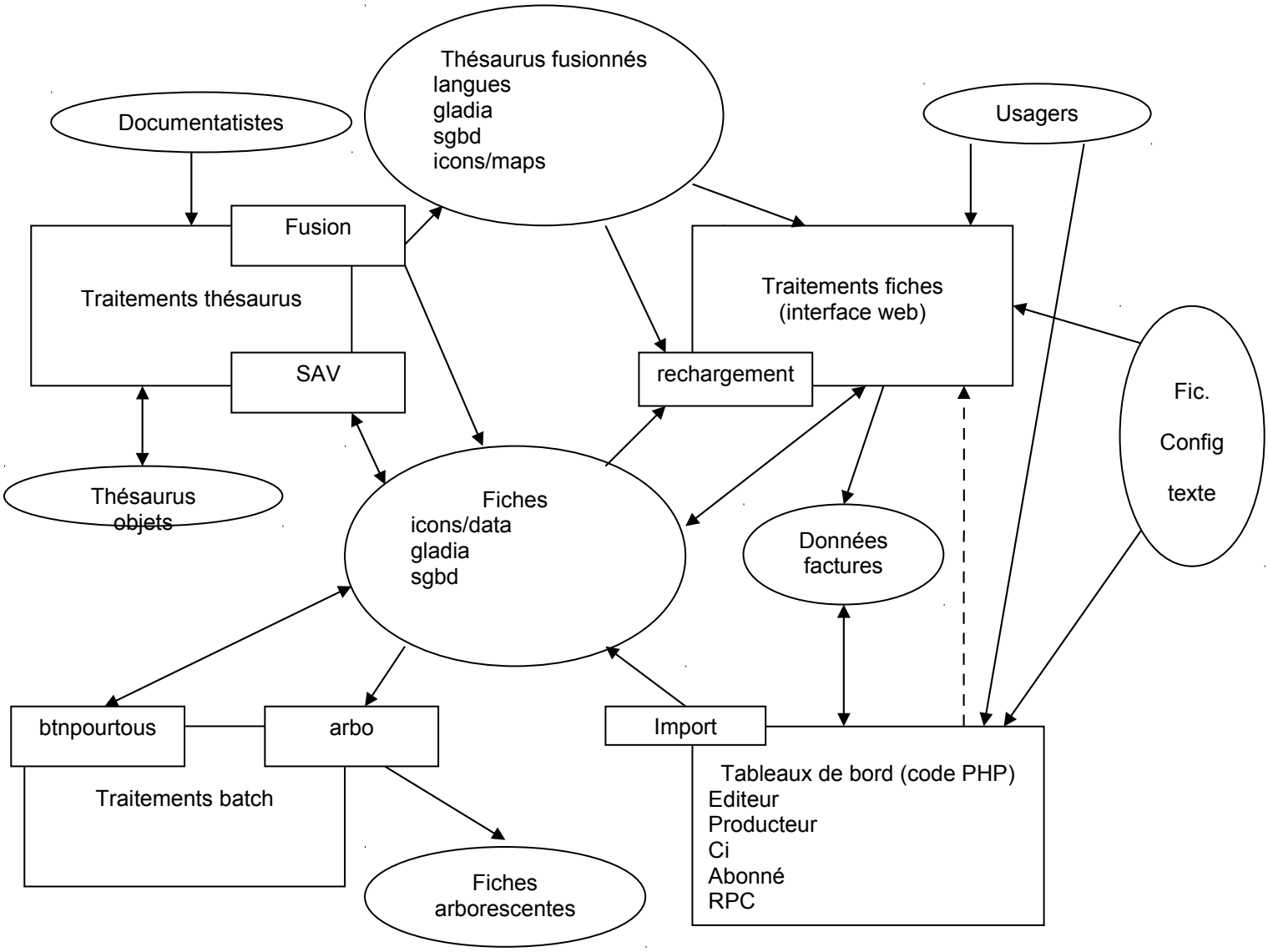

marinas-yachting.com
www.marinas-yachting.com/voile.html
S'identifier
WWWJDIC: Wor...
Wikipédia, l'ency...
Vidéos 3Taiki, tou...
localhost/~jpt/jap...
Autres favoris
Marinas Yachting
Bateaux à voiles d'occasion
Ajouter
Rechercher
Réponses
Comparer
L'annonce
Marinas
Annuaire
A VENDRE
Bateau à voile
Bateau à moteur
Bateau annexe
Emplacement
A LOUER
Bateau
Emplacement
Equipements
4 réponses sélectionnées, pour avoir les coordonnées et comparer, cliquez sur une photo
Caractéristiques
Constructeur
Achat
Usage
Localisation
Matériau coque
Tirant d'eau
Nombre de couchage
Marque bateau (1ère lettre)
Marque bateau
Prix
Moteur auxiliaire
Année de construction
Etat du bateau
Région, Canton, Etat
Département
Dans ou p
STRATIFIE
QUILLARD 1,01 A 1,50 METRE
CINQ
J
JEANNEAU
11500
10 CH
1973
TRES BON ETAT
BRETAGNE
FINISTERE
MORI
BOIS EPOXY
DERIVEUR 1,01 A 1,50 METRE
SIX
C
Marque commençant par C non référencée
62000
25 CH
1999
ETAT NEUF
BRETAGNE
FINISTERE
DOUARNEN DE TRE
STRATIFIE
QUILLARD 1,51 A 2,00 METRES
SIX
B
72000
20 CH
2000
TRES BON ETAT
POITOU-CHARENTE
CHARENTE-MARITIME
LA ROC
STRATIFIE
QUILLARD 1,51 A 2,00 METRES
SIX
C
Marque commençant par C non référencée
17500
25 CH
1977
TRES BON ETAT
AQUITAINE
GIRONDE
BORDI
Localisation
Pays
FRANCE
Catégorie
Type de bateau
MONOCOQUE HAUTURIER DE CROISIERE
Type de gréement
SLOOP
Descriptif
Longueur du bateau en mètres
8,1 A 9,9
Achat
Vendu par
PARTICULIER
Monnaie de la vente
EURO
(*) Vos choix successifs
Précédent
Nouvelle re
www.whatwhere.com/AW/AC/public/Fr/cons/attributs.html#

Annexe C : Le modèle Entités-relations de Métaguide



Les parties rosées ont été ajoutées pour la réalisation du projet. Il existe autant d'instances des tables a, d, f que de thésaurus.

Annexe D : Structure interne de Métaguide



Métaguide est constitué de quatre grands ensembles, représentés par des grands rectangles sur le schéma :

- les traitements des thésaurus (création, modification, suppression, mise en service)
- les traitements des fiches (création, modification, consultation, suppression)
- les traitements des utilisateurs (création, édittion, consultation, suppression, tableaux de bord spéciaux)
- les traitements batch (sauvegardes, conversion des fiches en pages html liées par familles de produits ou d'annonces, traitements spécifiques à certains clients)

Ces ensembles traitent des données représentées par des ovales, ou s'en servent pour communiquer entre eux.

- Les thésaurus objets sont édités par des documentalistes pour produire des « thésaurus fusionnés » qui servent à définir des fichiers de données utiles
- Les fiches sont le point central de l'activité. La donnée d'un thésaurus définit un type de fiches.

Annexe E : Le formulaire HTML de recherche libre

Il y a deux formulaires : l'un concerne les « sites appelants », l'autre est dans la page initiale de recherche guidée. Les codes HTML suivants sont adaptés des pages ayant servi à la recette.

Formulaire pour les sites appelants :

```
<form method="post" action="/cgi-bin/FULLTEXTSERVICE">
        <input type="hidden" name="LC" value="3" />
        <td>simulation de recherche depuis site externe:<br>(methode post)</td>
        <td><input type="text" name="FT" placeholder="tapez ici" /></td>
        <td>avec le client:</td>
        <td><input type="text" name="CI" value="BAMYedit" /></td>
        <td><input type="submit"></td>
</form>
```

Ici le point d'entrée dans Métaguide est la fonction « FULLTEXTSERVICE ». Le code client est transmis au serveur Métaguide dans la variable « CI » et les mots à rechercher dans la variable « FT » (Full Text Search).

Saisie intégrée à la recherche guidée :

```
Tapez des mots-clef:
<INPUT
    type="text"
    onChange="
        window.location.replace(
            '{$osm_var_list.cgi_session$}&FONC=TSEARCH&TS='+this.value);
        return false">
```

On remarque que l'URL appelé est une concaténation du numéro de session, qui est connu dans ce cas, d'un paramètre destiné au module contrôleur (la fonction à appeler), et de la chaîne de mots tapés.

Cet extrait provient du modèle de page ; c'est pourquoi on ne voit pas le numéro de session proprement dit mais la balise Osmose qui le représente.

Annexe F : La bibliothèque d'accès aux fiches

Cette bibliothèque est auto-documentée avec l'outil « doxygen ». Voici un extrait significatif concernant la bibliothèque de modification des fiches créée pour remplacer les traitements disparates.

Référence du fichier cl_fiche.c

Description détaillée

Classe "fiche permanente": creation, suppression, modif d'une fiche sur disque

---

Fonctions

char * **FicheCreation** (int do_it, int fd_donnees, ucle_t *ucle_fiche_glad, char *nomths, char *version, char *langue, char *numref, char *preinit1, char *preinit2, int use_fiche_adr, time_t date_suppr, ucle_t annonceur, char *opaque)
char * **FicheCommitCnacnus** (int fd_donnees, ucle_t ucle_fiche_glad, char *nomths, char *version)
char * **FicheCommitBornesAutres** (int fd_donnees, ucle_t ucle_fiche_glad, char *nomths, char *version, int numbit, char *val)
char * **FicheCommitResume** (int fd_donnees, ucle_t ucle_fiche_glad, char *nomths, char *version)
char * **FicheCommitAnnonceur** (int fd_ann, ucle_t ucle_fiche_glad, char *nomths, char *version)
int **FicheModifBegin** (char ftype, char *ucle)
char * **FicheModifRollback** (int fd)
char * **FicheSuppression** (char *ucle_data)

Documentation des fonctions

---

char* FicheCreation (int *do_it*, int *fd_donnees*, ucle_t * *ucle_fiche_glad*, char * *nomths*, char * *version*, char * *langue*, char * *numref*, char * *preinit1*, char * *preinit2*, int *use_fiche_adr*, time_t *date_suppr*, ucle_t *annonceur*, char * *opaque*)

FicheCreation insere une fiche dans le SGBD

Paramètres:

[in] *do_it* FALSE=>ne modifier aucun fichier ni SHM
[in] *fd_donnees* handle fichier donnees GLADIA
[out] *ucle_fiche_glad* cle GLADIA attribuee a la fiche
[in] *nomths* C-string, nom thesaurus "PRTHS"
[in] *version* C-string, version du thesaurus "0" ou "1"
[in] *langue* C-string, langue des donnees e.g. "Fr"
[in] *numref* numref de 7 caracteres
[in] *preinit1,2* preinits, NULL est converti en ""
[in] *use_fiche_adr* 0 si le contact est le CI 1 si il est spécifique
[in] *date_suppr* time_t, date de peremption en timestamp UNIX
[in] *annonceur* long, cle GLADIA de l'annonceur
[in] *opaque* texte, peut être NULL

Renvoie:

NULL si OK, sinon un message d'erreur SANS newline
Si OK, ucle_fiche_glad contient une cle gladia valide

---

char* FicheCommitAnnonceur (int *fd_ann*, ucle_t *ucle_fiche_glad*, char * *nomths*, char * *version*)

FicheCommitAnnonceur remplace les coordonnees associees à une fiche

Paramètres:

[in] *fd_ann* handle fichier annonceur GLADIA
[in] *ucle_fiche_glad* cle GLADIA attribuee a la fiche de donnee
[in] *nomths* C-string, nom thesaurus "PRTHS"
[in] *version* C-string, version du thesaurus "0" ou "1"

Renvoie:

NULL si OK, sinon un message d'erreur SANS newline
Dans tous les cas, le fichier GLADIA est fermé.

---

char* FicheCommitBornesAutres (int *fd_donnees*, ucle_t *ucle_fiche_glad*, char * *nomths*, char * *version*, int *numbit*, char * *val*)

FicheCommitBornesAutres remplace les valeurs de cnacnus d'une fiche en utilisant les champs GLADIA THESAUR, etc., préalablement valorisés

ATTENTION: devrait inclure gestion.c::modif_fiche_data() mais c'est pas fait pour l'instant car elle n'est appelee que dans gestion.c

Paramètres:

[in] *fd_donnees* handle fichier donnees GLADIA
[in] *ucle_fiche_glad* cle GLADIA attribuee a la fiche
[in] *nomths* C-string, nom thesaurus "PRTHS"
[in] *version* C-string, version du thesaurus "0" ou "1"

Renvoie:

NULL si OK, sinon un message d'erreur SANS newline
Dans tous les cas, le fichier GLADIA est fermé.

---

char* FicheCommitCnacnus (int *fd_donnees*, ucle_t *ucle_fiche_glad*, char * *nomths*, char * *version*)

FicheCommitCnacnus remplace les cnacnus d'une fiche en utilisant les champs GLADIA THESAUR, etc., préalablement valorisés

Paramètres:

[in] *fd_donnees* handle fichier donnees GLADIA
[in] *ucle_fiche_glad* cle GLADIA attribuee a la fiche
[in] *nomths* C-string, nom thesaurus "PRTHS"
[in] *version* C-string, version du thesaurus "0" ou "1"

Renvoie:

NULL si OK, sinon un message d'erreur SANS newline
Dans tous les cas, le fichier GLADIA est fermé.

---

char* FicheCommitResume (int *fd_donnees*, ucle_t *ucle_fiche_glad*, char * *nomths*, char * *version*)

FicheCommitResume remplace les champs fixes d'une fiche

Paramètres:

[in] *fd_donnees* handle fichier donnees GLADIA
[in] *ucle_fiche_glad* cle GLADIA attribuee a la fiche
[in] *nomths* C-string, nom thesaurus "PRTHS"
[in] *version* C-string, version du thesaurus "0" ou "1"

Renvoie:

NULL si OK, sinon un message d'erreur SANS newline

Dans tous les cas, le fichier GLADIA est fermé.

---

int FicheModifBegin (char *ftype*, char * *ucle*)

Prepare la modification d'une fiche.

Necessite: ThesaurusNomVersion, DirBibGl (pour OuvrirTable)

Paramètres:

[in] *ftype* 'd' ou 'a' ou 'f'
[in] *ucle* string, numero GLADIA de la fiche

Renvoie:

handle gladia si OK, -1 si erreur

---

char* FicheModifRollback (int *fd*)

Annule la modification d'une fiche.

Necessite: ThesaurusNomVersion, DirBibGl (pour OuvrirTable)

Paramètres:

[in] *fd* handle gladia

Renvoie:

NULL (i.e. pas d'erreur, pas de message d'erreur)

---

char* FicheSuppression (char * *ucle_data*)

FicheSuppression supprime une fiche et ses éléments

1 tables fiche et objet
2 tables en cascade
3 fichier gladia annonceur
4 fichier gladia donnees
5 boutons et photos ? ligne en SHM metaguide

PREREQUIS: ThesaurusNomVersion doit etre initialisé

Paramètres:

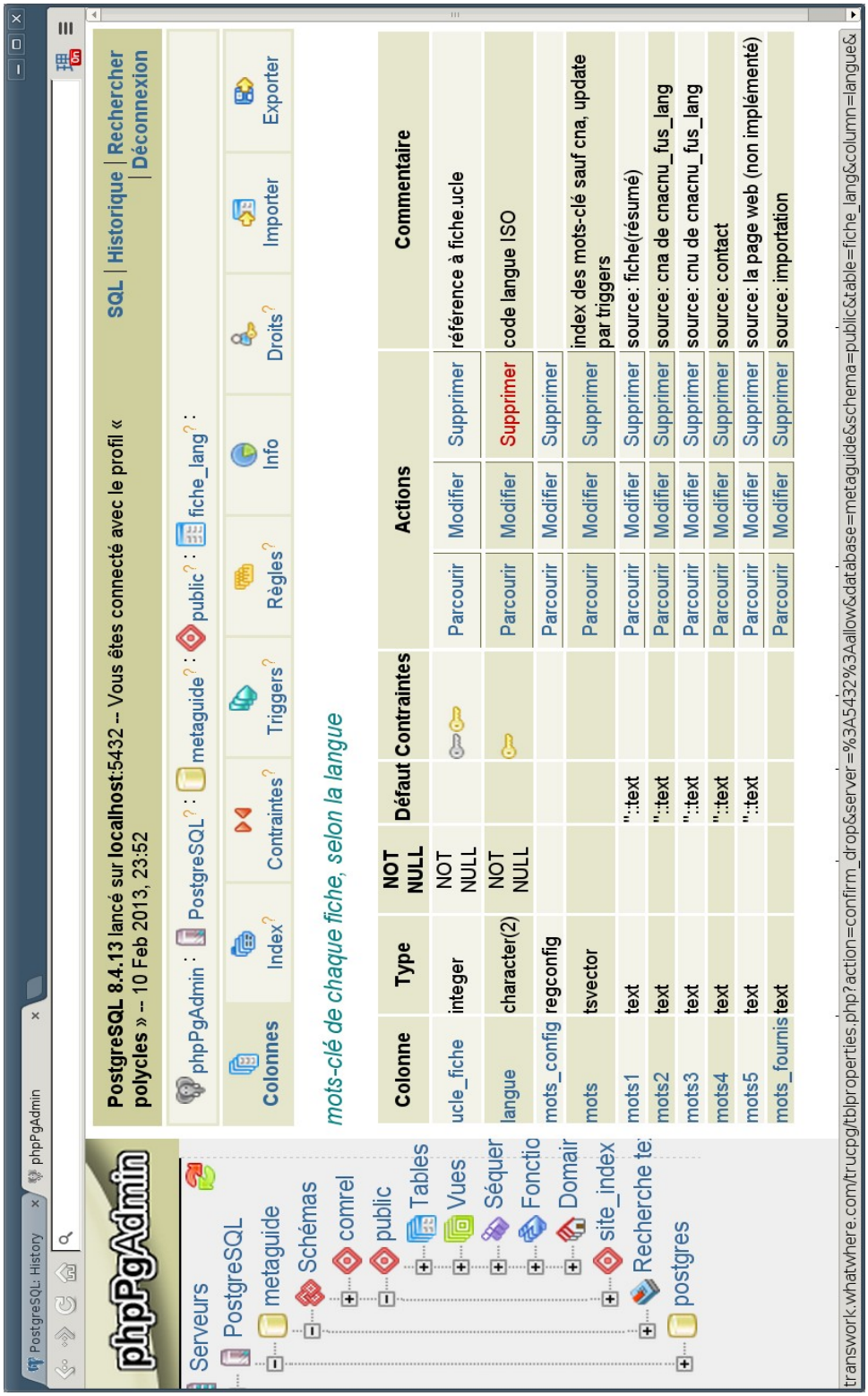
[out] *ucle_data* cle GLADIA attribuee a la fiche

Renvoie:

NULL si OK, sinon un message d'erreur SANS newline

Annexe G : Copie d'écran de phppgadmin

Dans cette copie d'écran, la structure de la table « fiche_lang » est affichée par phppgadmin. On peut y constater l'aspect graphique convivial et quelques possibilités de l'outil.



Colonne	Type	NOT NULL	Défaut	Contraintes	Actions				Commentaire
					Parcourir	Modifier	Supprimer	Droits	
ucle_fiche	integer	NOT NULL			Parcourir	Modifier	Supprimer		référence à fiche.ucle
langue	character(2)	NOT NULL			Parcourir	Modifier	Supprimer		code langue ISO
mots_config	regconfig				Parcourir	Modifier	Supprimer		
mots	tsvector				Parcourir	Modifier	Supprimer		index des mots-clé sauf cna, update par triggers
mots1	text		''::text		Parcourir	Modifier	Supprimer		source: fiche(résumé)
mots2	text		''::text		Parcourir	Modifier	Supprimer		source: cna de cnacnu_fus_lang
mots3	text		''::text		Parcourir	Modifier	Supprimer		source: cnu de cnacnu_fus_lang
mots4	text		''::text		Parcourir	Modifier	Supprimer		source: contact
mots5	text		''::text		Parcourir	Modifier	Supprimer		source: la page web (non implémenté)
mots_fournis	text				Parcourir	Modifier	Supprimer		source: importation

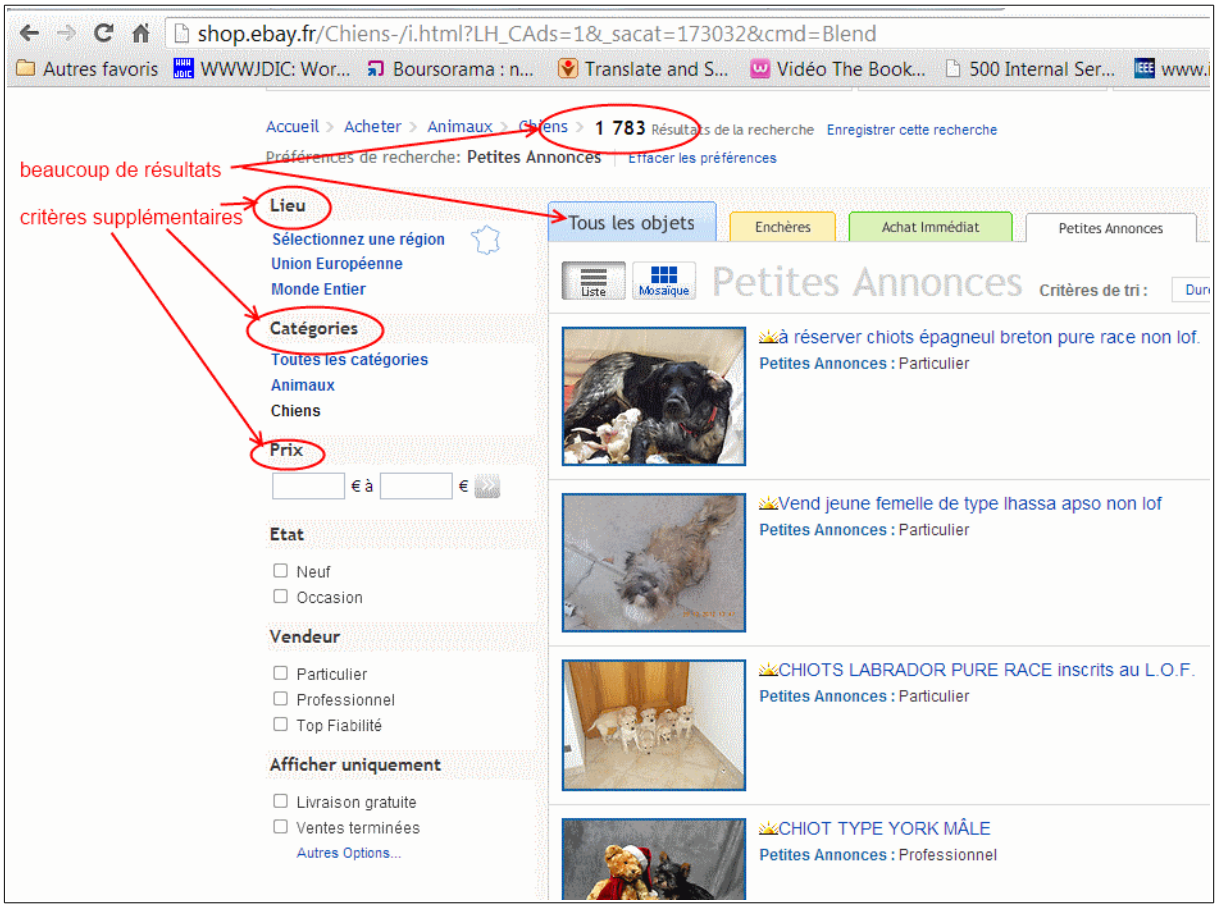
Annexe H : Exemple de checklist d'installation

Tâche 20 - Conversion Postgresql en version 8.4
Fait et testé dans une machine virtuelle : checklist des commandes pour la conversion

Installation de postgresql 8.4 en parallèle	• Supprimer cluster 7.4 et packages postgres 7.4 pg_dropcluster –stop 7.4 transwork aptitude purge postgresql-7.4 ; supprimer /etc/init.d/postgresql-7.4 • éviter conflit pendant l'install edit init.d/postgresql-8.1, Provides: postgresql => postgresql-81 • aptitude install postgresql-8.4 => libpq4{a} postgresql-8.4 postgresql-client-8.4{a} • ajouter squeeze-backports dans /etc/apt/sources.list à cause de http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=633801 • aptitude -t squeeze-backports install postgresql-common	OK OK OK OK OK
Préparations non intrusives	• pg_dropcluster --stop 8.4 main • vérifier les select/update/insert pour des conversions text implicites • tests unitaires sur un serveur de test en version 8.1 et en 8.4	OK OK OK
Arrêt du service	• avertissement des usagers • branchement sur la page maintenance • kill des process restants	7/4/12 OK OK
Migration	pg_upgradecluster -v 8.4 8.1 transwork /ww/pgsql84	OK
Reconfiguration à la 8.4	Dans 8.4/transwork/postgresql.conf - log_min_messages = notice - log_min_error_statement = notice pg_ctlcluster 8.4 transwork restart	OK
Changement encodage	dropdb metaguide dans cluster 8.4 transwork pg_dumpall dans cluster 8.1 édition du dump pour modifier l'encodage de SQL_ASCII en iso8859 psql <dump dans le cluster 8.4	OK
Complément	Redémarrage du service web	OK
Suppression de l'ancienne database	pg_dropcluster 8.1 transwork remettre log_min_error_statement = notice en commentaire	OK non
Sauvegarde	Sauvegarde de la configuration du cluster Sauvegarde de la liste des packages installés Préparation de sauvegarde des données (/ww/pgsql84 → /ww/pgsql) Sauvegarde du schéma Sauvegarde du système de développement	OK OK OK OK OK
Contrôles	Surveiller la première rotation du log Surveiller la première fusion tester l'accès consultation tester l'accès dépôt tester l'accès modification de fiche tester l'accès suppression Surveiller le premier redémarrage du système tester l'accès abonnés	OK OK OK non non non OK non

Annexe I : Les critères successifs sur e-Bay

Sur cet extrait d'une page du site e-bay, on peut constater qu'après la prise en compte des premiers critères de l'utilisateur, des résultats trop nombreux font apparaître une liste de critères supplémentaires.



Glossaire et acronymes

Bot	Néologisme. Contraction de « robot », logiciel automatique qui interroge les serveurs Web en simulant les actions d'un utilisateur humain. (voir http://fr.wikipedia.org/wiki/Bot_informatique et par exemple http://fr.wikipedia.org/wiki/Wikip%C3%A9dia:Bot.
Clé opaque	Information que le site appelant associe à toute fiche lors de l'importation, et qui lui est restituée à l'issue d'une recherche libre. Dans l'intervalle, cette information n'est utilisée pour aucun calcul ni test.
CGI	Common Gateway Interface, standard de fait qui spécifie comment les différents paramètres d'une requête HTTP (URL, adresse d'origine, etc.) sont transmis à une application extérieure au serveur HTTP.
CI	« Client intégré », abonné ayant des présentations spécifiques et éventuellement une possibilité d'insertion gratuite de fiches.
cna/cnu	Abréviation de contenant/contenu
Contenant/contenu	Couple attribut/valeur listé dans un thésaurus et pouvant être référencé par une fiche de données. Par exemple dans un thésaurus automobile : «Type de véhicule/Camion ».
CRUD	De l'anglais « Create/Read/Update/Delete », les quatre opérations de base sur une donnée ou un groupe de données : création, lecture, mise à jour, suppression.
Déclinaison	Variation d'un mot servant à l'adapter au contexte grammatical, sans impact direct sur le sens. Par exemple, les conjugaisons des verbes, les pluriels des noms communs, les féminins des adjectifs.
EC	M. Edmond Chaboche, inventeur de Métaguide.
Expression	Suite de mots ayant une signification propre, distincte des mots qui la constituent. Par exemple : « voiture de collection ».
Expression particulière	Expression dont le sens conduit à ne pas l'utiliser pour une recherche libre mais à déclencher une action séparée. Par exemple « vendre maison » pourrait conduire à créer une fiche au lieu d'en rechercher.
Full text	Concept de recherche dans une masse de données constituée de documents textuels en langage naturel ou approchant, et basé sur la prise en compte pour la recherche de tous les textes disponibles dans les données. Synonyme de recherche libre.
GLADIA	Système de gestion de fichiers « maison » qui permet de décrire des fichiers indépendants constitués de champs fixes, munis d'index B-Tree, d'une recherche multicritère et d'une recherche full-text phonétique sur les mots français écrits en majuscules.

HTTP	HyperText Transfer Protocol, protocole d'échange de requêtes et de réponses principalement utilisé entre les serveurs Web et les navigateurs.
Internaute	Néologisme. Usager d'Internet ; usager de services disponibles sur le Web.
ISAM	Indexed-Sequential Access Mode. Système de gestion de fichiers organisant les fiches, ou leurs clés, par blocs triés indépendamment et accessibles par des index en arbre.
Lien vers un site Web	L'un des champs descriptifs présents dans les fiches gérées par Métaguide. Ce lien est facultatif, c'est l'une des informations qu'il est possible de préciser dans la fiche.
MOA	Maître d'ouvrage
MOE	Maître d'œuvre
Mot-clé	Mot saisi par l'usager et recherché dans certains champs des fiches d'un fichier, pour extraire les fiches concernées.
Mot faible	Mot qui a une fonction principalement grammaticale (le, la, des, je, avec...) et qui n'apporte pas d'information utile dans une recherche « full text ». Ces mots sont habituellement éliminés avant la recherche.
Open source	Logiciel libre, dans le sens où l'obtention du logiciel exécutable emporte juridiquement la possibilité d'obtenir le code source, ce qui rend théoriquement l'utilisateur indépendant du producteur pour la maintenance et l'évolution (mais l'absence presque systématique de documentation interne limite fortement cette indépendance). En pratique, le code source concerné est presque toujours gratuit.
Osmose	Composant de Métaguide qui gère des listes chaînées à deux dimensions en mémoire centrale. Ce composant est utilisé dans la plupart des interfaces et en particulier dans la réception des requêtes HTTP, la génération de pages Web et la mémorisation des résultats des requêtes SQL.
Pay-per-view	Technique de vente en ligne consistant à débloquer l'accès à un affichage (fiche, film...) après paiement en ligne.
POSIX	Standard IEEE inspiré du système UNIX. Il spécifie un groupe d'interfaces logicielles et de programmes permettant de développer des logiciels portables.
Recherche libre	Concept de recherche dans une masse de données, basé sur la saisie de mots (dits « mots-clés ») ou de phrases en langage naturel. Synonyme de full-text.
Réponse	L'une des fiches sélectionnées par le processus de recherche, qu'elle soit guidée par thésaurus ou par mots-clés.
SGBD, SGBDR	Système de Gestion de Bases de Données (Relationnel). Le logiciel qui organise les bases, tables, données et droits d'accès, et qui en particulier sert les requêtes de lecture, écriture, suppression, verrouillage, etc.

Shell	Langage de commande de Linux et d'autres variantes d'UNIX, qui permet de passer des commandes sur une console mais aussi d'écrire des programmes interprétés sophistiqués.
Site appelant, Site Web appelant	Site Web qui utilise les services Métaguide en envoyant une requête constituée de mots à rechercher et attendant en retour une liste de codes de fiches trouvées. Les fiches ont été préalablement fournies à Métaguide par une interface d'importation. (voir §1.3)
Smartphone	Téléphone à écran graphique miniature, combiné avec un organiseur multifonction (agenda, tableur, messagerie...) et en général un accès internet.
Tuple (ou t-uplet)	liste ordonnée d'éléments. Dans le domaine des bases de données, cela désigne la liste des valeurs contenues dans une fiche (un enregistrement).
URL	Uniform Resource Locator, chaîne de caractères normalisée identifiant un serveur réseau et une ressource (document, traitement) sur ce serveur.
Usager, utilisateur	Dans ce document, le terme « usager » désigne l'utilisateur final, celui qui va effectuer des recherches sur les fichiers « métier » mis à sa disposition ou insérer des fiches. Le terme « utilisateur » désigne tous les utilisateurs de Métaguide, y compris les exploitants, les usagers, les intermédiaires le cas échéant.
Web	L'ensemble des serveurs d'information et de données connectés à Internet, et liés entre eux par des références URL croisées, en particulier via les moteurs de recherche. Certains l'appellent « la toile » en français.

Médiagraphie

[1] http://phppgadmin.sourceforge.net phppgadmin, interface graphique par navigateur pour le SGBD PostgreSQL.

[2] http://www.postgresql.org/docs/8.4/static/index.html Documentation de PostgreSQL, en anglais.

[3] http://docs.postgresql.fr/8.4/ Documentation de PostgreSQL, en français.

[4] https://www.virtualbox.org/ Logiciel de virtualisation permettant de simuler, sur un unique ordinateur, plusieurs ordinateurs exécutant des systèmes d'exploitation variés.

[5] Brevet français de M. Edmond Chaboche n°00 04702 / FR 2 807 849 – *Procédé et système de recherche et d'aide au choix*.

[6] https://en.wikipedia.org/wiki/Word-sense_disambiguation Aperçu des concepts de reconnaissance du sens des mots, références à d'autres articles et travaux.

[7] https://en.wikipedia.org/wiki/Concept_Search Aperçu des systèmes de recherche sémantiques.

[8] DUVERT F., 2010. *Composition sémantique pour la langue orale*. Thèse de Doctorat en Informatique, Université d'Avignon et des Pays de Vaucluse, 201 p.

[9] Rosenblum L., 2013. La fusion des sens. *Pour la Science*, mai 2013, pp 60-63.

[10] Projet LUNA, 2006-2009. Spoken Language Understanding in Multilingual Communication Systems, http://www.ist-luna.eu.