University Denis Diderot, Paris VII,
Case 7014 75205 Paris Cedex 13
Tel : +33(0)1.57.27.92.56
Fax : +33(0)1.57.27.94.09

Cahier de Charges
iFind

Equipe :
Isabelle Richard
Elias Abou Haydar
Mikael Ahl
Jeremie Rouach

Chef d'équipe :
Elias Abou Haydar

Table des matières

1 Introduction

1.1 Objectif du document de specification

Ce document de spécification produit des informations spécifiques et nécessaires pour définir efficacement les fonctionnalités, l'architecture et la conception du système afin de donner la direction à l'équipe de développement sur l'architecture du système à développer. Le document de spécification du produit est créé pendant la phase de planification du projet. Son public visé est le chef de projet, l'équipe de projet et l'équipe de développement et en partie le client. Les spécifications techniques et fonctionnelles de ce document sont réservées au chef de projet, l'équipe de projet et l'équipe de développement.

1.2 Portée du produit

Le logiciel iFind permet de rechercher un fichier dans un ensemble de
répertoires ciblés du système. Cette recherche peut se faire soit en indiquant le nom du fichier, soit en donnant une liste de mots contenus dans ce fichier.

1.3 Définitions, acronymes et abréviations

UI Acronyme de "user interface" (interface utilisateur).

Corpus Un corpus est un ensemble de documents, artistiques ou non (textes, images, vidéos, etc.) , regroupés dans une optique précise.

Fichier Contenant virtuel auquel est assigné un nom unique, permettant de classifier et de réunir en une même entité une séquence de données. Le fichier est stocké dans un système de fichier et les données qu'il contient sont généralement structurées en suivant un même format.

Document En informatique, le mot *document* est généralement synonyme de fichier. On parle ici de document électronique. Un document électronique est un contenu de médias électroniques (autres que les programmes d'ordin
-ateur ou des fichiers système) qui sont destinés à être utilisés soit dans une forme électronique ou comme sortie imprimée.

Indexation L'indexation permet de regrouper en un seul endroit toutes les données souhaitées. On crée des indexes, ce qui permet d'y accéder plus rapidement.

Index Un index est, en toute généralité, une liste de descripteurs à chacun desquels est associée une liste des documents et/ou parties de documents auxquels ce descripteur renvoie. Lors de la recherche d'information d'un usager, le système accèdera à l'index pour établir une liste de réponses.

Moteur de recherche Un moteur de recherche est un code logiciel qui est conçu pour rechercher des informations ou retrouver des ressources associées à des mots quelconques. Ici, on parle de moteur de recherche de type " Desktop " , car son champ d'action est limité à l'ordinateur sur lequel l'application est installée.

Requête En informatique, une requête est une demande de traitement. Dans notre cas, le terme est employé dans le contexte des bases de données, une requête correspondant à l'interrogation d'une base pour en récupérer une certaine partie des données.

Base de données Une base de données, usuellement abrégée en BD ou BDD, est un ensemble structuré et organisé, permettant le stockage de grandes quantités de d'informations afin

d'en faciliter l'exploration (ajout, mise à jour, recherche de données). Autrement dit, il s'agit d'un conteneur informatique permettant de stocker dans un même endroit l'intégralité des informations en rapport avec une activité. Une base de données permet de stocker un ensemble d'informations de plusieurs natures ainsi que les liens qu'il existe entre les différentes natures.

Expression régulière Les expressions régulières (aussi appellées expressions rationnelles) sont de chaines de caractères permettant de décrire un ensemble de variables par l'utilisation d'une syntaxe précise qui se retrouvent dans de nombreux langages et outils.

Démon Un démon ou daemon désigne un type de programme informatique, un processus ou un ensemble de processus qui s'exécute en arrière-plan plutôt que sous le contrôle direct d'un utilisateur.

1.4 References

IEEE Std 830-1998 Recommended Practice for Software Requirements Specifications

2 Aperçu Général et Guide de Conception

Cette section décrit les principes et les stratégies qui seront utilisées comme des lignes directrices lors de la conception et de la mise en œuvre du système.

2.1 Perspective du produit

iFind utilise une base de données construite à l'aide d'un moteur d'indexation et mise à jour dès qu'un fichier est modifié. La requête est envoyée au moteur de recherche via une interface graphique (GUI) (voir Figure 1). On utilise une interface graphique, pour permettre à l'utilisateur de rechercher ce dont il a besoin. La GUI est constituée d'un champ de saisie, d'un bouton "Chercher" ainsi que d'un explorateur qui permettra d'ouvrir les fichiers trouvés (extension). De base, l'explorateur contiendra un tableau dans lequel on affichera les résultats. Le champ de saisie reçoit une requête sous forme d'expression régulière ou des mots simples.

Dans cet exemple, la recherche envoie tous les fichiers contenant les mots "toto" ou "abc".

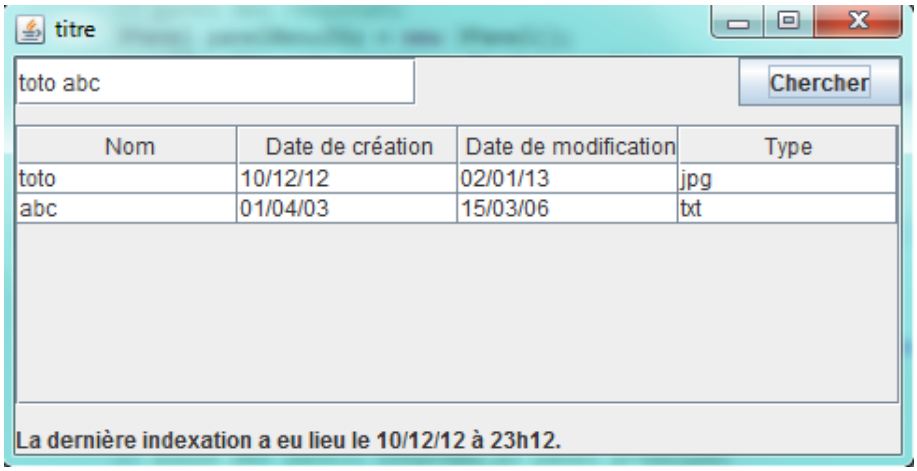


FIGURE 1 – Interface graphique du moteur de recherche.

2.2 Fonctionnalités du produit

Lors de la première utilisation, iFind lance un démon ayant pour tâche d'indexer un corpus ciblé. Ce démon va construire une base de données à l'aide de ces index. Un algorithme est appliqué pour identifier dans le corpus (en utilisant l'index), les fichiers qui correspondent le mieux aux mots contenus dans la requête, afin de présenter les résultats des recherches par ordre de pertinence. Cette base de données va jouer un rôle clé dans la phase de recherche de fichiers. Ensuite, à chaque création ou modification de fichier appartenant au corpus, le démon met à jour la base de données en fonction des modifications du fichier.

2.2.1 Utilisation normale

L'utilisateur entre une requête dans la barre de recherche en suivant la syntaxe du moteur de recherche (cf partie 2.3.2).
TODO (retour de la requête)

2.2.2 Utilisation avancée

La recherche avancée permet d'affiner la recherche de document en se basant sur un critère de type. Un utilisateur pourra rechercher des morceaux de musique par artiste, album ou titre du morceaux. Il pourra aussi recherches des images. Le moteur de recherche dans ce cas cherche à filtrer les résultats pour faire en sorte que les résultats affichés en sortie corresponde juste à des fichiers de type image. (jpg,jpeg,bmp)

Concernant les documents textuelles, l'interface de recherche de ces documents permet de rechercher un document par auteur du document, genre, extension et la date de modification. La pertinence des résultats obtenus dans ce cas depend de la disponibilité de ce information par rapport a ces documents. C'est à dire, si un genre de document n'est par fourni, et on souhaite rechercher un genre précis du documents.

Il y a des chances que le moteur de recherche ne puisse pas retrouver le document recherché même si celui ci existe sur le disque.

Dans le cas où le document recherché n'est pas présent, un message apparaîtra afin de guider l'utilisateur. De plus, la date et l'heure de la dernière indexation est indiqué.

La partie inférieure de la fenêtre permet de visualiser la liste des documents trouvés. Cette dernière se compose d'une colonne indiquant le nom du document, une deuxième indiquant sa date de création, une troisième indiquant sa date de modification (la plus récente) et enfin la dernière indiquant le type.

L'utilisateur peut également entrer une requête incluant des critères spéciaux sur les fichiers à rechercher :

- l'auteur
- la date de création
- la dernière date de modification
- le type
- la taille (pour les fichiers de type image)
- la durée (pour les fichiers de type musique ou vidéo)

Il est possible de paramétrer les fonctionnalités suivantes de l'indexation :

- Indexation ciblée : on délimite l'emplacement du corpus à indexer.
- Indexation filtrée : on peut choisir le type de fichier à indexer.

2.3 Caractéristique d'utilisation

2.3.1 Grammaire BNF

Dans ce paragraphe, on introduira la grammaire qui définira le langage de requêtes que on associe au moteur de recherche.

Un mot est formé de chiffres et/ou de lettres minuscules ou majuscules, possiblement accentuées.

Une requête doit obligatoirement commencer par un mot.

Dans le cas d'une recherche multiple, la liste de mots à rechercher est soit délimitée par des espaces (dans le cas d'une conjonction), soit délimitée par le marqueur **-or** (dans le cas d'une disjonction).

La requête peut être complétée par des options :

– la recherche par type de fichiers, avec le marqueur **-f**
– l'exclusion d'une liste de mots, avec le marqueur **-e**

Grammaire BNF

Dans la suite, les symboles terminaux seront notés en **gras**. Les symboles non-terminaux seront notés en *italique*. Une séquence entre crochets est optionnelle (comme par exemple "*[* word *]*").
Une séquence entre accolades se répète zéro fois ou plus, (comme par exemple "*{* , word *}*").
Sauf indication contraire, la sémantique d'une séquence est une liste dont la tête est le premier élément de la séquence.

char	≡	[a-z A-Z 0-9 àáâãäåçèéêëìíîïðòóôõöùúûüýÿ]
word	≡	**char**$^+$
query	::=	**word** *{* **-or word** *}* *[* *options* *]*
	|	**word** *{* **word** *}* *[* *options* *]*
options	::=	**-e word** *{* **word** *}*

2.3.2 Syntaxe du moteur de recherche

Casse des caractères	iFind ne tient pas compte de la casse des caractères. Exemple : Les requêtes *ibm*, *Ibm* et *IBM* renvoient le même résultat.
Lettres accentuées	Les lettres accentuées sont prises en charge par contre la recherche ne donne pas le même résultat, même si les différences sont souvent minimes.
Ordre des mots	On considère une relation d'ordre sur la requête, les mots de la requête sont ordonnés de gauche à droite. Exemple : *paris brest* donne un résultat différent de *brest paris*.
Conjonction	**Opération par défaut**. La requête est constituée d'une liste de mots séparés par des espaces. Exemple : *moteur recherche* La requête privilégiera les résultats contenant *moteur* **et** *recherche*, puis les résultats contenant seulement *moteur*, et enfin les résultats contenant seulement *recherche*.
Disjonction	La requête est constituée d'une liste de mots séparés par la marque **-or**. Les fichiers trouvés par la recherche contiennent un mot seulement parmi la liste, contrairement à la conjonction. L'opérateur doit être saisi avec un tiret obligatoirement.

	Exemple : *machin -or bidon* La requête privilégiera les résultats contenant seulement *machin*, puis les résultats contenant seulement *bidon*.
Exclure un mot	Tout mot ajouté après la marque **-e** dans la requête indiquera que les résultats contenant ce mot sont à exclure. Exemple : *moteur –e automobile essence* La requête donnera uniquemenet les résultats contenant *moteur* mais **ne contenant ni** *automobile*, **ni essence**. Remarque l'option -e doît être précédée par un au moins un mots
Expressions exactes	La recherche d'expressions exactes n'est pas possible.
Troncature	Il n'est pas possible de faire des recherches en utilisant la troncature sur iFind. Le moteur recherche toujours exactement le mot demandé. L'astérisque (*) ne peut pas être utilisée. iFind tient cependant parfois compte de la troncature, sans qu'il soit possible pour l'utilisateur de décider quand. Exemple : La requête *mot* ne cherche ni *mots* ni *moteur*.

2.3.3 Messages d'erreur possibles

Si la requête entrée par l'utilisateur est mal formée, le message suivant apparaît : "La requête entrée ne respecte pas la syntaxe. Cliquez sur Aide pour plus d'informations." (Cliquer sur Aide ouvre une petite fenêtre contenant les règles de syntaxe d'une requête.) L'utilisateur est alors invité à reformuler sa requête.

Si la base de données est corrompue et entraîne l'arrêt de la recherche et de l'indexation, l'utilisateur est invité à la réinitialiser : "La base de données a été corrompue et doit être intégralement reconstruite. Cette opération peut prendre du temps. Voulez-vous réinitialiser la base de données maintenant ? Si vous cliquez sur Annuler, la base de données ne sera pas reconstruite et iFind ne sera plus utilisable." ((OK) (Annuler) <- boutons)

2.4 Contraintes générales

La fraîcheur des résultats dépendra de la fraîcheur de la dernière indexation faite.
Le modification du contenu d'un fichier n'est prise en compte qu'au moment de sa fermeture. Si le contenu d'un fichier est modifié mais qu'une recherche est lancée alors que ce fichier n'a pas encore été quitté, la recherche ne prendra pas en compte son nouveau contenu.
Un fichier supprimé ne sera jamais retourné comme résultat d'une recherche, la suppression étant immédiatement traitée.

2.5 Dépendances

Le logiciel sera utilisable sur une distribution GNU/Linux.
JNotify et Hibernate3 possède une licence GNU LESSER GENERAL PUBLIC LICENSE Version 2.1, February 1999.

3 Spécification du Système

Cette partie du cahier de charge présentera les trois axes principaux du développement de ifind.

- Architecture du système.
- Charges du système.
- Modèles dy système.

3.1 Architecture du Système

Ce paragraphe présente un point du vue haut niveau de l'architecture préconisée du système et la distribution de fonctionnalités à travers les modules du système.

Notre système ce compose de trois sous-système :

- Moteur de recherche : Architecture Model View Controler
- Moteur d'indexation : Deamon
- Base de donnée indexée : Architecture en couche : PostgreSQL + surcouche de gestion de la base de donnée : Hibernate

3.1.1 Architecture du Moteur de recherche

BNF Nous avons commencé à réfléchir à la forme qu'aurait la requête de recherche. C'est pourquoi nous avons défini une syntaxe et à fortiori, une grammaire sous forme BNF.

Grammaire (BNF) :

query	→	word -or word [options]
query	→	word word [options]
options	→	-e word word
word	→	char*
char	→	[a-z A-Z 0-9 à é è ë ä ù ï ç]

Par la suite, nous avons défini plusieurs types abstraits de données permettant de modéliser la grammaire. Un type a forcément une ou plusieurs opérations de construction. Chaque type peut présenter des fonctions d'accès et de test ainsi que des constantes.

Type de données abstraites En effet, nous avons définit 6 types tels que :

Type : requête

_construction :	cons-requete-word : word → requête
	cons-requete-word-options : word * options → requête
	cons-requete-conj : conj → requête
	cons-requete-conj-options : conj * options → requête
	cons-requete-disj : disj → requête
	cons-requete-disj-options : disj * options → requête
_test :	est-word : requête → bool
	est-conj : requête → bool
	est-disj : requête → bool
	a-options : requête → bool
_accès :	get-word : requête → word
	get-conj : requête → conj
	get-disj : requête → disj
	get-options : requête → options

On garde get-conj() si est conj : get-word() puis recherche conjonction et get-disj() si est disj : get-word() puis recherche disjonction.

Type : conj

_construction :	cons-conj : word * conj → conj
	cons-conj-simple : word * word → conj
_test :	est-conj : conj → bool
_accès :	get-head : conj → word
	get-tail : conj → conj

Type : disj

_construction :	cons-disj : word * disj → disj
	cons-disj-simple : word * word → disj
	accès : getDisj()

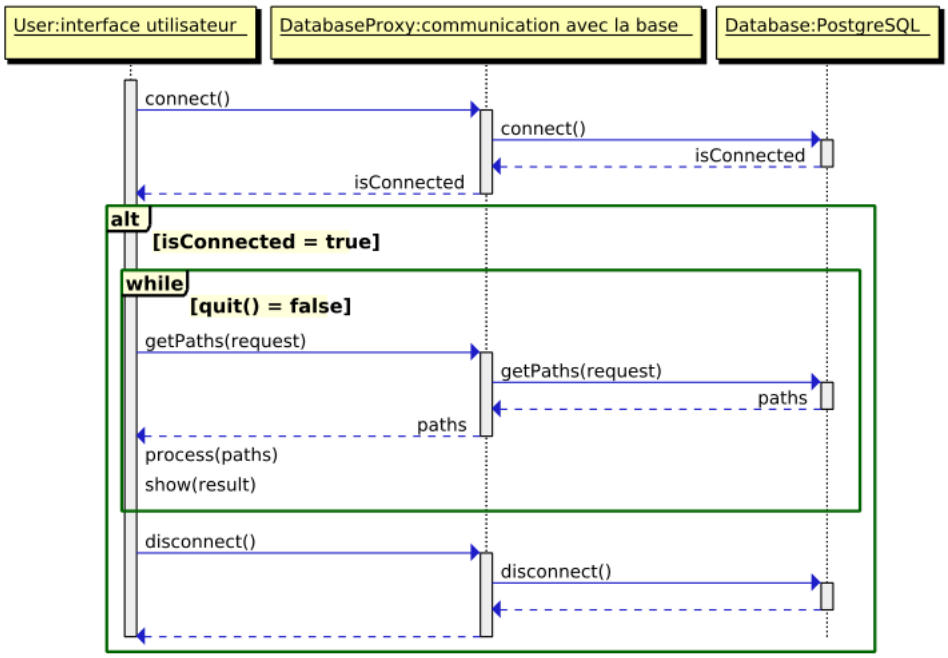
Type : word

_construction :	cons_word : string → word
_accès :	get-word : word → string

Type : options

_construction :	cons-options : word → options
	cons-options-simple : word * word → options
_test :	est-options : options → bool
_accès :	get-head : options → word
	get-tail : options → options

Model :
View : GUI
Controler : search-engine-core



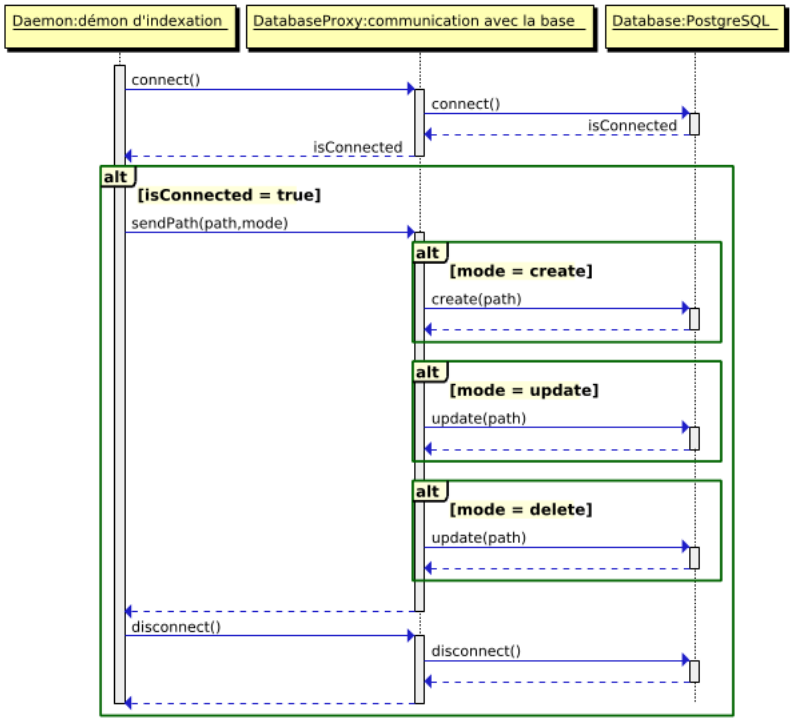
3.1.2 Architecture du moteur d'indexation

Il s'agit juste d'un démon parametrable qui construit en premier temps une base d'indexation ensuite il communique avec celle-ci à l'aide d'un protocole de communication par socket.

La communication entre la base d'indexation et le moteur d'indexation a pour but la mis-à-jour de la BD.

3.1.3 Architecture de la base d'indexation

La base d'indexation est une base de donnée PostgreSQL, en un premier temps, qui permettra au moteur de recherche de retrouver ce qu'il cherche.



3.2 Charges du Système

Ce paragraphe décrit les charges fonctionnelles du système en détail et d'autres charges non fonctionnelles :

- interface
- autres...

3.3 Modèles du système

Ce paragraphe décrit les divers modèles du système ainsi que leurs relations entre composantes du système et de son environnement.

3.3.1 Modèle de communication

le module de BD crée deux sockets Server :
L'instanciation de ces sockets doit se faire avec les informations lues à partir des fichier config-sample-Deamon.XML et config-sample-MoteurR.

On définit tout d'abord la socket entre la base d'indexation et le moteur de recherche. Le socket reste en écoute des requêtes en provenance du moteur de recherche, une requête est un fichier XML générée suite à une seule et unique recherche pour cela on doit impérativement ajouter un identifiant dans la DTD au retour. Chaque réponse génère un seul fichier XML qui sera envoyé au moteur de recherche on peut facilement retrouver l'ordre grâce au ID de chaque requête.

On distingue un cas particulier : par exemple l'envoie des résultats en morceaux , quand la recherche prend éventuellement du temps. Là aussi il faut ajouter des informations supplémentaires dans la DTD pour qu'on puisse rendre possible ce genre d'opérations.

On définit ensuite la socket entre la base d'indexation et le moteur d'indexation. Le socket reste en écoute de nouvelle information capturée par le démon (Daemon). Chacun peut choisir sa politique de mise à jour (envoie du fichier XML chaque 20 min par ex etc .) On doit se mettre d'accord sur le format d'échange pour ces deux modules.

Remarques : J'ai choisi les ports de façon arbitraires il y a 3 ports le premier et celui par défaut si jamais ce port n'est pas libre ont basculera vers le 2e etc.

Remarque 2 : La DTD bd2M. n'est pas encore assez riche pour gérer une recherche dont le nombre de mots-clés et supérieur à 1 comme on l'a vu au TD. Il faut prendre en considération tous les cas possibles, exemple : mot1 ou mot2 , mot1 et mot2, etc.

Je propose qu'on rajoute de nouveaux éléments en dehors de la balise search qui concerne la manière dont on va l'interpréter.