

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 | 2 |
| 1.1 | Objectif du document de specification | 2 |
| 1.2 | Portée du produit | 2 |
| 1.3 | Définitions, acronymes et abréviations | 2 |
| 1.4 | References | 3 |
| 2 | Aperçu Général et Guide de Conception | 4 |
| 2.1 | Perspective du produit | 4 |
| 2.2 | Fonctionnalités du produit | 4 |
| 2.2.1 | Utilisation normale | 4 |
| 2.2.2 | Utilisation avancée | 5 |
| 2.3 | Caractéristique d'utilisation | 5 |
| 2.3.1 | Grammaire BNF | 5 |
| 2.3.2 | Syntaxe du moteur de recherche | 6 |
| 2.3.3 | Messages d'erreur possibles | 7 |
| 2.4 | Contraintes générales | 7 |
| 2.5 | Dépendances | 7 |
| 3 | Spécification du Système | 9 |
| 3.1 | Architecture du Système | 9 |
| 3.1.1 | Architecture du Moteur de recherche | 9 |
| 3.1.2 | Architecture du moteur d'indexation | 11 |
| 3.1.3 | Architecture de la base d'indexation | 11 |
| 3.2 | Charges du Système | 13 |
| 3.3 | Modèles du système | 13 |
| 3.3.1 | Modèle de communication | 13 |
| 4 | Annexe | 13 |
| 4.1 | Communication du moteur d'indexation vers la base d'indexation (DTD) | 13 |

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'ordinateur 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 index, 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 appelées expressions rationnelles) sont de chaînes 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.

Evènement Un évènement est une action qui a lieu sur un fichier. Les différentes actions possibles sont : création, modification, renommage et suppression.

DTD La Document Type Definition (DTD), ou Définition de Type de Document, est un document permettant de décrire un modèle de document SGML ou XML. Le modèle est décrit comme une grammaire de classe de documents : grammaire parce qu'il décrit la position des termes les uns par rapport aux autres, classe parce qu'il forme une généralisation d'un domaine particulier, et document parce qu'on peut former avec un texte complet.

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 tous les dix événements. 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".

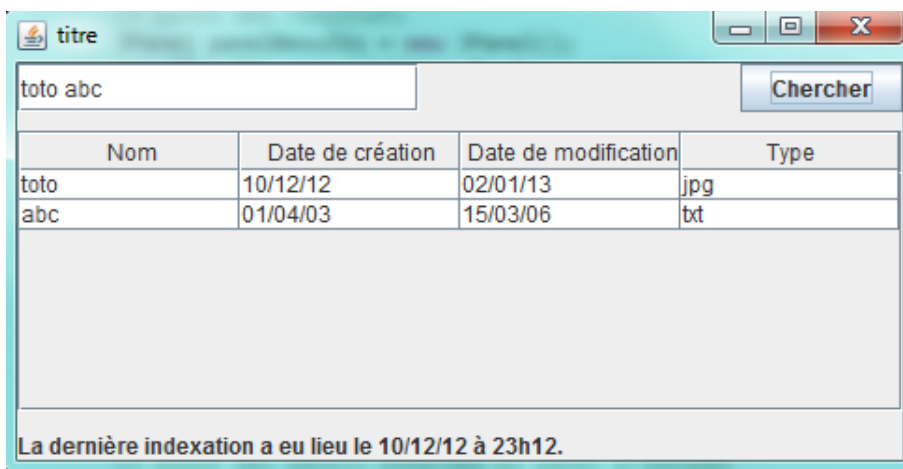


FIGURE 1 – Interface graphique du moteur de recherche.

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 envoyer l'ensemble des données à indexer à la base d'indexation, qui construira la base de données utilisée pour les recherches. Cette base de données va jouer un rôle clé dans la phase de recherche de fichiers. 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. Ensuite, tous les dix événements détectés par le démon, celui-ci envoie les nouvelles informations à indexer à la base d'indexation.

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 l'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      ≡    [ :alnum :]
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 <i>ibm</i> , <i>Ibm</i> et <i>IBM</i> 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 : <i>paris brest</i> donne un résultat différent de <i>brest paris</i> . |
| Conjonction | Opération par défaut. La requête est constituée d'une liste de mots séparés par des espaces. Exemple : <i>moteur recherche</i> La requête privilégiera les résultats contenant <i>mo-teur et recherche</i> , puis les résultats contenant seulement <i>moteur</i> , et enfin les résultats contenant seulement <i>recherche</i> . |
| 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. |

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

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." (Cliquez 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. L'indexation a lieu tous les dix événements (par exemple, dix créations de fichier). Il peut arriver qu'un fichier supprimé soit retourné comme résultat d'une recherche, mais seulement si le fichier a été supprimé il y a moins de dix événements.

2.5 Dépendances

Le logiciel iFind est utilisable sur une distribution GNU/Linux. Il est nécessaire d'installer PostgreSQL pour utiliser le logiciel.

JNotify 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 se compose de trois sous-systèmes :

- Moteur de recherche : Architecture Model View Controler
- Moteur d'indexation : Deamon
- Base de donnée indexée : Architecture en couche : PostgreSQL

3.1.1 Architecture du Moteur de recherche

Le moteur de recherche (MR) communique avec la base d'indexation (BI) par l'intermédiaire d'un flux XML au travers d'une Socket. Le flux XML envoyé par le MR est associé à un identifiant, qui permet de savoir à quelle requête est associée quelle réponse. L'envoi respecte la DTD (voir annexe).

BNF Nous avons défini une syntaxe et à fortiori, une grammaire sous forme BNF.

Grammaire (BNF) :

```
char      ≡    [ :alnum :]
word      ≡    char+
query     ::=  word { -or word } [ options ]
           |    word { word } [ options ]
options   ::=  -e word { word }
```

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

```

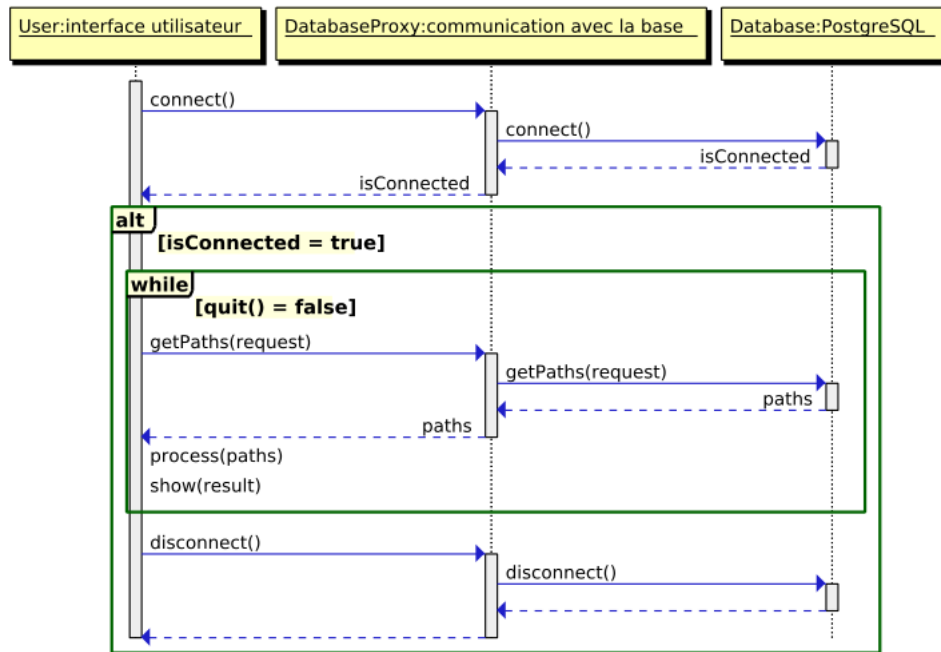
    _construction :   cons-options : word → options
                      cons-options-simple : word * word → options
    _test :   est-options : options → bool
    _accès :   get-head : options → word
              get-tail : options → options

```

Model :

View : GUI

Controler : search-engine-core



3.1.2 Architecture du moteur d'indexation

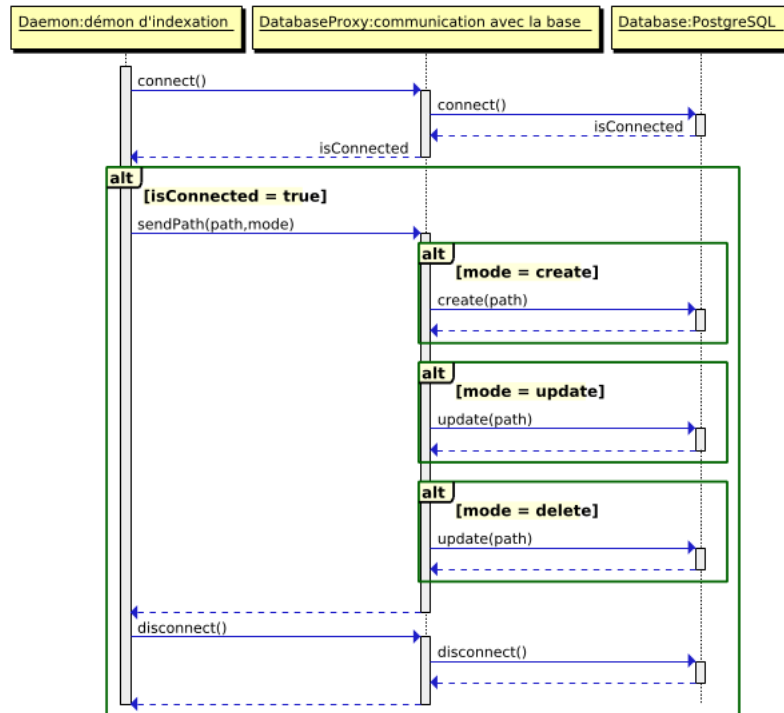
Le moteur d'indexation se compose de deux instances.

La première est lancée uniquement lorsqu'il faut entièrement créer l'index (lors de l'installation, par exemple).

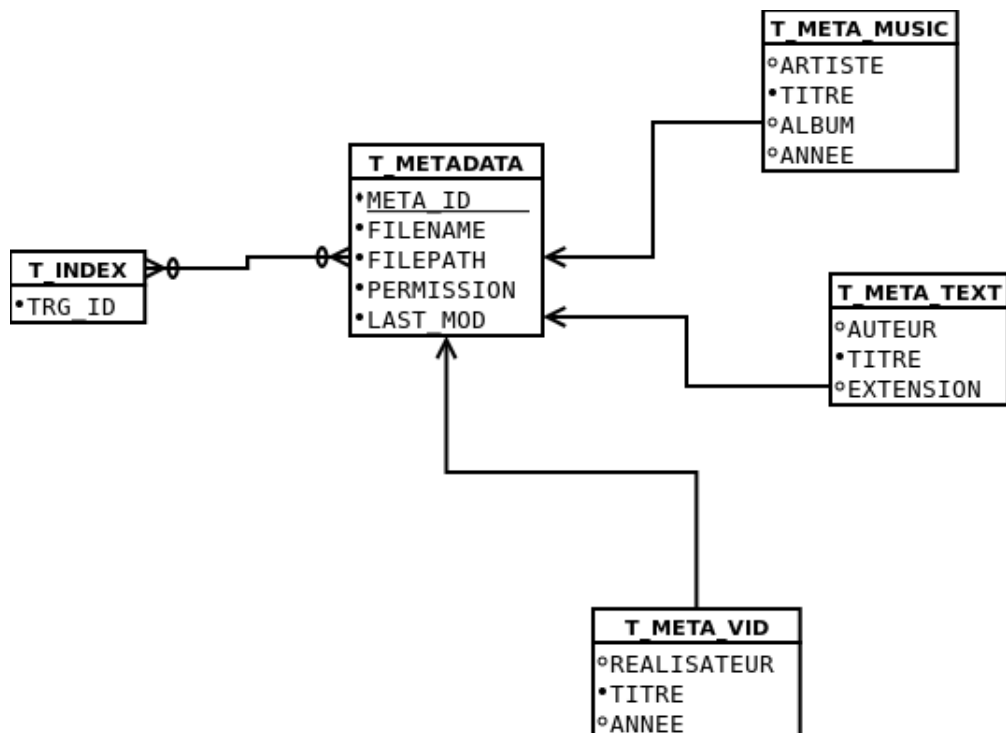
La deuxième, le démon, communique en permanence avec la base d'indexation afin de lui envoyer les différents événements qui ont eu lieu à l'intérieur du corpus et qui doivent être indexés. L'envoi se fait par l'intermédiaire d'un flux XML au travers d'une Socket. L'envoi respecte la DTD (voir annexe) établie par les chefs de projets pendant les réunions.

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.



Le modèle conceptuel de données (MCD) de la base de données est le suivant :



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 base d'indexation écoute sur deux ports différents, un pour la communication avec le moteur d'indexation, et un pour la communication avec le moteur de recherche.

Communication avec le moteur d'indexation La base d'indexation (BI) attend qu'un moteur d'indexation (MI) se connecte sur le port 40000. Lorsque la connexion est établie, la BI lit le flux XML envoyé par le MI. La connexion entre le MI et la BI n'est fermée que lorsque l'un des deux modules est fermé.

Communication avec le moteur de recherche La base d'indexation (BI) attend qu'un moteur de recherche (MR) se connecte sur le port 30000. Lorsque la connexion est établie, la BI lit le flux XML envoyé par le MR. La BI génère une réponse au flux et la renvoie au MR, via le port 30000 sous forme de XML.

4 Annexe

4.1 Communication du moteur d'indexation vers la base d'indexation (DTD)

```
<!-- règle générale :
chaque fichier Xml doit contenir la ligne suivante
<?xml version="1.0" encoding="UTF-8"?> -->

<!-- comme convenue toutes les balises de l'indexation sont facultatives -->
<!ELEMENT INDEXATION (RENOMMAGES?, MODIFICATIONS?, SUPPRESSIONS?, CREATIONS?)>

<!ELEMENT RENOMMAGES (FICHIERRENOMME)*>
<!ELEMENT MODIFICATIONS (FICHIERMODIFIE)*>
<!ELEMENT SUPPRESSIONS (FICHIERSUPPRIME)*>
<!ELEMENT CREATIONS (FICHIERCREE)*>

<!ELEMENT FICHIERRENOMME (PATH, NEWPATH)>
<!-- un fichier renommé n'est pas nécessairement un fichier modifié -->
<!-- un fichier modifié nécessite une re-indexation -->
<!ELEMENT FICHIERMODIFIE (PATH, DATEMODIFICATION, TAILLE, PROPRIETAIRE, GROUPE,
PERMISSIONS, INDEXAGE, NEWPATH?)>
<!ELEMENT FICHIERSUPPRIME (PATH)>
<!ELEMENT FICHIERCREE (PATH, format, DATECREATION, TAILLE, PROPRIETAIRE, GROUPE,
PERMISSIONS, INDEXAGE)>

<!-- les meta-données -->
<!ELEMENT PATH (#PCDATA)>
<!ELEMENT format (#PCDATA)>
<!ELEMENT DATECREATION (#PCDATA)>
<!ELEMENT DATEMODIFICATION (#PCDATA)>
<!ELEMENT TAILLE (#PCDATA)>
<!ELEMENT PROPRIETAIRE (#PCDATA)>
<!ELEMENT GROUPE (#PCDATA)>
<!ELEMENT PERMISSIONS (#PCDATA)>
<!-- liste des mots a indexer -->
<!ELEMENT INDEXAGE (MOT*)>
<!ELEMENT MOT (#PCDATA)><!ATTLIST MOT frequence CDATA #REQUIRED>
<!ELEMENT NEWPATH (#PCDATA)>

<!-- les id's seront utilisés pour la traçabilité et la détection d'éventuel
erreurs, elle sont tout de même facultatives -->
<!ATTLIST RENOMMAGES id CDATA #IMPLIED>
<!ATTLIST MODIFICATIONS id CDATA #IMPLIED>
<!ATTLIST SUPPRESSIONS id CDATA #IMPLIED>
<!ATTLIST CREATIONS id CDATA #IMPLIED>
```

4.2 Communication du moteur de recherche vers le moteur d'indexation (DTD)

```
<!-- Description de la requête de recherche -->
<!ELEMENT SEARCH (WORD, CONTENT?, PATHDIR?, PERM?, EXTENSION?, TIMESLOT?)>
<!-- Le mot à rechercher -->
<!ELEMENT WORD (#PCDATA)>
<!-- Un booléen qui dit si l'on fait une recherche de contenu (true) ou une
recherche sur les nom de fichier (false) -->
<!ELEMENT CONTENT (#PCDATA)>
<!-- Le nom du fichier à partir duquel on recherche -->
<!ELEMENT PATHDIR (#PCDATA)>
<!-- Les permissions du fichier à chercher -->
<!ELEMENT PERM (#PCDATA)>
<!-- L'extension des fichiers à chercher -->
<!ELEMENT EXTENSION (#PCDATA)>
<!-- Interval de temps -->
<!ELEMENT TIMESLOT (BEGIN, END)>
<!-- Debut de l'interval -->
<!ELEMENT BEGIN (DAY, MONTH, YEAR)>
<!-- Fin de l'interval -->
<!ELEMENT END (DAY, MONTH, YEAR)>
<!-- Le jour -->
<!ELEMENT DAY (#PCDATA)>
<!-- Le mois -->
<!ELEMENT MONTH (#PCDATA)>
<!-- L'année -->
<!ELEMENT YEAR (#PCDATA)>

<!-- L'id du resultat correspondant à une recherche aura le même id que la
recherche -->
<!ATTLIST SEARCH id CDATA #REQUIRED>
```

4.3 Communication de la base d'indexation vers le moteur de recherche (DTD)

```
<!-- Balise contenant les résultats de la requête search -->
<!ELEMENT RESULT (FILE*)>
<!-- Balise file correspond à un fichier resultat donc n balises files = n
résultats -->
<!ELEMENT FILE (NAME, PATH, PERM, SIZE, LASTMODIF?, PROPRIO?)>
<!-- Le nom du fichier -->
<!ELEMENT NAME (#PCDATA)>
<!-- Le chemin complet du fichier -->
<!ELEMENT PATH (#PCDATA)>
<!-- Les permissions du fichier -->
<!ELEMENT PERM (#PCDATA)>
<!-- La taille du fichier -->
```



```
<!ELEMENT SIZE (#PCDATA)>
<!-- La date de dernière modification du fichier -->
<!ELEMENT LASTMODIF (#PCDATA)>
<!-- Le propriétaire du fichier -->
<!ELEMENT PROPRIO (#PCDATA)>

<!-- L'id du resultat correspondant à une recherche aura le même id que la
recherche -->
<!ATTLIST RESULT id CDATA #REQUIRED>
```