INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES LYON

# SPECIFICATIONS DES BESOINS

## Génie logiciel – INSA Lyon 2017

### Objectif

Sur ce document, nous définissons les besoins auxquels l'application d'interprétation d'analyses médicales doit répondre. Il comporte donc le cahier des charges et les spécifications fonctionnelles et non fonctionnelles de notre application.

### Auteurs

Marc-Antoine Fernandes
Julia Lu Dac

**Version 2**
Date de création : 27/03/17
Dernière modification : 05/04/17

# Table des matières

# 1 Introduction

L'objectif premier de l'application est de déceler une liste de maladies potentielles sur une personne à partir d'une analyse médicale de son génome.

L'analyse médicale est sous la forme d'un tableau de mots. Une maladie est elle aussi sous la forme d'un tableau de mots. Ainsi, on repère un risque de maladie si tous les mots présents dans le tableau de mots de la maladie se situent dans le tableau de mots de l'analyse médicale. Autrement dit, soient :

- x = [x1,x1,...,xn] un tableau de n mots xi distincts représentant un génome
- y= [y1,y2,...,ym] un tableau de m mots yi distincts représentant une maladie.

Alors la personne dont le génome est x a le risque d'avoir la maladie y ssi :

$$\forall y_i \in y, \exists x_j \in x \; tel \; que \; y_i = x_j \; avec \; i \in [|1,m|] \; et \; j \in [|1,n|]$$

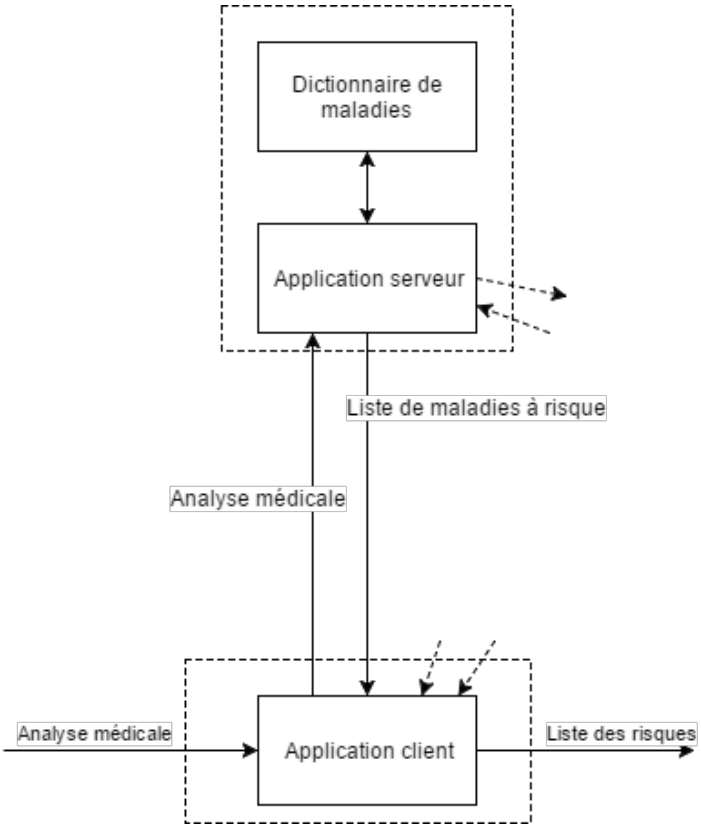
Exemple :

Entrée : l'analyse médicale = [1,2,3,4,5] et les maladies A : [6,1], B : [3,1,5], C : [4,1]

Sortie : la liste des maladies : [B, C]

Remarque : Il peut y avoir plusieurs risques de maladies détectées dans un même génome.

Voici une visualisation de l'application :




L'application à créer se divise en deux parties : la partie client qui servira d'interface à l'utilisateur. Et la partie serveur qui cherchera la présence potentielle de maladie dans un génome.

## 1.1 Côté client :

Le client est le laboratoire. Pour analyser des données, le laboratoire dispose de fichiers contenant les analyses médicales et d'un fichier contenant une liste de serveurs à interroger. L'application client fait des requêtes vers chacun des serveurs et récupère, pour chaque serveur, une liste de maladie (voir annexe I) ou, si le laboratoire veut tester la présence d'une maladie spécifique, un booléen (vrai si la maladie est présente, faux sinon) (voir annexe V).

### 1.1.1 Analyse complète

Dans le cas d'une analyse complète (recherche de maladie(s) potentielle(s)), une requête contient comme paramètre une analyse médicale (voir annexe I). A chaque interrogation, le serveur interrogé retourne une liste des risques de maladies qu'il a évalué sur l'analyse envoyée par le client.

### 1.1.2 Présence d'une maladie spécifique

Dans le cas d'un simple test de présence d'une maladie, une requête contient une analyse médicale ainsi qu'un nom de maladie. A chaque interrogation, le serveur interrogé retourne un booléen correspondant à la présence de la maladie.

## 1.2 Côté serveur :

Le serveur est la partie de l'application qui va traiter les analyses. Pour cela, il a besoin d'un dictionnaire contenant une liste de maladies avec, pour chaque maladie, une liste contenant elle-même des listes de mots. Cette liste correspond à un fichier au format JSON détaillé dans l'annexe II.

Le serveur évalue les risques d'un génome au travers d'une requête faite par un client. Il y a deux requêtes possibles : une première qui cherche l'ensemble des maladies potentiellement présentes dans une analyse, et une deuxième requête qui cherche la présence d'une maladie spécifique.

Ces requêtes sont détaillées respectivement dans les annexes I et V.

# 2 Exigences fonctionnelles

## 2.1 Partie serveur

### 2.1.1 Interprétation de l'analyse

Le système devra interpréter une analyse médicale et fournir une liste de maladies associée au génome présenté. Cela sera fait par l'application serveur.

Les génomes ainsi que les maladies sont représentés par une liste de mots alphanumériques.

Les maladies fonctionnent par système de clé/valeur, la clé est le nom de la maladie et la valeur est une liste de mot. Un dictionnaire peut contenir plusieurs fois une même maladie.

Si le génome contient l'intégralité d'une liste de mots de la maladie (de façon pas forcément ordonnée ni séquencée), la maladie associée doit être ajoutée à la liste des maladies détectées pour cette analyse.

### 2.1.2 Détection d'une maladie spécifique

Aussi, le système devra être capable de détecter si une certaine maladie est présente dans une analyse. Cette fonction est très similaire à l'interprétation de l'analyse. Au lieu de rechercher la présence de plusieurs maladies depuis une liste de maladies, le serveur vérifie la présence d'une maladie demandée par le client.

### 2.1.3 Exposition de la liste de maladies présente sur le serveur

Afin qu'un client sache quelles maladies sont présentes sur un serveur, le serveur doit exposer au client la liste des maladies qu'il dispose dans son dictionnaire. Le client peut obtenir cette liste grâce à une requête au serveur décrite en annexe VI.

## 2.2 Partie client

### 2.2.1 Chargement du génome

Le système, à travers l'application client, doit pouvoir charger un génome. Celui-ci sera charger depuis un fichier spécifique (voir annexe III).

### 2.2.2 Editer une liste de serveur

La liste des serveurs est une liste permettant à l'application client de traiter l'analyse. Cette liste est présente sous la forme d'un fichier spécifique (voir annexe IV). L'application doit permettre de modifier cette liste afin de pouvoir ajouter/modifier/enlever des serveurs.

# 3 Exigences non-fonctionnelles

## 3.1 Ergonomie

La prise en main doit s'effectuer de manière facile et l'application doit avoir une présentation conviviale.

## 3.2 Sécurité

La sécurité du système consiste en plusieurs points :

- Le respect de la vie privée : Le génome transmis ne doit pas être enregistré par l'application.
- Protection de l'information : les communications entre client-serveur doivent être sécurisées (au moyen par exemple de HTTPS).

## 3.3 Performance

L'application doit gérer sans soucis les charges que ce soit dans le nombre de requête ou dans la taille des données envoyées (dans la limite des critères définis ci-dessous).

Critères du système :

Une analyse médicale est composée de moins de 1M de mots et une maladie de 10% de ces 1M de mots (les maladies sont stockées dans les serveurs).

# 4 Annexes

## 4.1 Annexe I : Demande d'une analyse complète

Lorsqu'un client fait une requête au serveur afin d'interpréter une analyse complètement (tester toutes les maladies) :

Protocole utilisé : HTTP

Méthode : POST

URL: https://api.example.org/all

Paramètres :

- Une analyse médicale sous la forme JSON :

```
[
  "78zEBXhzfreugis",
  "ff8sg1PfkdgPKJfj8d",
  ...
]
```

Valeur de retour :

Une liste de maladies sous forme JSON :

```
[
  "Cancer du sang",
  "Diabete de type 1",
  ...
]
```

Code de retour :

200 – Tout est bon

400 – Erreur de format du paramètre

500 – Erreur durant le traitement de la requête

## 4.2 Annexe II : Format du dictionnaire des maladies

Le dictionnaire des maladies auquel accède l'application serveur est ici représenté par un fichier. C'est un fichier JSON ressemblant à ceci :

```
{
   “Cancer du sang” : [
          [
                 “Jqhzoy4rVY”,
                 “GH3CXOO”,
                 “riYY4aevK”
          ],
          [
                 “fds5”,
                 “EK445AGej”,
                 “MWZzTfdgf15IRPCA3”
          ]
   ],
   “Diabete de type 1” : [
          [
                 “4Wmmyyh5F4g880DVKE”,
                 “zUBEfLYTkU”,
                 “id49gOYuF”
          ]
   ],
   ...
}
```

## 4.3 Annexe III : Format d'un génome

L'application client peut charger uniquement un type de fichier spécifique. Ce fichier est lisible et formaté de cette façon :

```
FileData: genome
[
   "78zEBXhzfreugis",
   "ff8sg1PfkdgPKJfj8d",
   ...
]
```

La première ligne est une ligne d'en-tête, celle-ci est obligatoire afin de ne pas lire n'importe quel fichier.

## 4.4 Annexe IV : Format de la liste des serveurs

L'application client charge au démarrage de l'application un fichier contenant la liste des serveurs à consulter lors de l'analyse. Ce fichier est lisible et formaté de cette façon :

```
FileData: servers
https://api.example.org/, Example Dictionary
https://foo.bar/, Foo Dictionary
```

La première ligne est une ligne d'en-tête, celle-ci est obligatoire afin de ne pas lire n'importe quel fichier.

## 4.5 Annexe V : Demande d'une analyse spécifique

Lorsqu'un client fait une requête au serveur afin de vérifier la présence d'une maladie dans une analyse :

Protocole utilisé : HTTP

Méthode : POST

URL: https://api.example.org/one

Paramètres :

- Une analyse médicale sous la forme JSON :

```
[
  "78zEBXhzfreugis",
  "ff8sg1PfkdgPKJfj8d",
  ...
]
```

- Un nom de maladie sous la forme d'une chaine de caractère :

```
"Cancer du sang"
```

Valeur de retour :

Un booléen indiquant si la maladie est présente ou pas :

```
true/false
```

Code de retour :

200 – Tout est bon

400 – Erreur de format d'un des paramètres

418 – La maladie passée en paramètre n'existe pas

500 – Erreur durant le traitement de la requête

## 4.6 Annexe VI : Demande de la liste des maladies disponibles

Lorsque le client veut prendre connaissance des maladies qu'un serveur peut tester :

Protocole utilisé : HTTP

Méthode : GET

URL: https://api.example.org/list

Valeur de retour :

Une liste des maladies disponibles sous forme JSON :

```
[
  "Cancer du sang",
  "Diabete de type 1",
  ...
]
```

Code de retour :

200 – Tout est bon

500 – Erreur durant le traitement de la requête