Hes·so VALAIS WALLIS
Haute Ecole Spécialisée de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences Western Switzerland

Travail de bachelor 2010

Filière Informatique de gestion

Recherche d'informations médicales sur mobile

Etudiant : Samuel Duc

Professeur : Dr. Henning Müller

www.hevs.ch

1 Préface

Pour des professionnels, un accès à l'information actuelle est extrêmement important car de nouvelles connaissances sont créées quotidiennement. La littérature scientifique médicale est en grande partie disponible sur Internet et peut ainsi être exploitée par la recherche d'informations.

Durant les dernières années, la façon de rechercher a constamment évolué. Si la recherche textuelle reste toujours très présente, de nouvelle manière d'accéder à l'information basée sur du multimédia sont en pleine expansion. Dans un cadre médical, la recherche basée sur de la comparaison d'image peut être étonnamment efficace et efficiente. Les images peuvent être lu sur un écran de manière beaucoup plus rapide et pertinente que des documents textes et permettre ainsi de retrouver l'article recherché.

L'accès à Internet et donc aux sources de données était limité auparavant à un environnement fixe, celui des ordinateurs de bureau. Maintenant les opérateurs téléphoniques ont enrichi leurs produits et proposent un accès à Internet depuis un milieu mobile, les smartphones. Pour exploiter ces nouveaux horizons beaucoup d'applications pour téléphones mobiles, spécifiques à un domaine, ont été produites.

Ce rapport décrit la construction d'un système de recherche basé tant sur de la recherche textuelle que visuelle d'informations issues de la littérature scientifique de BioMedCentral. Cette solution a été développée pour permettre d'accéder à cette littérature depuis un smartphone. Un scénario basé sur le Web a été choisi pour permettre l'utilisation de cette application indépendamment de leur plateforme. Plusieurs contraintes liées à la taille de l'écran à la navigation avec un écran tactile ont été étudiées et prisent en compte lors du développement

Beaucoup de problèmes avec l'incompatibilité de certaine plateforme, de certain navigateur mobile ou de restrictions mises en place par le constructeur ont été recensées dans le texte.

2 Motivation personnelle

Ayant déjà effectué du développement native pour iPhone dans le cadre de mes études à la HES-SO Valais, je voulais approfondir cette matière. J'avais été surpris par la politique extrêmement restrictive des constructeurs et je voulais examiner les alternatives proposées pour créer des applications exécutables sur plusieurs plateformes.

Avec la quantité d'information que contient internet, les recherches devront être de plus en plus précises et pointues pour que le résultat satisfasse la requête. Les différentes alternatives pour retrouver du contenu étaient pour moi un atout supplémentaire. Je connaissais bien sûr la recherche basée sur du texte qui est une recherche standard actuellement mais celle basée sur du multimédia, comme la comparaison d'images dans ce projet, m'était inconnue.

3 Sommaire

4 Gestion de projet

4.1 Déroulement

Ce travail de bachelor a été réalisé en 360 heures, soit 43.5 jour/homme, réparties sur une période allant du 17 mai 2010 au 13 août 2010. Ce projet correspond donc aux exigences lié à un travail de bachelor en filière informatique de gestion de la HES-SO Valais.

4.2 Planification

Une planification lourde n'a pas été effectuée dans le cadre du développement. D'entente avec M. Müller, une organisation plus légère à été choisi car ce projet comportait une grande phase d'analyse et de recherche qui rendrait le planning initial trop imprécis.

Un simple calendrier relevait, dans les grandes lignes, cinq grandes phases par lesquelles passait ce projet. Celle-ci permettait de fixer des objectifs et des délais.

Phases	Jour / Homme	Dates
Initialisation	5	17.05 - 31.05
Cadrage	6	01.06 - 14.06
Design	8.5	15.06 - 12.07
Construction	19	13.07 - 06-08
Finalisation	5	09.08 - 13.08

4.3 Suivi

Un suivi hebdomadaire a été mis en place avec le responsable de ce projet. Lors de chaque séance le travail effectué durant la semaine était évalué, nous discutions des problèmes rencontrés et une planification des objectifs pour la semaine était définie.

Ce système de travail a permis de travailler de manière incrémentielle. Ainsi au début, le noyau du projet a été développé puis, selon les objectifs fixés, de semaine en semaine de nouvelles fonctionnalités venaient s'y greffer.

4.4 Cahier des charges

Analyse

Obligatoire

familiarisation avec le système existant de recherche,	✓
état de l'art sur les contraintes et possibilités des Smartphones,	✓
adaptation du système existant, avec une détection du browser et de la résolution, à une recherche d'info mobile.	✓

Optionnel

utilisation de la caméra pour chercher des images similaires,	✓
navigation spécifique pour Smartphone.	✓

Crawling HTML

Obligatoire

crawling de quelques journaux (15-20) en cherchant le texte, les images et la légende des images.	✓

Optionnel

plus de journaux,	✗
plus d'informations structurées.	✗

Indexation

Obligatoire

indexation des données obtenues avec Lucene et avec GIFT,	✓
communication avec les outils existants par sockets.	✓

Optionnel

optimisation de la recherche.	✗

Interface mobile

Obligatoire

interface mobile qui s'adapte à une résolution limitée.	✓

Optionnel

utilisation de l'appareil photo des téléphones mobiles,	✓
utilisation des fonctions tactiles du Smartphone pour la navigation.	✓

5 Introduction

5.1 Description du contexte

Pour des professionnels, un accès à l'information actuelle est extrêmement important car de nouvelles connaissances sont créées quotidiennement. La littérature scientifique médicale est en grande partie disponible sur Internet et peut être ainsi exploitée pour la recherche d'informations.

5.2 Objectifs

Ce travail de bachelor se déroule donc en plusieurs étapes. La première partie consiste à recueillir des sites qui proposent de la littérature scientifique médicale comme BioMedCentral, puis d'y extraire des articles, les images relatives ainsi que les descriptions des images. Durant la seconde étape, les différentes données amassées seront indexées selon leur type "article" ou "image".

La dernière partie consiste à créer une interface de recherche sur téléphone mobile qui inclut des images et des articles. Le système pourrait alors exploiter les différentes possibilités et contraintes d'un téléphone mobile de type Smartphone avec des données visuelles et textuelles.

6 Méthodes

6.1 Analyse de l'existant

Ce travail de bachelor s'appuie sur un projet "MedSearch[1]" qui a été réalisé par Ivan Eggel dans le cadre d'un mandat de l'Institut Informatique de Gestion de HES-SO. Le but de cette réalisation est de parcourir la littérature médiale disponible sur Internet puis de l'indexer et ensuite créer un moteur de recherche accessible depuis un navigateur internet standard.

MedSearch suit trois étapes principales. Durant la première étape le système s'occupe de parcourir la littérature médicale contenue sur le site internet de BioMedCentral[2]. Plusieurs éléments dont l'auteur, le résumé, les résultats, les images,… sont recensés dans chaque article qui compose cette littérature.

Durant la deuxième étape, les données recensées par le système pendant l'étape précédente sont indexées par un moteur de recherche. Lucene[3] a été choisi pour ce projet. Ainsi Lucene indexe les données récoltées pour chaque article de cette littérature de BioMedCentral.

La troisième étape est l'exploitation de ces données. L'utilisateur peut entrer un terme et Lucene se charge de trouver ce terme dans le contenu indexé par lui-même. En cas de succès, l'utilisateur pourra consulter cet article. Une autre fonctionnalité permet à l'utilisateur de rechercher un article par soumission et comparaison d'images. Pour cette possibilité l'outil GNU Image Finding Tools (GIFT[4]) est utilisé. De la même manière que pour l'indexation des données, les images sont indexées par cet outil. Lorsque l'utilisateur soumet une image celle-ci est comparée avec celles indexées par GIFT et les résultats similaires à l'image soumise sont présentés à l'utilisateur.

MedSearch est essentiellement basé sur le web pour être accessible par tous les navigateurs internet d'ordinateur de bureau. L'application a été programmée en Java pour ce qui est de la partie système et en JSF pour la partie visible à l'utilisateur. Glassfish est utilisé comme serveur d'application.

La partie d'indexation de la littérature médicale avec Lucene ainsi que celle concernant la recherche d'image similaire avec GIFT de MedSearch pourront être employées dans la version mobile de MedSearch qui sera développée dans le cadre de ce travail de bachelor. Au contraire la partie qui sera visible du client qui était développée en JSF pour MedSearch devra être complètement revue car beaucoup de caractéristiques diffèrent entre un ordinateur de bureau ou un smartphone.

[1] http://medgift.unige.ch:8080/MedSearch/faces/Search.jsp
[2] http://biomedcentral.com/
[3] http://lucene.apache.org/
[4] http://www.gnu.org/software/gift/

6.2 Analyse Smartphone

Au point précédant le fait qu'une application destinée à un appareil mobile doive posséder des caractéristiques et une accessibilité différentes d'une application destinée à un ordinateur de bureau a été relevée. Pour bien correspondre aux nécessités et besoins d'un appareil mobile, il est important de connaître la spécificité de celui-ci. Cette analyse présente de manière générale ces téléphones portables de type smartphone.

Qu'est-ce qu'un smartphone ?

Un smartphone est un concentré de technologie embarquée dans un téléphone portable. Outre les fonctions de communication standards d'un mobile, celui intègre aussi de nouvelles possibilités comme la connexion internet en continu via le réseau 3G ou WiFi, la gestion des informations personnelles (contact, agenda, notes), un navigateur internet, un logiciel de gestion des emails, un GPS, du multimédia (photos, vidéos, musiques). Le système d'exploitation offre généralement la possibilité à l'utilisateur d'installer des logiciels tiers. Cette dernière option a complètement révolutionné le téléphone portable.

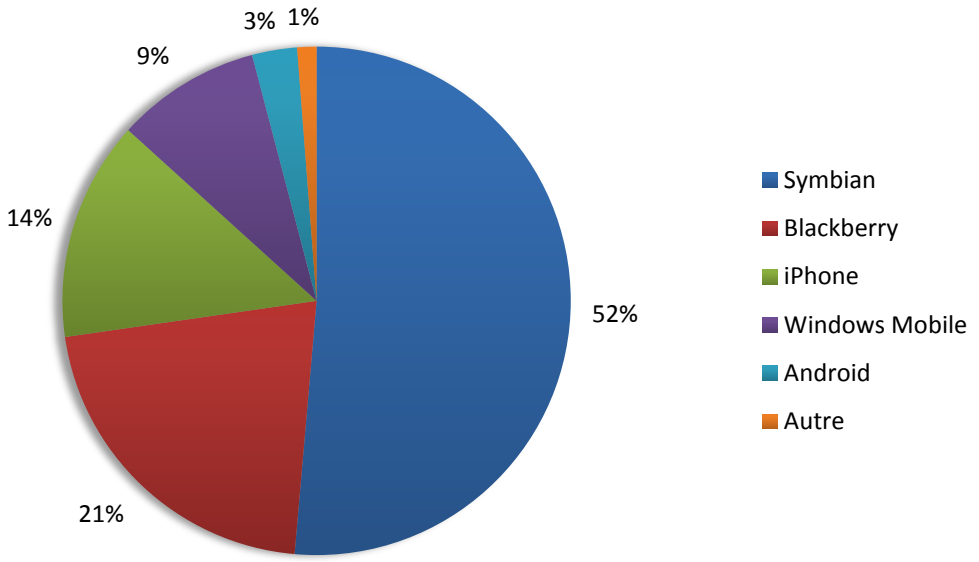
Aperçu du marché

Le marché d'un smartphone est bien distinct du marché du téléphone portable. En effet non seulement les produits sont différents mais aussi les constructeurs. Apple, HTC, RIM ne fabriquent uniquement des téléphones "intelligents" au contraire Nokia et Motorola sont présents sur les deux marchés. Ces deux fabricants ont diversifié leur gamme de produits pour aussi être présents sur le marché des smartphones.

Certaines agences de statistiques ont créées une division qui suit l'évolution du marché du smartphone. Celui-ci connaît une croissance fulgurante depuis cinq ans maintenant et d'autant plus ces trois dernières années depuis l'arrivée de l'iPhone d'Apple sur le marché.

Nous allons voir au moyen du graphique suivant les parts de marché des différents systèmes d'exploitation disponibles pour les smartphone. Du fait que nous allons analyser plus tard les navigateurs mobiles et que ceux-ci sont fournis avec le système d'exploitation, il était plus intéressant de porter notre étude sur les systèmes d'exploitation et non sur les fabricants.

Graphique 1. Répartition des parts de marché entre les systèmes d'exploitation mobiles

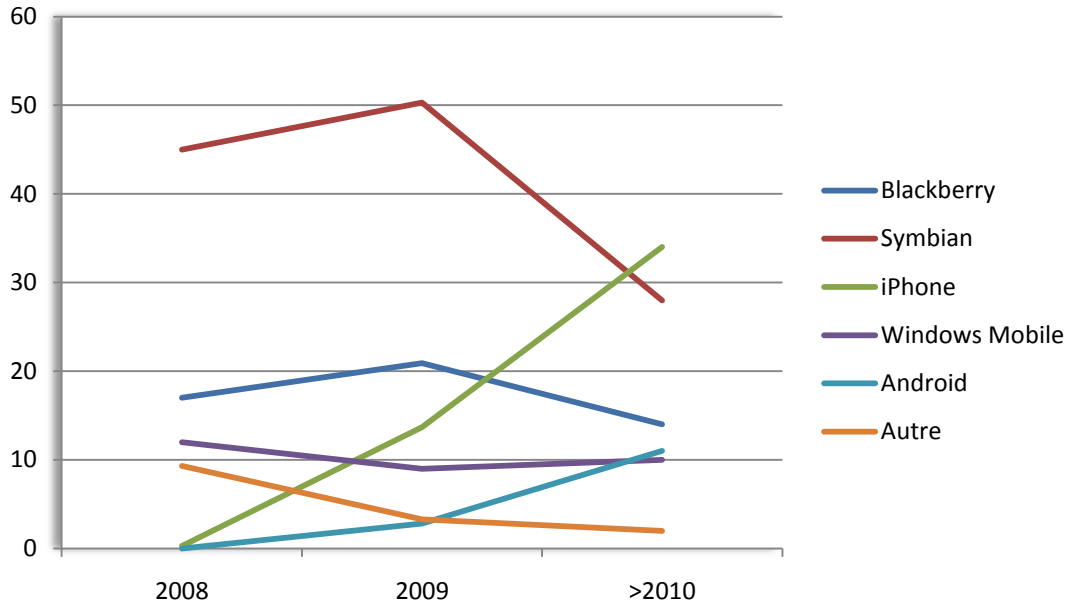


Source : Gartner (2009)

Il est intéressant de constater que Symbian développé par Nokia est actuellement le leader sur ce marché des smartphones. Mais tout évolue extrêmement vite et le point suivant va nous éclairer sur l'évolution de ce graphe. Blackberry occupe une part significative du marché mondial mais ce pourcentage provient essentiellement des Etats-Unis où ce produit est très repandu, contrairement au territoire européen ou asiatique.

Tendance du marché

Graphique 2. Tendance du marché des Smartphones 2008 - >2010



Source : jTribe – Mobile OS/ Browser Report, (Juillet 2009)

Bien entendu ces chiffres varient considérablement car le marché est en pleine effervescence. Symbian perd de plus en plus sa position de leader pour la fin de l'année 2010 et devrait se stabiliser autour des 28%. Au contraire l'iPhone d'Apple devrait atteindre les 34% de part de marché et ainsi devenir leader. Android de Google est aussi un système d'exploitation à suivre de près car, contrairement à l'iPhone OS, il est distribué par plusieurs fabricants de smartphone dont Motorola, HTC et promet donc de poursuivre sa croissance dans ce marché et de s'installer comme un des systèmes d'exploitation mobiles majeurs.

Evolution de l'Internet mobile

Si on devait citer la grande nouveauté qu'a apporté le smartphone au téléphone mobile "classique" c'est l'accès à internet. Ceci a été réalisable grâce à l'appui des opérateurs téléphoniques qui ont modernisé leur réseau de téléphonie (3G) et aussi par un récepteur WiFi interne au smartphone.

L'étude menée par Gartner, qui est une entreprise de conseils et de recherches dans le domaine des technologies avancées, explique bien l'importance que va occuper l'internet via smartphone dans le futur.

' Ils prévoient que dans les 3 prochaines années l'accès au Web par mobile dépassera l'accès par PC.

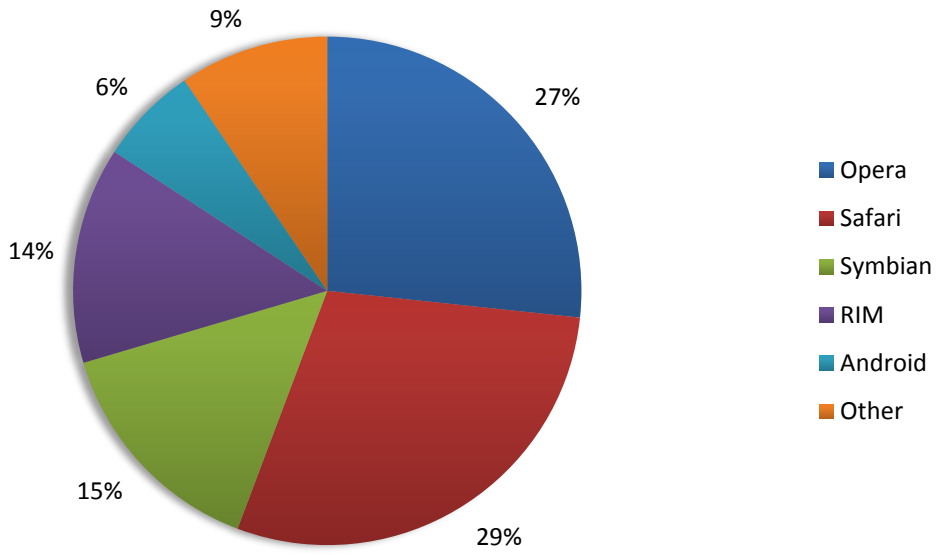
Ils ne disent pas que les téléphones portables remplaceront les ordinateurs comme principal moyen de naviguer sur le Web, mais l'explosion du marché des smartphones et l'amélioration de la navigation se traduiront d'ici 2013 que plus de personnes auront accès au web sur leurs téléphones mobiles que sur PC.'[5]

Analyse des mobiles browsers

Dans le cas d'une application web, cette analyse s'avère importante pour déterminer les navigateurs mobiles les plus utilisés du marché. Comme indiqué plus haut, ceux-ci sont livrés nativement avec le système d'exploitation du smartphone, mais l'utilisateur a le choix d'installer un navigateur "indépendant". Par exemple Opera n'est distribué par un système d'exploitation mais est disponible sur la plupart des plateformes.

Un problème principal d'une application web est la compatibilité avec le navigateur. Cette étude permettra donc de bien cibler les navigateurs clés du marché.

Graphique 3. Part de marchés des navigateurs web mobiles



Source : gs.statcounter.com (Mai 2010)

[5] http://ifonlyblog.wordpress.com/2010/01/14/gartners-mobile-predictions/

Il est intéressant de constater que les parts de marché ne sont absolument pas cohérentes dans toutes les régions du monde. Safari est utilisé dans 40% des accès à internet par un mobile en Europe et au Etats-Unis, mais **en Suisse Safari pèse plus de 73%.**

Cette étude est basée sur les statistiques d'accès par un navigateur mobile. Les résultats ne sont donc pas guidés par les parts de marché des systèmes d'exploitation. Ces deux statistiques ne sont donc pas liées dans leurs résultats

Une rapide analyse permet donc de dire que l'iPhone et Opera sont leaders dans la consommation de site web via un smartphone. RIM, Blackberry et Symbian ne génèrent que peu de trafic web mobile par rapport à leur position sur le marché.

D'après les statistiques établies ci-dessus, il est déjà possible de choisir une ligne directrice vers laquelle va s'orienter le développement de MedSearch Mobile. Selon l'évolution du marché des smartphones dans les années à venir iPhone OS d'Apple ainsi que Android de Google ressortent comme des valeurs sûres.

6.3 Analyse du type d'application

Application native

Pour une interface mobile, il existe plusieurs styles d'applications destinées à un smartphone. L'une est basée sur son système d'exploitation et s'installe donc comme une application-tierce, une autre est totalement basée sur son navigateur internet et enfin la troisième est une solution hybride des deux premières.

La première possibilité est de créer une application dite « native » ou application-tierce qui s'installera sur le smartphone. Le principal obstacle s'élève rapidement. En effet, l'application est concrètement reliée au système d'exploitation de l'appareil mobile ainsi, si elle est développée pour fonctionner sur un Android elle ne fonctionnera pas sur un iPhone et vice-versa. Avec une solution native on se limite donc à un seul type de smartphone.

Application web

La seconde possibilité propose une application entièrement basée sur du web. Au moyen de librairies JavaScript existantes, il est possible de créer une application web (webapp) qui est adaptée à un smartphone et qui peut ressembler à une application native.

Malheureusement les interactions avec le matériel physique (caméra, vocal,…) du smartphone n'est pas envisageable à ce jour. Les constructeurs bloquent intentionnellement toutes interactions avec le matériel hormis la boussole et la géo-localisation. Donc, avec ce type de solution, il n'est pas envisageable d'utiliser directement la caméra pour prendre une photo et la soumettre au système GIFT pour une comparaison.

Un avantage non négligeable par rapport à une application native est qu'aucun déploiement de l'application n'est nécessaire. En effet, vu que celle-ci est entièrement basée sur du web, on y accède simplement par une URL.

Une application web pour smartphone utilise non seulement le navigateur web mais aussi son moteur de rendu HTML pour créer des effets comme la transparence ou les arrondis des angles.

Les navigateurs sont basés sur ce moteur de rendu qui est leur cœur. Actuellement Safari (iPhone) et Android (Google) utilisent déjà WebKit[6]. Symbian (Nokia) et RIM (Blackberry) vont baser leur navigateur sur ce dernier dans leur prochaine version. Seul Opera utilise son propre moteur de rendu : Presto[7].

Dès lors, en créant une application web basée sur le moteur de rendu WebKit, la comptabilité est assurée avec les principaux navigateurs web mobiles du marché. Seul Opera ne sera pas compatible avec son moteur de rendu différent.

Application mixte

La dernière solution envisagée est de créer une application mixte. Cette technologie, très récente, permet de créer une application basée sur du web (HTML, CSS, JavaScript) pour, au final, générer une application native à l'aide d'un compilateur. Il existe une possibilité de compilation pour trois plateformes : Android, iPhone et Blackberry. Ce genre de solution est dite mixte car elle comporte une partie native : une fenêtre de navigateur web intégrée à l'application et une autre partie basée sur du web (application web).

La création d'une application mixte parait intéressante sur plusieurs points. Elle permet à la fois de répondre au critère de distribution multiplateforme et d'intégrer des éléments matériels du smartphone comme par exemple la caméra. Cette solution représente une vraie alternative à une application « native ».

Une étude[8] sur les principaux outils du marché avait déjà été menée par un groupe d'étudiants de la HES-SO dans le cadre d'un travail de module. Trois produits ont été testés : PhoneGap[9], Titanium Mobile[10] et Rhodes[11].

Parmi ces trois produits aucun ne s'est avéré être concluant pour du développement mobile. De manière générale, de part leur nouveauté, les aides ainsi que les sources sont rares. Ainsi, même pour une application de test, les résultats n'étaient pas concluants. De plus la norme de validation d'application d'Apple ne permet pas de développer une application créée en dehors de leur propre environnement de production.

[6] http://webkit.org/
[7] http://www.opera.com/
[8] Frameworks mobiles Hes-so (2009 – 2010)
[9] http://www.phonegap.com/
[10] http://www.appcelerator.com/
[11] http://rhomobile.com/products/rhodes/

6.4 Recherche sur smartphone

Une récente étude[12] sur les perspectives de la recherche sur mobile a été menée par un groupe de travail européen. Cette étude permet d'évaluer les différences entre un système de recherche sur un PC et sur un téléphone mobile. Le tableau suivant reprend dans les grandes lignes ces différences.

Tableau 1. Comparaison entre un environnement PC et mobile

ATTRIBUT	PC	SMARTPHONE
Taille de l'écran	Grand	Très petit
Capacités d'entrée	Bonne	Limitée
Personnalisation et ciblage	Raisonnable	Très bonne
Vitesse de connexion	Rapide et en amélioration	Raisonnable et en amélioration
Optimisation des sites	Bon	Pauvre mais en amélioration
Localisation	Raisonnable	Très bon
Mode de consommation	Etendu.	Court
Prix	Forfaitaire	Compteur ou forfait limité
Degré d'ouverture	Complet	En principe fermé mais ouverture partielle
Accessibilité	Stationnaire	Mobile

Source : Zoller (2007) & Prospects of Mobile Search Studies

Certaine des différences relevées ci-dessus peuvent impliquer un changement majeur dans la manière de développer une application. L'application MedSearch Mobile devra prendre en compte tous ces critères et s'adapter à l'environnement mobile auquel elle est destinée.

[12] Prospects of Mobile Search – Instute for Prospective Technological Studies

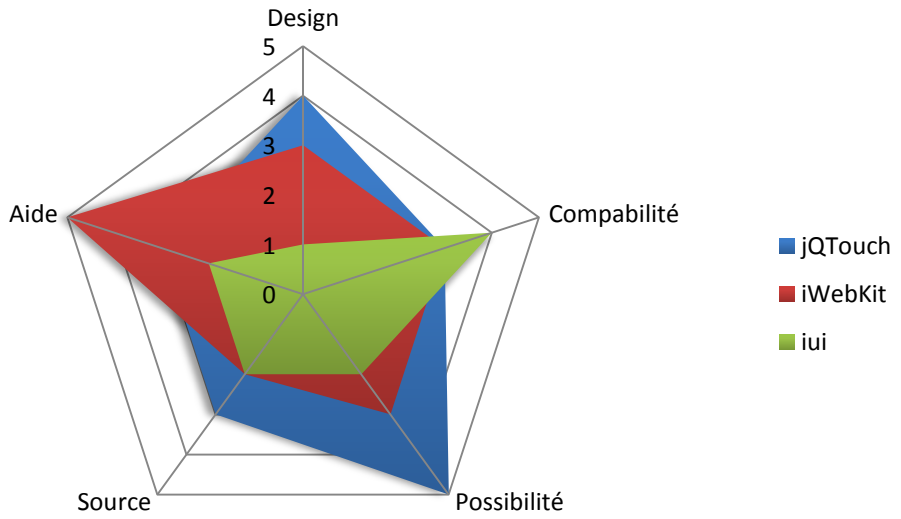
6.5 Analyse d'une libraire d'application web

Il n'y a pas encore de beaucoup de produits mis à disposition par la communauté pour construire des applications web mobiles. Après quelques recherches, les principaux noms qui ressortent sont : iui, jQTouch, iWebkit. De manière générale ces produits sont récents et donc les sources et l'aide sont très peu développées pour l'instant.

Ces Framework sont conçus de la même manière qu'un site web. Ils utilisent une librairie JavaScript, du HTML et du CSS. Le JavaScript est essentiel dans ce type de développement car il permet d'envoyer des requêtes asynchrones et aussi d'ajouter quelques effets simples pour que l'application web se rapproche d'une application native.

Le *graphique 4* nous montre une brève analyse par comparaison de ces Framework de développement d'application web mobile.

Graphique 4. Radar des différentes libraires pour une application web



Source : Mobile Web Application Frameworks Overview, Adrian Kosmaczevski

La Constatation est assez claire. Aucun produit ne s'avère être complet dans les cinq critères retenus pour cette comparaison. JQTouch offre beaucoup de possibilités car il est basé sur la librairie jQuery qui, elle, est très rependue. En revanche, toutes ces possibilités qu'offre ce produit, le rende moins compatible pour les navigateurs internet. Iui est un Framework simple voir même simpliste qui ne permet de créer qu'un design ressemblant à celui qu'Apple fournis pour leur application native. Il n'offre que peu de possibilités et de soutien de la communauté. L'atout majeur d'iWebkit est l'aide proposée pour s'initier à ce Framework. Malheureusement, les sources ne sont pas autant présentes et de même qualité.

6.6 WebService

Le point clé pour pouvoir réutiliser la même base que celle du projet MedSearch consiste à trouver un moyen optimum de communication entre la partie visuelle de l'application destinée au client et la partie intelligence de celle-ci. Dans un milieu très hétérogène, le moyen standard est bien souvent de mettre en place un Web Service.

Dans le domaine des Web Service, deux différentes technologies peuvent être utilisées dans le cadre de ce projet. Soit une architecture REST [13] qui se distingue par l'utilisation du protocole http, soit une architecture SOAP [14] qui créer un nouveau protocole de communication. En créant son propre protocole SOAP impose l'envoie d'un nombre important d'informations, ce qui a comme conséquence d'alourdir considérablement les échanges.

Pour utiliser un web service SOAP il est nécessaire que le client importe une libraire spécifique à cette communication. Au contraire REST est simplissime, un appel au web service se fait par URL, de la même manière que l'on appelle un site web. Lors d'un appel d'un site web, le serveur retourne au client du HTML. Dans le cas du REST un appel à l'URL retourne des données ordonnées en XML.

Même pour les paramètres, un simple ajout du paramètre dans l'URL permet de le passer au service web. De plus la communication par le protocole HTTP permet de diminuer amplement le poids des données échangées.

[13] Representational State Transfert
[14] Simple Object Access Protocol

6.7 Choix des outils et solutions

Choix du type d'application

Dans le time-box imparti, il n'est pas envisageable de créer une solution native pour les principaux systèmes d'exploitation ressortis de l'étude de marché. Dès lors, la solution d'une application-tierce est abandonnée.

Malgré les possibilités intéressantes que nous offre la technologie "mixte", cette solution a été écartée par l'impossibilité de créer une solution satisfaisante. De plus la crainte que l'application ne puisse être déployée car elle ne répondrait pas au norme de validation des constructeurs a joué en défaveur de ce type d'application.

De part tous les avantages qu'offre une application basée entièrement sur du web et la possibilité d'utilisation sur les principaux smartphones, cette solution a été choisie. Le seul point négatif est que l'utilisation du matériel physique du smartphone est impossible directement, mais la tendance laisse croire que les constructeurs vont autoriser l'utilisation de ceux-ci à court ou moyen-terme comme ils le permettent déjà pour la boussole ou la géo-localisation.

Choix d'une libraire d'application web

Le *Graphique 4* parle de lui-même. Iui a été abandonné rapidement car il conviendrait plutôt à une application très simple et avec un design peu aboutit.

Quelques tests on été effectués sur les deux autres Frameworks. IWebkit est principalement axé sur le développement mobile pour iPhone et dans le cas de MedSearch Mobile il était plus intéressant de développer pour les principaux smartphones du marché et de ne pas se limiter à une plateforme.

JQTouch correspond mieux à l'utilisation vers laquelle est orientée MedSearch Mobile : du multiplateforme. Sa librairie de base jQuery est, en plus, un réel atout pour ce Framework. JQuery est une libraire de JavaScript extrêmement complète, avec beaucoup de sources et une communauté très active. JQTouch a donc été préféré pour le développement de l'application web mobile.

WebService

Dans le cadre de ce projet un Web Service de type REST correspond entièrement aux besoins. La simplicité de sa mise en place et son utilisation pèse en sa faveur. Le SOAP nécessite une mise en place du côté client de l'application tandis que le REST est un simple appel d'URL. La librairie JavaScript sur laquelle est basée jQTouch se combine parfaitement avec une architecture REST, grâce à sa technologie d'appel asynchrone.

Dans un environnement mobile l'échange de données doit être limité. Comme nous l'avions vu dans le *Tableau 1*, l'opérateur limite en quantité l'échange de données (actuellement 250 Mo à 1 Go). Toutes données superflues sont donc néfastes pour le client.

En résumé, cette architecture est simple à mettre en place et surtout ne requiert pas de changements majeurs sur la partie du projet MedSearch qui va être réutilisée.

7 Résultats

7.1 Architecture

L'architecture du projet MedSearch Mobile se présent selon l'Image 1. Elle est découpée en trois axes.

La première partie, « Middleware », est la partie d'intelligence de l'application. Tout le noyau qui composait le projet MedSearch a pu être repris. Tout d'abord l'index qui référence les articles issus de la littérature médicale de BioMedCentral est le même pour les deux applications.

Toutes les méthodes de recherche dans l'index à l'aide de Lucene ont pu être réutilisées, il en est de même pour celles qui géraient la comparaison d'image semblable avec GIFT. Ainsi pour deux interfaces différentes (web PC et web Smartphone) la même intelligence a pu être utilisée. Toute cette partie à été développer en langage Java et tournait sous une forme de Servlet.

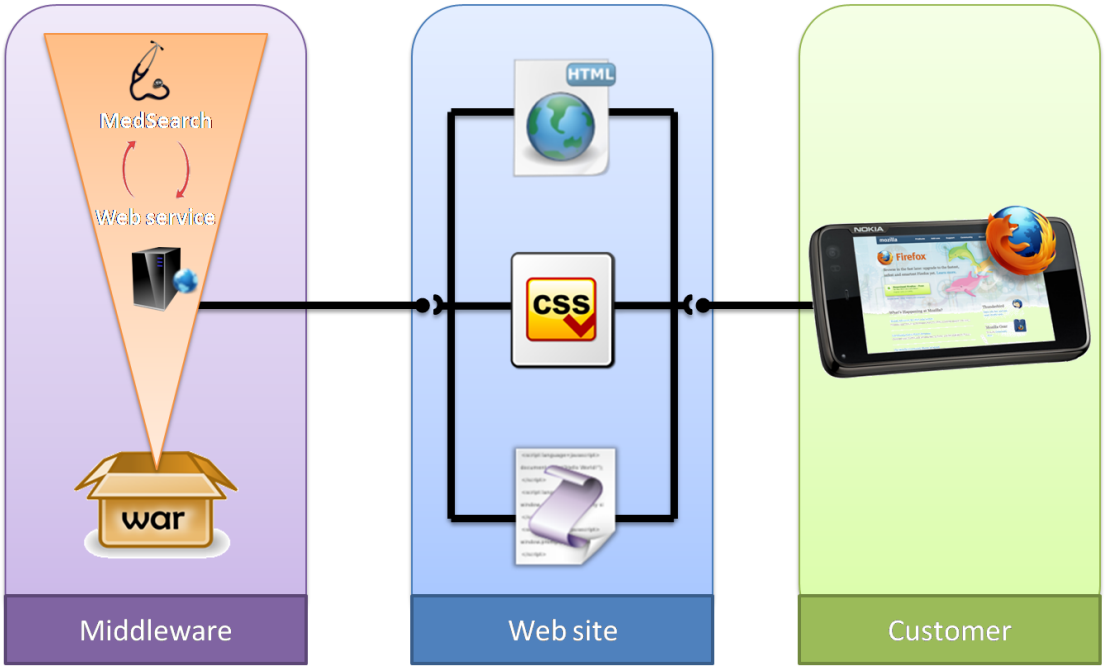
Le changement intervient dans la façon de communiquer entre l'interface client et cette partie middleware de l'application. Pour le projet MedSearch, le JSF permet d'interagir directement avec le code JAVA mais dans le cas de MedSearch Mobile JavaScript ne le permet pas. Le web service permet donc de faire ce pont.

L'axe bleu représente le serveur web qui hébergera la partie HTML, JavaScript et CSS de l'application web. La communication entre la partie Middleware, qui contient le web service, et la partie JavaScript qui sera client du web service se fera par XML[15]

Le troisième axe symbolise simplement l'utilisateur final qui peut se connecter grâce au navigateur internet de son smartphone à la plateforme MedSearch Mobile.

[15] Extensible Markup Language

Image 1. Architecture de MedSearch Mobile



Web service en REST

La librairie java utilisée : Jersey[16], permet justement de mettre en place des web services de type REST, notez que le terme RESTful est aussi utilisé parfois. Pour créer une méthode web avec Jersey il suffit d'ajouter quelques arguments avant celle-ci.

Code 1. Web méthode avec Jersey

```
@GET
@Path("/chemin/{id}")
@Produces("text/xml")
public String maWebMethode(@PathParam("id") String id)
```

Jersey permet, tout comme le HTML, de passer des variables soit par la méthode GET c'est-à-dire par l'URL soit par la méthode POST. Là les variables ne sont pas visibles car elles passent par l'entête HTML. Avec la variante `@GET`, l'indication `@PathParam` indique que la variable est dans le chemin ou l'URL et ainsi l'id de type String contiendra la valeur de cette variable. Au contraire en mode `@POST` il faut indiquer `@FormParam` pour récupérer cette variable.

[16] `https://jersey.dev.java.net/`

L'argument `@Path` indique le chemin par lequel est appelé cette web méthode par le client dans ce cas un appel sur `http://monServer:8080/chemin/identifiant1` va directement appeler *maWebMethode* et lui passer identifiant1 comme variable. {id} indique le nom de la variable qui sera recherché par le `@PathParam("id")`.

Cette méthode Java retour un objet de type String, et justement l'argument `@Produces("text/xml")` indique que la valeur de retour sera du XML sous forme de texte. Pour le projet MedSearch Mobile, le XML qui sera renvoyé au client est construit et non générer. Cela permet de maitriser les balises de ce langage structuré et ainsi d'éliminer les balises inutiles car chaque octet passé au client mobile réduit sa limite de données que lui impose son opérateur téléphonique (cf. *Tableau 1*). Une attention toute particulière a été donnée à l'optimisation de ce flux de retour.

Communication avec le client du Web Service

La librairie jQuery permet de générer des appels asynchrones. Cette fonction est très utile pour le client en JavaScript du web service. Simplement en se servant de cette fonctionnalité de jQuery, une requête peut être passée au web service, sans aucun rechargement de page comme c'est le cas en HTML.

Code 2. Appel d'un web service avec jQuery

```
$.get("http://monServer:8080/chemin/id1"),function(xmlInput){
//Traitement du XML
}
```

Le `$` représente une instance de jQuery et le `get("…")` la fonction get de jQuery. Dans le cas d'une fonction post comme il été le cas précédemment il faut indiquer `$.post("…")`.La variable passée en paramètre de la fonction get est l'URL d'accès à la web méthode. Au retour de cette méthode jQuery capte l'objet XML dans la variable xmlInput puis les données sont traitées dans la fonction.

Pour des raisons de sécurité[17] les navigateurs bloquent les requêtes[18] asynchrones appelées par du JavaScript vers un serveur autre que celui où est hébergé l'application web. Dans le cas de ce projet cela est problématique car la partie middleware, commune avec le projet MedSearch, n'est pas hébergée sur le même serveur que la partie web. La communication entre le web service et son client JavaScript est donc bloquée par le navigateur

[17] Same origin policy
[18] Cross-Domain AJAX calls

Pour contourner ce problème un proxy écrit en PHP a été utilisé (Md Emran Hasan). L'extension curl de php est utilisée pour rediriger l'URL d'appel au web service. jQuery appelle donc le proxy par une url interne et c'est le proxy qui se charge d'accéder à l'URL externe, celle du web service. De cette manière le navigateur internet ne détecte pas que le JavaScript fait référence à une URL externe. La syntaxe d'appelle du web service par le client JavaScript varie donc quelque peu :

Code 3. Appel d'un web service avec jQuery avec un proxy

```
$.get("http://localhost/proxy.php?action=monServer:8080/chemin
/id1&method=get")
```

L'appel par jQuery à la méthode get se fait donc sur le proxy qui lui va se charger de transférer cette requête vers le web service. Le type de requête est à nouveau précisé au proxy. Ce moyen permet donc de contourner les protections vers les URL extérieures mises en place par les navigateurs internet envers le JavaScript.

7.2 jQuery & jQTouch

JQuery est une bibliothèque JavaScript qui permet une utilisation simplifiée des commandes JavaScript. Ce Framework possède plusieurs fonctionnalités parmi lesquelles :

Recherche et modification des éléments html

Lors de la construction de la maquette HTML, des espaces peuvent être laissés vides, ils seront remplis dynamiquement par après. Par exemple une zone `<div id="maDiv" />` : du texte pourra être inséré lors de la détection d'un évènement.

Gestion des évènements

JQuery permet d'écouter plusieurs sortes d'évènements :

- Évènements page et fenêtre (onload, onresize,…),
- Évènements souris (onclick, onmousedown,…),
- Évènements clavier (onkeypress,…),
- Évènements formulaire (onblur, onsubmit,…).

Pour illustrer cette fonctionnalité, une image est insérée à notre maquette html avec le code suivant `<img src="monImage.gif" id="monImage" />`.

Lorsque l'utilisateur cliquera sur cette image un événement « click » sera déclenché. JQuery captera cet événement puis appellera la fonction qui a été créé pour répondre à cet évènement.

Code 4. Ajout de texte dynamique avec jQuery lors d'un évènement

```
$("#monImage").click( function(){ $("#maDiv").html("voici le texte) })
```

Dans ce code, jQuery écoute les événements de type click sur la base `id="monImage"`. Lors du déclenchement la fonction qui ajoute « voici le texte » dans la `DIV id="maDiv"` crée plus-haut est exécutée.

Manipulation des feuilles de style

Le style d'une balise peut également être modifié dynamiquement. La balise suivante indique un message à l'utilisateur. `<div class="normal" id="box">Faites une sélection</div>`. L'utilisateur effectue les sélections qu'il désire mais en cas d'erreur le message ainsi que le style de ce message peuvent varier.

Code 5. Ajout de texte avec un style différent

```
$("#box").html("Attention aux erreurs").addClass("error")
```

AJAX

La partie qui concerne l'envoie de requête asynchrone a déjà été traitée sous le point « Communication avec le client du Web Service ».

Ajout d'extension

Une multitude d'extensions sont proposées avec cette bibliothèque. Des effets de Drag-and-Drop, des formulaires interactifs, des widgets, des médias et bien d'autres extensions encore sont simplement implémentable à la version de bases de jQuery.

Pour ce projet jQTouch, qui est une extension de jQuery pour le développement d'applications web mobile, a été utilisée. Lors de l'initialisation de ce plugin, différentes options permettent de le configurer selon son utilisation.

Code 6. Initialisation du plugin jQTouch

```
$.jQTouch({
                slideSelector: '#home li a, .slide',
                icon: 'icone.png',
                addGlossToIcon: false,
                statusBar: 'default',
                fullScreen: true,
                preloadImages: [
                            'image1.gif',
                      'image2.gif'
                 …          ]
}) ;
```

Cette liste présente une partie de la configuration possible de cette extension. Le **SlideSelector** permet de définir et d'ajouter implicitement un effet de « slide » lors de transition entre deux pages. **L'icône** permet de créer notre propre icône et de l'ajouter sur le bureau du smartphone pour créer un lien direct vers notre application web, un effet de « gloss » est ajouté en fonction de la valeur de **addGlossToIcon**. Il est également possible de choisir la couleur de la **barre de statut** qui reste visible au sommet de l'écran du smartphone lors de l'exécution de notre application. Ainsi cette dernière peut mieux s'adapter à notre design. L'option **fullScreen** permet de lancer l'application sur l'écran en entier si le système d'exploitation le permet et d'éviter ainsi les barres de navigations ajoutées par le navigateur internet. La dernière option présentée **charge les images** indiquées au début du lancement de l'application de manière à améliorer la réactivité de l'application durant son utilisation.

Beaucoup d'autres options sont disponibles lors de l'initialisation de jQTouch. La plupart des effets créés pour une application native sont aussi disponibles pour cette version web. Seul le sliding qui est un effet standard a été utilisé dans le cadre de cette application.

JQTouch permet aussi de mettre en place un ou plusieurs thèmes. Ceux-ci sont composés du squelette HTML de l'application web ainsi que des fichiers de styles CSS.

7.3 Design

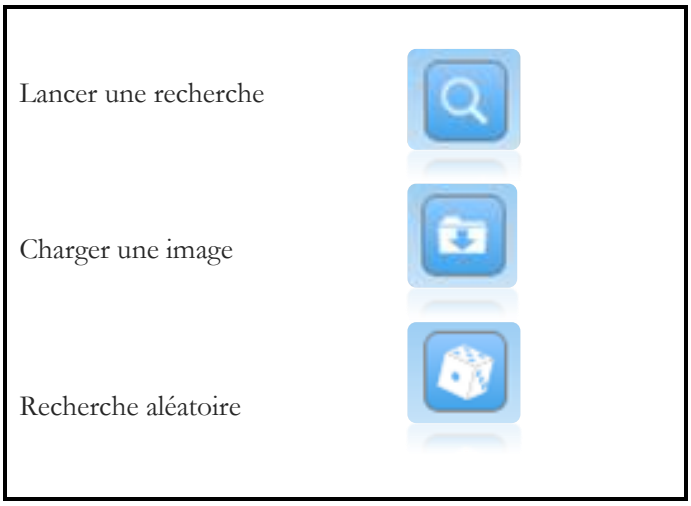
Le design de l'application joue un rôle primordial sur un appareil mobile. Il doit savoir mêler l'utile à l'agréable. Les couleurs suivent la ligne graphique du projet MedSearch et sont dans la palette des couleurs bleu à blanc que l'on associeraient au domaine médical.

Comme vu lors du comparatif du *Tableau 1*, la taille de l'écran d'un smartphone est relativement limitée. C'est pourquoi dans ce cas, le bandeau utilitaire qui permet de lancer des recherches est extrêmement limité dans son contenu et par sa taille. En effet il n'occupe qu'une petite partie du haut de l'écran pour laisser le maximum de place aux résultats des recherches.

Image 2. Design de l'application

Une astuce, utilisée pour tenter de limiter au maximum la taille, est de remplacer les textes qui prennent beaucoup de place par une simple icône significative. Ceci est d'une grande aide dans le développement pour mobile mais l'accent doit être porté sur le sens de ce que représente l'icône.

Dans l'application MedSearch Mobile les symboles suivants ont été préférés à leur équivalence textuelle.



CSS3

L'utilisation du CSS3 a permis d'arrondir le design de l'application. En effet cette technique se base sur le moteur de rendu du navigateur internet. Cette nouvelle génération de style permet d'arrondir les angles des éléments ou de créer un dégradé uniquement avec les styles et sans l'usage d'images.

Code 7. Arrondis avec WebKit

```
.monStyle {
   -webkit-border-radius: 6px;
}
```

Cet exemple montre un arrondi que l'on applique avec la classe `.monStyle`. Dans ce cas, cette exigence de style ne sera interprétée que par un navigateur doté de webkit comme moteur de rendu.

Encore plus loin, webkit permet de choisir les angles qui subiront cet effet comme utilisé pour la partie onglets de MedSearch Mobile où seulement un angle était arrondi et les autres restaient droits.

Code 8. Arrondis avec webkit avec choix des angles

```
-webkit-border-top-left-radius: 6px;
-webkit-border-bottom-left-radius: 6px;
-webkit-border-bottom-right-radius: 6px;
```

Cette nouvelle façon de définir les styles permet aussi de créer un dégradé et d'en choisir le style, l'orientation ainsi que ses couleurs. Ceci permet de se passer des images qui prennent du temps à charger et surtout optimiser le transfert de données.

Code 9. Dégradé avec webkit

```
background: -webkit-gradient(linear, 50 0, 50 30,
from(RGB(255,255,255)), to(RGB(235,239,249)));
```

Beaucoup d'autres nouveautés sont à constater, mais pour l'instant le CSS3 n'est pas encore reconnu comme une norme et n'est pas compatible avec tous les navigateurs internet.

7.4 Fonctionnalité de l'application

L'application MedSearch Mobile, tout comme MedSearch, permet deux types de recherche. La première, dite textuelle, permet de rechercher un mot ou une citation dans l'index qui répertorie les articles de la littérature médicale. Ce genre de recherche est devenu un standard sur le web.

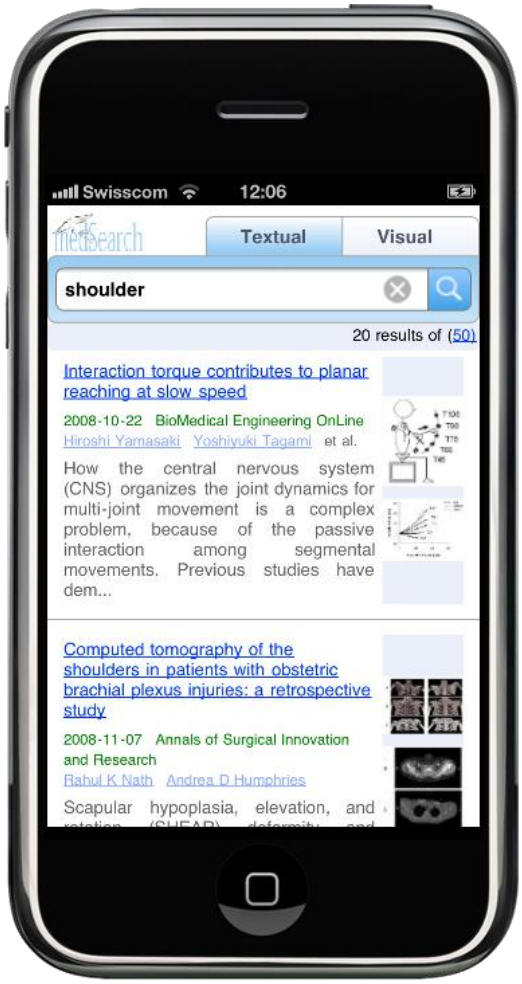
Rechercher textuelle

Dans ce cas, le mot « shoulder » a été entré dans la zone de recherche. Pour limiter la transmission de données vers le mobile seul les vingt premiers résultats de la recherche sont renvoyés mais cinquante résultats sont proposés à l'utilisateur par un simple clic sur ce nombre.

Une petite croix a été ajoutée à côté de l'icône de recherche. Celle-ci permet de directement vider le champ de recherche en un seul clic. Ce petit élément utilitaire[19] n'est pas proposé par jQTouch mais il se repend de plus en plus dans les interfaces mobiles.

[19] http://hogtownconsulting.com/clearquery/index.html

Image 3. Exemple de recherche textuelle



Les résultats de la comparaison avec l'index du mot « shoulder » sont affichés directement sur la partie inférieure de l'écran. Sur la droite un petit aperçu des deux premières images qui illustrent l'article s'affichent si celui-ci en contient.

Dans l'espace de gauche le titre de l'article est indiqué, ainsi que la date de parution et la revue dans laquelle il a été publié. Les deux premiers auteurs sont également présents avec possibilité de cliquer sur leur nom pour rechercher tous les articles écrits par cet auteur.

Une partie du résumé figure aussi sur cette page. Seuls les premiers caractères sont présents. Ainsi l'utilisateur pourra juger si cet article correspond à ses attentes. Si c'est le cas, il a la possibilité de passer sur un écran consacré à cet article par un simple clic sur le titre.

Image 4. Vue de détail d'un article

Sur cet écran l'utilisateur peut se faire une meilleure vision de l'article avec un résumé plus complet et la liste de tous les auteurs. La totalité des illustrations sont également présentes au-dessous du résumé.

Pour finir, l'utilisateur a le choix de revenir en arrière à l'écran des résultats ou se diriger vers l'article complet soit en version internet soit en version PDF.

Recherche visuelle

La deuxième possibilité de recherche, dite visuelle, se base sur une recherche multimédia. L'utilisateur peut afficher les images de façon aléatoire et rechercher les cas qui l'intéressent. L'autre possibilité est la comparaison d'images. Actuellement, tous les smartphones sont équipés d'une caméra embarquée. Malheureusement les constructeurs ne donnent pas d'accès direct à celle-ci depuis une application web, mais la démocratisation de ce type d'application les obligent à revoir l'accès aux ressources physiques du smartphone à moyen terme.

Pour cette application un prototype a été développé en utilisant une application native disponible gratuitement sur iPhone. Celle-ci permet de poster des images pris avec la caméra sur un serveur. « Picup » est disponible uniquement sur iPhone et par conséquent ce type de recherche est temporairement limité à cette interface.

L'application appelle Picup grâce au protocole mis en place : fileupload://. Le navigateur iPhone, Safari, reconnait que ce protocole est lié à Picup et l'application se lance. L'utilisateur a la possibilité de prendre une image et de la déposer sur un serveur dédié. Une fois le transfert effectué Picup retourne en tâche de fond et l'application MedSearch Mobile apparait à nouveau à l'écran avec une icône indiquant que l'image a été chargée. Malgré toute cette activité, ce processus est pratiquement transparent pour l'utilisateur qui, lui, n'a pratiquement pas l'impression de changer d'application.

Image 5. Exemple de recherche visuelle

L'utilisateur a encore le choix du nombre d'images à afficher (10, 20, 50, 100). Puis, il peut procéder à la recherche qui va comparer avec GIFT les images issues des articles de la littérature avec celle prise par l'utilisateur à l'étape précédente.

Le résultat de la comparaison des images similaires est proposé à l'utilisateur sous forme de petites vignettes

L'événement d'une inclinaison de son téléphone portable sur le côté est détecté par jQTouch. Ceci permet à l'application de s'adapter à l'orientation du téléphone portable et d'offrir plus de confort à l'utilisateur en adaptant aussi la taille et la disposition des vignettes d'aperçu.

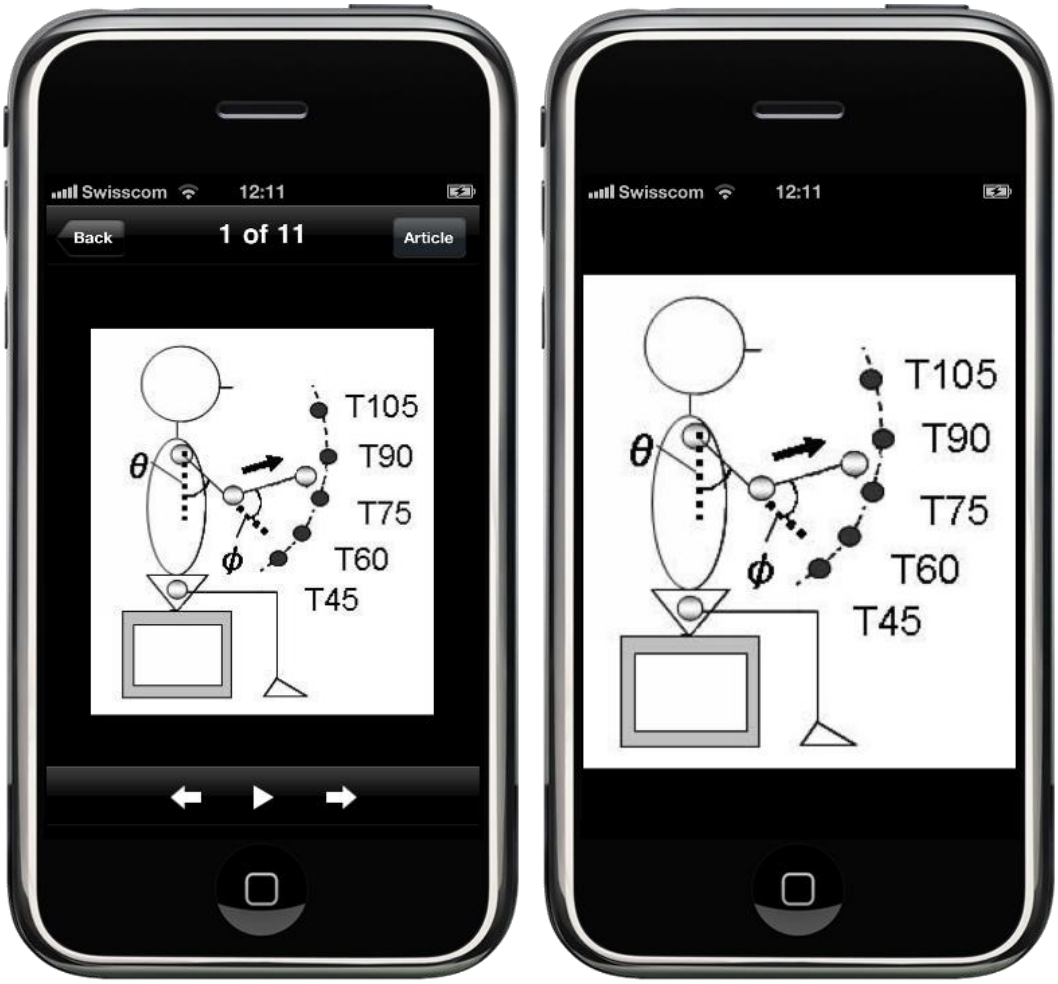
Image 6. Exemple de rechercher visuelle avec écran en position latéral

Galerie photo dynamique

Pour limiter l'utilisation de la bande passante ainsi que pour optimiser l'affichage d'information à l'écran, toutes les images ont été réduites et seulement une vignette est visible à l'utilisateur sur les différents écrans de l'application. L'agrandissement de ces illustrations est une nécessité mais dans le cadre d'un appareil mobile il est judicieux de proposer un moyen efficace de visualisation.

Pour cela, une extension de jQTouch a été employée. JQTouchPhotoGallery permet de créer une galerie photos ressemblante à celle d'un iPhone. Cette solution permet de regrouper toutes les images d'une même page et de construire la galerie dynamiquement. Cette simplification est parfaitement adaptée à une interface mobile car l'utilisateur n'a pas besoin de zoomer ou de cliquer sur chaques photos mais il peut naviguer à travers les images dans la galerie. Ce moyen est disponible pour toutes les images résultant d'une recherche textuelle ou visuelle.

Image 7. Galerie d'images



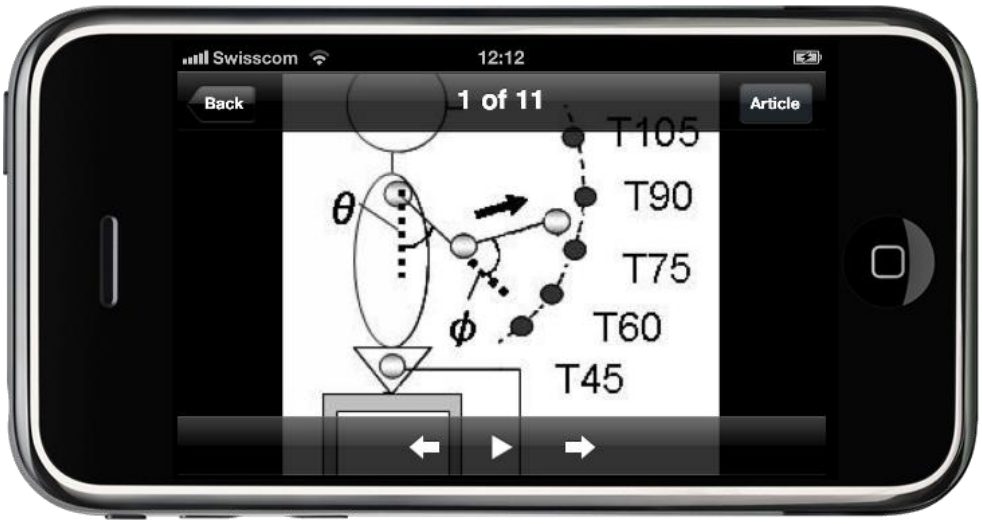
Une simple tape sur l'écran permet de cacher les barres de navigation et de profiter du mode plein-écran. Les fonctionnalités de zoom sur les images sont accessibles avec l'utilisation de deux doigts. Un rapprochement de ceux-ci zoom sur la zone ciblée et un éloignement permet de dé-zoomer.

L'extension JQTouchPhotoGallery[20] a été adaptée pour ce projet. Ainsi, un lien au sommet à droite permet d'accéder directement à l'article de la littérature médicale d'où est issue l'illustration. Dans le cas d'une recherche visuelle aléatoire, grâce à une image qui intéresse l'utilisateur celui-ci peut accéder directement à l'article concerné.

[20] http://code.google.com/p/jqextensions/wiki/JQTouchPhotoGallery

La navigation entre les différentes images contenues dans la page est très aisée. Soit la barre inférieure permet avec ces flèches d'avancer ou de reculer d'une image, soit un défilement avec le doigt permet de passer d'une image à l'autre. Une possibilité de slideshow est aussi disponible. Celle-ci fait défiler les images avec un arrêt de 5 secondes sur chaque élément.

Image 8. Galerie d'images en mode paysage



Tout comme la librairie jQTouch, cette galerie a été très récemment développée (Juin 2010). Cette version comporte encore quelques dysfonctionnements et certaines parties, comme la vitesse de transition lors de rotation de l'écran *Image 8*, doivent encore être optimisées. Des corrections seront certainement apportées lors de nouvelles versions de cette extension.

Navigation

L'une des particularités de jQTouch est que toute la partie HTML de l'application web se trouvent dans le même et unique fichier. Ainsi les boutons de navigation précédent et suivant du navigateur n'ont aucun effet sur l'application.

Toutes les différentes pages de l'application, comme la partie recherche textuelle ou visuelle, ne sont en fait que des régions.

Code 10. Structure HTML de MedSearch Mobile

```
<body>
      <div id="jqt">
            <div id="home" class="current">…</div>
            <div id="article">…</div>
      <div id="visual">">…</div>
…
      </div>
  </body>
```

Chaque page est, en faite, représentée par une balise `DIV` :

- Home : page de recherche textuelle
- Article : page de détail d'un article
- Visual : page de recherche visuelle

Depuis la page de recherche textuelle un clique sur l'onglet « visual » affiche à l'utilisateur la page de recherche visuelle. Pour effectuer ceci en HTML standard, une balise de lien serait `utilisée <a href="visual.php">Visual</a>` mais avec comme dans ce cas le fichier est le même il suffit de rendre visible la partie visual et de cacher la partie home en utilisant du JavaScript et jQuery en ajoutant `onClick="showVisual()"`.

Code 11. Navigation entre les pages avec jQuery

```
function showVisual() {
   $("#home").hide();
   $("#visual").show();
}
```

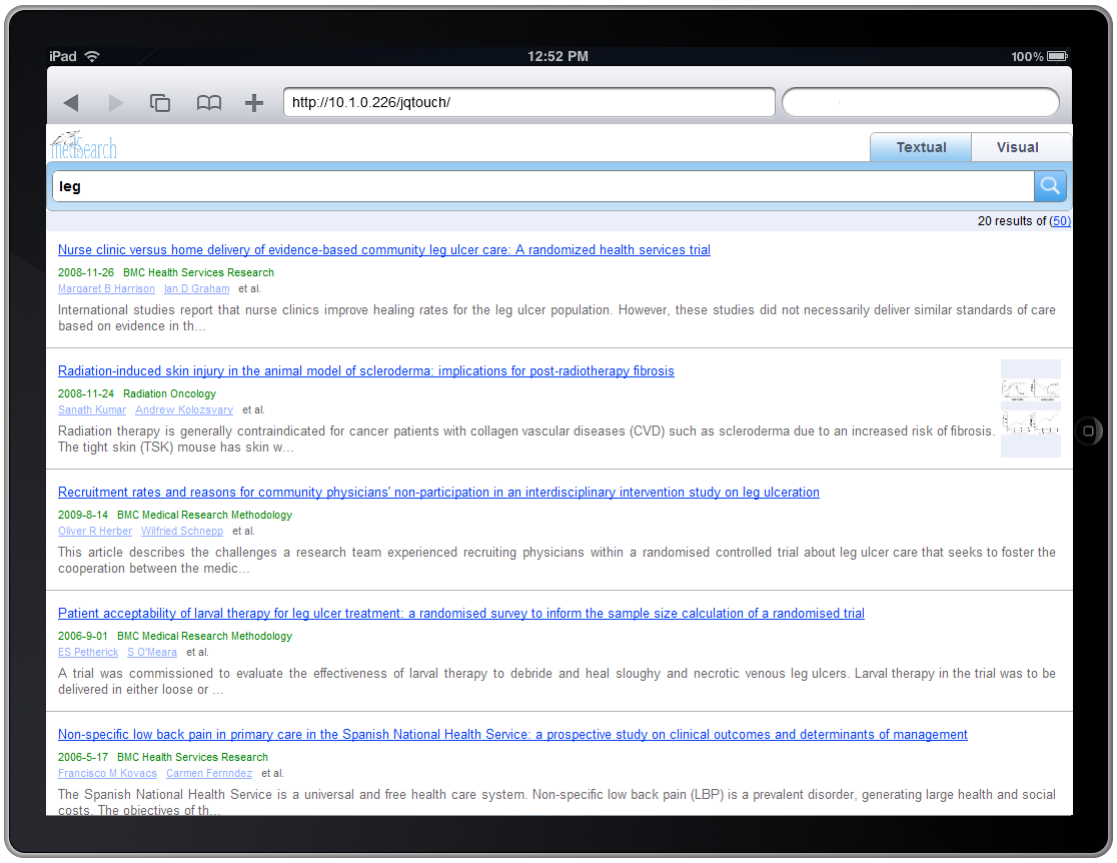
Cette fonction indique à jQuery (`$`) de cacher la `DIV` avec l'id home et d'afficher à la place celle avec l'id visual. Cette solution permet de ne pas avoir à attendre le chargement de cette nouvelle page car le fichier est le même et donc déjà chargé ; la transition est immédiate. Par la même occasion un grand confort est offert à l'utilisateur car il peut changer de page. Les données présentes sur la page cachée sont toujours en mémoire. Il pourra ainsi passer d'un onglet à l'autre et retrouver exactement la page qu'il a laissée.

7.5 Particularité de l'application

Support sur iPad

Grâce à l'emploi de la librairie jQTouch et au respect des normes de développement mobile, MedSearch Mobile s'adapte également parfaitement à une interface iPad.

Image 9. MedSearch Mobile sur iPad



Son mode de fonctionnement est parfaitement identique que sur un smartphone. Toutes les fonctionnalités sont disponibles sauf la prise de photos pour une comparaison d'images avec GIFT car l'iPad ne possède pas de caméra.

Par ailleurs, pour une recherche visuelle lors de comparaison avec une image stockée sur l'appareil et lors de résultats aléatoires, les vignettes de prévisualisation des images sont adaptées à la taille de l'écran. Ainsi, des vignettes sur écran iPad (9' pouces) sont donc plus grandes en taille que celles d'un iPhone (3.5' pouces) qui possède un écran plus petit.

Optimisation

Une application native a la particularité d'être téléchargée une seule fois sur l'appareil mobile de destination puis elle s'exécutera sur celui-ci. Dans le cadre d'une application web cela est quelque peu différent. Comme le principe d'un site internet, les fichiers HTML, CSS et JavaScript sont téléchargés à chaque accès par URL à cette ressource ou à chaque rafraichissement de page, puis mis en cache par le navigateur.

Cette issue apporte un avantage de taille car, lorsque l'utilisateur se connectera à l'application web, il récupérera automatiquement les fichiers les plus récents et ainsi il sera assuré de toujours consommer la dernière version. Pour une application de type native, il doit constamment la mettre à jour pour utiliser la dernière version.

Une application web n'exige pas de mises à jour côté client, mais le grand désavantage est que les fichiers qui la composent sont téléchargés à chaque nouvelle ouverture. Pour améliorer la rapidité de chargement et pour répondre aux exigences d'un téléphone mobile, comme vu au *Tableau 1*, une version minimisée des fichiers CSS et JavaScript a été générée au moyen de YUI Compressor[21].

En supprimant les données redondantes, les commentaires, les espaces inutiles et les retours à la ligne, le fichier qui contient une partie du JavaScript a passé de 12'955 Ko à 8'146 Ko soit un ratio de compression de 37%. Pour la partie CSS les chiffres sont moins éloquents mais la taille a tout de même pu être optimisée de 28%.

Code 12. Exemple de fichier CSS avant compression

```
body {
    background: rgb(255,255,255);
}

body > * {
    background: rgb(255,255,255);
}

h1, h2 {
    font: bold 18px Helvetica;
    text-shadow: rgba(255,255,255,.2) 0 1px 1px;
    color: rgb(76, 86, 108);
    margin: 10px 20px 6px;
}
```

[21] http://refresh-sf.com/yui/

Code 13. Exemple de fichier CSS après compression

```
body{background:#fff;}body>*{background:#fff;}h1,h2{font:bold
18px Helvetica;text-shadow:rgba(255,255,255,.2) 0 1px
1px;color:#4c566c;margin:10px 20px 6px;}
```

Détection du navigateur internet

Dans le cadre de MedSearch et de MedSearch Mobile plusieurs détecteurs de version et de navigateurs internet ont été recherchés. Cette solution permet d'atteindre en amont l'utilisateur lors d'accès à MedSearch. S'il accède à MedSearch depuis un appareil mobile il sera rediriger vers la version MedSearch Mobile et s'il accède depuis un PC il obtiendra la version standard.

La meilleure solution trouvée est d'utiliser une classe PHP qui reconnait directement le navigateur internet. Cette classe va encore plus loin car elle permet de connaitre aussi la version du navigateur, s'il est mobile et encore la plateforme sur laquelle il est exécuté.

Dans un premier temps, toutes ces fonctionnalités ne sont pas utilisées mais elles seraient intéressantes si notre application fonctionne avec la version courante ou supérieure d'un navigateur. Ainsi, un message pourrait avertir l'utilisateur de le mettre à jour. Si des fonctionnalités ne sont possibles que pour une sorte de navigateur ou de plateforme bien précise, nous pourrions rediriger les navigateurs qui n'ont pas Webkit comme moteur de rendu ou supprimer le bouton d'accès à Picup si la plateforme n'est pas iPhone.

Code 14. Exemple de détection du navigateur internet

```
$browser = new Browser();
if( $browser->getBrowser() == Browser::BROWSER_FIREFOX &&
$browser->getVersion() >= 2 ) {
   echo 'You have FireFox version 2 or greater';
}
```

7.6 Problème rencontré

L'un des objectifs de ce projet était de faire un choix judicieux concernant la portabilité de l'application vers la plateforme mobile. La création d'une application native offre beaucoup d'avantages par rapport à l'interaction avec les nombreuses possibilités du téléphone mobile mais elle se limite à son déploiement sur une unique plateforme. Cette solution limite de manière significative le nombre d'utilisateurs potentiels.

Une plateforme de type mixte était une excellente alternative car elle combine les avantages du natif pour l'exploitation des ressources du téléphone et le web pour son inter-comptabilité. Mais cette solution est pour l'instant beaucoup trop fragile, de part sa nouveauté et le manque de transparence concernant la validation par le constructeur d'application-tierce.

L'application web était donc la seule solution plausible qui répondait à cette possibilité de déploiement multiplateforme. Bien que, pour l'instant, l'absence de normes concernant les styles CSS3 limite l'accès à un navigateur basé sur le moteur de rendu Webkit. Mais la capacité d'exploiter les possibilités du smartphone est extrêmement limitée. L'utilisation d'application-tiers pour pouvoir prendre une photo et la déposer sur un serveur limite encore plus la multitude de possibilité que peut offrir l'appareil.

Au final, il n'existe, actuellement, pas vraiment de bonnes solutions pour développer une application qui pourrait tirer profit du matériel du téléphone et être déployée sur plusieurs plateformes. La création de normes et une politique claire en matière de validation d'application permettraient de rendre la plateforme mixte comme une bonne solution à court terme. Mais à plus long terme le web devrait rester comme une valeur sûre. Les constructeurs veulent garder main mise sur la distribution des applications et les ventes de celle-ci c'est pourquoi l'accès aux ressources est limité depuis le web. D'un point de vue technique l'accès aux données de géo-localisation est déjà possible mais tout le reste est bloqué volontairement. Une politique plus ouverte des constructeurs et la création de nouvelles normes dans le domaine du web mobile permettraient de créer de nouvelle ouverture dans le domaine du développement d'applications mobiles.

8 Conclusion

8.1 Analyse des résultats obtenus

Malgré tous les problèmes que posent un développement d'application mobile basé sur le web, l'application fonctionne parfaitement et rempli tous les critères définis par le cahier des charges. L'application s'adapte parfaitement au milieu mobile, pour lequel elle est destinée, par sa simplicité d'utilisation, son optimisation de l'espace sur l'écran, sa navigation vis-à-vis d'un écran tactile, la minimalisation d'utilisation de la bande passante et sa rapidité d'exécution.

De part sa compatibilité avec les navigateurs internet des principaux fabricants du marché des smartphonse, cette application remplit parfaitement le rôle qui lui est demandé. Elle suit la tendance de Google ou de YouTube qui, eux, favorisent un développement d'une application multiplateforme au lieu de plusieurs développements mono-plateforme.

L'adaptation à un environnement mobile d'un système qui permet de rechercher textuellement ou visuellement du contenu issu de la littérature de BioMedCentral est donc réussie.

8.2 Avantage et inconvénients de la solution proposée

La solution offre plusieurs avantages de tailles. Comme déjà décrit plus haut le fait qu'elle soit basée uniquement sur le web permet une portabilité vers tous les navigateurs internet basés sur Webkit. De plus, ce dernier s'impose comme la référence en matière de moteur de rendu car toutes les principales marques de smartphones (iPhone, Android, Blackberry) se basent dessus pour leur propre navigateur interne.

Vu que les pages sont rechargées à chaque nouvelle utilisation, l'utilisateur n'a absolument aucune mise à jour à effectuer. Dans le cas d'une application native à chaque changement de Firmware du constructeur ou de l'ajout de fonctionnalités une mise à jour est nécessaire. Dans ce cas, un simple déploiement sur le serveur web se répercute immédiatement sur tous les clients et tout cela sans avoir à gérer les problèmes de rétrocompatibilités.

Au contraire, une application basée sur le web n'a que des accès limités vers le matériel physique du téléphone mobile et une partie des plus-values que celui-ci pourrait apporter sont réduites à néant. Il existe quelques possibilités de contourner ce problème comme-ici l'utilisation de Picup, mais ces solutions ne sont que temporaires et fragiles et surtout liées à la plateforme du smartphone.

Bien que limité au strict minimum, le transfert de donnée via la bande passante du téléphone est capital dans ce type d'application car celle-ci est rechargée à chaque utilisation. Le forfait mobile qui limite donc le nombre de données peut donc poser problème. Comme cité dans la rubrique « problème rencontré », actuellement la bonne solution pour développer des applications mobiles n'existe pas.

8.3 Opinion personnelle et critique

Je suis quelque peu déçu que la communauté de développement et les constructeurs n'arrivent pas à trouver un terrain d'entente. Apple est connu pour sa rigueur dans la distribution d'applications notamment par la validation de celles-ci et la distribution par leur propre canal, App Store, uniquement. Tandis qu'Android est plutôt connu pour le contraire et offre beaucoup de liberté au propriétaire. Mais au final, lors d'une application web tous les deux sont sur le même pied d'égalité et bloquent une grande partie des fonctionnalités. Pour chacun d'entre eux, la vente d'application native représente un marché très lucratif et par conséquent l'arrivée des applications les dérange quelque peu.

D'un point de vue médicale et non technique l'application peut être très intéressante, cette recherche par comparaison offre réellement de nouvelles perspectives. L'outil GIFT devrait être optimisé pour avoir une meilleure comparaison d'images et ainsi profiter de résultats encore plus pertinents et précis.

Une application pour mobile fait tout son sens car d'après les statistiques, la navigation web par smartphone va rattraper, voir même dépasser, l'accès depuis les ordinateurs de bureau.

8.4 Amélioration

Nous avons vu plus haut, que la galerie image de jQTouch a été développée il y a moins d'un mois. Il faudra donc suivre ce produit car de nouvelles versions vont certainement corriger des bugs mineurs et l'optimiser dans sa performance. En particulier lors de la détection de rotation de l'écran.

Il faudrait également suivre l'évolution des applications web et particulièrement le choix des constructeurs de les intégrer et de donner accès aux possibilités qu'offre un smartphone. En particulier l'accès direct à la caméra qui permettrait d'éviter de passer par une application-tierce.

L'outil GIFT utilisé pour la comparaison d'images devrait faire l'objet d'une attention particulière car les résultats lors de recherche d'images similaires ne sont pas optimums.

Le marché des smartphones, et également celui des tablettes PC comme l'iPad, est en pleine effervescence. Il serait nécessaire de suivre attentivement les nouveaux produits qui vont sortir sur le marché et leur compatibilité avec cette application. Même si Webkit tend à devenir le standard des moteurs de rendu, il faudra constamment suivre les directions des constructeurs.

9 Remerciements

Je tiens à remercier chaleureusement ici toutes les personnes qui ont contribués, de près ou de loin, à la réalisation de mon travail de Bachelor.

Je porte une attention particulière à :

M. Henning Müller, pour m'avoir suivi et encadrer durant la réalisation de ce travail. Merci pour ta précieuse collaboration et tes sages conseils.

M. Ivan Eggel, assistant à la HES-SO Valais, créateur de MedSearch pour m'avoir épaulé durant la partie initiale du projet ainsi que pour l'assistance technique.

M. Yann Bocchi, professeur à la HES-SO Valais, pour ses conseils concernant le développement d'applications mobiles.

M. Xin Zhou, du service d'informatique médicale du HUG, pour la mise en place des comptes et du serveur.

Merci aussi aux personnes qui m'ont aidé dans la retouche stylistique et la relecture de ce document

Merci enfin à mon amie et à ma famille pour leur présence, leur soutien et leur encouragement tout au long de mes études.

10 Déclaration sur l'honneur

Je déclare, par ce document, que j'ai effectué le travail de bachelor ci-annexé seul, sans autre aide que celles dûment signalées dans les références, et que je n'ai utilisé que les sources expressément mentionnées. Je ne donnerai aucune copie de ce rapport à un tiers sans l'autorisation conjointe du RF et du professeur chargé du suivi du travail de bachelor, y compris au partenaire de recherche appliquée avec lequel j'ai collaboré, à l'exception des personnes qui m'ont fourni les principales informations nécessaires à la rédaction de ce travail et que je cite ci-après :

- M. Henning Müller, professeur chargé du suivi
- M. Ivan Eggel, assistant à la HES-SO Valais

Sierre, le 13 août 2010

Samuel Duc

11 Sources

11.1 Bibliographie

Frameworks mobiles

HES-SO Valais 2009-2010

Carol Hermann, Yann Métrailler, Alexandre Tacchini, Matteo Selva, Thomas Frick.

Prospects of Mobile Search

Instute for Prospective Technological Studies 2010

José Luis Gòmez-Barroso, Ramòn Compañò, Claudio Feijòo, Margherita Bacigalupo, Oscar Westlund, Sergio Ramos, Ajit Jaokar, Federico Àlvarez, Rudy De Waele, Gema Mateos-Barrado and Marià Concepciòn Garcìa-Jiménez.

Mobile Medical Image Retrieval,

Article soumis pour le SPIE Medical Imaging, Orlando - Florida, USA 2011

Samuel Duc, Adrien Depeursinge, Ivan Eggel, Henning Müller.

11.2 Webographie

Analyse

- http://www.slideshare.net/lis186/smartphone-market-trends (09.06.2010)
- http://distractable.net/coding/iphone-android-web-application-frameworks (19.05.2010)
- http://phonegap.com/
- http://rhomobile.com/
- http://jqtouch.com/
- http://www.appcelerator.com/
- http://code.google.com/p/iui/

Web service

- http://www.clever-age.com/veille/clever-link/soap-vs.-rest-choisir-la-bonne-architecture-web-services.html (22.06.2010)
- http://blog.goyello.com/2009/09/17/make-it-easy-test-it-easily-restful-web-services-with-java-and-soapui/ (17.06.2010)

Développement

- http://www.siteduzero.com/tutoriel-3-7930-presentation.html (11.06.2010)
- http://www.siteduzero.com/tutoriel-3-4525-presentation-du-dhtml.html (11.06.2010)
- http://www.siteduzero.com/tutoriel-3-160891-jquery-ecrivez-moins-pour-faire-plus.html (11.06.2010)
- http://www.reynoldsftw.com/2009/03/tutorial-from-php-to-xml-to-jquery-and-ajax/ (16.06.2010)
- http://api.jquery.com/jQuery.post/ (16.06.2010)
- http://www.reynoldsftw.com/2009/03/tutorial-from-php-to-xml-to-jquery-and-ajax/ (14.06.2010)
- http://en.wikipedia.org/wiki/Same_origin_policy (15.06.2010)
- http://www.phpfour.com/blog/2008/03/cross-domain-ajax-using-php/ (15.06.2010)
- http://www.rgagnon.com/javadetails/java-0598.html (25.06.2010)
- http://github.com/ardell/jQTouch-Gallery (28.06.2010)

Divers

- http://chrisschuld.com/projects/browser-php-detecting-a-users-browser-from-php/ (06.07.2010)
- http://hogtownconsulting.com/clearquery/index.html#search-results(29.07.2919)
- http://refresh-sf.com/yui/ (03.08.2010)

12 Tables

Tableaux

Images

Graphiques

Portion de code

13 Annexe

13.1 Cahier des charges

Introduction

Recherche d'informations médicales sur mobile

Pour des professionnels, un accès à l'information actuelle est extrêmement important car de nouvelles connaissances sont créées quotidiennement. La littérature scientifique médicale est en grande partie disponible sur Internet et peut être ainsi exploitée pour la recherche d'informations.

Ce travail de bachelor se déroule donc en plusieurs étapes. La première partie consiste à recueillir des sites qui proposent de la littérature scientifique médicale comme BioMedCentral, puis d'y extraire les articles, les images relatives ainsi que les descriptions des images. Durant la seconde étape, les différentes données amassées seront indexées selon leur type "article" ou "image".

La dernière partie consiste à créer une interface de recherche sur téléphone mobile qui inclut des images et des articles. Le système pourrait alors exploiter les différentes possibilités et contraintes d'un téléphone mobile de type Smartphone avec des données visuelles et textuelles.

Analyse

Analyse de l'existant

Un système de recherche similaire à mon travail a été développé par Ivan Eggel (MedSearch - IIG) mais avec un client de type browser.

Analyse Smartphone

Durant cette partie, le marché des Smartphones sera étudié pour en ressortir les tendances actuelles, et ainsi définir le type d'application client (web-mobile, native ou mixte) et les plateformes où le produit sera proposé. Une recherche des possibilités "matérielles" des Smartphones (caméra, tactilité,…) sera aussi menée pour permettre de créer une réelle valeur ajoutée au produit. De plus, une analyse des besoins pour une recherche d'informations sur téléphone mobile sera aussi nécessaire pour répondre au mieux aux contraintes d'un Smartphone.

Obligatoire:

- familiarisation avec le système existant de recherche,
- état de l'art sur les contraintes et possibilités des Smartphones,

- adaptation du système existant, avec une détection du browser et de la résolution, à une recherche d'info mobile.

Optionnel:

- utilisation de la caméra pour chercher des images similaires,
- navigation spécifique pour Smartphone.

Crawling HTML

Pour cette partie la base de l'application développée par Ivan pourra être repris. Néanmoins, une étude du code et une modification de la nature des données seront indispensable. En effet, les données nécessaires à notre système ne sont pas forcément les-mêmes que celui basé sur un browser.

Améliorations possibles

Actuellement le crawling se lance de manière manuelle. Il serait intéressant que celui-ci soit automatisé et que les nouvelles sources soient directement indexées par notre outil d'indexation.

Le moteur est "limité" à du parsing sur du HTML, or le parsing est très risqué comme moyen d'extraction de données car il dépend de la structure même du site. Il serait intéressant que l'on puisse extraire des données de différents formats de document comme du .doc ou du .pdf…

Obligatoire:

- crawling de quelques journaux (15-20) en cherchant le texte, les images et la légende des images.

Optionnel:

- plus de journaux,
- plus d'informations structurées.

Indexation

Cette phase permet de faire la liaison entre les données extraites des différentes sources de données et l'interface du moteur de recherche (partie client).

Comme avec le travail d'Ivan l'outil open source Lucene sera utilisé pour la recherche de texte, GIFT sera utilisé pour la partie de recherche sur des images.

Un des enjeux important sera de trouver un moyen de communication entre cette partie et la partie client (interface mobile).

Obligatoire:

- indexation des données obtenues avec Lucene et avec GIFT,
- communication avec les outils existants par sockets.

Optionnel:

- optimisation de la recherche.

Interface mobile

Le résultat de l'analyse sur les Smartphones permettra de définir la solution vers laquelle le projet sera développé ainsi que les plateformes où l'application sera développée.

Selon le choix, une orientation du projet devra être choisi car un client "web mobile" n'offre pas les mêmes possibilités et contraintes qu'une application native ou qu'une application mixte.

Également la technologie et l'environnement de développement dépendront directement de cette orientation. Une analyse et un comparatif des solutions de développement disponibles seront donc nécessaires afin de garantir au maximum le succès de la création de l'application cliente.

La solution développée devra être optimisée de manière à répondre aux critères déterminés par l'analyse des besoins pour une recherche d'informations sur téléphones mobiles.

Améliorations possibles

Une réelle plus-value pourrait être ajoutée en utilisant directement la caméra du Smartphone pour prendre une photo et immédiatement effectuer la recherche sur les images.

Le Smartphone nous offre un écran tactile ; une manière plus ludique et pratique pourra être implémentée, par exemple, pour l'affichage du résultat de la comparaison d'images.

Obligatoire:

- interface mobile qui s'adapte à une résolution limitée.

Optionnel:

- utilisation de l'appareil photo des téléphones mobiles,
- utilisation des fonctions tactiles du Smartphone pour la navigation.