Université SAAD DAHLEB de Blida

Faculté des Sciences

Département de l'Informatique

Mémoire présenté par :

- AIT SAADI Nadir
- ZERNADJI Halim

En vue d'obtenir le diplôme de Master

Domaine : Mathématiques et Informatique MI
Filière : Informatique
Spécialité : Informatique
Option : Ingénierie du Logiciel

Sujet :

Traduction et Normalisation Automatique du Langage SMS vers la Langue Française

Proposé et encadré par :

M[r] : SIDEMOU Mohamed Redha

2012/2013

Remerciement

En premier lieu nous remercions Allah le Tout Puissant, le Miséricordieux, qui nous a donné l'opportunité de mener à bien ce travail.

A Mr.SIDOUMOU promoteur de cette thèse, nous le prions de trouver ici le témoignage de notre reconnaissance, et notre plus profonde gratitude à tous jamais, pour l'aide précieuse et les conseils judicieux qu'il nous a prodiguée tout au long de cet humble travail. Encore grand merci.

A Mr.Cherif Zahar qui nous a encouragés depuis le début.

Au Professeur LEBBAH Yahia, professeur de l'université d'Oran, qui nous a guidés et conseillés tout au long de la réalisation de ce travail, pour son encouragement et sa disponibilité sans aucune limitation. Qu'il trouve ici l'expression de toute notre reconnaissance.

Au service de scolarité du département d'informatique, et surtout le bureau 1

Aux membres du jury qui nous ont fait l'honneur d'accepter d'évaluer notre thèse.

Hommages respectueux.

Dédicace

Je dédie ce travail,

A mon père '' رحمه الله '' : tu me manque beaucoup, ton souvenir resta, reste et restera à jamais gravé dans ma mémoire.

Paix à ton âme.

A ma mère : Maman tu m'as mis au monde, et depuis tu n'as pas cessé de me chérir, de m'encourager, de t'occuper de moi, de mettre à ma portée ce qu'il y'a de meilleur, tu n'as épargné aucun effort pour me rendre heureux.

A mes frères, Mohamed, Samir, Mounir, Hamza et Redha.

A mes chères sœurs, Badira, Fahima et Hanane.

A mes neveux et nièces que j'adore plus que tout.

A mes chers amis(es)...

Pour témoigner de la fraternité qui nous associé.

Une spéciale dédicace a ces personnes qui comptent déjà énormément pour moi, et pour qui je porte beaucoup de tendresse et de respect.

A mesdemoiselles et messieurs : CHERIF Zahar, GUENDOUZ Amina, SIDOUMOU Mohamed Reda, OUATTASSI Salima, BOUSTIA Narimène, MEZZI Melyara, LEBBAH Yahya, TOUBAL SGHIR Ismail, ZEBOUCHI Anouar, MOUZAI Boualem.

A mon cher ami, mon binôme, Nadir Ait Saadi chez qui m'a supporté durant ces dernières années.et chez qui j'ai trouvé l'entente dont j'avais besoin.

A toutes l'équipe de PC-MAX

À toutes les personnes que j'aime.

À tous ceux que j'ai omis de citer.

Halim ZERNADJI

Dédicace

Un Hommage à mon père, tu me manque, et si je suis arrivé jusque là, c'est surtout pour toi.

Je dédie ce modeste travail ;

A ma tendre mère, pour son soutien inconditionnel, son sacrifice, sa tendresse, et son amour infinis. J'espère pouvoir te rendre fière de moi et être à la hauteur de tes attentes.

A ma grand mère Yaya !

A ma chère sœur adorée, la perle de ma famille, qui m'a bercé, élevé, conseillé et surtout qui a toujours été compréhensive et disponible au moment ou j'en avais le plus besoin.

A mes deux Frères Farid et Mustapha.

A ma belle sœur Nabila.

A mon beau frère, Yahia, qui n'a jamais arrêté de me porter son aide et de me soutenir.

A mes neveux et nièces, Zakaria, Yousra, Hachemi, Wafaa, Asma, Othmane. Et surtout au tout dernier, Ahmed, qui a apporté joie et bonheur à la maison.

A mes deux anges gardiens Melyara et Samira, qui me donne force et foie chaque jour.

A mon collègue, ami et frère, Boualem, pour tout le soutien moral durant les deux dernières années.

A mes amis, Rabie, Salim, Nabil, Mahdi, Fayçal, Malik et Scaloop.

A Amina, Doli, Zahra et Salima.

Une spéciale dédicace à Anouar pour son coup de pouce magique.

A toute l'équipe PC-MAX !

*A mon binôme **Halim** qui m'a supporté durant toute une année alors que je ne me supporté pas moi-même.*

A ma « Binôme »…

Ait Saadi Nadir

Résumé

Dans le but d'obtenir le diplôme de Master en Informatique, nous nous sommes intéressés - afin de faciliter la tâche de lecture des messages téléphoniques écrits aux personnes mal voyantes – dans un premier lieu, à l'étude des techniques de normalisation du langage SMS en langage naturel. Cette étude nous a permis de retenir les automates sur lesquels nous nous sommes focalisés et réussit à mettre au point un travail qui a donné des résultats très encourageons pour la tâche convoitée. Nous envisageons par la suite d'intégrer ce travail dans le cadre d'un environnement mobile afin de le concrétiser.

MOTS-CLÉS : Le SMS, Automates à états fini déterministe, Traduction, normalisation, synthèse vocale.

Abstract

In the aim of getting the Master degree in Computer Science, we have been interested - in order to facilitate the task of reading mobile text messages to visually impaired people - in the first place, to the study of different technics of normalization of SMS language to natural language. This study has allowed us to retain the automaton on which we focused. The use of automaton gave very encouraging results for the coveted work. We plan eventually to include this work in the context of a mobile environment to concrete the work.

KEYWORDS: The SMS, Automates with finite state, translate, normalization, speech synthesis.

ملخص

من أجل الحصول على شهادة الماستر في الإعلام الآلي ، وضعنا اهتمامنا في تسهيل قراءة الرسائل القصيرة في الهاتف النقال من أجل فئة ضعاف البصر. في المقام الأول، ومن خلال دراستنا لتقنية تعديل الرسائل القصيرة إلى اللغة الطبيعية تمكنّا من الحصول على عدة أنظمة التشغيل الآلي.

العمل المقدم في هذه المذكرة يقوم بتطوير نظام من الأنظمة التي أعطى نتائج جيدة ومشجعة جدا في هذا المجال للحصول على هذه الوظيفة المرموقة، في النهاية نخطط لتنفيذ هذا العمل في بيئة متنقلة حتى يتسنى للمستخدمين التمتع بخدماته.

الكلمات الرئيسية: الرسالة القصيرة ، آلة الحالات المحدودة ، التركيب الصوتي .

Table des matières

Tables des illustrations

Liste des Tableaux

Introduction générale

Introduction Générale

Présentation du travail

Le marché de la téléphonie mobile est en pleine expansion depuis quelques années. L'utilisation de ces appareils a permis la prolifération d'un nouveau langage dit SMS (Short Message Service). Celui-ci consiste à réduire un texte en utilisant par exemple des abréviations ou la phonétisation des mots. Cependant ce langage n'est pas normalisé et il n'est employé que par une partie des utilisateurs. De plus de nouvelles règles sont continuellement inventées ce qui le rend très dynamique et très complexe à être assimilé de prime à bord.

Aujourd'hui, le lancement sur le marché public d'appareil mobile de plus en plus intelligents, nous ouvre l'horizon pour imaginer des solutions à divers problèmes de la vie quotidienne visant à faciliter cette dernière. Parmi ces problèmes, on trouve le problème de lecture des SMS par les personnes mal voyantes. Et c'est exactement à ce genre de problèmes que nous nous sommes intéressé afin de tenter d'y apporter une solution.

Problématique

Dans le monde de l'informatique, les nouvelles technologies ont donné de l'importance aux personnes handicapées, c'est dans ce but que des technologies comme la synthèse vocale et la reconnaissance vocale ont vu le jour, mais qui ont toujours quelques lacunes. Si on prend comme exemple, un malvoyant utilisant une application de synthèse de vocale pour son téléphone mobile, et qui reçoit un SMS, le mobile n'arrivera pas à lire correctement le message reçu à cause des différentes structures du langage SMS.

Du coup ce travail consiste à rendre facile la lecture des SMS, en appliquant un traitement automatique pour traduire les SMS vers la langue française. Afin d'arriver à cela, il faut commencer par une étude analytique de la structure du langage SMS et définir les règles dynamiques.

Vu que le langage SMS n'a pas de règles strictes et claires, car ça change d'une région à une autre et d'une époque à une autre, la définition des règles s'est avéré un vrai casse-tête, l'analyse de presque 100 SMS récoltés a aidé à l'élaboration d'une petite base de tests nous permettant de définir quelques règles. En effet, vu la non coopération des opérateurs téléphoniques algériens pour permettre la collecte d'un nombre important de SMS de manière automatique, nous avons dépêché une issue de secours consistant en une collecte manuelle qui a permis de constituer cette fameuse base de 100 SMS.

Objectifs

L'objectif de ce travail est de trouver une solution pour traduire le langage SMS vers la langue française, cette démarche est différente de la méthode classique qui consiste à chercher dans une table de données la correspondance entre un mot "SMS" et un mot de la littérature.

L'idée est d'implémenter les règles de traduction et de générer un automate qui englobera tous les mots de la langue française (thésaurus) pour faciliter la recherche. Après la reconstitution du message en langue naturelle, nous lui appliquerons une synthèse.

Pour concrétiser nos objectifs et rendre les résultats plus tangibles, le système mis au point est capable d'offrir les services de traduction aux utilisateurs via ordinateur. Plus tard, nous envisageons l'intégration de l'application dans un environnement mobile (les Smartphones en l'occurrence).

Organisation du mémoire

Afin d'atteindre les objectifs cités ci-dessus, notre mémoire s'articulera autour de quatre (04) chapitres, le premier présente l'état de l'art, il définit les éléments nécessaires du projet, les travaux déjà existants dans le domaine de la traduction et la normalisation des langages SMS avec une synthèse étalant les avantages et les inconvénients de ces derniers.

Le second chapitre, s'intéresse au traitement de langage SMS, où l'on analyse sa structure et la définition des règles d'écriture sur lesquels ce projet se base et aussi à la conception générale du projet ainsi qu'une architecture plus détaillée.

Le troisième chapitre détaille l'implémentation de l'application en définissant les différents systèmes d'exploitation, et sur quel choix s'est porté notre projet, les outils utilisés pour la réalisation.

Le dernier chapitre comprend les tests de l'application, ainsi que les avantages et les inconvénients de celle-ci.

Chapitre I

Etat de l'art

Chapitre 1 : L'Etat de l'art

Introduction

La normalisation SMS a jusqu'à présent été abordée selon trois angles différents : correction orthographique, traduction automatique et reconnaissance de la parole. Chaque approche, basée sur des postulats différents, gère efficacement certains des phénomènes présents dans les SMS.

Dans ce chapitre, on va définir quelques concepts de base et résumer les travaux antérieurs concernant la normalisation.

Concepts de base

1.1 Le SMS

Les SMS (Short Message Service) [1] correspondent à des petits messages écrits de 160 caractères maximum envoyés d'un téléphone portable à un autre. Ce mode de communication écrite issu de la téléphonie mobile s'est développé au cours des années 1990 et a connu par la suite un essor considérable, touchant toutes les couches de la population et plus particulièrement la population adolescente.

La pratique des SMS a donné naissance à une écriture expressive, abrégée et phonétique s'éloignant à bien des égards des usages orthographiques habituels.

I.2 Automates à états fini déterministe

Un automate (ou machine) à états fini [2] est un modèle abstrait utilisé pour représenter le fonctionnement d'un programme ou de tout autre dispositif susceptible d'être décrit en termes d'états et de transitions entre des états (circuit électronique, ascenseur, machine à café…).

Le modèle de l'automate est assez commode pour donner une représentation graphique favorable à l'intuition de certains processus, dont la description sans tel procédé serait trop complexe. En outre la famille des automates finis à états ouvre la voie vers les langages réguliers, dont on démontre qu'ils leur sont équivalents, et les langages réguliers, sont un formalisme puissant avec des applications pratiques très variées, notamment pour le traitement de données textuelles.

Un automate fini est constitué d'états en nombre fini ; il passe d'un état à un autre par une transition ; le diagramme de l'automate décrit les transitions possibles.

L'automate évolue (passe d'état en état) en fonction d'un mot qui lui est soumis ; il comporte un état initial, à partir duquel sont parcourus des arcs de transition déterminés pas les symboles successifs du mot soumis en entrée.

I.3 La synthèse vocale

La synthèse vocale [4] est une technique informatique qui permet de créer de la parole artificielle à partir de n'importe quel texte.

Un système de synthèse à partir du texte (TTS) est une machine capable de lire à priori n'importe quel texte à voix haute, que ce texte ait été directement introduit par un opérateur sur un clavier alphanumérique, qu'il ait été scanné et reconnu par un système de reconnaissance optique des caractères (OCR), ou qu'il ait été produit automatiquement pas un système de dialogue homme-machine. Un tel système diffère fondamentalement d'autre machines parlantes en ceci qu'il est destiné à donner lecture de phrases qui n'ont en principe jamais été lues auparavant. Il est en effet possible de produire automatiquement de la parole en concaténant simplement des mots ou des parties de phrases préalablement enregistrées, mais il est clair dans ce cas que le vocabulaire utilisés doit rester très limité et que les phrases à produire doivent respecter une structure fixe, afin de maintenir dans des limites raisonnable la quantité de mémoire nécessaire à stocker les éléments vocaux de base.

Les Travaux existants

2.1 Dictionnaire de SMS

Il existe sur Internet des dictionnaires de SMS (*www.dictionnaire-sms.com*, *www.mobilou.info/10kosms.htm* , *www.1fo.co*) (Consulté le 12 mai 2013).

Cependant, ces dictionnaires représentent une ressource limitée puisqu'ils sont parfois dépassés vue la rapidité de l'évolution du langage SMS. De plus, ils ne prennent compte qu'un synonyme ou deux. En effet, il existe pour un même mot une vaste quantité de variant, exemple pour ***demain*** → *2m1, dmain, dmin, 2main, dem1, 2min, dems...etc.*

2.2 Correcteur automatique proposé par Vienney et Melian

Le correcteur automatique appliqué au SMS de S. Vienney et C. Melian [7] se découpe en cinq étapes : la lecture du texte source, la segmentation du SMS, la transcription en français standard, un module d'analyses morphologiques, syntaxiques et sémantiques et enfin, la proposition d'un texte cible, correction du texte source.

Certains phénomènes de construction du langage SMS ne sont pas simples à prendre en compte pour réaliser une segmentation. Si on prend comme exemple, l'agglutination et l'utilisation de sigles : comment faire pour découper « jallais » (j'allais) ou « tkt » (t'inquiètes) ? Le module de transcription en français standard nous apparait très complexe. Il se fonde « sur un ensemble de règles de transcription traitant l'ensemble des phénomènes de néographies [...] puis il calcule des hypothèses de transcription avec une analyse lexicale et combinatoire »

Les transcriptions qui résultent de ce module vont être validées ou non par l'analyse morphosyntaxique et sémantique.

A la suite de ces traitements, le système transmet à l'utilisateur un texte en français standard. Cependant, quelques ambigüités demeurent comme, par exemple, avec des systèmes de traitement automatique de l'oral : les pois sont verts/ les poissons verts.

2.3 Système de normalisation des SMS d'Yvon

Après avoir observé une proximité entre les formes d'écriture utilisées dans les SMS et la langue orale, F. Yvon [8] propose un système

de normalisation des SMS inspiré des systèmes de reconnaissance de la parole.

Ce système est optimisé pour un système de vocalisation de SMS puisqu'il passe par une étape de phonétisation.

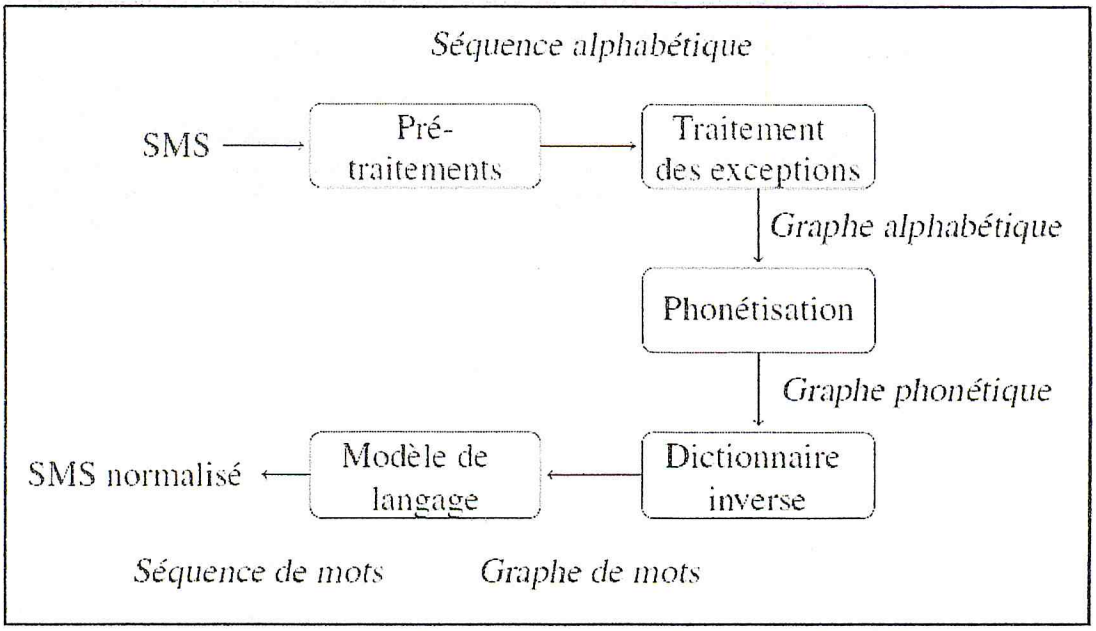


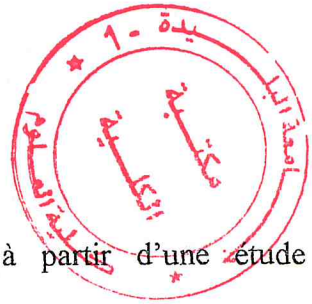
Figure 1 : Schéma du système [8].

Présentation des modules du système de F. Yvon [8]

Dans un premier temps, le SMS passe dans le module de traitement. Ce dernier converti le message en entrée en un automate fini. Puis il réalise des opérations de normalisation telles que le traitement des chiffres, l'insertion de marques de débuts et fin de phrase et fin de mots, la suppression de la ponctuation et des majuscules.

Le second module va traiter les exceptions en utilisant un dictionnaire des formes abrégées avec leur correspondance en français standard.

Exemple : pr → pour.



Ce dictionnaire a été élaboré manuellement à partir d'une étude de corpus.

Le troisième module de modèle orthographique vise à modéliser les récritures graphiques en traitant les phénomènes tels que l'absence d'accents, les fautes de frappes, l'écriture phonétique et les rébus. Il comporte deux transducteurs, l'un pour établir les être phonétisé en fonction de son contexte. Ce module crée une liste de phonétisation possibles. Les exceptions détectées au module précédent vont être phonétisées à partir d'un dictionnaire de prononciation.

Le quatrième module correspond à l'accès dictionnairique. Une fois la liste de phonétisations possibles établie, le module utilise un dictionnaire de prononciation qui associe une séquence de phonèmes en séquences de mots. Si les mots du message sont connus, ils seront associés aux mots du dictionnaire, par contre, s'ils sont inconnus, ils seront associés à « <unk> ».

Le cinquième module applique un modèle de langage statistique de type n-gram pour faire ressortir la séquence de mots la plus probable.

Avantages et Inconvénients du système

F. Yvon (2008) concède lui-même que son système pourrait être amélioré. Il faudrait traiter les accords, les dépendances longue distance, le

2.4 Le traitement automatique des SMS de Beaufort

Le système proposé par Beaufort [10] repose entièrement sur des lexiques, des modèles de langue et des règles de réécriture compilés en machines à états finis (finite-state machines, FSMs) et combinés avec le texte à traiter par composition. Les FSMs et leurs propriétés fondamentales sont expliqués dans [11] et [12]. Nous utilisons no propres outils à états finis : une bibliothèque de FSMs et son compilateur associé.

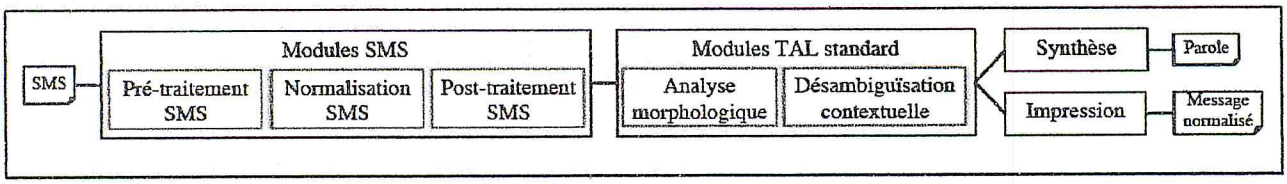


Figure 2 : Architecture du système [10]

Comme l'illustre la figure 2, un SMS passe d'abord au travers de trois modules SMS qui en normalisent les parties bruitées. Deux modules TAL réalisent ensuite une analyse morphosyntaxique du texte normalisé.

Le dernier module, enfin, dépend du type de sortie désiré : soit un synthétiseur, qui construit le signal de parole correspondant au texte normalisé sur la base de son analyse linguistique, soit un module d'impression, qui produit le texte normalisé et lui applique les règles typographiques fondamentales (majuscule en début de phrase, présence ou absence d'espaces entre les unités du texte, etc.) en se basant sur les unités détectées par les modules de pré- et de post-traitement SMS.

Le module de prétraitement. Au sein d'un texte, ce module repère exclusivement les séquences suivantes : fins de paragraphes

Le module de prétraitement. Au sein d'un texte, ce module repère exclusivement les séquences suivantes : fins de paragraphes et de phrases, URL, numéros de téléphone, dates, unités de mesure et de temps, monnaies…etc.

Toute autre séquence de caractères est considérée comme une unité bruitée et subit l'étape de normalisation. Ceci rapproche la méthode de la métaphore orientée correction.

Le module de normalisation : Les modèles et les lexiques utilisés ici sont tous appris. Inspirée de la métaphore du canal bruité.

Ce modèle est le résultat de nombreux tests, qui ont mis en évidence le fait que distinguer les séquences connues et inconnues améliore considérablement l'efficacité du système, sans nuire aux performances.

Le module de post-traitement : Ce dernier module SMS n'est appliqué qu'à la version normalisée des unités bruitées, afin d'y identifier toute séquence non alphabétique et de l'isoler dans une unité distincte.

A ce stade, par exemple, un point devient une « ponctuation forte ». Les grammaires locales utilisées ici sont plus complètes que celles du prétraitement, car elles peuvent – et doivent – détecter les séquences numériques et alphanumériques, les champs de données (comme les numéros de cartes bancaires), les ponctuations et les symboles.

Avantages et Inconvénients du système

Les séparateurs manquants (Pensa ms → Pense à mes) ou superflus (G t → J'étais) sont globalement bien gérés.

Le prétraitement est utile, puisque les unités non ambiguës ne sont pas modifiées.

Les erreurs sont souvent contextuelles : elles concernent le genre (quel(le)), le nombre (bisou(s)), la personne ([tu t']inquiète(s)) ou le temps (arrivé/arriver).

Conclusion

Dans ce chapitre nous avons étalez les différentes approches existantes pour la traduction et la normalisation du langage SMS. Ce qui nous a permis de constater les résultats de chaque approche, de les comparer et d'en sortir avec une synthèse. Cependant, dans l'ensemble, normaliser un SMS reste un problème complexe et non résolu : les meilleurs systèmes, en effet, descendent difficilement en dessous des 11% d'erreurs au mot. Dans le prochain chapitre, nous allons nous intéresser à la présentation et la conception de notre solution.

Chapitre II

Solution Proposée

Chapitre 2 : Solution proposée

Introduction

Afin de présenter notre approche nous allons commencer par résumer notre solution, puis de définir les concepts de base de traitement du langage SMS. Par la suite, nous présenterons l'architecture générale de notre système ainsi que la conception détaillée et la logique de la solution proposée. Enfin, nous détaillerons le schéma global de traitement ainsi qu'un extrait de l'automate généré.

Solution proposée

La Solution proposé dans ce cadre de travail est beaucoup plus proche de l'axe correction orthographique que de l'axe Traitement de langage et traduction littéraire. Le but est d'arriver à un résultat acceptable, rapidement et efficacement, et cela en générant un automate à états finis englobant les mots de la langue française, pour une recherche optimisé, puis de classer chaque mot du SMS dans sa catégorie, faire la normalisation et chercher le mot le plus proche dans l'automate. Une fois la normalisation faite, on applique la synthèse vocale sur le résultat obtenu.

Traitement du langage SMS

Les procédés d'écriture employés dans les SMS sont divers et nombreux. Dans cette analyse, nous exposerons quelques-uns des phénomènes relatifs à l'écriture des langages SMS étudiés.

Ces procédés sont expliqués ci-dessous :

3.1 L'écriture rébus

Un rébus, littérairement, désigne un jeu de mots basés sur l'emplacement graphique des lettres.

Dans le langage SMS, un « rébus » [5] désigne le procédé d'écriture par lequel certaines séquences de lettres sont remplacées par un arrangement de chiffres et/ou de lettres correspondant au même phonème (un phonème est un Elément sonore distinctif du langage) [9] que la séquence en question. Ce phénomène est considéré souvent comme un des plus caractéristiques du Langage SMS.

On distingue deux types de rébus,

3.1.1 Le rébus de lettres

Ce qui consiste à remplacer un son par une lettre, avec la même valeur sonore qu'on lui confère quand on la dénomme.

Concernant ce phénomène, la lettre est souvent mise en majuscule pour marquer sa valeur particulière.

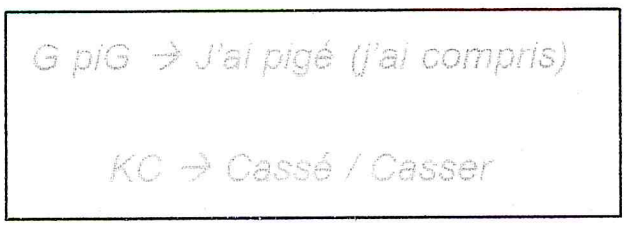


Figure 3 : Exemple de rébus de lettre

3.1.2 Le rébus de chiffres

Ça part du même principe que le rébus de lettre sauf que dans ce cas là, c'est la valeur phonétique de la dénomination chiffrée qui est utilisé.

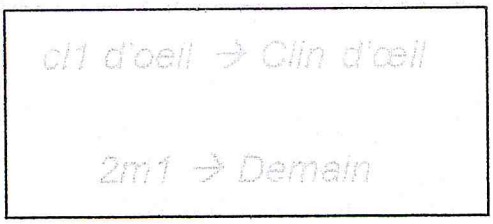


Figure 4 : Exemple de rébus de chiffre

Il se peut aussi qu'on trouve des mots qui comportent une composition des deux types de rébus (lettre et chiffre)

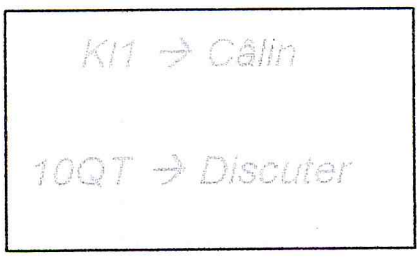


Figure 5 : Exemple de rébus chiffre et lettre

3.2 Squelettes consonantiques

Nous considérons comme squelettes consonantiques [5] les mots dont les voyelles ont été supprimées, réduisant ainsi la forme à une succession des consonnes principales du mot. Nous savons depuis longtemps, que les consonnes ont une valeur informative plus forte que les voyelles. Le mot français écrit est fortement charpenté autour des consonnes, dont certaines n'ont pas de contrepartie phonique. Exemples :

Slt = salut

partt = partout (notons que le « t » muet final a été conserver)

tjrs = toujours (de la même manière ici le « r » pourtant prononcé n'est pas conservé alors que le « s » muet l'est)

3.3 Les agglutinations

Nous appelons enfin « agglutination » **[6]** la formation d'un mot par la réunion de deux ou plusieurs unités lexicales (jpouré = « je pourrai »).

Il est à noter que nous avons traités uniquement le cas des agglutinations binaires (combinaison de deux unités).

Forme Agglutinée	Exemple de patrons observés	Exemple de formes
« JE »	J + pronom J + verbe	jte (Je te) jsui (Je suis)
« QUE »	K + article défini K + pronom démonstratif K + pronom personnel	kle (Que le) kce (Que ce) ktu (Que tu)
« CE / SE »	S + pronom S + verbe	ske (ce que) svoir (se voir)
« ME »	M + pronom M + verbe	mle (me le) mparl (me parle)

Tableau 1: Exemple d'agglutination

On appelle troncation [6] le procédé par lequel on crée un nouveau mot en supprimant une ou plusieurs syllabes d'un mot plus long. On subdivise les troncations par apocope, la perte de lettre ou de syllabe finale, et par aphérèse, la suppression de lettres ou de syllabe au début du mot.

Exemples :

Ordinateur = ordi
Salut = lut
Téléphone = phone.

Architecture Générale du système

L'architecture générale du système est résumée dans la figure qui suit :

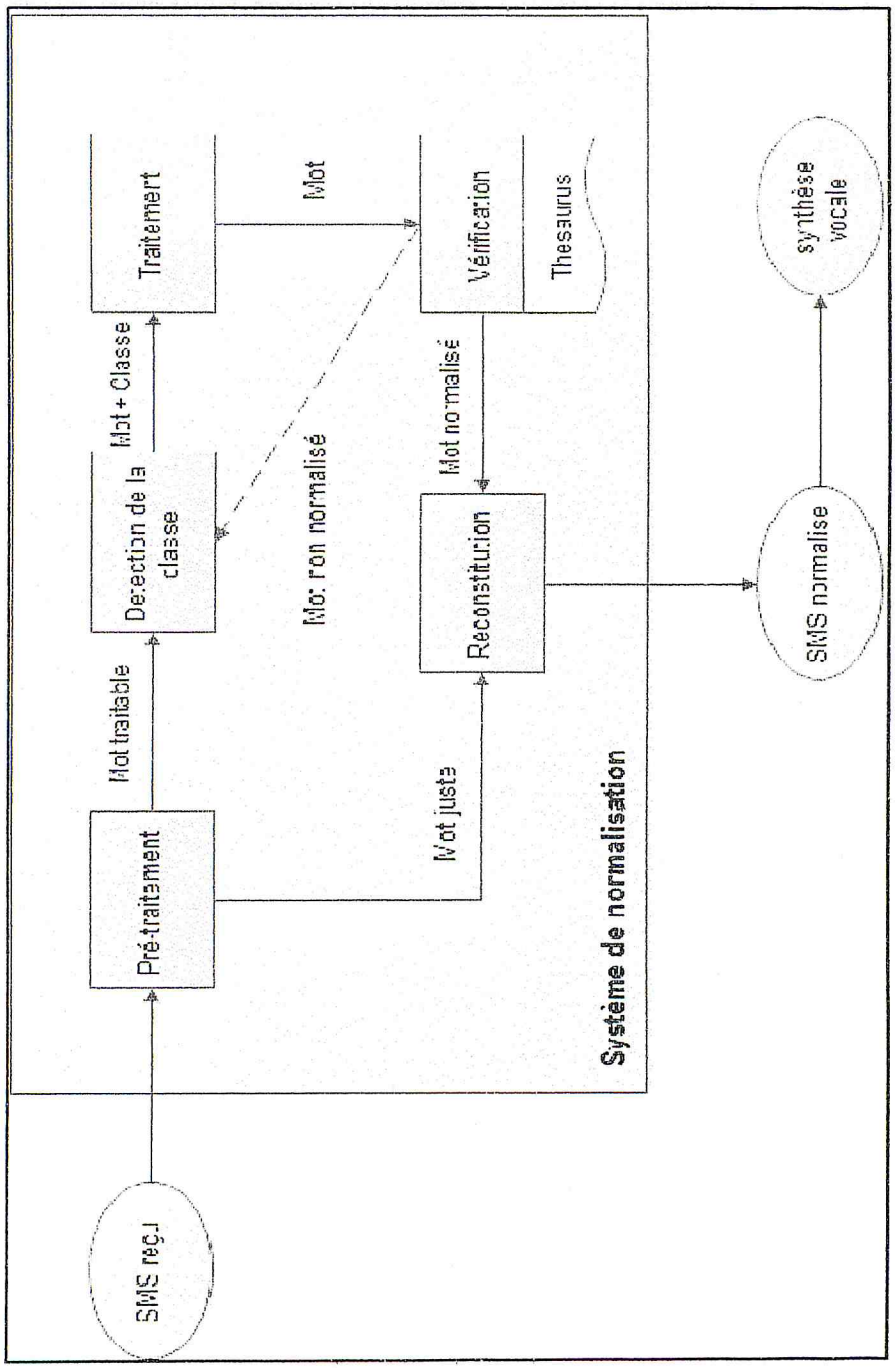


Figure 6 : Architecture générale du système

La Figure ci-dessus montre la structure de l'application réalisée. Le processus global de reconstruction d'un SMS reçu passe par cinq (05) modules :

A- Le prétraitement

La première, le prétraitement, s'occupe de deviser le SMS reçu mot par mot et d'écarter les mots écrits juste.

Dans le cas où l'on reçoit un SMS *« slt belle amie, on svoi 2m1 ».*

Le module prétraitement va prendre soin de diviser le message pour pouvoir le traiter et définir les mots qui vont subir des changements.

La division se fait grâce au Regex [13] (Regular Expression), une expression régulière définit un modèle de recherche pour les chaînes. Les expressions régulières peuvent être utilisées pour rechercher, modifier et manipuler du texte. Dans notre cas c'est pour découper le message en utilisant le méta-caractère « \W+ » qui veut dire que le traitement se fera sur les caractères non-mots, c'est-à-dire les espaces vides. Donc pour chaque espace croisé le système enlève le mot et le place dans une liste.

On aura,

slt
belle
amie
on
svoi
2m1

Après ça le système définit si le mot a besoin d'être traiter ou non.

	slt	*belle*	*amie*	*on*	*svoi*	*2m1*
	Traitable	Juste	Juste	Juste	Traitable	Traitable

Tableau 2 : Prétraitement d'un SMS reçu

B- Détection de classe

Le module de détection de classe, consiste à définir quel traitement doit se faire pour le mot en question. Nous avons trois (03) classes principales qui s'occuperont de traiter les rébus, Squelettes et agglutination et pour détecter la classe, le mot va être analysé, si il contient des chiffres ou des lettres en majuscule, alors c'est un rebus, sinon on vérifie si c'est un squelette consonantique ou une troncation et si il n'appartient pas aux deux premières classes alors il est forcément une agglutination.

slt	Non	Oui	/
svoi	Non	Non	Oui
2m1	Oui	/	/

Tableau 3 : Détection de la classe

C- Traitement

Après la détection de la classe, le module traitement exécute l'algorithme approprié pour chaque mot et fait la normalisation du mot qui va être juste ou non,

Dans ce module, trois (03) traitements sont possibles,

Traitement des rébus

Si le mot est un rébus, l'algorithme va décortiquer le mot en symbole, tout ce qui est chiffre et lettres en majuscule va subir une conversion à partir d'un fichier qui regroupe les rébus les plus vues. C'est ainsi qu'un *« 2 »* va devenir *« de / deu »* et un *« G »* va devenir *« gé / jé / geai / jai ».*

Après conversion, plusieurs propositions vont être générées par le système, et il va choisir le mot qui sera juste dans la langue française.

2m1	Demin	Demain
	Demain	
	Demein	
	deumin	
	deumain	
	deumein	
	deumun	

Tableau 4 : Exemple de traitement de rébus

Traitement de squelettes et Troncations

Si le mot n'est pas un rébus, le module traitement va balayer un fichier à la recherche du mot qui convient

slt	Salut
svp	S'il vous plait
stp	S'il te plait
qqch	Quelque chose
lgtps	longtemps
rdv	Rendez vous
bcp	beaucoup
ss	suis
jspr	J'espère
msg	message

Tableau 5 : Liste des Squelettes les plus fréquents

Ordi	Ordinateur
Ciné	Cinéma
lu	Salut
tel	Téléphone
phone	Téléphone
télé	Télévision
Diapo	Diaporama
Manif	manifestation
re	reviens

Tableau 6 : Liste des Troncations les plus fréquentes

Traitement des Agglutinations

En troisième lieu, si le mot n'ai pas juste, ne contient pas de rebus, n'est pas un squelette et commence par un *« j,k,s,t »*, alors on décompose le mot en lui retirant la première lettre (le j deviendra *je,* le k deviendra *que*...) et on vérifie la 2e partie du mot.

Pour notre exemple *« slt belle amie, on svoi 2m1 »*. Le mot svoi deviendra alors **se voi**

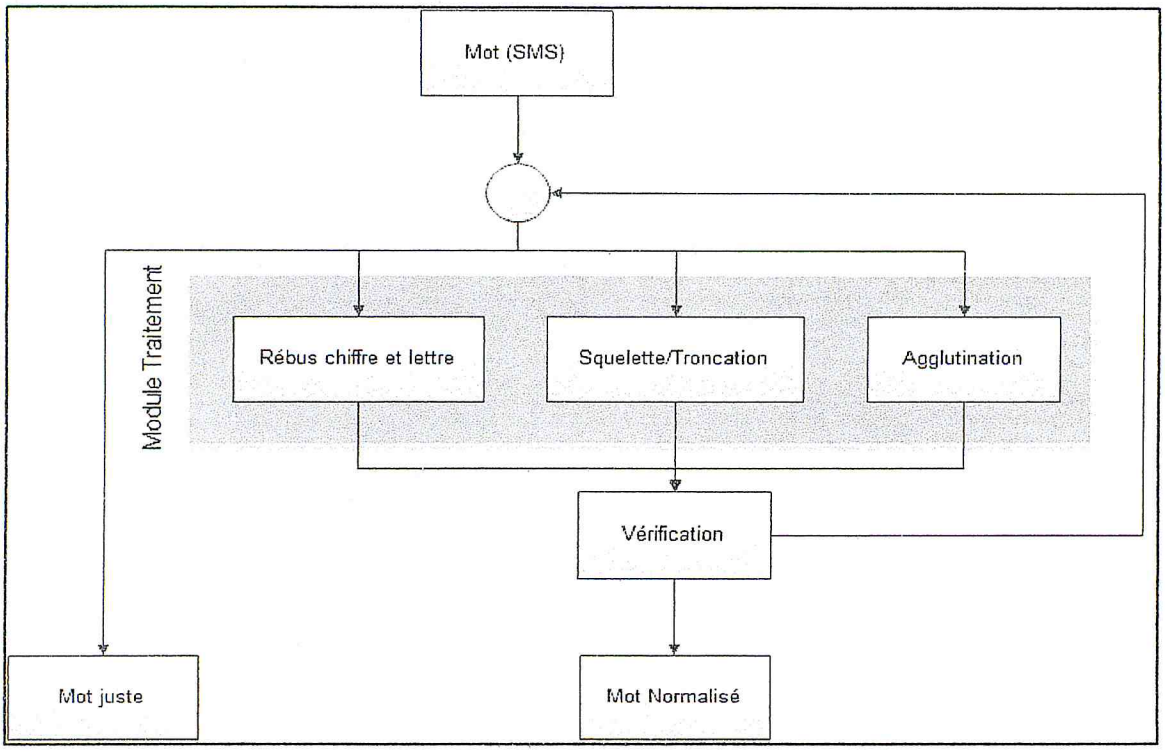


Figure 7 : Schéma de traitement

D- Vérification

Après avoir traité le mot on passe au module de vérification, où le système va confirmer si le résultat obtenu est un mot juste ou non. En effet il y a des cas où le mot doit avoir plus d'un traitement, surtout pour les agglutinations, qui ont besoin d'un deuxième traitement pour le traduire. La traduction se fait grâce à un automate à état fini que nous avons mis au point et qui génère plus de 336 000 mots de la langue française (thesaurus), son but est de rendre la vérification très rapide et de définir si le mot est juste ou pas.

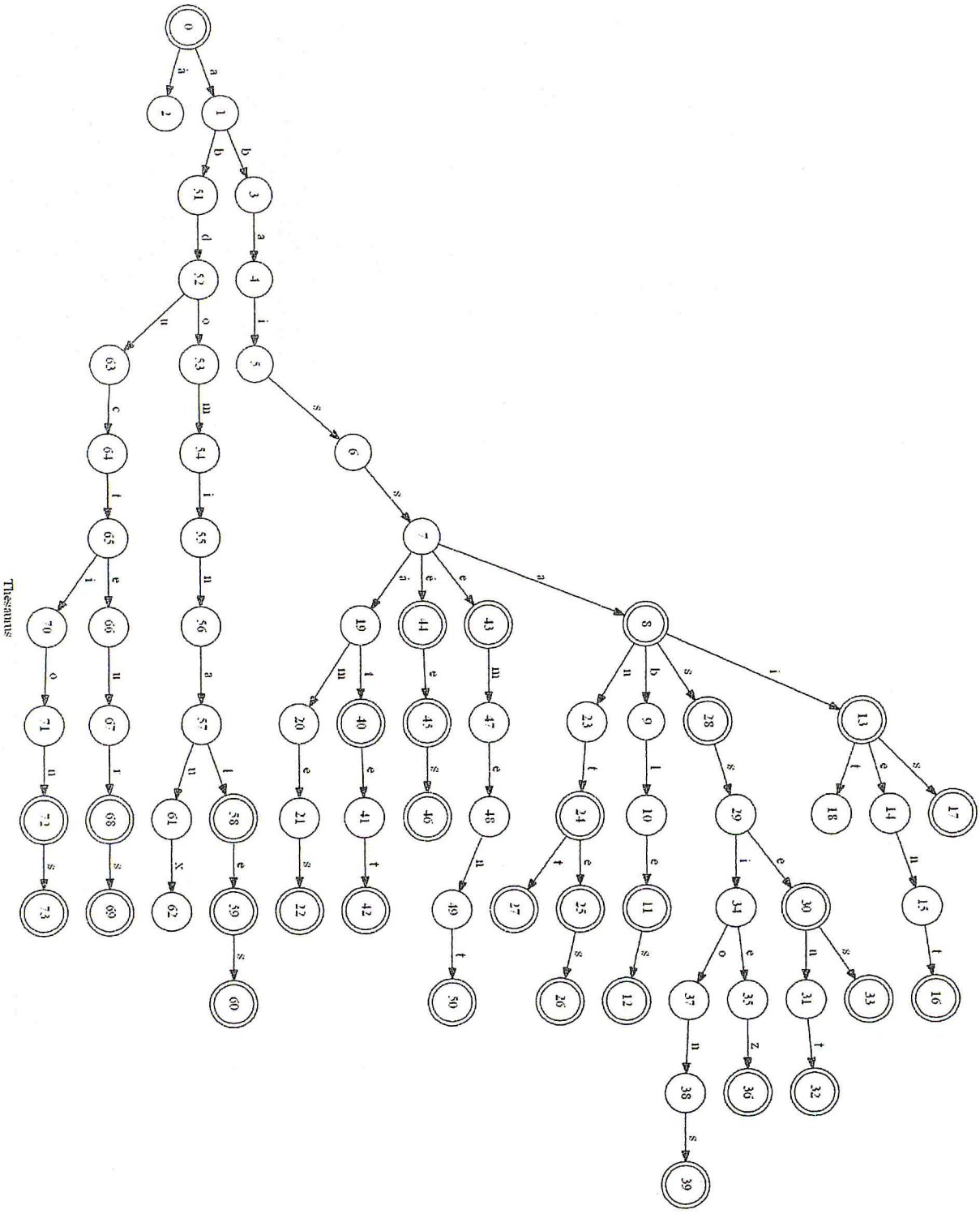


Figure 8: un extrait de l'automate généré

E- Recomposition

Après vérification de tous les mots, le système reforme le SMS reçu sous sa forme normalisée, et sous un environnement mobile, le SMS va être synthétisé.

Algorithme Général du Système

```
Algorithme SMStoFr
Variables
mot : String ;
liste : Liste de String ;

Début
RecupererSMS() ;
DecomposerSMS();
Pour chaque mot dans liste
 faire
      Mot ← listeSMS.mot
      Si MotJuste() ;
           alors
                inserer (mot, listeRetournée)
           Sinon
                Si motRebus()
                alors
                     traitementRebus() ;
                 Sinon
                     Si motSquelette()
                     alors
                     traitementSquelette() ;
                     Sinon
                          Si
                          motAgglutination() ;
                               alors

 traitementAgglutination() ;
                     fsi
                fsi
           fsi
      fsi
finPour

recomposerSMS();
afficherSmsJust ();
FIN
```

Conclusion

Dans ce chapitre, nous avons présenté les fonctionnalités de notre système. Nous avons commencé par une conception architecturale globale de la solution proposée, puis détaillée insistant sur le module de traitement. Dans le chapitre suivant, nous allons aborder l'implémentation de notre système, les outils qui nous ont servis pour sa réalisation, ainsi que les résultats obtenus.

Chapitre III
Implémentation

Chapitre 3 : Implémentation

Introduction

Dans ce chapitre, nous présentons les différents systèmes d'exploitations utilisés dans le marché du mobile ; les raisons du choix du système pour le projet ; approfondir son étude. De plus, faire la connaissance des outils de réalisations du projet.

Systèmes d'exploitation mobiles

Un système d'exploitation mobile [14] est un ensemble de programmes responsable de la gestion des opérations, du contrôle, de la coordination, de l'utilisation du matériel et du partage des ressources d'un dispositif entre divers programmes tournant sur ce dispositif.

Parmi les systèmes d'exploitation nous avons,

Windows mobile

Le Windows Phone [15] (mobile) est un système d'exploitation spécialement développé pour un usage mobile. Développé par Microsoft, le Windows Phone (anciennement Windows Mobile) offre donc une interface principalement conçue pour les Smartphones qui leur donne en outre accès à la plate-forme d'applications de Microsoft, Market Place.

Le 15 février 2010 Microsoft a lancé un nouveau système d'exploitation pour mobile, Windows Phone 7. Il intègre des

fonctionnalités média sociaux tel que Facebook et Twitter. Et en 2012 Microsoft a lancé le Windows Phone 8. [14]

L'iOS [14] Anciennement appelé « iPhone OS » est un système d'exploitation conçu par Apple, qui est dérivé deMac OS X. Il fonctionne sur iPhone et iPad et a pris une part significative du marché.

La technologie BlackBerry de la companie Canadienne RIM (Research In Motion) [16], permet de recevoir et envoyer des courriers électroniques ainsi que de se brancher à internet via un terminal mobile de poche.

Le mode de compression utilisé réduit le poids du message, ce qui facilite la synchronisation de ses courriers électroniques avec le serveur de messagerie électronique. Cette synchronisation s'éffectue via le réseau de téléphonie mobile sur lequel l'appareil est connecté (GSM, GPRS, UMTS...). Ainsi, envoyer un email s'apparente à la simplicité d'envoyer un SMS ou un MMS.

Android [17] est un OS pour téléphone mobile et tablette tactile, promu par Google et l'Open Handset Alliance qui comprend plus de 35 constructeurs, fournisseurs de logiciel, et opérateurs. Il concurrence des plateformes telles que l'iOS d'Apple, Windows Mobile de Microsoft, RIM OS intégré dans les BlackBerry de Research In Motion, WebOS d'HP Bada de Samsung, ou encore Symbian et MeeGo de Nokia.

Leur plateforme est un OS basé sur GNU/Linux entièrement gratuit, sous licence open source Apache 2. Le kit de développement (SDK) et le code source d'Android sont disponibles depuis novembre 2007 en version 1.0. La version actuelle des sources est la 2.2 (Froyo), et la version 3.0 (Gingerbread) devrait sortir normalement en novembre 2010.

Android a une très grande communauté de développeurs qui produisent des applications diverses et variées pour étendre les fonctionnalités du système d'exploitation. Il y a actuellement plus de 100 000 applications sur l'Android Market, ce qui en fait le deuxième environnement de développement le plus populaire, derrière iOS. Les applications sont 15 écrites pour la plupart en Java, et peuvent utiliser le hardware et les fonctionnalités du système via des librairies Java développées par Google.

Le fait que le code soit complètement ouvert, permet à de nombreux développeurs de le modifier pour y ajouter des fonctionnalités ou corriger des bugs, et ainsi de proposer à la communauté des builds personnalisés, souvent plus avancés que les versions officielles proposées par les constructeurs.

Raisons du choix d'Android

Avec l'augmentation du nombre de Smartphones qui vendent sur le marché mondial mobile, le nombre de plates-formes de système d'exploitation pour développer des applications pour les Smartphones a également augmenté. Android remplace progressivement iOS en termes de système d'exploitation le plus populaire pour le développement des applications pour les Smartphones. Bien que le système d'exploitation de Google a fait face à la rude concurrence des autres grandes plates-formes

contemporaines, il est apparu que le système d'exploitation le plus vendu de 2011 et 2012, Voici les raisons de notre choix :

1- Facile langage de programmation aux développeurs Android ont besoin d'apprendre le langage de programmation très courante de Java pour créer et efficaces succès des applications mobiles. C'est la raison pour laquelle la plupart des développeurs app à travers le monde veut travailler sur la plateforme Android,

2- Plateforme de développement ouverte Depuis Android est une plate-forme de développement d'application ouverte.

3- Système d'exploitation polyvalent La polyvalence de la plate-forme Android app permet aux concepteurs de construire des applications dynamiques mobiles qui servent à des fins multiples après avoir été installé dans l'appareil. La nature polyvalente de la plate-forme Android permet également aux développeurs pour en savoir plus à partir du processus de conception.

4- Testés Environnement Android OS a l'un des meilleurs environnements de test app par rapport aux rivaux principaux systèmes d'exploitation. Les outils de test disponibles en vertu de cette plate-forme sont soigneusement répertoriés. L'IDE Android offre la meilleure qualité de modèle de code source.

La Figure ci-dessous représente les parts du marché des OS mobile dans le monde en 2012 et 2013

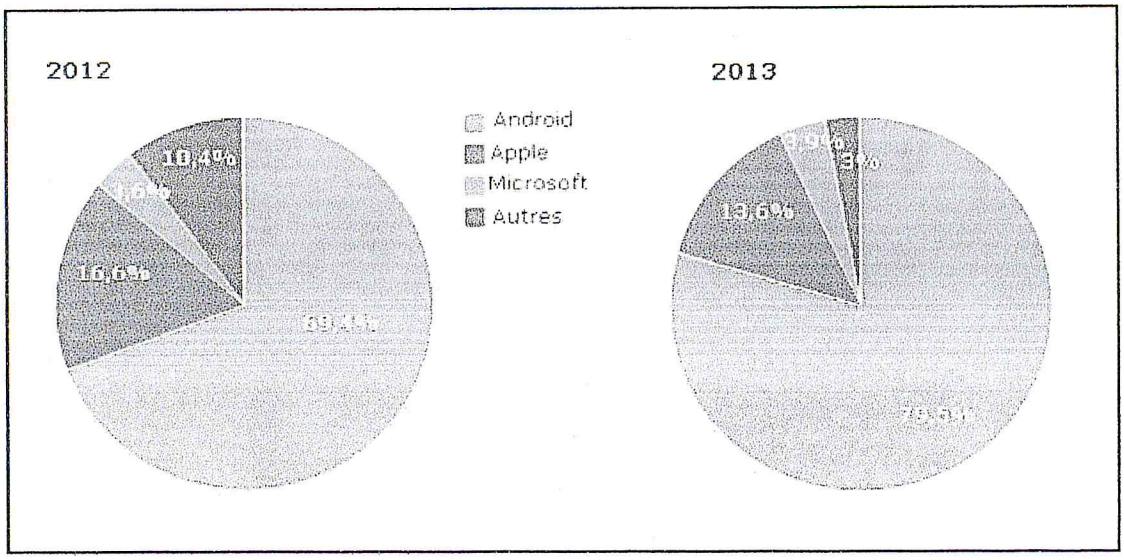


Figure 9 : Parts du marché des Os mobile dans le monde [18]

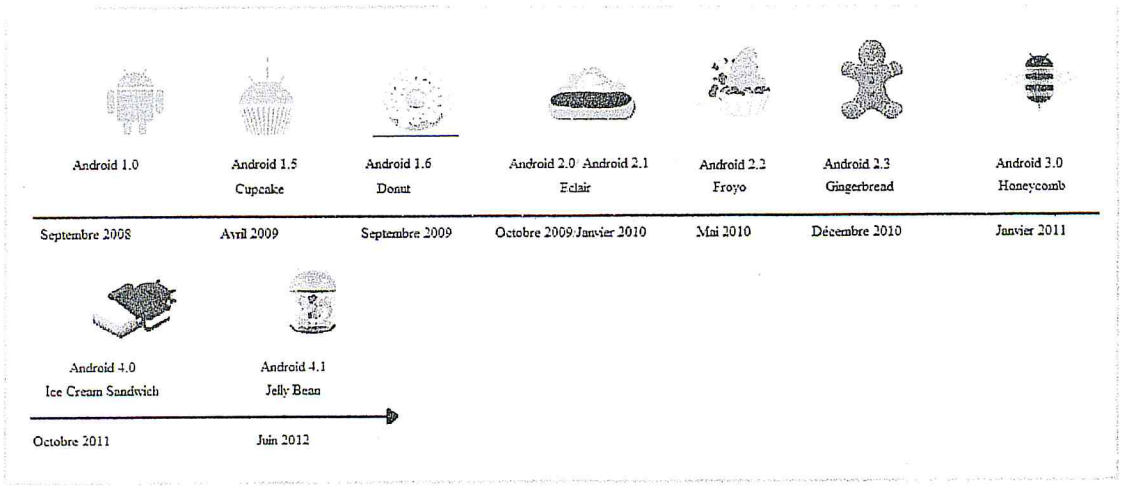


Figure 10 : Evolution des versions d'Android [19]

Installation du kit de développement JAVA (JDK)

On procède à l'installation de l'environnement JAVA Oracle JDK

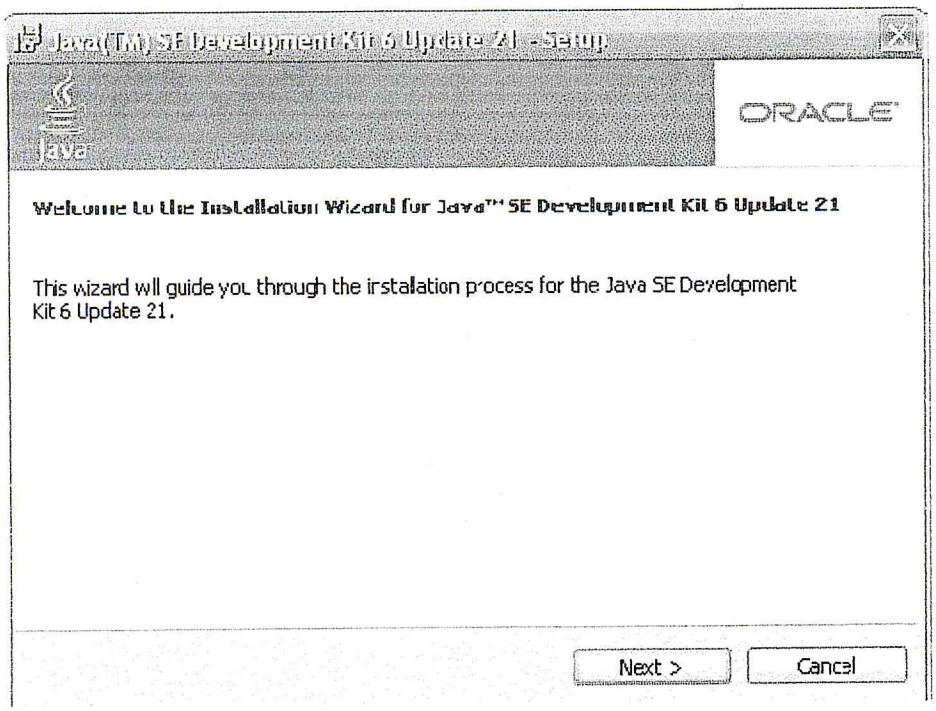


Figure 11 : l'environnement JAVA Oracle JDK

Ce projet été implémenté en java, l'environnement le plus adéquat pour la programmation java est l'Eclipse IDE qui est un environnement de développement intégré libre (le terme Eclipse désigne également le projet correspondant, lancé par IBM) extensible, universel et polyvalent, permettant potentiellement de créer des projets de développement mettant en œuvre n'importe quel langage de programmation. Eclipse IDE est principalement écrit en Java (à l'aide de la bibliothèque graphique SWT, d'IBM), et ce langage, grâce à des bibliothèques spécifiques, est également utilisé pour écrire des extensions. [20]

On continue avec l'installation des outils nécessaires comme suit :

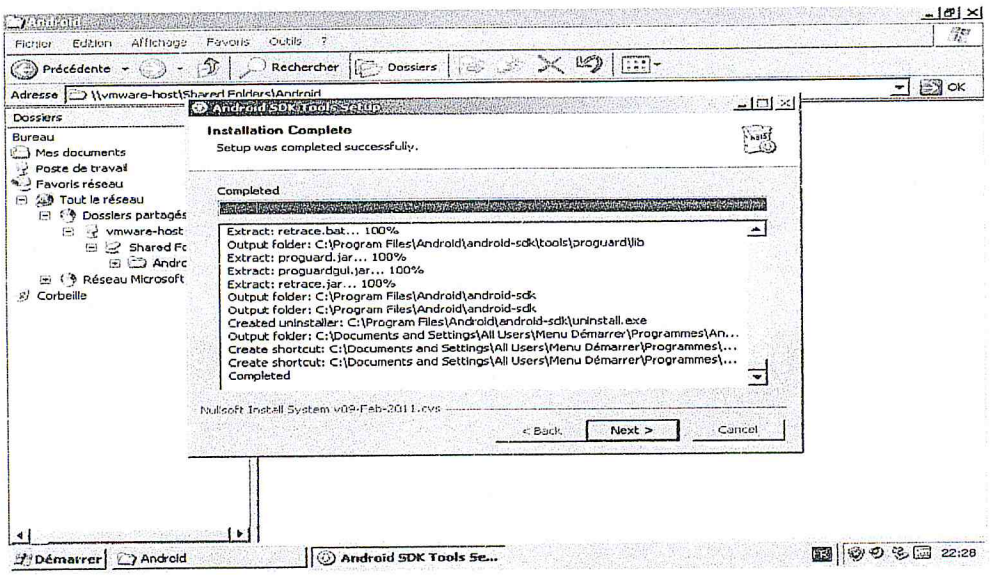


Figure 13 : Installation de SDK

Le SDK est l'outil le plus important, il permet de télécharger tous les outils indispensables pour le développement des applications, il permet aussi de télécharger toutes les versions d'Android, ainsi que les versions de Google APIs pour intégrer les fonctionnalités liées aux services Google tels

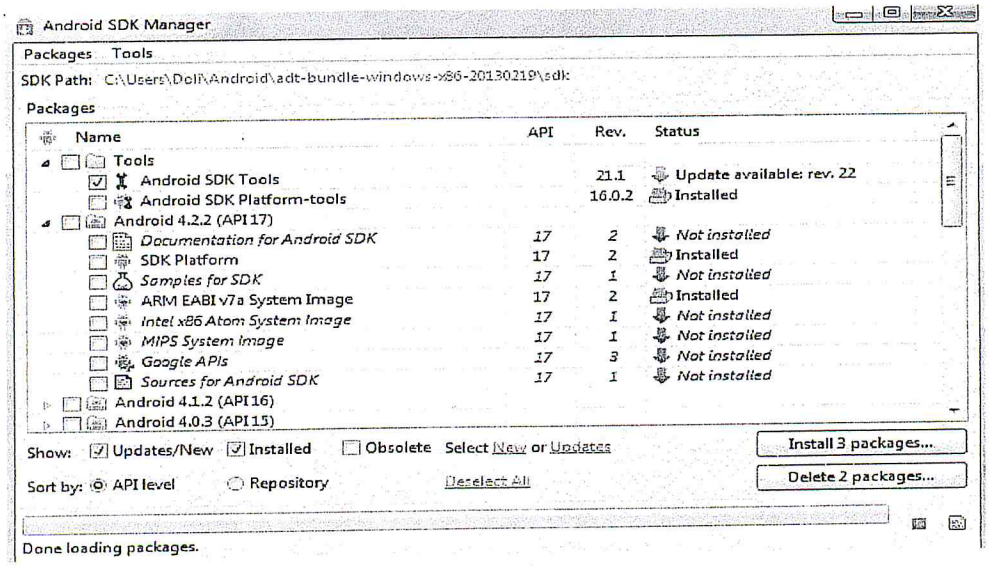


Figure 12: installations de déférentes versions d'Android

On continue avec l'installation des outils nécessaires comme suit :

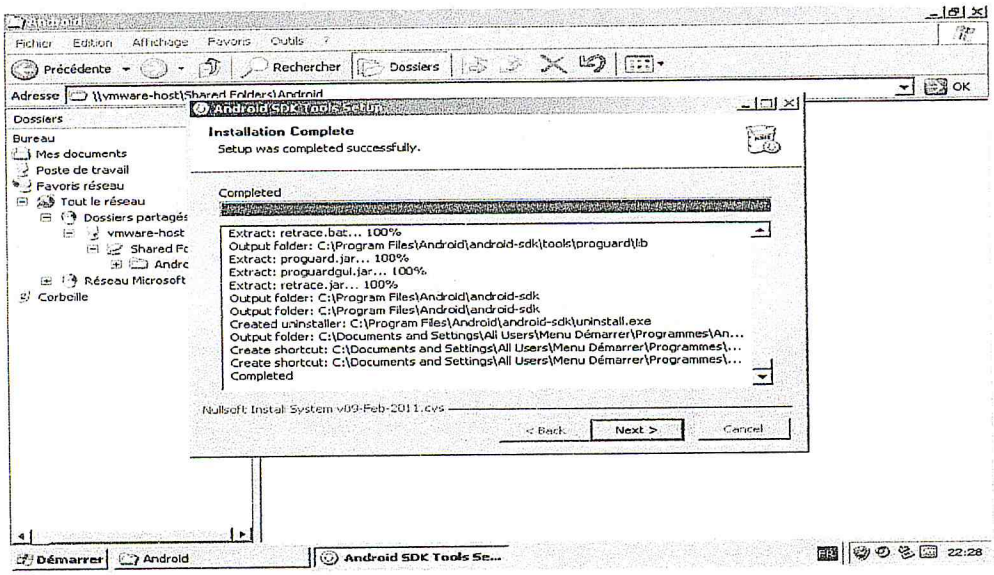


Figure 12 : Installation de SDK

Le SDK est l'outil le plus important, il permet de télécharger tous les outils indispensables pour le développement des applications, il permet aussi de télécharger toutes les versions d'Android, ainsi que les versions de Google APIs pour intégrer les fonctionnalités liées aux services Google tels que MAP etc. On peut aussi télécharger de la documentation Java Doc. La figure suivante est l'interface du SDK Manager qui nous permet d'installer les différentes versions d'Android ainsi que d'autres outils.

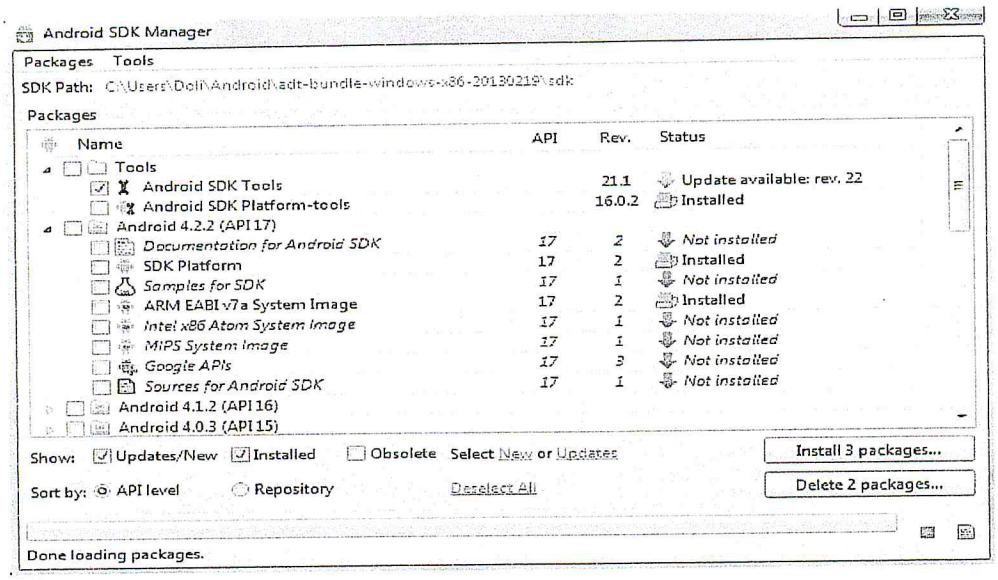


Figure 13: installations de déférentes versions d'Android

ADT pour Eclipse

L'ADT (Android Developement tools) est très complets et facile à l'utilisation et à la manipulation de la conception graphique d'interface utilisateur, debug distant sur un téléphone, la gestion de l'architecture de fichier d'une application etc. Dans le cadre de notre projet, nous avons utilisé la dernière version d'Eclipse, Juno et le plugin ADT de Google

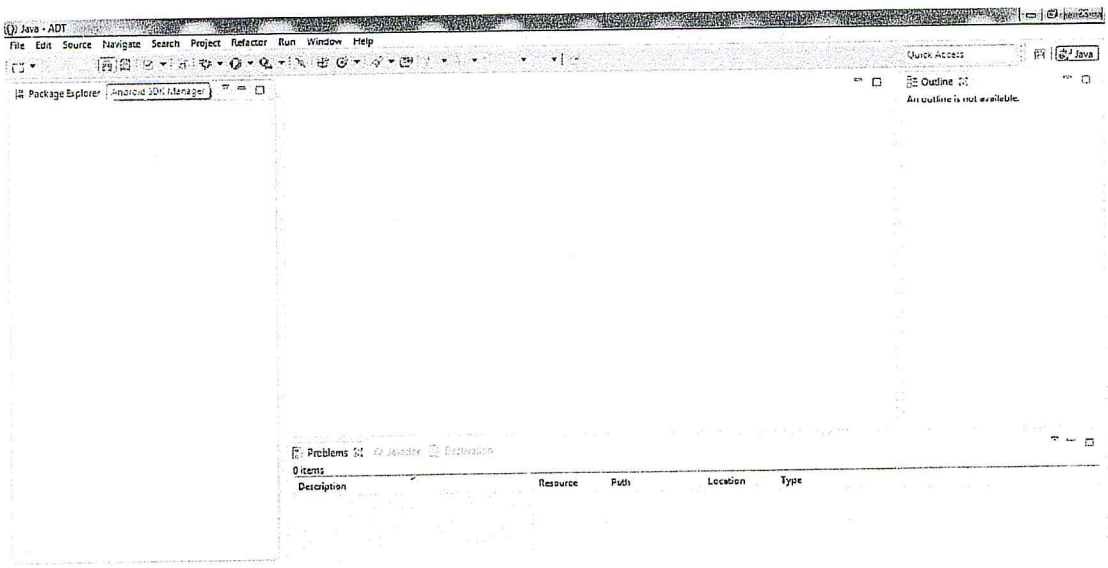


Figure 14 : L'interface de l'ADT

Résultats de la réalisation des scénarios

Ce projet a pour but de faciliter la lecture d'un SMS, pour pouvoir réaliser ce projet nous avons créé deux zones de texte, une pour saisir un SMS et l'autre pour afficher sa traduction, ainsi qu'un bouton de traduction et un bouton pour la synthèse vocale.

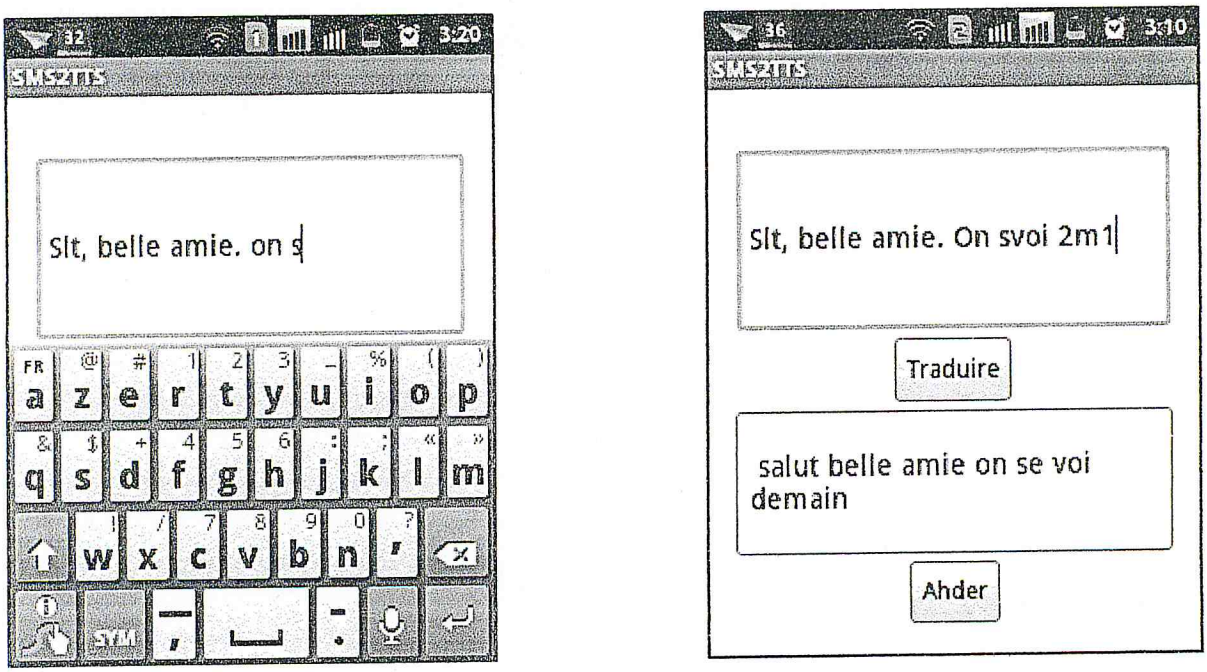


Figure 15 : Capture d'écran de l'application sous Android

Conclusion

Ce dernier chapitre a présenté les différents systèmes d'exploitation mobiles connus, une présentation du système Android, ainsi que les outils de réalisation du projet. Et enfin, un aperçu de l'interface du projet sous Android.

CHAPITRE IV

Tests et Validation

Chapitre 4 : Tests et validations

Introduction

Pour évaluer le degré de pertinence de la normalisation et la reconstruction des SMS reçus grâce à notre système, nous avons procédé à son évaluation. Et ce, en utilisant quelques mesures inspirées du domaine de recherche d'Information **[21]**, comme : précision, rappel, overall et f-mesure (dites mesures de performance du système).

Définition des mesures de performance utilisées

Les mesures utilisées pour évaluer la qualité des messages reconstruits sont principalement les mesures de calcul de la pertinence en recherche d'information, telles que la ***précision*** et le ***rappel***.

Le calcul de ces mesures, est basé sur la comparaison entre les normalisations effectuées par notre système que nous noterons **S** et l'ensemble des normalisations correctes qu'un humain peut détecter que nous noterons **H.**

- Les normalisations *correctes trouvées* par le système sont appelées « *the true positives (TP)* » et sont calculées ainsi :

$$TP = S \cap H$$

- Les normalisations *incorrectes trouvées* par le système sont appelées « the false positives (FP) » et sont calculées ainsi :

$$FP = S - S \cap H$$

- Les normalisations *correctes omises* par un système sont appelées « the false négatives (FN) » et sont calculées ainsi :

$$FN = H - S \cap H$$

- La *précision* est une mesure d'exactitude, elle varie entre [0,1] elle est calculée de la manière suivante :

$$\boldsymbol{Precision} = \frac{|TP|}{|TP + FP|} = \frac{S \cap H}{S}$$

- Le rappel est une mesure de perfection, elle varie entre [0,1], elle est calculée de la manière suivante :

$$\boldsymbol{Rappel} = \frac{|TP|}{|TP + FN|} = \frac{S \cap H}{H}$$

Les valeurs obtenues par le calcul des normalisations sont comparables par le biais de la précision et du rappel, en fait, *le rappel peut prendre des valeurs importantes aux dépens de la précision*, en retournant toutes les normalisations possibles.

En même temps, *la précision peut prendre des valeurs importantes aux dépens du rappel*, en ne retournant que les normalisations correctes cependant peu nombreuses.

- C'est pour ces raisons qu'il est préférable de prendre en considération les deux mesures simultanément via une mesure qui combine le rappel et la précision telles que : la *F-mesure* qui se calcule de la manière suivante :

$$\boldsymbol{F - Mesure} = \frac{2 * (\boldsymbol{Rappel} * \boldsymbol{Précision})}{\boldsymbol{Rappel} + \boldsymbol{Précision}}$$

La *F-mesure* est une mesure globale de la qualité des normalisations produites, elle varie entre [0,1], cette mesure alloue la même importance à la précision et au rappel.

- Une autre mesure de la qualité qui combine le rappel et la précision : *l'overall* qui se calcule de la manière suivante :

$$\boldsymbol{OVERALL = Rappel(2 - \left(\frac{1}{Précision}\right))}$$

L'overall peut prendre des valeurs négatives si le nombre de fausses normalisations (FP) trouvées par le système dépasse le nombre de normalisations correctes (TP) trouvées par le système.

Dans la plupart des cas ***l'overall*** est plus petit que le rappel et la précision, ce qui rend difficile d'atteindre un ***overall*** supérieur à **0.5**.

- Une mesure pour évaluer *le pourcentage d'erreurs* du système automatique est le *Fallout* qui se calcule de la manière suivante :

$$\boldsymbol{Fallout = \frac{FP}{FP + TP}}$$

Jeu de données

Afin de valider notre système, nous avons choisit un jeu de données qui consiste à prendre un SMS constitué de 17 mots. Et ce, par le calcul des mesures de performance citées précédemment. Enfin, nous allons afficher les résultats.

Le sms sur lequel nous avons effectuer le test est le suivant,

« *slt belle amie, jspr ktu va bien, dsl pr hier G pa pu vnir on svoi 2m1*»

Les mots du SMS	Normalisation	Normalisation du

Les mots du SMS	Normalisation d'un humain	Normalisation du système
slt	**salut**	***salut***
belle	**belle**	***belle***
amie	**amie**	***amie***
jspr	**j'espère**	***j'espère***
ktu	**que tu**	***que tu***
va	**vas**	***va***
bien	**bien**	***bien***
dsl	**désolé**	***désolé***
pr	**pour**	***pour***
hier	**hier**	***hier***
G	**j'ai**	***geai***
pa	**pas**	***pa***
pu	**Pu**	***pu***
vnir	**venir**	***vnir***
on	**on**	***on***
svoi	**se voit**	***se voi***
2m1	**demain**	***demain***
Correspondances produites par le Système : |S| = 06		
Correspondances produites manuellement : |H| = 09		

Tableau 7 : Comparaison entre la normalisation d'un humaine et du système

Le tableau précédent montre les normalisations faites par un être humain et celles effectuées par notre système. Pour le calcul des bonnes normalisations, nous nous sommes intéressés aux mots normalisés qui seront correctement synthétisés en message vocal par notre système. Et de ce fut les normalisations grammaticales non effectuées comme « pas », ou « vas » ont volontairement été omises car elles ne rentrent pas dans le cadre de notre problématique principale.

Mesure de performance du système

TP	FP	FN	Précision	Rappel	*F-mesure*	OVERALL	Fallout
0,66	0	0,33	1	0,66	0.79	0.66	0

Tableau 8 : Résultat des mesures

Le tableau précédent montre que notre algorithme donne des résultats plutôt satisfaisant pour ce jeu de données (pour le moins complexe) avec un taux de précision élevé, un bon rappel, et un faible taux d'erreur (fallout). Par la suite, nous envisageons d'effectuer d'autres tests avec un plus grands nombre de SMS.

Conclusion

Dans ce chapitre, nous avons testé l'application, et mesurer la performance du système, qui ont donné des résultats assez satisfaisant, et aussi nous avons calculé le temps d'exécution, ce qui a donner un traitement qui se fait en ***1.3 sec.***

Conclusion Générale

Conclusion et perspective

Le projet que nous avons réalisé consistait à trouver une solution facile, utile et rapide pour la normalisation des SMS. Le principal objectif était que le système doit être en mesure de traduire un SMS vers la langue française selon des règles préétablis

Dans le cadre de la recherche d'une solution possible, nous avons retenu les automates à états finis, pour la pertinence des résultats qu'ils permettent d'obtenir dans le cas où ils sont bien définis. Une étude approfondie des travaux existants, nous a permis de constater la complexité du projet, car tous les travaux avaient des résultats satisfaisants mais à des taux de correction faible.

La première étape était de définir les règles de normalisation, vue que le langage SMS évolue rapidement, nous avons rencontré pas moins de 20 règles, mais on s'est focaliser sur les 3 règles les plus pertinentes pour notre travail. Après avoir définit les règles, nous avons opté pour les automates finis pour généré un corpus de mot juste pour notre comparaison.

Ce projet très enrichissant, nous a donné l'occasion de nous plonger dans un problème concret, où nous avons pu mettre à profit les connaissances acquises durant notre cursus, et avons acquis de nouvelles compétences, plus particulièrement dans le domaine des traitements de langage. Le début a été très difficile, ce qui est principalement dû à un manque de documentation, et aussi au fait que le domaine touché n'a pas trop évolué durant les 10 dernières années. Cependant, ces difficultés nous ont permis d'acquérir un esprit de synthèse, ce qui est loin d'être négligeable. Le projet nous a de plus permis d'approfondir nos

connaissances sur la programmation en JAVA, implémenter les automates à état finis, ainsi que l'acquisition de connaissances sur la programmation sous un environnement Android.

Néanmoins comme tout projet, le notre n'est pas exhaustif, il serait intéressant d'enrichir ce travail par :

- une application plus ergonome sous Android ;
- La gestion des fautes grammaticales ;
- L'intégration d'une unité de mesure pour faire sortir les mots les plus probables.

Bibliographie

Bibliographie

[1] : T.M. Tran, M. Trancart and D. Servent, « Littéracie, SMS et troubles spécifiques du langage », Institut de Linguistique Française 2008.

[2] : Laurent Bloch, « Initiation à la programmation avec Scheme », Editions TECHNIP 2009.

[3] : http://android-france.fr/android-cest-quoi/ (Consulté le 02 Septembre 2013).

[4] : T. Dutoit, « Introduction au traitement automatique de la parole », Faculté Polytechnique de Mons, 2000.

[5] : Cédrick Fairon,Jean-René Klein,Sébastien Paumier , « étude d'un corpus informatisé à partir de l'enquête », Presses université de Louvain, 2010.

[6] : Rémi Bove, « Étude de quelques problèmes de phonétisation dans un système de synthèse de la parole à partir de SMS », RÉCITAL, 2005.

[7] : VIENNEY, S., MELIAN, C., (2004), « La correction automatique du langage des nouvelles formes de communication écrite », BULAG N°29, 2004.

[8] : Yvon et Kobus, « Application d'un algorithme de traduction statistique à la normalisation de textos », Université de Montréal, 2008.

[9] : http://www.linternaute.com/dictionnaire/fr/definition/phoneme/ (consulté le 15/08/2013)

[10]: Richard Beaufort, Sophie Roekhaut, Louise-Amélie Cougnon, Cédrick Fairon, Une approche hybride traduction/correction pour la normalisation des SMS, TLAN 2010.

[11]: John E. Hopcroft and Jeffrey D. Ullman. « Introduction to Automata Theory, Languages and Computation. » Addison-Wesley, 1979.

[12]: Emmanuel Roche and Yves Schabes, editors. Finite-state language processing. MIT Press, 1997.

[13]: http://www.vogella.com/articles/JavaRegularExpressions/article.html (consulté le 12/09/2013).

[14] : http://fr.slideshare.net/Rouinihouss/introduction-aux-systmes-dexploitation-mobile (consulté le 15/09/2013)

[15] : http://news.idealo.fr/news/205636/definition-windows-phone.html (consulté le 15/09/2013)

[16] : http://www.maisondugsm.com/a/encyclopedie/definition/4/systeme_d_exploitation.html (consulté le 15/09/2013)

[17] : Fauvarque Florian, Réalisation d'une application de visualisation de données sur Android, INRIA (2010)

[18] : PERELSTEIN, Laszlo, « Android équipe quatre téléphones sur cinq vendus dans le monde », La Tribune, du 9 août 2013 « En ligne »,http://www.latribune.fr/technos-

medias/electronique/20130809trib000779898/android-equipe-quatre-telephones-sur-cinq-vendus-dans-le-monde.html.(Consulté le 15/09/2013).

[19] : Android- Help « En ligne ». http://www.android-help.fr/article/les-versions-Android (Consulté le 15/09/2013).

[20] : Stéphane Huot, « Introduction à Eclipse », iUT Orsay, 2008.

[21] : Mme Ghomari Leïla (Née Zemmouchi), Alignement d'ontologies de domaine : Etude comparative syntaxique versus sémantique, cas d'application COMA++ VS OWL-CTXMatch, Mémoire pour l'obtention du diplôme de Magister en Informatique, à l'Ecole Supérieur d'Informatique (E.S.I) Alger, Algérie (2008-2009)