MASTER HANDI

Domaine : Sciences Technologie, Santé (STS)

Mention : Ingénierie et Cognition

Spécialité : Technologie et Handicap

Mémoire de stage M2

Développement de nouvelles fonctionnalités d'un agent virtuel d'écriture pour un aveugle

Joël Stéphane RAVELOJAONA

Directeur de stage : Gérard UZAN

Lieu du stage : Laboratoire THIM – Université Saint Denis

Coordonnateur : J. LOPEZ KRAHE

Paris, septembre 2012

A. AVANT PROPOS

A ce jour, l'accès à l'écrit (lecture et écriture) pour les personnes déficientes visuelles reste encore un grand défi. Et pourtant, 1,7 million de personnes souffrent d'une déficience visuelle dont : 560 000 malvoyants légers[1]; 932 000 malvoyants moyens ; 207 000 malvoyants profonds, dont environ 61 000 aveugles complets[2].

Moins de 1 % des déficients visuels (8 000 personnes environ) se servent d'interfaces d'ordinateurs (reconnaissance vocale, écran tactile, synthèse vocale).

15 % des aveugles ont appris le braille, 10 % l'utilisent pour la lecture et 10 % pour l'écriture également. L'apprentissage du braille est plus rare chez les malvoyants profonds (3 % environ) et les malvoyants moyens (1 % environ).

Pourtant, à ce jour, aucune solution n'a pu remplacer pleinement l'imprimé traditionnel et la lecture optique. De l'écriture braille à la synthèse vocale, les techniques mises au point répondent aux besoins spécifiques de catégories de lecteurs ou aux nécessités liées à des contextes donnés.

Cette grande diversité d'usages s'accompagne d'une complémentarité des outils. Un même lecteur déficient visuel pourra avoir recours à des adaptations différentes en fonction des circonstances, ou des ressources disponibles.

Tout matériel susceptible de pouvoir être exploité par l'utilisateur non voyant ou mal voyant et permettant d'effectuer les mêmes tâches (dactylographie, traitement de textes, traitement de données, tableur, gestion de données...) que les voyants et utilisant les mêmes logiciels semble le mieux indiqué pour faciliter l'intégration professionnelle de l'opérateur déficient visuel. Outre l'allégement du travail attendu, les outils informatiques contribuent grandement à améliorer la vitesse d'exécution des nombreuses tâches à effectuer. Ainsi, ce matériel permet à l'aveugle d'assurer des tâches jusque là inaccessibles pour lui, faute de moyens techniques et/ou technologiques ; et non faute de moyens intellectuels. Il est préférable d'avoir un logiciel qui soit fortement optimisé pour un aveugle mais permettant l'utilisation partagée avec un voyant. (biblio)

Le logiciel que nous allons étudier ci-après est connu sous le nom de FRANCOPHONEME ® Trio, crée par Gérard UZAN, dont la première version a vu le jour en 1990, puis avec le concours d'autre auteur depuis 1999, jusqu'à aujourd'hui. C'est un logiciel d'accompagnement à l'activité de l'utilisateur aveugle.

---

[1] Classification selon l'OMS : Déficients visuels légers de 3/10e à 1/10e

[2] Selon l'enquête HID (Handicaps-invalidités-dépendance) de l'Insee - Numéro 87-88 - décembre 1998.

Il s'agit d'un accompagnement en terme de mémoire et en terme d'action *« contextualisée* », c'est à dire une action proposée ou réalisée de façon contextuelle.
La pratique d'un traitement de texte nécessite un "investissement intellectuel".
Un traitement de texte est composé de quatre processus principaux:

- de saisir du texte,
- de le mettre en forme,
- de le sauvegarder,
- de l'imprimer éventuellement.

Lors de la saisie, on peut taper incessamment des textes sans se soucier de sa mise en forme. D'ailleurs, cela est conseillé lorsqu'on commence l'apprentissage d'un traitement de texte. Par exemple, sans faire de retour chariot à la fin de chaque ligne.
Les fautes de frappe peuvent être corrigées, des modifications peuvent être sur tout endroit à tout moment avant sa sortie à l'imprimante ou sa diffusion à d'autre personne.
Il nous arrive aussi de sélectionner une partie du texte, de la copier, couper ou supprimer, de la remplacer, la corriger, de la ré-remplacer, de l'insérer, de la réinsérer.
Après la saisie, il faut mettre en forme le document : création des titres, paragraphes, retours chariot, la tabulation. De modifier la typographie de certains mots ou lettres : mettre en gras, souligner, mettre en italique.
Une fois le texte saisi et mis en forme, il doit être sauvegardé, ou transféré vers d'autres applications.

L'exploitation d'un traitement de texte nécessite la mémorisation des nombreuses « fonctions » du logiciel utilisé. D'un point de vue ergonomique, chaque « fonction » est définie par :

- une commande (la ou les touches de clavier utilisées),
- une procédure (la ou les conditions de mise en œuvre),
- un résultat (la ou les modifications de l'état du système).

Toutes les fonctions sont dépendantes de la position sur l'écran. En premier lieu, il est primordial de connaître où, comment, quand sont distribuées, les informations affichées à l'écran.

Pour le voyant la position d'un objet se définit par rapport à la position des autres objets situés dans l'espace ou dans le plan et non par rapport à ses coordonnées. Or pour se représenter une configuration spatiale ou visio-spatiale, le non voyant a besoin non seulement de connaître la nature des objets qui la composent, mais aussi les coordonnées des objets les plus pertinents et leur position par rapport aux autres objets.

Pour le non voyant, il s'agit d'une double tâche : trouver les stratégies les plus économiques pour utiliser les possibilités de correction, de déplacement, de copie, de

recherche, de coupe, d'ajout, de tri, ... du traitement de texte en consultant au bon moment et au bon endroit de l'écran l'information à faire lire par la synthèse vocale.

L'aide vocale permet de se représenter ce qui se passe à l'écran et dans l'application en cours mais le repérage pour retourner, revenir, insérer, rechercher, lire partiellement, reprendre, demande parfois de nombreuses manipulations.

De plus, elle ralentit le rythme de travail d'un utilisateur non-voyant et rend parfois pénible l'attente de l'information recherchée.

D'autant plus qu'un texte en création est rarement pensé et écrit dans une séquence unique de la première à la dernière lettre.

S'appuyant sur une attention, un contrôle permanent et une capacité de pointage visuel, ces outils sont difficilement exploitables par les aveugles où en perdent toutes leurs pertinences en tant qu'outils d'aide à l'écriture.

Ce projet reprend l'idée du projet que Ferber a appelé « Kénétique » : « Qui consiste à concevoir et réaliser des univers ou des organisations d'agents artificiels capables d'agir, de collaborer à des tâches communes, de communiquer, de s'adapter, de se représenter l'environnement dans lequel ils évoluent et de planifier leurs actions, pour répondre à des objectifs définis » [Ferber 1995].

Dans le cadre du stage, l'objectif est :

- Développer une assistance de gestion(s) de pointeur(s) et de mémoire(s) propre(s).
- Faciliter la rédaction et/ou correction de texte en première rédaction ou en correction pour accroitre l'efficience et l'efficacité de l'écriture sur ordinateur par des aveugles.

Tables des matières

B. ETAT DE L'ART

B.1. INTERACTION HOMME MACHINE

L'interaction Homme-Machine ou interaction Personne Système se réfère à l'ensemble des phénomènes cognitifs, matériels, logiciels et sociaux mis en jeu dans l'accomplissement des tâches sur support. Tandis que l'interface désigne le dispositif physique, support de l'interaction. [Calvary 2002].

B.1.1 COLLABORATION HOMME MACHINE (CHM)

La réalisation et la diffusion d'interface souple, fiable et conviviale n'est plus une luxe, mais est devenue une exigence. Aujourd'hui, avec la possibilité d'utiliser d'autres modes de communication que le clavier / écran / souris ; par exemple la parole; il s'avère évident que les acquis des disciplines relativement étrangères à la culture informatique comme la linguistique ou la psychologie cognitive et sociale ou encore l'ergonomie doivent être intégrés dans les modèles informatiques pour réaliser une véritable communication homme machine. Ceci à pour but d'adapter cette collaboration aux besoins communicationnels et aux modes de raisonnement des utilisateurs.

Trois entités interactives interagissent dans cette collaboration : l'opérateur (utilisateur), la machine et l'environnement commun (application), et enfin l'objet (machine, ordinateur, matériel ou non) sur lesquels portent l'activité du système opérateur / machine. C'est cet échange d'informations entre l'opérateur et la machine, qui constitue la communication.

En ce sens, la CHM ne peut être réduite à un simple échange d'informations, elle doit assumer la coordination entre deux processus intelligents qui se déroulent l'un dans le cerveau de l'utilisateur, l'autre dans la machine, que ce soit pour la coordination des travaux collectifs aussi bien que pour la stimulation de tâches ou l'aide coopérative (assistance).

B.1.2 INTERFACE HOMME MACHINE (IHM)

Récente, la notion d'interface est issue de la physique, domaine qui permet d'appréhender son mode opératoire. Elle correspond au contact entre au moins deux objets de natures différentes. Elle y est modélisée comme « un sous-système doté d'une structure, réseau de processeurs, et assurant des activités spécifiques (de transfert et de codage en général) » [Le Moigne, 1994]. En informatique, les interfaces facilitent l'échange d'informations en mobilisant les avancées en matière d'ergonomie, de graphisme et de technologie.

Par quelles propriétés peut-on juger de la bonne qualité d'une interface? Par son esthétique, par sa commodité d'emploi, par la cohérence et les présentations, par l'adéquation des modes d'expression, la facilité d'usage, la rapidité d'apprentissage, les aides mnémoniques, le naturel ?

Une interface est dite de qualité, quand elle est plus « conviviale » ou affordante[3]. [GIBSON et al. 1977]

Par analogie, il faut bien reconnaître que la satisfaction est plus grande quand on a devant soi un bureau bien rangé, ou d'effectuer des commandes pour lesquelles il n'est pas nécessaire de retenir une syntaxe complexe et avoir aux moments opportuns des aides ou des mises en garde anticipatives. Il est de fait qu'il y a des applications dont on a envie de se servir et d'autres non.

B.1.3 SYSTEMES MULTI-AGENTS (SMA)

Les SMA, systèmes multi-agents sont utilisés pour modéliser des systèmes complexes. Dans un SMA, les agents interagissent selon des modèles d'interaction plus ou moins élaborés, normalisés, plus riches que dans des modèles classiques.

La réification des messages est l'une des principales richesses du SMA, c'est-à-dire, la transformation, transposition d'une abstraction (message système ou concept), en objet ou message concret (perçu par l'utilisateur).

Cet enrichissement rend plus dynamique et flexible, l'organisation du système, tout en préservant l'autonomie des agents, de l'utilisateur et de leur opérabilité. [Luck et al. 2003] [Jennings et al. 1998].

Les SMA sont difficiles à mettre en œuvre, car il faut faire appel à d'autres disciplines, essentiellement lorsque les SMA sont du type *Agent Communication Language* (ACL), auxquels pourra faire référence une sémantique de la communication, voire de raisonner sur la base de la sémantique associée, selon des modèles cognitifs plus ou moins complexes.

B.2. L'INFORMATION PARLEE CHEZ L'AVEUGLE

B.2.1 MEMOIRE DE TRAVAIL

La mémoire de travail ou mémoire à court terme est la capacité de stockage de l'information utile, verbale ou visuo-spatiale pendant un temps relativement court (empan). Elle est impliquée dans le raisonnement, la compréhension et les apprentissages. Elle est sujette à l'erreur lorsque la

[3] En ergonomie le terme « affordant » se réfère à la « capacité d'un objet à suggérer sa propre utilisation », par exemple, sans qu'il ne soit nécessaire de lire un mode d'emploi.

quantité d'information et la demande cognitive sont élevées (travailler dans le bruit). Elle est aussi dépendante de l'âge.

B.2.2 GESTION MENTALE DE LA LECTURE.

Il est important de suivre de près la sensibilité du lecteur aux sons et aux découpages qu'il fait des mots qu'il découvre. Le lecteur doit laisser le temps aux mots pour qu'ils prennent sens et que le texte existe dans sa tête au fur et à mesure d'une lecture ponctuée et lente. La compréhension se construit et peut exiger des relectures des phrases ou des passages entiers.

B.2.3 LECTURE AUDITIVE

Les aveugles, pour pallier à l'absence de vue, sont contraints d'utiliser des stratégies et des processus vicariants. Bon nombre d'entre eux lisent de manière auditive, que ce soit à partir d'enregistrement vocal, ou de plus en plus à l'aide d'ordinateurs équipés de systèmes de synthèse vocale. [RICHAUME, 1990]

B.2.4 IMPORTANCE DE LA PROSODIE

Il est généralement admis que la musique de la voix, la prosodie, facilite le traitement cognitif des messages oraux. En effet, de nombreux auteurs soulignent le rôle fondamental que joue la prosodie dans le traitement cognitif de la parole. La prosodie informe l'auditeur, par exemple, sur la segmentation et la hiérarchie syntaxique. Elle comporte un caractère prédictif qui permet à l'auditeur d'anticiper sur la suite de la phrase. Elle décompose la phrase en unités mnésiques et allège la charge cognitive. Il est légitime de penser que la prosodie agit de manière prépondérante sur les performances du lecteur, dans la lecture auditive. [CALLAMAND, 1987]

C. PRESENTATION DU LOGICIEL

Le logiciel est né sous hypercard utilisant le langage de programmation hyperTalk (cf. : Annexe 1 :) sous MacOs. Durant plusieurs années, les développeurs ont privilégié l'utilisation des bibliothèques propres à HyperCard. Or Hypercard ne verra plus le jour sur les versions Mac OS X. Une mise à jour importante a été décidée sur le logiciel en le transférant sous l'environnement de « Rev Revolution », utilisant le langage « script ».

Vocalisation et synthèse vocale

Le logiciel utilise comme sortie : l'affichage visuel à l'écran (wysiwyg[4]), une sortie vocale, une sortie sous forme de fichier, et éventuellement, une impression.

C.1. NOTION DE LIVRE ET DE PAGE

Extension : .fpe

Par analogie, un livre est un classeur Excel ou a l'apparence d'un site web; composé de une ou plusieurs page(s) (comme une feuille Excel)

Il faut différencier la page imprimée (feuillet) de la notion de page du logiciel, qui ressemble à une page web. Quand on remplit la page, on ne passe pas automatiquement à la page suivante. Par contre, le logiciel pouvait coïncider la page imprimée avec la page à l'écran.

En plus, le dispositif bénéficie de la notion de l' hyper lien : c'est-à-dire, à n'importe quelle page d'un livre, on peut directement faire ouvrir n'importe quelle autre page d'un autre livre.

Une page peut avoir soit le statut d'un mail, soit le statut d'un texte tout court. Ce qui a pour avantage que les mails peuvent être intégrés dans le document.

C.2. NOTION DE POINTEUR

Il existe habituellement trois pointeurs principaux, le pointeur d'insertion, le pointeur souris et le pointeur de lecture. Mais dans le logiciel, il existe en plus d'autrse types de pointeur, le pointeur de recherche, le pointeur de marquage, et enfin le pointeur des précédents : précédent d'insertion, précédente recherche, précédente marquée. C'est ce jeu de pointeur qui va faire une avancée

[4] Wysiwyg : « What you see is what you get » est une interface utilisateur qui permet de composer visuellement le résultat voulu, typiquement pour un logiciel de mise en page, un traitement de texte ou d'image. C'est une interface «intuitive» : l'utilisateur voit directement à l'écran à quoi ressemblera le résultat final.

dans la façon de travailler. Le pointeur souris n'est pas utilisé par l'aveugle, mais est totalement disponible pour l'accompagnement voyant.

C.2.1.1 MEMOIRE DE POINTEUR D'ECRITURE.

La mémoire de pointeur du scribe se fait au niveau de la page. C'est-à-dire, chaque page comporte sa propre mémoire d'écriture. A chaque fois qu'on revient sur une page, le pointeur se met à l'endroit au moment où on a quitté la page.

L'agent scribe a la mémoire de pointeur d'insertion par page, c'est celui qui suit ce qu'on fait lors d'une écriture dans un texte. Il dispose en plus une mémoire de pointeur temporaire. C'est un pointeur qui se met en mémoire automatiquement, lorsqu'on modifie quelque part (dans un livre ou dans une page) un texte, il va mettre en mémoire, le pointeur précédent d'insertion dans la page courante.

On peut citer aussi le pointeur d'insertion, le pointeur de synchronisation ; le pointeur de marquage et de reprise (permettant à l'utilisateur de marquer un point d'intérêt) et le pointeur d'insertion avant synchronisation avec le lecteur (c'est la position du pointeur avant qu'on synchronise un pointeur avec le lecteur). Cela permet par exemple lorsqu'on rédige à un endroit donné de lire le texte, de faire une correction, et lorsque cette correction est faite, de revenir immédiatement au pointeur sans devoir naviguer avec les flèches de direction. (Le voyant fait ce processus très rapidement avec ses yeux et la souris, car la souris est un système de pointage direct.)

C.2.1.2 REMARQUES SUR LES POINTEURS

Le logiciel a une particularité, c'est qu'il considère l'affichage visuel, comme une visualisation de ce que fait le logiciel pour la personne déficiente visuelle, mais la visualisation n'est pas là pour être vocalisée.

On remarque que les dialogues oraux des boites de dialogue sont différents du contenu textuel de ses boites. La partie orale doit compléter la partie écrite et réciproquement. Et la partie orale doit être plus brève, mais d'autant plus brève que l'utilisateur appelle la commande.

Quand on est en présence des pointeurs qui sont mémorisés, il faut que ces pointeurs soient constamment actualisés, lorsqu'on fait des actions sur le document. Donc, il y a tout un travail de réactualisation des pointeurs, et c'est ce travail de réactualisation qui constitue aussi une partie du programme. Cf. (D.3.9, page 21)

Le pointeur de lecture tient compte du contenu lexico-syntaxique du texte et non pas seulement en terme de position dans une chaine de caractère à l'écran.

.

C.3. NOTION D'AGENT

Le logiciel possède trois agents : le scribe, le secrétaire et le lecteur

Par défaut, l'agent scribe utilisant la voix de « Junior », le secrétaire : celle de « Ralph » et enfin le lecteur : celle de « Fred », mais l'utilisateur peut choisir une voix pour chacun des agents (analogie entre un acteur prenant un rôle dans une pièce). Chaque agent a sa propre fonction et rôle dans l'interaction. Dans quelques cas particuliers, on peut jouer sur la variation des paramètres de la voix, pour mettre en évidence un changement d'état ou lors d'un changement de la nature d'un item (passage d'un dossier à un document). Comme dans le cas d'une épellation, c'est le lecteur qui lit le mot ou la phrase à épeler, et le secrétaire épelle chaque lettre du mot.

La mémoire du pointeur évoquée plus haut est une partie de la mémoire des agents, qui comporte également des contenus, des règles, des index, etc.

C.3.1 LE SCRIBE.

C.3.1.1 L'ECRITURE : L'ECHO « LOGIQUE »

Lors d'un écho : c'est à dire, la restitution vocale de ce qu'on vient juste de taper au clavier, l'utilisateur ne passe pas par raccourci clavier du caractère, en mot et en phrase, mais le logiciel bascule automatiquement en fonction de ce que fait l'utilisateur.

Par exemple : quand l'utilisateur tape il se met automatiquement en mode mot par mot. Il est un accompagnement à l'écho mental, cognitif de ce que la personne est en train d'écrire. Donc, il est systématiquement mot par mot, mais quand on revient en arrière, pour corriger un mot, il devient un écho lettre par lettre sur le mot.

Puis dès qu'on repasse au mot suivant, il redevient automatiquement un écho mot par mot.

Quand on arrive à la fin d'une ligne, ou paragraphe, et qu'on va au paragraphe suivant, il devient un écho de la fin de paragraphe, c'est-à-dire, il prend les derniers[5] mots du paragraphe.

L'écho d'une zone ne se déclenche que s'il y a modification.

Le scribe est automatiquement bilingue.

C.3.1.2 AUTRES FONCTIONNALITES DU SCRIBE

Avec les flèches, il permet d'écouter le début d'une phrase jusqu'au point d'insertion, et du point d'insertion jusqu'à la fin d'une phrase. Nous parlons bien ici de la « phrase », la ligne visuelle

[5] Deux ou plusieurs mots (pas plus d'une phrase)

n'a pour le logiciel aucune importance, sauf qu'il avait[6] la possibilité lorsqu'on met le point d'insertion quelque part, et de pouvoir dire à combien de centimètre se trouve le pointeur des quatre bords de la page telle qu'elle serait imprimée.

Il permet aussi d'étendre cette fonctionnalité au paragraphe (du début du paragraphe jusqu'au pointeur et du pointeur jusqu'à la fin du paragraphe).

C.3.1.3 LE SCRIBE ET LA MESSAGERIE :

Lorsque le statut de la page devient un message, le scribe va avoir un comportement particulier en fonction du contenu des rubriques. Par exemple, si dans un champ de liste d'adresses mail, s'il n'y qu'une seule adresse, il va l'énoncer en entier, s'il y a deux, il va énoncer les noms…, en présence de plusieurs adresses, il va énoncer le prénom/nom du premier et « n-1 » autres. Si on appuie sur ***« autre »***, il va énoncer la liste des autres items. Et si on continue toujours d'appuyer sur celui qui est en train d'être dit, il va énoncer le nom de domaine. Et enfin, si on appuie encore, il va l'épeler.

C.3.1.4 LE SCRIBE ET LA SELECTION :

Il a un comportement spécifique pour la sélection : on marque le point du début de la sélection, on se déplace n' importe où dans le texte et on marque la fin de la sélection. Il est à noté que le Scribe est compatible avec le pointage de la souris. Il est totalement compatible si un utilisateur voyant utilise en partage, ce qui est à l'écran avec un non voyant.

C.3.2 LE LECTEUR

C.3.2.1 FONCTION

Il a pour fonction de lire avec le plus de tolérance et de pertinence un texte qui lui est soumis. Avec tolérance parce qu'il n'écorche pas les mots, même lorsque par exemple, en majuscule, ils ont perdu l'accent, ou qu'ils sont découpés d'une façon bizarre ; ou lors d'une présence d'élément de césure. Entre autre, les pluriels en « ent » et les adverbes en « ent », ou bien le substantif « sens » et le verbe sentir conjugué au deux premières personnes du présent de l'indicatif « sens ». Il doit aussi tenir compte des imperfections des logiciels d'OCR, par exemple le mot « 1'image » sera lu comme « l'image ».

[6] Fonction non remise à jour sur le nouveau logiciel.

Il est très coordonné au secrétaire dans les tâches de recherches.

Il possède des mémoires mobiles et des mémoires par page et mobile à l'intérieur d'une page. Il peut donc lire un texte depuis le début, ou à partir de l'endroit où on a interrompu la précédente lecture, ou le début d'une tête de chapitre et faire des aller retour entre un contenu courant d'un chapitre et sa tête de chapitre (ou celle du chapitre suivant).

C.3.2.2 NAVIGATION RAPIDE DU LECTEUR

Il détecte automatiquement et sans balise, à partir de l'analyse de contenu la présence de tête de chapitre. Il permet ainsi de créer virtuellement une table des matières et de naviguer dans cette table des matières. Il permet aussi de naviguer directement avec un « effet de zoom » dans sa navigation. C'est-à-dire que si on part d'une phrase dans un paragraphe donné, il va de phrase en phrase à l'intérieur du paragraphe cerné, puis ira de paragraphe en paragraphe. Dans ce cas, il lit les débuts de phrase de chaque paragraphe dans une longueur variable qui dépend de leur contenu, pour garder la compréhension du texte lu. Ceci étant réalisé en exploitant les mots connecteurs : proposition, conjonction de coordination et de subordination, pronoms relatifs, articles, auxiliaires, etc.

La notion de chapitre intègre des aspects absolus (taille, longueur, etc.), nombre de mots, nature des mots, mais également leurs composantes relatives par rapport aux paragraphes qui précèdent ou suivent.

On peut donc remonter un chapitre, et redescendre à son texte de départ. Et entre temps, si on a modifié le contenu du texte entre la tête de chapitre et la position actuelle, le logiciel anticipe cette modification et on retrouve facilement la position de fin de lecture lors de la reprise de la lecture.

C.3.2.3 LE LECTEUR : UN AGENT BILINGUE

Le lecteur est automatiquement bilingue : anglais et français. Dans un texte comportant des mots en français et en anglais, il a la possibilité de détecter la langue. Il va adapter son mode de lecture par rapport à la langue qu'il a détectée. Il a aussi une particularité d'offrir la possibilité, lors du choix de la langue, de lui forcer à lire un texte uniquement en anglais ou en français. Il dispose aussi d'une langue prioritaire, où en cas d'ambigüité ou d'indécision, par défaut, il choisit la langue maternelle de l'utilisateur.

C.3.2.4 INTERRUPTION D'UNE LECTURE

Lorsqu'on interrompt le lecteur, Il s'arrête immédiatement tant qu'on n'est pas à la fin de la phrase, mais en fin de phrase il considère l'arrêt et continue sa lecture jusqu'à la fin de la phrase

dite. Permettant à l'utilisateur d'être concentré sur le contenu et non sur le point d'arrêt tout en arrêtant bien en fin de phrase, lui laissant l'écoute complète de la phrase et non pas plus loin.

Enfin l'une des particularités importantes du lecteur est qu'il navigue aussi bien dans le livre, et dans la presse papier. En mode développeur, c'est le lecteur qui renvoie le message d'erreur.

C.3.3 LE SECRETAIRE

Il assiste tout ce qui est gestion des dossiers et de fichiers, il assure toutes les fonctions d'édition : couper, copier, coller, insérer, remplacer. Il assiste toutes les commandes nécessitant le déclenchement d'actions au niveau de l'ordinateur, du système, ou du pilotage d'une autre application.

Il possède aussi toutes les fonctions de « recherche » en coordination avec le lecteur. Ces fonctions de recherches sont dans une logique de « zooming » ou « logique à zoom ». la recherche s'effectue en premier temps dans ce qui est affiché (la page courante), puis, dans le livre courant, ensuite dans les livres ouverts, et enfin dans tous les livres ou documents.

Le secrétaire distingue six catégories d'objet : la page courante, le livre courant, les livres ouverts, les documents et les fichiers ouverts, les documents disponibles dans le volume, les fichiers disponibles via internet.

C.3.3.1 L'AUTO – PROGRAMMATION

Il dispose d'une fonction d'auto-programmation, permettant à l'utilisateur de se mettre en mode développeur et d'avoir un accès direct à l'ensemble des codes du programme. Il dispose aussi, de fonctions en cas d'erreur, de revenir aux codes précédents, pour chacun des objets.

C.3.3.2 MODE D'INTERACTION DU SECRETAIRE :

- **Le barillet « QCM »** :

Comme les autres fonctions, le QCM génère des plages d'interactions, qui sont pour chacune d'elles, un espace temporel dans lequel un monde est possible. Du coup, il faut que le monde soit ordonné, de telle façon que le monde le plus pertinent soit présenté en premier. Ici, le mot « monde » se rapporte à : « monde d'action », « monde de données », « monde d'objet à l'écran », etc. et surtout un monde de « tâches » à accomplir. Et essentiellement, le QCM propose des actions à réaliser, et qu'il va accomplir.

C'est une fonction générique, on peut l'appeler n'importe où dans le programme.

Sa particularité, c'est qu'elle n'a pas de raccourci clavier. Certes, c'est un menu contextuel, mais en oral, la notion de menu contextuel n'est pas la même, c'est qu'on est temporel. Alors qu'en visuel, les yeux peuvent aller rapidement à un menu visuel, il faut toujours que la première proposition soit la bonne. Donc, il faut être le plus pertinent possible. A l'oral, il y a deux règles à respecter, le premier : **que le nombre de propositions** ne soit pas trop élevé (inferieur ou égale à sept , et de préférence inférieur ou égale à cinq), le deuxième c'est que **l'ordre de présentation** soit toujours le plus pertinent[7].

L'autre particularité, c'est que la touche qui sert à appeler le secrétaire, en appui maintenu, choisit la première proposition.

Le secrétaire prend des précautions particulières concernant toutes les tâches destructives, il fait objet d'une demande de confirmation voire, une reconfirmation.

C.3.3.3 LANGUE DU SECRETAIRE

Le secrétaire est bilingue, à l'écrit comme à l'orale il adopte la langue maternelle de l'utilisateur, ici limité entre le français et l'anglais, mais la structure du programme permet facilement l'ajout rapide d'autre langues.

C.4. SYNTHETISEUR VOCAL

Il lit directement des textes en anglais sans être obligé de passer par la lecture phonétique. Il est automatiquement bilingue (français, anglais),et est sur un modèle neuronale.

Fonctionnement :

Lors de la lecture d'un bloc de texte, ce bloc va être découpé, en phrase ou en une unité minimale. Après, on effectue un test de langue sur ce bloc. En parallèle, on transforme les lettres avec accent en leur correspondance (symbole non accentué).

(Par exemple « é » va devenir « ee », ainsi , « références » deviendra « reeference ».

Si la langue est « anglaise » on va faire lire directement la langue par le synthétiseur, et dans le cas du français, le texte est soumis à une transformation phonétique, le synthétiseur lira alors phonétiquement les textes.

[7] En visuel, l'ordre n'a pas trop d'importance parce que l'œil pointe directement là où l'intérêt de l'utilisateur l'emporte. Et contrairement à l'oral, il faut que la structure soit toujours le même. Et que la liste ne bouge pas. Éventuellement, on grise ce qui n'est pas utilisable, alors qu'en auditif, on perd du temps.

C.5. LES PRINCIPALES COMMANDES DU LOGICIEL

C.5.1 TOUCHE « COMMANDE », CMD, OU POMME

La touche « cmd » permet de valider le choix en cours d'énonciation. Dans le cas d'un « QCM », le maintien de cette touche force le choix du premier ITEM. En mode lecture, l'appui long force le lecteur à lire un texte depuis le début.
Associée avec d'autres touches du clavier, elle assure les principaux raccourcis clavier du logiciel. Sur un PC, cette touche est remplacée par la touche CNTRL.

C.5.2 TOUCHE « MAJUSCULE », MAJ, OU SHIFT

Elle permet d'annuler un processus en cours. Par exemple l'interruption d'une lecture en cours, l'énonciation d'un choix, une recherche en cours.

C.5.3 LES TOUCHES DIRECTIONNELLES

Elles permettent la navigation dans un texte.
L'appui sur la touche pointant vers la droite ou gauche permet de naviguer de caractère en caractère. Combinée avec la touche ALT, la navigation se fait du pointeur en cours jusqu'au début ou fin d'une phrase.
Tandis que celle pointant vers le haut ou bas assure la navigation de mot en mot. Quand le pointeur se trouve en à l'intérieur d'un mot, l'appui sur ces touches permet de naviguer du pointeur en cours jusqu'au début ou fin du mot. Associée avec la touche ALT, la navigation se fait du pointeur en cours jusqu'au début du paragraphe ou vice-versa.

C.5.4 TOUCHE « CONTROLE », CTRL

Utilisé sous windows, la touche contrôle devient la touche commande.

D. DEROULEMENT DU STAGE

D.1. SUPPORT ET LANGAGE DE TRAVAIL.

Le matériel de développement est un MacBook Air doté d'un processeur Intel Core 2 Duo, fonctionnant sous Mac OS X 10.6.8 (Mac OSX Snow Leopard). Deux autres machines ont été utilisées pour tester le programme, dont un MacMini Intel (sous OSX Snow Leopard) et un PC sous Windows 7.

Les principaux logiciels de développement sont :

- Rev-Studio version 4.0.0, utilisant le langage « script » (Live code) ;
- L'éditeur AppleScript, utilisant le langage « AppleScript »
- Le « Terminal » du mac utilisant des lignes de commande UNIX.

D.2. REMISE EN ETAT DU LOGICIEL.

Lors de la prise en main, le logiciel comporte des bugs informatiques importants, qui ne me permettent même pas d'essayer ses fonctionnalités. Outre les nécessités de réécriture liées à quelques changements de syntaxe entre hyperCard et Revolution, les modifications des niveaux de sécurité du MacOs, les changements d'écriture (structure) des chemins d'accès des fichiers, le portage a été effectuée selon différentes pratiques entraînant à la fois une remise en fonctionnement de certaine fonctionnalité et une perte qualitative de ces mêmes fonctionnalités :

- La « *commentarisation* » de l'ancien code et leur réécriture systématique de certains intervenants, dont la compréhension insuffisamment fine de leur fonctionnalité ont produit avec les meilleures intentions des appauvrissements soit au niveau fonctionnel, soit dans la logique ou comportement des agents, soit dans la multiplication inutile des raccourcis claviers. (exemple : cf Annexe 4 : command arrowKey avant correction., page 27 et Annexe 5 : page 39)
- Le développement parfois inachevé ou réaliser dans la précipitation ou encore la transmission de l'avant dernière version plutôt que la dernière ont généré parfois des bugs ou des scories de disfonctionnement parfois difficile à détecter : (Exemple : test de fonctionnement d'échappement laissé dans le code entraînant des dysfonctionnements importants.)

Des analyses approfondies, suivies par processus d'identification d'erreurs (d'amont en aval) ont été mis en place en vue d'apporter une correction.

D’après la recommandation de «Rev Revolution », il faut déclarer au préalable les variables locales. Jusqu’ici, seules les variables globales étant déclarées. Ceci a pour objectif, d’assurer la réinitialisation de ces variables lors du lancement du logiciel

La difficulté repose sur le nombre relativement nombreux des variables locales. Il faut porter une attention particulière sur les fautes d’orthographes concernant les variables. Effectivement, une variable globale mal écrite pourrait être interprétée par le logiciel comme une variable locale, et vice-versa.

Il est à noter que le nombre de commandes ou fonctions du logiciel avoisine les cinq cents. Et chaque fonction utilise en moyenne une demi-douzaine de variables.

D.3. FONCTIONS OU COMMANDES CORRIGEES OU REFAITES COMPLETEMENT

D.3.1 LA FONCTION « INKEY » OU « TOUCHE PRESSEE »

Cette fonction permet de retourner le caractère d’une touche appuyée durant le déroulement du programme. Les caractères interceptés seront après usage tapés au point d’insertion.

On distingue les codes touches, les codes caractères et les codes ASCII. Par souci de compatibilité entre les systèmes d’exploitation et d’interopérabilités entre les différents types de matériel, cette fonction a été refaite complètement.

Une fonction interne à « Rev Revolution » renvoie le code de la touche appuyé, par exemple, la « touche a (minuscule) » envoie le code « 113 ». Et on exploite ce code pour l’associer à un caractère (dans notre cas, c’est le caractère « a minuscule ».

Or, on distingue deux types de clavier, les claviers dits « AZERTY » et les claviers « QWERTY ». Donc, pour un clavier « AZERTY », on associe à « 113 » le caractère « a », tandis que pour un clavier « QWERTY », on lui associe le caractère « q ».

La deuxième difficulté est que la disposition des touches n’est pas la même sur un clavier utilisé dans les systèmes d’exploitation Windows et MacOs. Prenons le cas de la touche tiret «-» d’un MacOs dont l’équivalent sous Windows est la touche « = » pour un clavier « AZERTY ». D’où, l’importance de traiter les quatre cas possibles, le clavier « AZERTY du MacOS, les claviers. Son équivalent de Windows, le clavier « QWERTY » de MacOs et enfin son équivalent de Windows. Cf. (Figure 1, page 49) et (Figure 2, page 49)

D.3.2 LA COMMANDE « VOCALISER BILINGUE »

Cette commande a comme entrée un bloc de texte à lire. Ce bloc va être découpé en phrase (Cf. : *D.3.5 La fonction de découpage, page 21*), puis en unité. Une transformation de caractère en symbole est effectuée sur cette unité, parallèlement à cela, on lance une commande qui permet d'identifier la langue. Si la langue est « anglaise », on va faire lire le texte directement par le synthétiseur vocal, et dans le cas du français, le texte est soumis à une transformation phonétique. Le synthétiseur lira phonétiquement le texte. Une erreur intempestive a été décelée lors de l'interruption d'une lecture, en effet, quand on décide d'interrompre une lecture, soit le synthétiseur continue de lire tout le texte, d'où une perte de temps, soit qu'il s'arrête tout de suite au milieu d'un mot, d'où la perte du sens du mot, voire de la phrase. Le règle est la suivant, en présence d'une virgule, il s'arrête tout de suite à la fin du mot, en présence d'autre ponctuation, il continue de lire jusqu'à la ponctuation. (*cf Figure 3, Logigramme simplifiée de la commande « Vocaliser » page 50*).

D.3.3 LA FONCTION «TEXTE EN TEXTE PHONETIQUE »

Cette fonction transforme le texte en texte phonétique. Elle fait appel à une autre fonction qui renvoie l'ensemble des règles du caractère pointé, puis sélectionne la règle qui sera appliquée et éventuellement quelques caractères qui lui succèdent et retourne également, de combien le pointeur doit être déplacé.

On récupère le code phonétique du texte, puis on le renvoie vers la commande « Vocaliser bilingue » en y associant les valeurs de hauteur, débit, volume, emphase. (cf Annexe 6 : La fonction «Texte en texte phonétique » page 48)

Cette fonction a été corrigée durant le stage, en effet, au lancement, le logiciel plante quand la règle envoie vide, par exemple, certaines variables muettes. Si par malheur au lancement, les règles qui sont applicables sont des règles dont la chaine phonétique est vide, il plante. Une gestion de l'erreur a été mise en place.

D.3.4 LA FONCTION TEST DE LANGUE

Elle permet de détecter automatiquement la langue dans un texte. Elle consiste à mettre un score pour chaque langue sur la base des formes idiomatiques inexistantes ou quasi-inexistantes dans l'autre langue.

Cette fonction a été mise à jour en enlevant, corrigeant ou en rajoutant des éléments dans la base idiomatique. Cette fonction a été dégradée par un mode de réseau sémantique à faible utilité. C'est une structure par réseau sémantique de type « if » « or » « else » et que par la suite, nous l'avons modifié en mode « scoring » qui était à l'origine.

D.3.5 LA FONCTION DE DECOUPAGE

Cette fonction extrait du premier caractère d'un bloc de texte à lire, jusqu'à un élément de ponctuation qui délimite la première phrase ou le premier syntagme.
Elle a été complètement réécrite. Une nouvelle fonctionnalité a été apportée à cette fonction, notamment dans le cas où on est en présence d'un bloc de mot ne comportant pas un élément de ponctuation.

D.3.6 LA FONCTION « SYMBOLE EN LETTRE »

Elle permet de transformer un symbole en lettre ou de désaccentuer en correspondance avec la règle phonétique. Par exemple, le symbole « $ » serait transformé en « dollars ». Une correction majeure a été faite sur cette fonction, avec l'intégration d'autres symboles non traités auparavant. D'autant plus que la normalisation d' « UNICODE, FT8, » a durci les codes ASCII (0 à 255).

D.3.7 LA COMMANDE « MONTRE »

C'est une commande qui permet de récupérer l'heure et la date du système. En effet, le logiciel annonce automatiquement l'heure. On a aussi la possibilité d'appeler cette commande à partir d'u raccourci clavier. Cette commande a été corrigée et réactivée dans la nouvelle version du logiciel.

D.3.8 LES COMMANDES ET FONCTIONS DE TRAITEMENT DE CHEMIN, FICHIER, DOSSIER

La plupart des fonctions relatives à la gestion d'un fichier ou dossier ont été réécrite. Entre autre, la récupération d'un chemin d'un dossier ou d'un fichier, la récupération d'un nom de fichier, la mise à jour du catalogue du logiciel.

D.3.9 LES COMMANDES RELATIVES A LA « MARQUE PAGE MANUELLE »

Le logiciel permet de marquer manuellement une page, indépendamment du livre. La fonction vérifie si la page est marquée ou non. Par le biais de la fonction « QCM » il propose à l'utilisateur de marquer ou de démarquer, de reprendre le précédent marquage ou le marquage suivant, selon la position dans la page. Et enfin de retourner dans la page où on s'est retrouvé avant le déplacement vers une page marquée. S'ensuive la gestion de l'actualisation automatique de la longueur du champ. En effet, en modifiant le texte en amont de la marque page, (ajout ou suppression), la position du marquage se décalera automatiquement et se retrouve toujours sur le mot qu'il a marqué.

D.3.10 LES COMMANDES RELATIVES A LA REPRISE APRES LECTURE

Quand on rédige un texte, il nous arrive de lire un autre texte dans une autre page ou dans un autre livre. Cf. C.2.1.1 Mémoire de pointeur d'écriture. Page 11 . Cette fonction a été complètement réécrite, ainsi que les fonctions de réactualisation permanente de la position du pointeur

D.4. LES FONCTIONS NOUVELLEMENT CREEES.

D.4.1 LES COMMANDES LIEES A LA CORRECTION D'ORTHOGRAPHE

D.4.1.1 DISTANCE DE DAMERAU-LEVENSHTEIN »

La correction d'orthographe est basée sur l'algorithme de « Distance de Damerau-Levenshtein » [LEVEINSHTEIN 1964] et [DAMERAU 1964]. (cf : Annexe 2 : Fonction Damerau-Leveinshtein page 25). C'est une fonction qui consiste à calculer le nombre minimum d'opérations nécessaires pour transformer une chaîne de caractères en une autre, où une opération est définie comme l'insertion, la suppression ou la substitution d'un simple caractère, ou comme une transposition de deux caractères. Quand la fonction renvoie 0, les deux chaines de caractères sont identiques. Ainsi, cette opération compare un bloc de texte et une liste de mots dans un dictionnaire. Toute une batterie de processus est effectuée en amont de cette opération.

E. SYNTHÈSE ET CONCLUSION

Au cours de ce stage de Master, nous avons posé comme objectif de concevoir d'autre fonctionnalité de l'agent virtuel d'écriture pour un aveugle. Ce qui nécessite le rétablissement de toutes les autres fonctionnalités en amont et en aval du logiciel.

La première étape a été consacrée à la maintenance des codes, de rendre le logiciel opérationnel.

Il 'est difficile d'expliquer que la maintenance et le rétablissement du logiciel est un travail de première nécessité. Par analogie, dans une voiture, on peut avoir les plus géniales des fonctionnalités, mais quand le moteur est en panne, ces fonctionnalités ne servent à rien. Ainsi, on peut réaliser le plus beau noyau informatique du monde, l'algorithme le plus sophistiqué qui soit, si au bout du compte l'aveugle ne puisse pas s'en servir, ces fonctionnalités ne servent à rien. Et la première étape consiste à une reprise et accroissement de fonctionnalité, au plus proche de l'utilisation par un aveugle.

La seconde consiste à améliorer le comportement des trois agents, nécessitant la réécriture et ou la correction de plusieurs fonctions relatives au bon fonctionnement de ces agents et de récupérer certaines fonctionnalités (surtout graphique).

Du point de vue de la mise en œuvre, l'application est opérationnelle, Néanmoins, des corrections et optimisations sont encore à faire afin de la rendre plus robuste et plus efficace. Il y a aussi une réécriture de commentaires[8] pour la compréhension et la maintenance ultérieure.

En ce moment, le logiciel fonctionne uniquement sous « MacOs », il faut continuer les processu s d'interopérabilité (pour qu'il puisse communiquer avec les autres logiciels) et de compatibilité (pour qu'il puisse être utilisé sous d'autre système d'exploitation : Windows, linux, Android, Ios, ..). (Difficile à réaliser dans la durée du stage, mais facilement réalisable).

Il pourrait être intéressant d'établir une documentation logicielle, composée d'une documentation technique[9], architecture et conception (vue d'ensemble sur le logiciel, documentation du code, algorithmes, interfaces programmation, API) et de manuel pour l'utilisateur (prise en main, fonctionnement, liste des raccourcis…).

[8] Sauf dans la partie messagerie (existence des commentaires)

[9] Qui existait dans les années 90.

F. ANNEXE

Annexe 1 : Hypercard

HyperCard est un programme et un environnement de programmation simple développé par Apple qui ne fonctionne que sous Mac OS versions 9 et précédentes. Il ressemble un peu à un logiciel de bases de données, car il conserve des informations, mais à l'inverse des bases de données traditionnelles il est graphique, très flexible et trivialement facile à modifier quand on a compris la hiérarchie des piles. En plus, HyperCard inclut un langage de script, HyperTalk, puissant et facile à utiliser pour manipuler ces données.

Il fut commercialisé avec le Système 5 en 1987, et ne fut retiré de la vente qu'en mars 2004, bien que son développement ait été stoppé plusieurs années auparavant.

HyperCard est basé sur le concept de piles de cartes virtuelles.

Les cartes sont composées globalement de deux objets :

- le fond
- les calques qui utilisent les outils graphique (boutons, champs, images, sons, etc.) du fond et ajoutent les leurs. Plusieurs niveaux autorisent ou non des modifications devenant définitives, l'état historique étant sauvé comme nouvelle carte.

Les utilisateurs peuvent construire des bases de données en ouvrant l'éditeur de fond et des éléments de dessin pour contenir les diverses pièces de données. Par exemple, un carnet d'adresse peut être construit simplement en ajoutant quelques champs de textes pour contenir le nom et l'adresse. Une fois complété l'utilisateur ajoute simplement une nouvelle carte (en tapant commande-n) et tape dans les champs. Le fond peut être changé à n'importe quel moment, ce que les systèmes traditionnels gèrent mal. Les opérations de base comme la recherche, l'ajout ou la suppression sont incluses dans l'environnement.

Un script dans le langage HyperTalk permet de modifier ou étendre le système dans une certaine mesure. Cependant, plus la taille de pile augmente, moins celle-ci est réactive. On peut écrire une commande en anglais simplifié, écrire les chiffres sous forme ordinale ou numérique, et tous les objets utilisés peuvent posséder un nom.

Ajouter des scripts était facile : on « option-cliquait » n'importe quel élément dans la pile et un éditeur apparaissait. Le script pouvait être édité, sauvé et utilisé immédiatement. HyperCard 2.0 ajouta un débogueur puissant et simple d'emploi.

On pouvait augmenter la puissance d'HyperCard par des modules de commandes externes, ou XCMDs (petits fragments de codes en paquets dans une fourchette de ressources). Pendant la pointe de popularité de la fin des années 1980, des vendeurs proposèrent des milliers de XCMDs : compilateurs, systèmes graphiques, accès à des bases de données, connexion Internet,

HyperCard peut aussi servir à organiser des présentations hypertextes interactives très artistiques. Avant la disponibilité de PowerPoint, il fut utilisé comme logiciel de présentation. []

Annexe 2 : **Fonction Damerau-Leveinshtein**

```
function Levenshtein motCorrect , motAverifier
--Joel Mai 2012
--Il s'agit de la distance de Damerau-Levenshtein, car on a incorporé la transopsition.
--Celle de Leveinshtein étant limitée aux 3 premières opérations.
/*
On calcule le nombre minimum d'opérations nécessaires pour transformer une chaîne de
caractères en une autre,
où une opération est définie comme l'insertion,
la suppression ou la substitution d'un simple caractère,
ou comme une transposition de deux caractères.
*/
local i,j,mot1,mot2,long1,long2,distance,cout
global distanceLevenshtein
put motCorrect into mot1
put motAverifie into mot2
put length(mot1) into long1
put length(mot2) into long2
repeat with i=1 to long1
--On crée une matrice de i ligne et de j colonne
put i into distance[i,0]
end repeat
repeat with j=1 to long2
put j into distance[0,j]
end repeat
/*
combine distance using return comma
*/
repeat with i=1 to long1
repeat with j=1 to long2
if char i of mot1=char j of mot2 then
put 0 into cout
else
put 1 into cout
end if
--substitution d'un caractère de Mot1 en un caractère de Mot2
--ajout dans Mot1 d'un caractère de Mot2
--suppression d'un caractère de Mot1
put min(distance[i-1,j]+1,distance[i,j-1]+1,distance[i-1,j-1]+cout) into distance[i,j]
if ((i>1) and (j>1) and (char i of mot1=char j-1 of mot2) and (char i-1of mot1=char j of mot2)) then
--transposition entre 2 caractères
put min(distance[i,j],distance[i-2,j-2]+cout) into distance[i,j]
end if
end repeat
end repeat
-- on récupère la cellule issue du i-ème ligne et du j-ème colonne
put distance[long1,long2] into distanceLevenshtein
return distanceLevenshtein
end Levenshtein
```

Annexe 3 : **Commande Epeler on epeler phraseAepeler**

```
local nombreDeMot,compteurLettre,compteurMot,motAepeler,compteurCaract
put phraseAepeler into listeAEpeler
put number of words of listeAEpeler into nombreDeMot
repeat with compteurMot=1 to nombreDeMot
put word compteurMot of phraseAepeler into motAepeler
put length(motAepeler) into nombreDeCaractere
if the shiftKey is down then
beep
exit repeat
put empty into phraseAepeler
end if
put empty into ponctuationadire
if last char of motAepeler is in ".;?!," then
put last char of motAepeler into ponctuationadire
delete last char of motAepeler
end if
annoncer motAepeler,motAepeler,60
wait 9
wait until revIsSpeaking() is false
repeat with compteurCaract=1 to nombreDeCaractere
wait until revIsSpeaking() is false
put char compteurCaract of motAepeler into caractereAepeler
annoncer caractereAepeler,caractereAepeler,160
wait 6
end repeat
if ponctuationadire is not empty then
annoncer ponctuationadire,ponctuationadire,160
end if
wait 6
end repeat
end epeler
```

Annexe 4 : command arrowKey avant correction.

```
on arrowkey X
  global langue_parlee
  global isuntitremessage
  global PointeurEnCours,EspacesTapes,DeplacementSilencieux
  global VuePersonneFPHONEME,EchoLettre,nomVoix
  global TextePrononce,vitvoix,VitEchoVoix,VolVoix,VolEchoVoix,HautVoix,HautEchoVoix,OCRactif,secretaire,nomEchoVoix
  global AvantRegle,Regle,ApresRegle,Phonetic,LongRegle,LongAvantRegle,LongApresRegle,ref
  global AvantRegleTest,RegleTest,ApresRegleTest,PhoneticTest,LongAvantRegleTest,LongRegleTest,LongApresRegleTest
  global PointeurAvantLecture,DernierPositionnement
  global modeConference
  --answer langue_parlee
  --revStopSpeech
  if the selectedfield=empty then exit arrowkey

  --  if the selectedfield is "field 13" and isuntitremessage is true then
  --    global pointeurEnCours
  --    put fld "titre" into tt
  --    put line (word 2 of the SelectedLine) of tt into ligne
  --    put the length of ligne into numcharline
  --    put offset(":", ligne) into posContenu
  --    put char 1 to posContenu of ligne into nomChamp
  --    put offset (nomchamp,tt) into numcharavline
  --    --put word 2 of the selectedChunk into q_m
  --    --put q_m into q_e
  --    --put q_m into q_f
  --    do "put " & the selectedfield & " into chtemp"


  --    if X="left" then
  --      --  ------     ----

  --      put the selectedchunk into pointeurEnCours
  --      delete word 2 to 3 of pointeurEnCours
  --      subtract 1 from word 2 of pointeurEnCours
  --      --put char (word 2 of pointeurencours) of field "titre" into texteProno nce
  --      ----------




  --      ----------
  --      if word 2 of pointeurencours <=numcharavline +posContenu then
  --        put "début" into texteProno nce
  --        put "char " & numcharavline +posContenu & " of field 13" into pointeurencours
  --        annoncer texteProno nce
  --        put "oui" into Echolettre
  --      else
```

```
--          put "oui" into EchoLettre
--          put char (word 2 of pointeurencours) of chtemp into texteDit
--          put VoixMinMaj(texteDit) into nomVoix
--          if deplacementSilencieux=empty then annoncer texteDit,texteDit
--        end if
--        ---------------------------

--        select after pointeurencours
--        ---------------------------------------
--      end if

--      if x is "right" then
--        ---langue_parlee
--        --answer word 2 of the SelectedLine
--        put the selectedchunk into pointeurEnCours
--        delete word 3 to 4 of pointeurEnCours
--        --subtract 1 from word 2 of pointeurEnCours
--        --do "select "&PointeurEnCours
--        --put the selectedtext into textePrononce
--        --put char (word 2 of pointeurencours) of field "titre" into textePrononce
--        ----------

--        ----------
--        if word 2 of pointeurencours >=numcharavline +numcharline then
--          put "fin" into textePrononce
--          put "char " & numcharavline +numcharline -1& " of field 13" into pointeurencours
--          annoncer textePrononce
--        else
--          put "oui" into EchoLettre
--          put char  (word 2 of pointeurencours) of chtemp into TexteDit
--          put VoixMinMaj(texteDit) into nomVoix
--          if deplacementSilencieux=empty then annoncer TexteDit,TexteDit
--        end if



--        --annoncer textePrononce
--        select after pointeurencours
--      end if
--      --------
--      --
--      if X="up" then
--        --answer langue_parlee
--        put fld "titre" into ttt
--        put length(ttt) into longueur_ttt
--        put the selectedchunk into pointeurEnCours
--        delete word 2 to 3 of pointeurEnCours
--        put word 2 of the selectedChunk into q_m
```

```
--      put q_m into q_e
--      put q_m into q_f
--      if word 2 of pointeurencours <=numcharavline +posContenu then
--        put "début" into texteProno nce
--        put "char " & numcharavline +posContenu & " of field 13" into pointeurencours
--        annoncer texteProno nce
--        select after pointeurencours
--        exit arrowkey
--      end if
--      put char q_m-1 of ttt into c
--      if c is in ","&return and (c=char q_m-2 of ttt or c=return) then
--        put empty into Nbspace
--        repeat while char q_m-1 of ttt=c
--          subtract 1 from q_m
--          add 1 to nbSpace
--        end repeat
--        subtract 1 from q_m
--        if nbspace<>empty then
--          if c=space then
--            put " espace" into aa
--            put " space" into ab
--          else
--            subtract 1 from NbSpace
--            put " ligne vide" into aa
--            put " empty line" into ab
--          end if
--          if nbspace>1 then annoncer nbspace&aa,nbspace&ab,90 -- secretaire<90
--          else annoncer aa,ab,90 -- secretaire<90
--          do "select char q_m of "&the selectedfield
--          put the selectedchunk into PointeurEnCours
--          delete word 2 to 3 of PointeurEnCours
--          do "select after "&pointeurEnCours
--          exit arrowkey
--        end if
--      end if
--      repeat until char q_m-2 of ttt is in ","&return or q_m < 3
--        subtract 1 from q_m
--      end repeat
--      --      if shifttouche=1 or optiontouche=1 then
--      --        repeat until char q_m-2 of chtemp is return or q_m < 3
--      --          subtract 1 from q_m
--      --        end repeat
--      --      end if
--      if q_m-1 <= numcharavline +posContenu then put numcharavline +posContenu+2 into q_m
--      do "select char q_m-1 to q_e-1 of "&the selectedfield
--      put the selectedchunk into PointeurEnCours
--      put the selectedtext into TexteProno nce
--      if last char of texteprono nce is in "," & " " & return then delete last char of texteprono nce
```

```
--      put VoixMinMaj(textePrononce) into nomVoix
--      if deplacementSilencieux=empty then VocaliserBilingue
--      delete word 3 to 4 of PointeurEnCours
--      subtract 1 from word 2 of PointeurEnCours
--      do "select after "&pointeurEnCours
--    end if
--    ------------------------------------------------
--    if X is "down" then
--      put field "titre" into ttt
--      put length(ttt) into longueur_ttt
--      put the selectedchunk into pointeurencours
--      delete word 2 to 3 of pointeurencours
--      put the word 2 of pointeurencours into q_m
--      if q_m+1 >=numcharavline +numcharline then
--        put "fin" into textePrononce
--        put "char " & numcharavline +numcharline -1& " of field 13" into pointeurencours
--        annoncer textePrononce
--        select after pointeurencours
--        exit arrowkey
--      end if

--      put char q_m+1 of ttt into c
--      --put char q_m+2of ttt into c1
--      if c is in ","&return then
--        put empty into Nbspace
--        if char q_m+1 of ttt is space then
--          repeat while char q_m+1 of ttt=c
--            add 1 to q_m
--            add 1 to nbSpace
--          end repeat
--          --subtract 1 from q_m
--        end if
--        --          --put q_m into q_f
--        if c=space then
--          put " espace" into aa
--          put "space" into ab
--        end if

--        if nbspace>1 then annoncer (nbspace)&aa,(nbspace)&ab,90 -- secretaire<90
--        else annoncer aa,ab,90 -- secretaire<90
--        do "select char q_m of "&the selectedfield
--        put the selectedchunk into PointeurEnCours
--        delete word 2 to 3 of PointeurEnCours
--        do "select after "&pointeurEnCours
--        exit arrowkey
--        --do "select after char q_m of "&the selectedfield
--        --          put the selectedtext into the selection
--        --          put the selectedChunk into PointeurEnCours
```

```
--          --          delete word 3 to 4 of PointeurEnCours
--          --          select after PointeurEnCours
--          --exit arrowkey
--        end if

--        put q_m into q_f
--        repeat until char q_m+2 of ttt is in ","&return or q_m>longueur_ttt
--          Add 1 to q_m
--        end repeat
--        --add 1 to q_m
--        do "select char (q_f+1) to (q_m+2) of "&the selectedfield
--        if the selectedtext is not " " then
--          put the selectedchunk into pointeurEnCours
--          put the selectedtext into texteProno nce
--          if last char of texteprononce is in "," & " " & return then delete last char of texteprononce
--          --        --put the selectedtext into field "texte"
--          if texteProno nce is not in return&space then
--            put VoixMinMaj(texteProno nce) into nomVoix
--            VocaliserBilingue
--          end if
--          --do "select char q_m of "&the selectedfield
--          put the selectedchunk into PointeurEnCours
--          delete word 2 to 3 of PointeurEnCours
--          put word 2 of pointeurencours into nnnn
--          if char (nnnn) of ttt is return then  subtract 1 from word 2 of pointeurencours
--          do "select after "&pointeurEnCours
--          --        --        --answer PointeurEnCours
--          --        --        delete word 2 to 3 of PointeurEnCours
--          --        put the selectedchunk into pointeurEnCours
--          --        delete word 2 to 3 of PointeurEnCours
--          --        --subtract 1 from word 2 of PointeurEnCours
--          --        if word 2 of pointeurencours+1 >=numcharavline +numcharline then
--          --          put "fin" into texteProno nce
--          --          put "char " & numcharavline +numcharline -1& " of field 13" into pointeurencours
--          --          annoncer texteProno nce
--          --        end if
--          --        --select after PointeurEnCours
--          --        --put last char of the selectedtext into last char of the selection
--          --        select after pointeurencours
--          exit arrowkey
--        end if
--      end if

--    else
if the commandKey is down then

  put the ticks into temps
  --global PointeurAvantLecture,DernierPositionnement
```

```
  put word 2 of the selectedline into wsc
  if dernierPositionnement is not in "devant derrière début fin" or¬
  (wsc>1 and wsc < number of lines of fld "texte" and¬
  wsc-1<>wsc<>line 1 of fld "débutLecture") then
    put the selectedchunk into PointeurAvantLecture
  end if
  put line 1 of fld "débutLecture" into p
  put line 3 of fld "débutLecture" into pm
  if p=empty or p>number of lines of fld "texte" then put 1 into p
  wait until the ticks>temps+48 or the commandkey is up
  if X is in "Up Down" then
    if X="Up" then put QCM("avant,début,reprise","at,begin,back") into c1
    else if X="Down" then put QCM("après,fin,reprise","after,end,back") into c1
    if c1<>"annulé" then
      put c1 into dernierPositionnement
      if c1="avant" then
        if pm=empty then select before line p of fld "texte"
        else select before char (item 1 of pm) of line p of fld "texte"
      else if c1="début" then select before char 1 of fld "texte"
      else if c1="après" then
        if pm=empty then select after line p of fld "texte"
        else select after char (item 2 of pm) of line p of fld "texte"
      else if c1="fin" then select after last char of fld "texte"
      else if c1="reprise" then do "select "&PointeurAvantLecture
    end if
  else if x is in "Left Right" then
    --gauche,droite
    if number of marked cds=0 then annoncer "aucune page marquée","no marked page"
    else
      if X="Left" then go previous marked card
      if X="Right" then go next marked card
    end if
  end if
  exit arrowkey
end if

--global modeConference
if modeConference = "oui" then
  send "deplacementConference X" to bg btn "lecteur"
  exit arrowkey
end if

if the shiftkey is down then put 1 into shifttouche
else put empty into shifttouche
if the optionkey is down then put 1 into optiontouche
else put empty into optiontouche
--  global PointeurEnCours,EspacesTapes,DeplacementSilencieux
--  global VuePersonneFPHONEME,EchoLettre,nomVoix
```

```
-- global TextePrononce,vitvoix,VitEchoVoix,VolVoix,VolEchoVoix,HautVoix,HautEchoVoix,OCRactif,secretaire,nomEchoVoix
-- global AvantRegle,Regle,ApresRegle,Phonetic,LongRegle,LongAvantRegle,LongApresRegle,ref
-- global AvantRegleTest,RegleTest,ApresRegleTest,PhoneticTest,LongAvantRegleTest,LongRegleTest,LongApresRegleTest

if VuePersonneFPHONEME="voyant" or the selection<>empty then
  if the optionkey is down then wait until the optionkey is up
  pass arrowkey
end if
put empty into TexteDit

put the selectedline into ligne
put the selectedChunk into PointeurEnCours
delete word 3 to 4 of PointeurEnCours
do "put " & the selectedfield & " into chtemp"
put length(chtemp) into longueur_champ
--
put word 2 of the selectedChunk into q_m
put q_m into q_e
put q_m into q_f
if vitvoix is a number then put round(vitvoix * 1.1) into VitEchoVoix
else put 200 into VitEchoVoix
if X="right" then

  if shifttouche=1 or optiontouche=1 then
    repeat until char q_m to q_m+1 of chtemp is in ". ; ? ! " or char q_m+1 of chtemp= return or q_m > length(chtemp)
      Add 1 to q_m
    end repeat
    if q_m<>q_f then
      do "select char q_f to q_m of "&the selectedfield
      if last word of the selectedField is number of bg fld "titre" and ¬
      the visible of bg btn "Mél" is true and bg fld "État Mél" is not empty then
        --
      else
        put the selectedtext into TextePrononce
        put VoixMinMaj(textPrononce) into nomVoix
        if textePrononce is not in space & return then VocaliserBilingue
      end if

      if optiontouche=1 then
        subtract 1 from word 2 of PointeurEnCours
        do "select after "&PointeurEnCours
      else
        put the selectedtext into the selection
        put the selectedchunk into PointeurEnCours
      end if
    end if
    exit arrowkey
  end if
```

```
   put "oui" into EchoLettre

   if char q_m of chtemp is return then put "ligne vide" into texteDit
   else put char q_m of chtemp into TexteDit


   put VoixMinMaj(texteDit) into nomVoix

   if deplacementSilencieux=empty then annoncer TexteDit,TexteDit

   if q_m>length(chtemp) then


      put secretaire into nomEchoVoix

      if revIsSpeaking() is false then annoncer "Stop","Stop",160, 1, "libre" --scribe>150
      --wait until revIsSpeaking() is false
   end if
   Add 1 to q_m
   pass arrowkey
end if
--OCR dr,autr annul
put empty into OCRactif

if X="left" then
   if shifttouche=1 or optiontouche=1 then
      repeat until char q_m-2 to q_m-1 of chtemp is in ". ; ? ! " or char q_m-2 of chtemp = return or q_m < 3
         subtract 1 from q_m
      end repeat
      do "select char q_m-1 to q_e-1 of "&the selectedfield
      put the selectedtext into TextePrononce
      put VoixMinMaj(textPrononce) into nomVoix
      if TextePrononce is not in space & space & return then VocaliserBilingue
      put the selectedchunk into pointeurEnCours
      if optiontouche=1 then
         delete word 2 to 3 of pointeurEnCours
         put last char of the selectedtext into last char of the selection
      else
         delete word 3 to 4 of pointeurEnCours
         subtract 1 from word 2 of pointeurEnCours
         do "select "&PointeurEnCours
         put the selectedtext into the selection
      end if
      exit arrowkey
   end if
   if q_m < 3 then
```

```
    if revIsSpeaking() is false then annoncer "début","begin",90, 1, "libre" --secretaire<90

    put 2 into q_m
  else
    if char q_m-2 of chtemp is return then put " ligne vide" into texteDit
    else put char q_m-2 of chtemp into texteDit
    put VoixMinMaj(texteDit) into nomVoix
    if deplacementSilencieux=empty then annoncer texteDit,texteDit
    put "oui" into EchoLettre
  end if
  subtract 1 from q_m
  put "char "&q_m-1&" of "&the selectedfield into pointeurEnCours
  pass arrowkey
end if

if X="down" then
  put char q_m of chtemp into c
  if c is in space & return then
    put empty into Nbspace
    repeat while char q_m+1 of chtemp=c
      add 1 to q_m
      add 1 to nbSpace
    end repeat
    put q_m into q_f
    if c=space then
      put " espace" into aa
      put "space" into ab
    else
      put " ligne" into aa
      put "line" into ab
      if nbspace>0 then
        if c=return then subtract 1 from NbSpace
        put " vide" after aa
        put " empty " before ab
      end if
    end if
    if nbspace>1 then annoncer (nbspace+1)&aa,(nbspace+1)&ab,90 -- secretaire<90
    else annoncer aa,ab,90 -- secretaire<90
    do "select char q_m of "&the selectedfield
    put the selectedtext into the selection
    put the selectedChunk into PointeurEnCours
    delete word 2 to 3 of PointeurEnCours
    exit arrowkey
  end if
  --
  repeat until char q_m of chtemp is in space&return or q_m>longueur_champ
    Add 1 to q_m
  end repeat
```

```
  if shifttouche=1 or optiontouche=1 then
    repeat until char q_m of chtemp is return or q_m>longueur_champ
      Add 1 to q_m
    end repeat
  end if
  do "select char q_f to q_m of "&the selectedfield
  put the selectedchunk into pointeurEnCours
  put the selectedtext into textePrononce
  if textePrononce is not in return&space then
    put VoixMinMaj(textePrononce) into nomVoix
    VocaliserBilingue
  end if
  if optiontouche=1 then
    delete word 3 to 4 of pointeurEnCours
    subtract 1 from word 2 of PointeurEnCours
    do "select "&PointeurEnCours
    put the selectedtext into the selection
    exit arrowkey
  end if
  delete word 2 to 3 of PointeurEnCours
  put last char of the selectedtext into last char of the selection
  if q_m >= longueur_champ then
    put secretaire into nomEchoVoix
    if revIsSpeaking() is false then annoncer "Stop","Stop",90, 1, "libre" --secretaire<90
    direLaPage
  end if
  exit arrowkey
end if


if X="up" then
  if q_m < 3 then
    --          arrowkey left
    --          exit arrowkey
    if shifttouche=1 or optiontouche=1 then
      repeat until char q_m-2 to q_m-1 of chtemp is in ". ; ? ! " or char q_m-2 of chtemp = return or q_m < 3
        subtract 1 from q_m
      end repeat
      do "select char q_m-1 to q_e-1 of "&the selectedfield
      put the selectedtext into TextePrononce
      put VoixMinMaj(textPrononce) into nomVoix
      if TextePrononce is not in space & space & return then VocaliserBilingue
      put the selectedchunk into pointeurEnCours
      if optiontouche=1 then
        delete word 2 to 3 of pointeurEnCours
        put last char of the selectedtext into last char of the selection
      else
        delete word 3 to 4 of pointeurEnCours
```

```
        subtract 1 from word 2 of pointeurEnCours
        do "select "&PointeurEnCours
        put the selectedtext into the selection
      end if
      exit arrowkey
    end if
    if q_m < 3 then
      if revIsSpeaking() is false then annoncer "début","begin",90, 1, "libre" --secretaire<90

      put 2 into q_m
    else
      put char q_m-2 of chtemp into texteDit
      put VoixMinMaj(texteDit) into nomVoix
      if deplacementSilencieux=empty then annoncer texteDit,texteDit
      put "oui" into EchoLettre
    end if
    subtract 1 from q_m
    put "char "&q_m-1&" of "&the selectedfield into pointeurEnCours
    pass arrowkey
  end if
  put char q_m-1 of chtemp into c
  if c is in space&return and (c=char q_m-2 of chtemp or c=return) then
    put empty into Nbspace
    repeat while char q_m-1 of chtemp=c
      subtract 1 from q_m
      add 1 to nbSpace
    end repeat
    subtract 1 from q_m
    if nbspace<>empty then
      if c=space then
        put " espace" into aa
        put " space" into ab
      else
        subtract 1 from NbSpace
        put " ligne vide" into aa
        put " empty line" into ab
      end if
      if nbspace>1 then annoncer nbspace&aa,nbspace&ab,90 -- secretaire<90
      else annoncer aa,ab,90 -- secretaire<90
      do "select char q_m of "&the selectedfield
      put the selectedchunk into PointeurEnCours
      delete word 2 to 3 of PointeurEnCours
      do "select after "&pointeurEnCours
      exit arrowkey
    end if
  end if
  repeat until char q_m-2 of chtemp is in space&return or q_m < 3
    subtract 1 from q_m
```

```
    end repeat
    if shifttouche=1 or optiontouche=1 then
      repeat until char q_m-2 of chtemp is return or q_m < 3
        subtract 1 from q_m
      end repeat
    end if
    do "select char q_m-1 to q_e-1 of "&the selectedfield
    put the selectedchunk into PointeurEnCours
    put the selectedtext into TextePrononce
    put VoixMinMaj(textePrononce) into nomVoix
    if deplacementSilencieux=empty then VocaliserBilingue
    if optiontouche=1 then
      delete word 2 to 3 of PointeurEnCours
      do "select after "&pointeurEnCours
      put last char of the selectedtext into last char of the selection
      exit arrowkey
    end if
    delete word 3 to 4 of PointeurEnCours
    subtract 1 from word 2 of PointeurEnCours
    do "select after "&pointeurEnCours
    exit arrowkey
  end if

--end if
end arrowkey
```

Annexe 5 : Commande arrowKey corrgée.

```
on arrowkey X
  --Joel Aout 2012
  global langue_parlee
 global isuntitremessage
 global PointeurEnCours,EspacesTapes,DeplacementSilencieux
 global VuePersonneFPHONEME,EchoLettre,nomVoix
 global TextePrononce,vitvoix,VitEchoVoix,VolVoix,VolEchoVoix,HautVoix,HautEchoVoix,OCRactif,secretaire,nomEchoVoix
 global AvantRegle,Regle,ApresRegle,Phonetic,LongRegle,LongAvantRegle,LongApresRegle,ref
 global AvantRegleTest,RegleTest,ApresRegleTest,PhoneticTest,LongAvantRegleTest,LongRegleTest,LongApresRegleTest
 global modeConference
 global RepriseApresLecture,LgChampsRepriseAl,PointeurAvantLecture,DernierPositionnement,blocRepriseAl

 local tt,pp,ligne,qq,rr,contenulignett,temps,wsc,p,pm,sc,c1,shifttouche,optiontouche,TexteDit
 local longueur_champ,q_m,chtemp,q_e,q_f,c,nbSpace,aa,ab
 if the selectedfield=empty then exit arrowkey
 --La section qui suit gère le cas particuliers du champs titre utilisé dans le cadre du mail
 --Dans lequel chaque paragraphe est un sous champs qui ne retourne pas à la ligne
 --chacun des sous champs correspondants à une rubrique
 --Destinataire, sujet, bcc : etc...
 if there is a bg field "titre" and ( the last word of the selectedField is number of bg fld "titre" and  the visible of bg btn "Mél" is true and bg fld "État
Mél" is not empty ) then
  global pointeurEnCours
  put field "titre" into tt
  put word 2 of pointeurEncours into pp
  put  word 2 of the SelectedLine into ligne
  put offset (line ligne of tt, tt) into qq
  put line ligne of tt into contenulignett
  if  X is in "top left" then
   put offset(": ",contenulignett) into rr
   add rr to qq
   if pp <= qq then
    if revIsSpeaking() is false then
     annoncer "debut","begin",70
    end if
    exit arrowkey
   end if
  else if X is in" bottom right" then
   put length(contenulignett) into rr
   add rr to qq
   subtract 1 from qq
   if pp >= qq then
    if revIsSpeaking() is false then
     annoncer "stop","stop",70
    end if
    exit arrowkey
   end if
```

```
    end if
  end if

  ----------
  --Les touches haute et basse utilisées avec la touche command ou control permettent de synchroniser le scribe avec le lecteur.
  --La touche flèche haut met le pointeur au début de la dernière phrase qui vient d'être lue par le lecteur
  -- la touche
  -- le niveau de segmentation est la phrase
  -- cela correspond aux propositions "avant" ou "après" les propositions débuts ou fin pointent directement en debut ou en fin fin du champs texte.
  --"reprise" permet de revenir à la situation initiale du pointeur juste avant la synchronisation
  --les touches droites ou gauches permettent d'accéder ou de créer des marque pages ou signets sur une page du livre.
  --(Proposition précédente marquer, suivante marquer, marquer démarquer.
  --La proposition retour permet de revenir à une page marquée, lorsque de celle-ci l'utilisateur a navigué sur des pages marquées.
  --La proposition travail permet d'accéder à une page dans laquelle sont effectuées un travail de rédaction ou de modification d'une certaine importance.


  if (the commandKey is down  and the optionKey is up and the shiftKey is up) or (the controlKey is down and the optionKey is up and the shiftKey is up) then
    put the ticks into temps


    --put line 1 of fld "débutLecture" into p
    put line 1 of fld "débutLecture" into numParagraphe

    if numParagraphe is empty then put 1 into numParagraphe

    --put line 3 of fld "débutLecture" into pm
    put line 3 of fld "débutLecture" into numsPhrase
     if numsPhrase is empty then put "0,0" into numsPhrase


    put line 1 to numParagraphe of fieldTexte into variableTemporaire
    put length(variableTemporaire) into variableTemporaire

    put item 1 of numsPhrase into NumDebPhraInParag
    put item 2 of numsPhrase into NumFinPhraInParag

    put variableTemporaire + NumDebPhraInParag into PointeurDebPhraLu
    put variableTemporaire + NumFinPhraInParag + 1 into PointeurFinPhraLu

    wait until the ticks>temps+33 or the commandkey is up

    if X is in "Up Down" then
      if X="Up" then put QCM("avant,début,reprise","at,begin,back") into c1
      else if X="Down" then put QCM("après,fin,reprise","after,end,back") into c1
      if c1<>"annulé" then
```

```
                put c1 into dernierPositionnement

                if c1="avant" then

                  --c'est un test pour mémorisation d'inhibition
                  --il faut mémoriser que dans le cas où le pointeur d'insertion ni le début du texte ni la fin du texte ni le début de la phrase lu ni la
fin de la phrase lu
                  if word 2 of the selectedchunk >1  and word 2 of the selectedchunk is not NumDebPhraInParag and word 2 of the selectedchunk is not
NumFinPhraInParag and word 2 of the selectedchunk < length(field"texte")  then
                    put the selectedchunk into RepriseApresLecture


                    put word 2 of RepriseApresLecture into PositionRepriseAl
                    put char 1 to PositionRepriseAl of field "texte" into blocRepriseAl
                    put length(field "texte") into LgChampsRepriseAl

                  end if

                  --Avant de synchroniser, il faut mettre en memoire la position du pointeur d'insertion.
                  --Inserer le pointeur d'insertion avant la dernière phrase lue par le lecteur
                  select before char ( NumDebPhraInParag) of field "texte"

                else if c1="début" then
                  if word 2 of the selectedchunk >1  and word 2 of the selectedchunk is not NumDebPhraInParag and word 2 of the selectedchunk is not
NumFinPhraInParag and word 2 of the selectedchunk < length(field"texte")  then
                    put  the selectedchunk into RepriseApresLecture


                    put word 2 of RepriseApresLecture into PositionRepriseAl
                    put char 1 to PositionRepriseAl of field "texte" into blocRepriseAl
                    put length(field "texte") into LgChampsRepriseAl

                  end if
                  select before char 1 of field "texte"

                else if c1="après" then
                  if word 2 of the selectedchunk >1  and word 2 of the selectedchunk is not NumDebPhraInParag and word 2 of the selectedchunk is not
NumFinPhraInParag and word 2 of the selectedchunk < length(field"texte")  then
                    put the selectedchunk into RepriseApresLecture


                    put word 2 of RepriseApresLecture into PositionRepriseAl
                    put char 1 to PositionRepriseAl of field "texte" into blocRepriseAl
                    put length(field "texte") into LgChampsRepriseAl


                  end if
                  select after char (NumFinPhraInParag) of field "texte"
```

```
            else if c1="fin" then
              if word 2 of the selectedchunk >1  and word 2 of the selectedchunk is not NumDebPhraInParag and word 2 of the selectedchunk is not
NumFinPhraInParag and word 2 of the selectedchunk < length(field"texte")  then
                put the selectedchunk into RepriseApresLecture


                put word 2 of RepriseApresLecture into PositionRepriseAl
                put char 1 to PositionRepriseAl of field "texte" into blocRepriseAl
                put length(field "texte") into LgChampsRepriseAl

              end if
              select after last char of field "texte"

            else if c1="reprise" then
              if word 2 of RepriseApresLecture< length(field"texte") then
                do "select after "&RepriseApresLecture
              else
                select after field"texte"
              end if
            end if
          end if
        else if x is in "left right" then
          if x ="Left" then marquepage "precedente"
          else if x ="right" then marquepage "suivante"
        end if
        exit arrowkey
      end if

    global modeConference
    if modeConference = "oui" then
      send "deplacementConference X" to bg btn "lecteur"
      exit arrowkey
    end if

    if the shiftkey is down then put 1 into shifttouche
    else put empty into shifttouche
    if the optionkey is down then put 1 into optiontouche
    else put empty into optiontouche

    if VuePersonneFPHONEME="voyant" or the selection<>empty then
      if the optionkey is down then wait until the optionkey is up
      pass arrowkey
    end if

    put empty into TexteDit
    put the selectedline into ligne
    put the selectedChunk into PointeurEnCours
```

```
delete word 3 to 4 of PointeurEnCours
do "put " & the selectedfield & " into chtemp"
put length(chtemp) into longueur_champ
put word 2 of the selectedChunk into q_m
put q_m into q_e
put q_m into q_f
if vitvoix is a number then put round(vitvoix * 1.1) into VitEchoVoix
else put 200 into VitEchoVoix
if X="right" then
  if shifttouche=1 or optiontouche=1 then
    repeat until char q_m to q_m+1 of chtemp is in ". ; ? ! " or char q_m+1 of chtemp= return or q_m > length(chtemp)
      Add 1 to q_m
    end repeat
    if q_m<>q_f then
      do "select char q_f to q_m of "&the selectedfield
      if last word of the selectedField is number of bg fld "titre" and the visible of bg btn "Mél" is true and bg fld "État Mél" is not empty then
        --
      else
        put the selectedtext into TextePrononce
        put VoixMinMaj(textPrononce) into nomVoix
        if textePrononce is not in space & return then VocaliserBilingue
      end if

      if optiontouche=1 then
        subtract 1 from word 2 of PointeurEnCours
        do "select after "&PointeurEnCours
      else
        put the selectedtext into the selection
        put the selectedchunk into PointeurEnCours
      end if
    end if
    exit arrowkey
  end if


  put "oui" into EchoLettre
  if char q_m of chtemp is return then put "ligne vide" into texteDit
  else put char q_m of chtemp into TexteDit
  put VoixMinMaj(texteDit) into nomVoix
  if deplacementSilencieux=empty then annoncer TexteDit,TexteDit
  if q_m>length(chtemp) then
    put secretaire into nomEchoVoix
    if revIsSpeaking() is false then annoncer "Stop","Stop",160, 1, "libre" --scribe>150
  end if
  Add 1 to q_m
  pass arrowkey
end if
```

```
put empty into OCRactif

if X="left" then
  if shifttouche=1 or optiontouche=1 then
    repeat until char q_m-2 to q_m-1 of chtemp is in ". ; ? ! " or char q_m-2 of chtemp = return or q_m < 3
      subtract 1 from q_m
    end repeat
    do "select char q_m-1 to q_e-1 of "&the selectedfield
    put the selectedtext into TextePrononce
    put VoixMinMaj(textPrononce) into nomVoix
    if TextePrononce is not in space & space & return then VocaliserBilingue
    put the selectedchunk into pointeurEnCours
    if optiontouche=1 then
      delete word 2 to 3 of pointeurEnCours
      put last char of the selectedtext into last char of the selection
    else
      delete word 3 to 4 of pointeurEnCours
      subtract 1 from word 2 of pointeurEnCours
      do "select "&PointeurEnCours
      put the selectedtext into the selection
    end if
    exit arrowkey
  end if
  if q_m < 3 then
    if revIsSpeaking() is false then annoncer "début","begin",90, 1, "libre" --secretaire<90

    put 2 into q_m
  else
    if char q_m-2 of chtemp is return then put " ligne vide" into texteDit
    else put char q_m-2 of chtemp into texteDit
    put VoixMinMaj(texteDit) into nomVoix
    if deplacementSilencieux=empty then annoncer texteDit,texteDit
    put "oui" into EchoLettre
  end if
  subtract 1 from q_m
  put "char "&q_m-1&" of "&the selectedfield into pointeurEnCours
  pass arrowkey
end if

if X="down" then
  put char q_m of chtemp into c
  if c is in space & return then
    put empty into Nbspace
    repeat while char q_m+1 of chtemp=c
      add 1 to q_m
      add 1 to nbSpace
    end repeat
    put q_m into q_f
```

```
  if c=space then
    put " espace" into aa
    put "space" into ab
  else
    put " ligne" into aa
    put "line" into ab
    if nbspace>0 then
      if c=return then subtract 1 from NbSpace
      put " vide" after aa
      put " empty " before ab
    end if
  end if
  if nbspace>1 then annoncer (nbspace+1)&aa,(nbspace+1)&ab,90 -- secretaire<90
  else annoncer aa,ab,90 -- secretaire<90
  do "select char q_m of "&the selectedfield
  put the selectedtext into the selection
  put the selectedChunk into PointeurEnCours
  delete word 2 to 3 of PointeurEnCours
  exit arrowkey
end if
--
repeat until char q_m of chtemp is in space&return or q_m>longueur_champ
  Add 1 to q_m
end repeat
if shifttouche=1 or optiontouche=1 then
  repeat until char q_m of chtemp is return or q_m>longueur_champ
    Add 1 to q_m
  end repeat
end if
do "select char q_f to q_m of "&the selectedfield
put the selectedchunk into pointeurEnCours
put the selectedtext into texteProno nce
if texteProno nce is not in return&space then
  put VoixMinMaj(texteProno nce) into nomVoix
  VocaliserBilingue
end if
if optiontouche=1 then
  delete word 3 to 4 of pointeurEnCours
  subtract 1 from word 2 of PointeurEnCours
  do "select "&PointeurEnCours
  put the selectedtext into the selection
  exit arrowkey
end if
delete word 2 to 3 of PointeurEnCours
put last char of the selectedtext into last char of the selection
if q_m >= longueur_champ then
  put secretaire into nomEchoVoix
  if revIsSpeaking() is false then annoncer "Stop","Stop",90, 1, "libre" --secretaire<90
```

```
    direLaPage
  end if
  exit arrowkey
end if
if X="up" then
  if q_m < 3 then

    if shifttouche=1 or optiontouche=1 then
      repeat until char q_m-2 to q_m-1 of chtemp is in ". ; ? ! " or char q_m-2 of chtemp = return or q_m < 3
        subtract 1 from q_m
      end repeat
      do "select char q_m-1 to q_e-1 of "&the selectedfield
      put the selectedtext into TextePrononce

      put VoixMinMaj(textPrononce) into nomVoix
      if TextePrononce is not in space & space & return then VocaliserBilingue
      put the selectedchunk into pointeurEnCours
      if optiontouche=1 then
        delete word 2 to 3 of pointeurEnCours
        put last char of the selectedtext into last char of the selection
      else
        delete word 3 to 4 of pointeurEnCours
        subtract 1 from word 2 of pointeurEnCours
        do "select "&PointeurEnCours
        put the selectedtext into the selection
      end if
      exit arrowkey
    end if
    if q_m < 3 then
      if revIsSpeaking() is false then annoncer "début","begin",90, 1, "libre" --secretaire<90

      put 2 into q_m
    else
      put char q_m-2 of chtemp into texteDit
      put VoixMinMaj(texteDit) into nomVoix
      if deplacementSilencieux=empty then annoncer texteDit,texteDit
      put "oui" into EchoLettre
    end if
    subtract 1 from q_m
    put "char "&q_m-1&" of "&the selectedfield into pointeurEnCours
    pass arrowkey
  end if
  put char q_m-1 of chtemp into c
  if c is in space&return and (c=char q_m-2 of chtemp or c=return) then
    put empty into Nbspace
    repeat while char q_m-1 of chtemp=c
      subtract 1 from q_m
      add 1 to nbSpace
```

```
      end repeat
      subtract 1 from q_m
      if nbspace<>empty then
        if c=space then
          put " espace" into aa
          put " space" into ab
        else
          subtract 1 from NbSpace
          put " ligne vide" into aa
          put " empty line" into ab
        end if
        if nbspace>1 then annoncer nbspace&aa,nbspace&ab,90 -- secretaire<90
        else annoncer aa,ab,90 -- secretaire<90
        do "select char q_m of "&the selectedfield
        put the selectedchunk into PointeurEnCours
        delete word 2 to 3 of PointeurEnCours
        do "select after "&pointeurEnCours
        exit arrowkey
      end if
    end if
    repeat until char q_m-2 of chtemp is in space&return or q_m < 3
      subtract 1 from q_m
    end repeat
    if shifttouche=1 or optiontouche=1 then
      repeat until char q_m-2 of chtemp is return or q_m < 3
        subtract 1 from q_m
      end repeat
    end if
    do "select char q_m-1 to q_e-1 of "&the selectedfield
    put the selectedchunk into PointeurEnCours
    put the selectedtext into TextePrononce
    put VoixMinMaj(textePrononce) into nomVoix
    if deplacementSilencieux=empty then VocaliserBilingue
    if optiontouche=1 then
      delete word 2 to 3 of PointeurEnCours
      do "select after "&pointeurEnCours
      put last char of the selectedtext into last char of the selection
      exit arrowkey
    end if
    delete word 3 to 4 of PointeurEnCours
    subtract 1 from word 2 of PointeurEnCours
    do "select after "&pointeurEnCours
    exit arrowkey
  end if
end arrowkey
```

Annexe 6 : La fonction «Texte en texte phonétique »

```
function FPHONEMETexteToPhonetic TexteNonPhonetique
  local T,longeurT,R,
  local chainePointee,NumerosRegleTrouve,longChaine,positionChaine,alphabetTraite
  put TexteNonPhonetique into T
  put empty into longeurT
  global AvantRegle,Regle,ApresRegle,Phonetic,LongRegle,LongAvantRegle,LongApresRegle
  global AvantRegleTest,RegleTest,ApresRegleTest,PhoneticTest,LongAvantRegleTest,LongRegleTest,LongApresRegleTest
  put length(T) into longeurT
  put 1 into positionChaine
  put 1 into longChaine
  put empty into chainePointee
  put empty into NumerosRegleTrouve
  put space & "&abcdefghijklmnopqrstuvwxyz0123456789+-*/\=#^'_$[(){}§@.:,?!:<>%`"&quote into alphabetTraite
  repeat while (positionChaine <= longeurT)
    --if ( the controlKey is down) then exit repeat
    put char positionChaine of T into chainePointee
    if chainePointee is in alphabetTraite then
    --contient la liste des caratcere qui est traité par la fonction.
    --tout caractere non traité est ignoré par déplacement du pointeur dans la boucle avant traittement
    ---add 1 to positionChaine
    ---else
    --La fonction BlocReglesToBlocCar  renvoie l'ensemble des regles du caractère pointé..
    BlocReglesToBlocCar(chainePointee)
    --A l'intérieur du bloc de règle de la caractère pointée, la fonction selectionne la règle qui sera appliquée sur le caractère pointée et
eventuellement quelques cararcteres quui lui succedent
    --elle retourne &galmt de combien le pointeur doit être deplacé
    put SelectionRegle(T,positionChaine) into NumerosRegleTrouve
    put line NumerosRegleTrouve of PhoneticTest after R
    put line NumerosRegleTrouve of LongRegleTest into longChaine
    add longChaine to positionChaine
else
  add 1 to positionchaine
end if
    if R is empty then
      put "\/" into R
      exit repeat
    end if

    --la ligne suivante met la partie phonetique de la règle
    --r contient la chaine phonetique dite par revSpeack
    --end if
  end repeat
  --put antiRobot(R) into R
  return R
end FPHONEMETexteToPhonetic
```

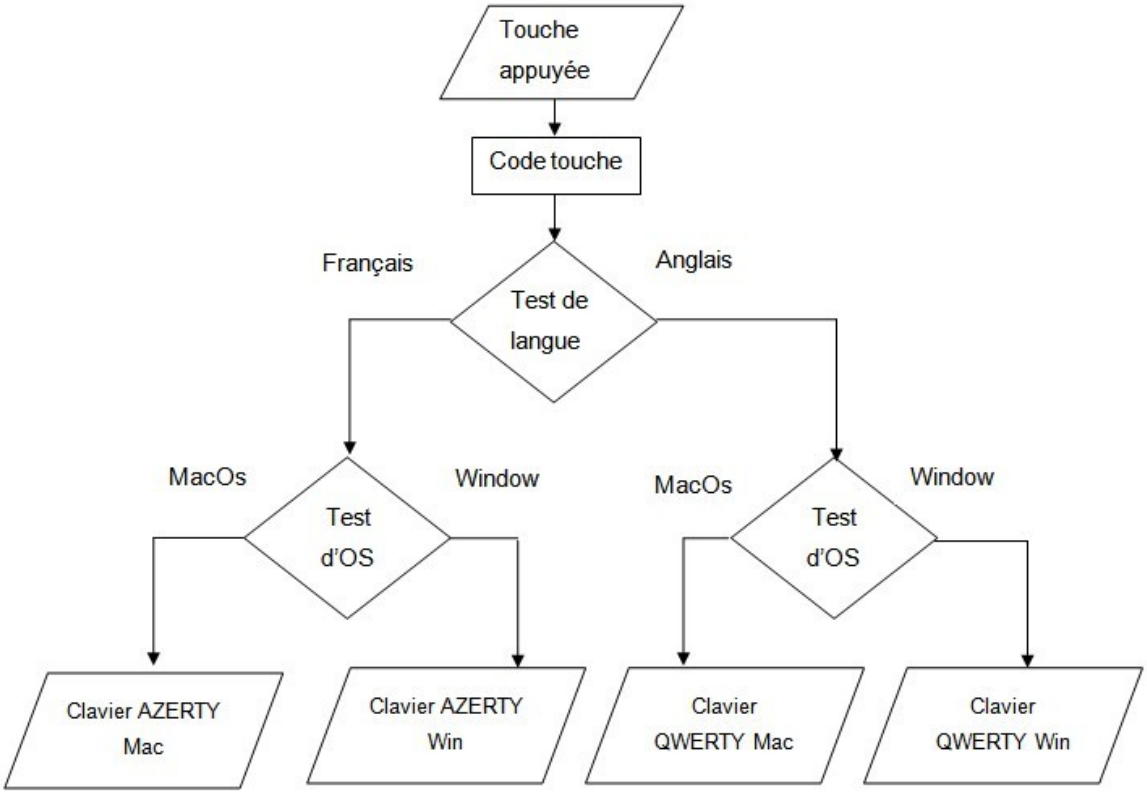

Figure 1 Logigramme de la fonction « Touche pressé »

```
put "32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,7
put " !"&quote&"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}←↑→↓" into clavierqwertyPc
put " 1%3457ù908+;):!à&é"&quote&"'(-è_cMm.=/§2QBCDEFGHIJKL?NOPARSTUVZXYW^*$6°<qbcdefghijkl,noparstuvzxyw¨>£←↑→↓" into clavierazertyPc
put " 1%3457ù908_;):=à&é"&quote&"'(§è!çMm.-/+2QBCDEFGHIJKL?NOPARSTUVZXYW^*$6°<qbcdefghijkl,noparstuvzxyw¨>°←↑→↓" into clavierazertyMac
put " !"&quote&"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[]^_`abcdefghijklmnopqrstuvwxyz{~}←↑→↓" into clavierqwertyMac
put " ´‰"'[»ÙÁØÛ-…}÷≠øëë"&quote&"'{¶«¡ÇÓµ•—\±„Ω∫¢ΔÊ·fÎÎìÍ¡ŒÏÆ,∑™ª√ÅŸ›ôŸ€å]≤‡ß©∂êƒfiÎìÍÊ¬∞∞~œπæ®Ò†°◊Â≈Ú‹Ô≥¥" into clavierAvecAltAzertyMac


--Les touches "SHIFT, ALT et CAPS LOCK" peuvent être appuyées en même temps qu'une autre touche de clavier
--ce sont des touches de modification
--Ces touches (SHIFT = 65505, ALT = 65507, ET LE CAPS LOCK = 65509)
--la fonction keysdown retourne la valeur d'une ou plusieurs de ces touches lorsqu'elles sont simultanément appuyées (séparé par une virgule)
--exemple SHIFT - A donnera (65505, 113) en clavier azerty
--les valeurs des touches de modification sont les mêmes quelque soit le clavier

switch klangue&versionOs
  case "en"&"MacOS"
    put clavierqwertyMac into listecaracteres
    exit switch
  case "fr"&"MacOS"
    put clavierazertyMac into listecaracteres
    put clavierAvecAltAzertyMac into listeCaracteresAlt
    exit switch
  case "en"&"WinOs"
    put clavierqwertyPc into listecaracteres
    put clavierAvecAltAzertyMac into listeCaracteresAlt
    exit switch
  case"fr"&"WinOs"
    put clavierazertyPc into listecaracteres
```

Figure 2 Copie d'écran d'une partie du code de la fonction « inkey » Correspondance entre les codes touches

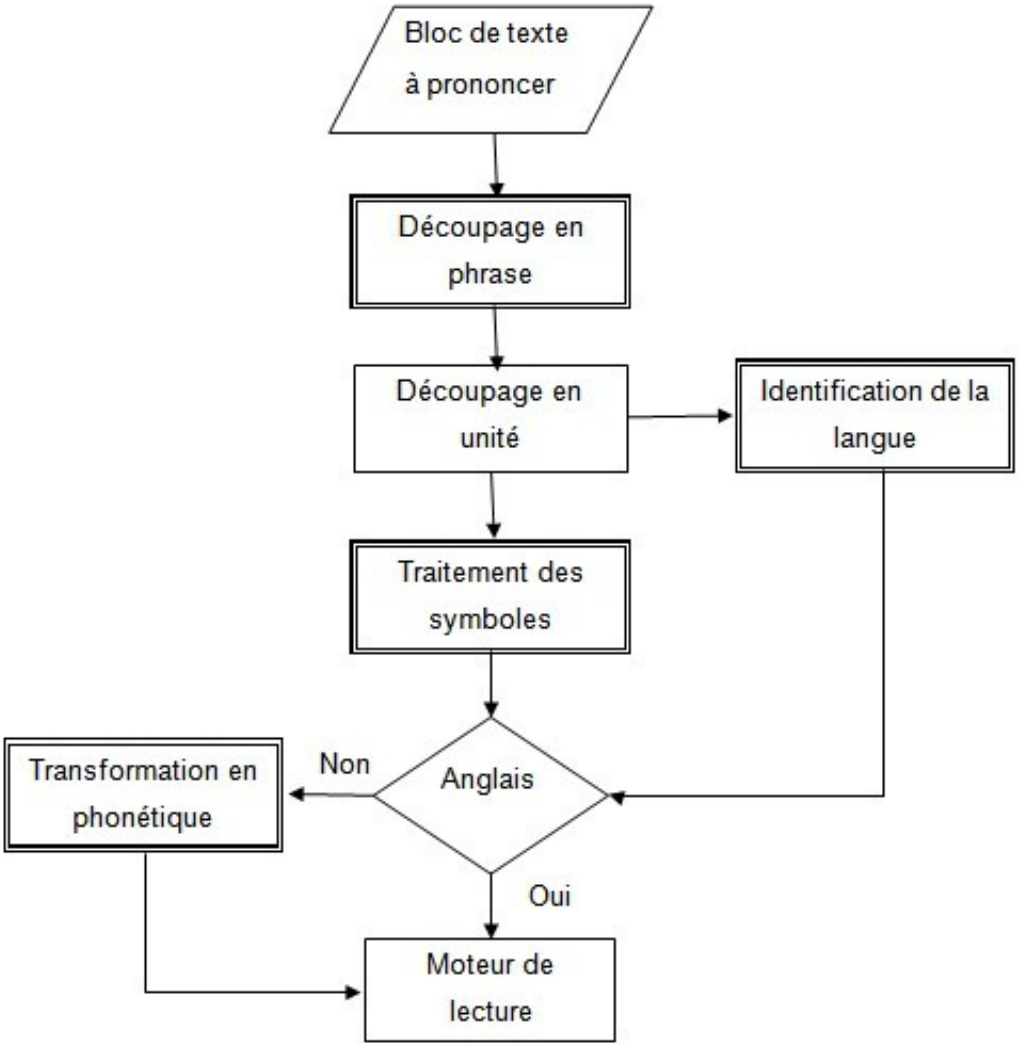

Figure 3 Logigramme simplifiée de la commande « Vocaliser »

G. LISTE DES REFERENCES BIBLIOGRAPHIQUES

- Callamand, M., *Analyse des marques prosodiques de discours.* Etudes de Linguistique Appliquée. Numéro 66, Pages 49-70. Paris, 1987.
- Calvary, Gaëlle, Ingénierie de l'interaction Homme-Machine: *Rétrospective in Traité des sciences et techniques d'information*. Pages 19-63, Hermès, 2002.
- Damerau, Fred J., A *technique for detection and correction of spelling errors*, Communication of the ACM, Volume 7, Issue 3. 1964
- Ferber J. *Les systèmes multi-agents*, InterEditions, 1995 (p8)
- Gibson (1977), Shaw R.; Bransford J., *The Theory of Affordances. In Perceiving, Acting, and Knowing*, Eds. Robert Shaw and John Bransford, Minnesota; 1977
- Jennings N., Sycara K., *Wooldridge M., A Roadmap of agent research and development,* Minneapolis, USA, ACM, 1998.
- Le Moigne, Jean-Louis, *La Théorie du Système Général, Théorie de la Modélisation,* PUF, 1994
- Leveinshtein, Vladmir, *Binary code capable of Correcting Deletions, Insertions and Reversals*. Soviet Physics Doklady, Vol.10, 1966, (p 707).
- Luck M., Mc Burney P., Preist C., *Agent Technology : Enabling next generation computing*, Agntilink II, 2003.
- Nogier, Jean-François, *Ergonomie du logiciel et design web*, 3e édition, Dunod, 2005
- Richaume-Crinquette A., *L'accès à l'information parlée chez l'aveugle*. Thèse de doctorat, I.F.R de psychologie, Université de Lille 3, Villeneuve d'Asq, 1990.

H. INDEX

I. TABLES DES ILLUSTRATIONS ET ANNEXES.