TABLE DES MATIERES

LISTE DES FIGURES

INTRODUCTION

Depuis le siècle dernier, la technologie a connu une croissance quasi exponentielle et touche presque toutes les branches de la vie industrielle, économique et sociale, chacun dans son domaine spécifique. Elle a accéléré le développement et la croissance économique de beaucoup de pays, il en est de même jusqu'à améliorer le bien être de la .population

La technologie, ce domaine qui aide à améliorer et à augmenter les différentes productions industrielles et qui tend à faciliter les tâches antérieurement difficiles à faire ou bien requérant le concours de plusieurs acteurs, trouve encore son application plus avancée en développant en son sein l'électronique, qui est un domaine de la physique appliquée exploitant les variations de grandeurs électriques (courant, tensions, charges, etc.) en vue d'une utilisation bien spécifique.

Devant ce développement à pas de géant de la technique, l'électrique et l'électronique, les besoins deviennent exigeants et nombreux voire complexes, d'où en fait la naissance de l'informatique qui est une science du traitement automatique de l'information, rendant encore plus aisée toutes les opérations tant dans les industries que dans la vie courante de tout homme. En effet dans toutes les branches possédant des informations qu'il faudrait gérer méthodiquement et rapidement soit en vue d'une prise de décision immédiate ou bien en vue d'effectuer une tâche avec toute la précision voulue, l'Informatique joue alors un rôle prépondérant.

A l'Ecole Supérieure Polytechnique d'Antananarivo, le département ELECTRONIQUE a développé un logiciel pédagogique appelé ACCEAO (Analyse et Conception de Circuits Electroniques Assistées par Ordinateur). Ce logiciel a été conçu par les étudiants sous l'environnement MS-DOS, en utilisant le langage de programmation Borland C++ version 3.1.

En effet, pour concrétiser les acquis au terme de la formation, le département s'est fixé un objectif visant à mettre en place ce logiciel destiné à être utilisé aussi bien dans les Lycées Techniques qu'à l'Université. Lequel logiciel combinera deux grands avantages. Il a un intérêt pédagogique, car il synthétise l'application des connaissances acquises tout au long de la formation d'une part, il concrétise réellement ce dont on attend des étudiants

notamment dans la vie active d'autre part. Enfin il a un intérêt technique permettant grâce à cette création ou cette conception de vérifier personnellement le savoir faire des étudiants.

Le travail accompli dans ce mémoire intitulé « ACCEAO version 4.3 » est la révision, et l'amélioration des versions 4.x [1] [2] [3]. Une importante partie du travail impliqué est la conception d'une nouvelle méthode d'implantation de la souris dans le logiciel. C'est un travail initié dans la version 4.2, mais qui a connu des difficultés d'utilisation et des dysfonctionnements.

De ce fait, ce mémoire de fin d'étude est organisé de la manière qui suit :

- Le premier chapitre traite la Maintenance des logiciels, en particulier l'ACCEAO,
- La seconde partie est consacrée à l'implantation de la souris avec les corrections des erreurs,
- En fin, le guide d'utilisation du logiciel ACCEAO version 4.3 est détaillé dans le chapitre 3.

CHAPITRE I :

MAINTENANCE D'UN LOGICIEL

1-1 Introduction

La vie d'un logiciel commence lorsqu'une organisation (par exemple l'entreprise) décide, pour répondre à ses besoins, à le construire. Elle passe par diverses activités de conception et réalisation. Puis elle connaît une période plus ou moins longue d'utilisation et d'adaptation. Enfin, elle se termine lorsque l'Entreprise estime que le logiciel est devenu obsolète.

Cette simple définition de la vie d'un logiciel n'est pas facile à réaliser. Il faut une bonne organisation, et la prise de conscience du fait que le développement du logiciel n'est pas simplement l'écriture de programmes sur une durée variant de quelques heures à quelques jours, d'ou la notion de cycle de vie (datant de 1970). Pour ce faire, ce cycle de vie contient de nombreuses étapes distinctes y compris la maintenance. Ces étapes jouent chacune des rôles importants dans la période de développement et d'utilisation d'un logiciel [4].

1-2 Les différents étapes d'un cycle de vie d'un logiciel [4]

L'activité de développement d'un logiciel peut être perçue comme un long raisonnement mené durant un certain temps. Ce long raisonnement est habituellement décomposé en de nombreuses étapes. Il existe plusieurs types de cycle de vie mais ils adoptent les mêmes définitions des différentes étapes (Fig 1.1).

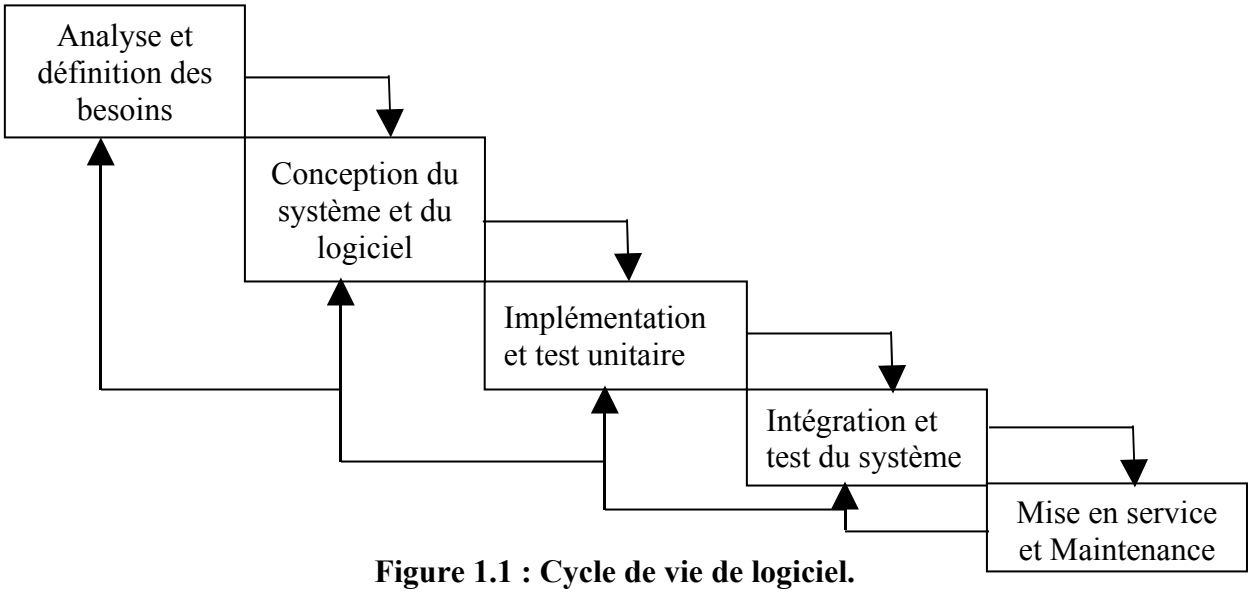

Figure 1.1 : Cycle de vie de logiciel.

a- Analyse et définitions des besoins

C'est un dialogue entre le client et le concepteur pour définir l'ensemble des fonctionnalités désirées et les contraintes à respecter. On essaie d'arriver à une compréhension totale du problème. Un cahier des charges exprime les besoins.

b- Conception du système et du logiciel

En partant de l'analyse des besoins, on décompose le système en deux parties : « matériel » et « logiciel ». Ce processus est appelé « conception du système ».
C'est le processus qui consiste à représenter les diverses fonctions du système d'une manière qui permettra d'obtenir rapidement un ou plusieurs programmes réalisant ces fonctions.

c- Implémentation et tests unitaires

Lors de cette étape, on réalise un ensemble d'unités de programme écrites dans un langage de programmation exécutable.

Les tests unitaires permettent de vérifier que ces unités répondent à leurs spécifications.

d- Intégration et tests du système

On intègre les unités de programme et on réalise les tests globaux pour être sûr que les besoins logiciels ont été satisfaits. Le système est alors livré au client.

e- Mise en oeuvre et maintenance

C'est l'étape la plus longue du cycle de vie d'un logiciel. L'activité de maintenance consiste à corriger les erreurs qui n'ont pas été découverts lors des étapes antérieures du cycle de vie; et aussi à améliorer la réalisation des unités du système et à augmenter ses fonctionnalités au fur et à mesure que de nouveaux besoins apparaissent.

Cette étape est très intéressante car le but de ce chapitre est de décrire les activités de la maintenance et sa place dans le cycle de vie d'un logiciel.

1-3 Généralité sur la maintenance d'un logiciel

Historiquement, le terme « maintenance »a été appliqué au processus de modification d'un logiciel après sa livraison.

Ces modifications peuvent être de simples changements destinés à corriger des erreurs de programmation, des changements plus profonds destinés à corriger des erreurs de conception, ou des modifications et extensions fondamentales qui correspondent à un changement dans la définition des besoins [2].

a- Les différents types de maintenance

Cependant, on continue d'utiliser le terme « maintenance » pour désigner toute modification à un programme destinée à corriger des erreurs ou à offrir de nouvelles fonctionnalités.

Dans la maintenance d'un logiciel, on peut distinguer :

- La maintenance adaptative,
- La maintenance corrective et
- La maintenance perfective.

***(i)* Maintenance adaptative**

Ce type de maintenance consiste à adapter le logiciel aux changements de son environnement c'est à dire, on adapte le logiciel au nouvel environnement technique. Cela est indispensable lorsque pendant la période d'utilisation, on se déplace vers un autre système d'exploitation.

(ii) La maintenance corrective

La maintenance corrective consiste à corriger des erreurs qui ont pu être faits pendant la conception et la réalisation du logiciel.

Plusieurs types d'erreurs peuvent être commises : mauvaise conception du logiciel, l'erreur des calculs, etc.

Ces erreurs sont détectées par ceux qui exploitent le logiciel ou par ceux qui assurent la maintenance.

(iii) La maintenance perfective

La maintenance perfective consiste à effectuer les changements demandés par les utilisateurs ou par les programmeurs du système, l'ajout de module pour les besoins des utilisateurs.

Une analyse effectuée par Lientz et Swanson [2] établit une répartition des diverses formes de maintenance :

- ✓ 65% de la maintenance perfective
- ✓ 18% de la maintenance adaptative
- ✓ 17% de la maintenance corrective

b- Le cycle de maintenance

Il y a plusieurs cycles de maintenance, qui commencent toujours par l'établissement de la liste des priorités de réparation et par la désignation des équipes de maintenance.

A la fin, ils aboutissent à une version du système corrigée (pour la maintenance corrective) ou évoluée (pour la maintenance perfective) ou compatible à un environnement (pour la maintenance adaptative).

(i) **Le modèle « maintenance urgente »**

C'est le modèle que nous avons utilisé pour maintenir le logiciel ACCEAO.

Ce modèle de maintenance est représenté à la Fig.2.

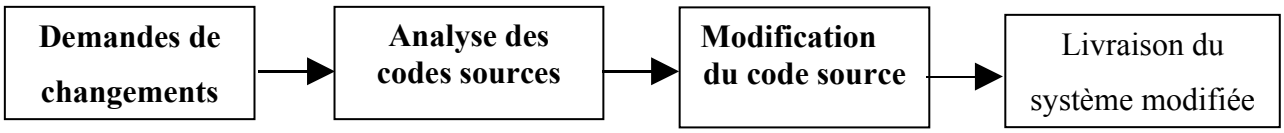

Figure 1.2 : Modèle de maintenance urgente d'un logiciel

Demandes de changements : Dans cette étape, on met clairement les fonctionnalités à corriger et les nouveaux besoins à ajouter.

Analyse des codes sources : Pour pouvoir trouver facilement les principales erreurs à corriger ou les besoins à ajouter, cette phase consiste à détecter l'architecture globale du logiciel, et les principaux algorithmes utilisés.

Modifications des codes sources : C'est la modification du programme proprement dit pour avoir les nouvelles fonctionnalités qu'on veut obtenir à partir de l'analyse des codes sources.

Livraison du système modifiée : Cette phase consiste à valider le nouveau produit. Une nouvelle période d'utilisation du logiciel recommence.

L'avantage de ce modèle est la rapidité de l'opération, et l'absence de règles ou de normes à respecter.

***(ii)* Autres modèles de maintenance**

Les grands systèmes se diffèrent par leur complexité. A titre d'exemple, on peut citer le système d'exploitation.

Les modèles de maintenance suivants sont beaucoup plus précis (formels) et plus détaillés par rapport au précédent. Ils se prêtent aux grands systèmes logiciels.

Ce sont :

- le modèle de TAUTE,
- le modèle IEEE.

- Modèle de TAUTE [2]

Ce type de modèle de maintenance est représenté à la Fig. 1.3. Les différentes significations des différentes étapes du modèle de TAUTE sont les suivants :

Demandes de changements : Les demandes qui sont de la part des développeurs du logiciel ou des utilisateurs.

Estimation : Cette étape consiste à estimer les dépenses ou les coûts de maintenance.

Planification : C'est la définition de la version du produit après la maintenance c'est à dire les fonctionnalités à changer, à modifier ou à ajouter.

Programmation : Pour traduire en langage de programmation les changements et les améliorations qu'on avait fait.

Tests : Ces sont les tests de la nouvelle version du logiciel obtenu.

Documentation : Les documents servent de supports aux revues de validation du produit.

Acceptation : Les clients et l'équipe de maintenance valident ensemble le nouveau produit et trouve le résultat ensemble.

Opération : C'est l'utilisation ou l'exploitation de produit par les utilisateurs.

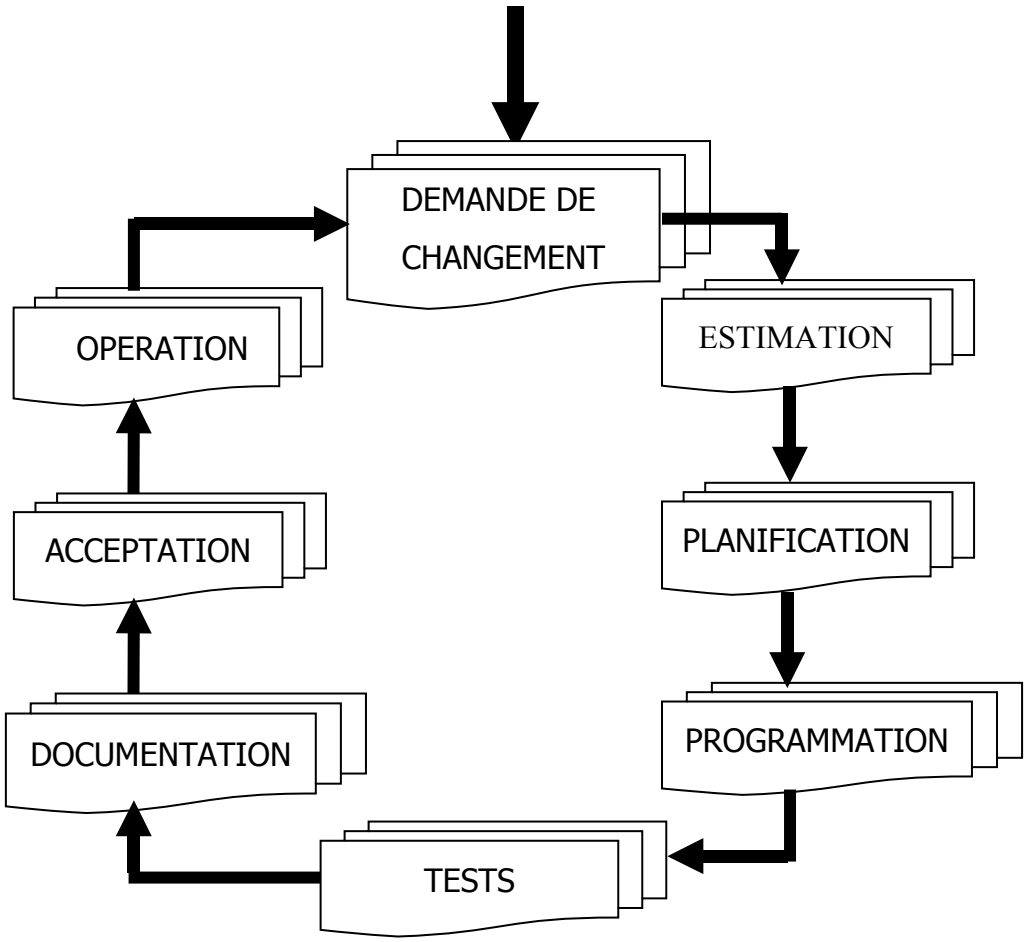

Figure 1.3 : Le modèle de TAUTE pour la maintenance d'un logiciel

- Le modèle IEEE *[2]*

Le modèle de IEEE comme montré à la Fig.1.4, est un modèle comparable au modèle de TAUTE. Les descriptions des différentes étapes sont les suivantes :

Classification : La nouvelle version du produit est définie par cette étape.

Analyse : Les nouvelles fonctionnalités à ajouter sont définies dans cette phase d'analyse.

Conception : Dans cette étape, on effectue la modification, l'amélioration et la correction des erreurs, ainsi que la construction d'un algorithme.

Implantation : Cette phase permet d'entrer dans la programmation proprement dite. Il existe deux types de programme à faire : la programmation des nouvelles fonctions selon les besoins et la correction de l'ancien programme qui contient d'erreur.

Acceptation : La réalisation et les tests unitaires sont faites dans cette partie, pour connaître la bonne fonctionnalité du logiciel.

Livraison : C'est la remise d'une nouvelle version et l'exploitation de ce nouvel produit.

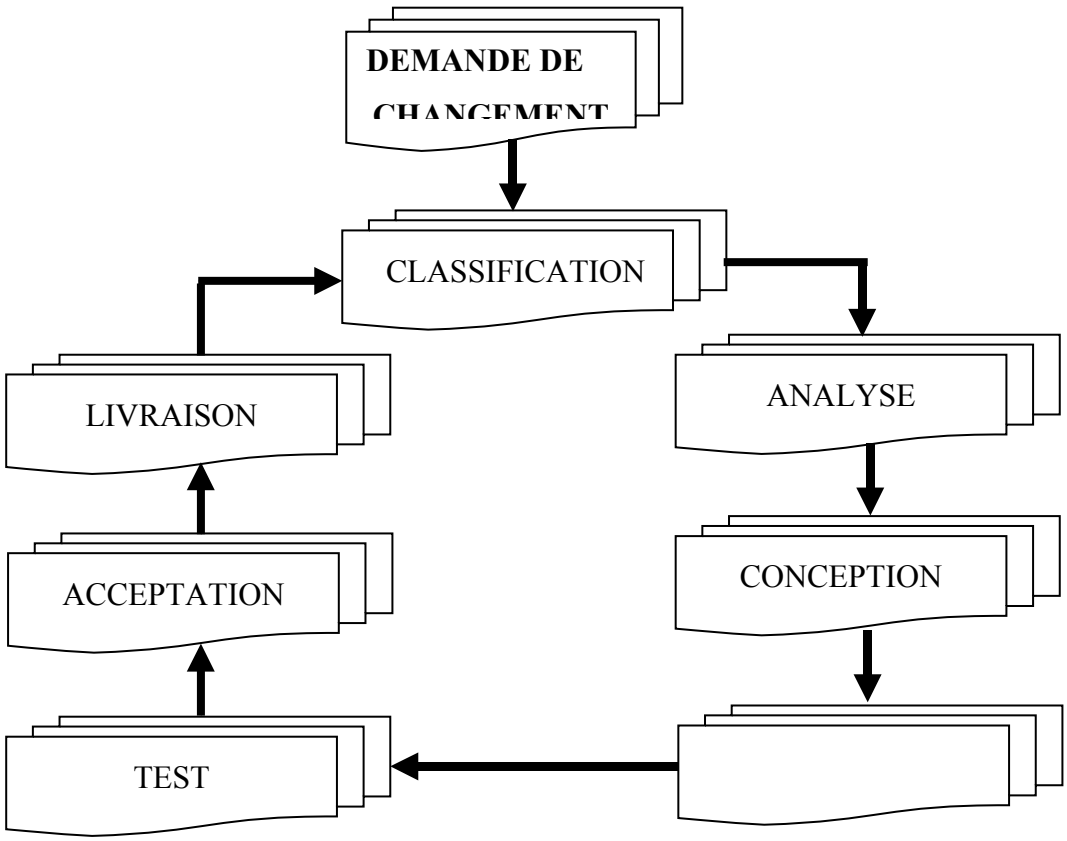

Figure1.4 : Le modèle IEEE

c- Les facteurs affectant les coûts de maintenance

Les coûts de maintenance peuvent être minimisés si l'on tient compte des besoins des programmeurs, des maintenances lors de la conception et de la réalisation du logiciel. L'estimation des coûts de maintenance est très délicate. Ces coûts dépendent d'un grand nombre de facteurs qui ne sont pas liés aux caractéristiques techniques du logiciel.

Les exemples qui suivent peuvent être cités.

- Le type d'application : Si l'application est nouvelle, les besoins des utilisateurs évolueront progressivement comme leurs expériences du système**.**

- **La stabilité du personnel** : Les coûts de maintenance sont plus faibles lorsque le programmeur initial du système est lui-même chargé de la maintenance.

- La durée de vie du système : Quelques systèmes encore en exploitation ont été écrits au début des années soixante. Il est certain que les coûts de maintenance ont tendance à augmenter avec l'âge des systèmes.

- La dépendance d'un programme par rapport à son environnement extérieur :

Citons par exemple l'effet d'un changement de la législation concernant les impôts et taxes sur les programmes « comptabilité et gestion des stocks ». De tels changements sont relativement courants.

- La stabilité du matériel : L'évolution constante des matériels conduit à modifier les logiciels de façon significative.

A côté de ces critères non techniques, il faudra également considérer d'autres critères pour évaluer les coûts de maintenance.

(i) La modularité :

Il doit être possible de modifier une unité sans remettre en cause une autre unité du système.

(ii) Le langage de programmation :

Il est généralement plus facile de maintenir un logiciel écrit dans un langage de haut niveau.

(iii) Le style de programmation :

La clarté d'un programme le rend beaucoup plus facile à maintenir.

(iv) la qualité des tests et de la validation :

Les coûts de maintenance diminuent avec le temps passé à valider un système. Les coûts de maintenance dûs à la correction des erreurs dépendent du type d'erreurs. Les erreurs de codage sont généralement beaucoup plus faciles à corriger que les erreurs de conception. De ce fait, la correction des erreurs de codage peut être faite à moindre coût. Par contre, la correction des erreurs de spécification, qui sont complexes, coûte plus cher.

(v) La qualité de documentation :

Etant donné la multiplicité des facteurs ci-dessus, il est quasi impossible d'établir un modèle d'évaluation des coûts de maintenance. On peut toutefois l'établir, pour une type bien connu d'application pour le quel on dispose d'informations concernant des projets passés.

CHAPITRE II :

IMPLANTATION D'UNE SOURIS ET CORRECTION DES ERREURS

Un travail d'implantation de souris a été commencé dans la version 4.2, mais la méthode utilisée a résulté au blocage du système après quelques utilisations et l'impossibilité d'utiliser le clavier dans certains cas. La correction des erreurs et l'adoption d'une autre méthode ont été nécessaires pour résoudre ces problèmes, afin d'avoir une bonne fonctionnement de la souris implantée dans le logiciel et pour rendre facile la manipulation du logiciel ACCEAO.

2-1 Les différences entre la version 4.2 et la version 4.3

Lorsqu'on parle d'une nouvelle version d'un logiciel, on pense directement à une version plus évoluée du logiciel considéré. La révision et l'amélioration d'une version peuvent être considérées comme évolution. Car dans ce cas, les différences entre deux versions successives sont inévitables.

a- Bref aperçu de la version 4.2

En tant que simulateur des circuits électroniques, l'ACCEAO version 4.2 conserve toutes les fonctionnalités des versions antérieures [3], telles que :

- Le traçage des diagrammes asymptotiques (de Bode),
- La possibilité de concevoir un amplificateur à transistors,
- L'analyse en régime continu,
- L'analyse en régime transitoire…

Ce logiciel possède aussi des avantages, en ne citant que sa possibilité de tourner même avec une machine bas de gamme (comme les IBM-PC, 286) [1].

b- Améliorations apportées par l'ACCEAO version 4.3

La version 4.3 résout les problèmes rencontrés à l'utilisation de la souris cités plus haut en exploitant une méthode complètement différente et effectuée les modifications y afférentes pour les besoins des utilisateurs et des concepteurs. Une nouvelle interface graphique qui ne perturbe ni la partie analyse ni la conception permet :

- d'accéder aux menus et sous-menus,
- de saisir les valeurs caractéristiques d'un composant,
- de déplacer les composants sélectionnés,
- d'accéder dans le digramme de Bode,
- de parcourir les fichiers d'aide,

à l'aide de la souris sans perturber le fonctionnement du clavier.

2-2 Implantation de la souris dans le logiciel ACCEAO sous DOS

Lorsque dans un logiciel donné, on peut faire usage d'une souris, la manipulation de certaines actions du programme devient plus pratique et plus confortable. C'est une des motivations de la version 4.2, et reprise par la version 4.3.

a- Principe

Il a été utile et nécessaire d'insérer le nouvel fichier **souris.cpp** (ANNEXE) dans le programme. Ce fichier contient toutes les fonctions qui gèrent les événements et les mouvements de la souris. Toutes les fonctions se rapportant à la souris sont écrites en assembleur et utilisent l'**interruption 33h** [5][6].

Pour invoquer cette **interruption**, la fonction **geninterrupt ()** [5][6] est exploitée.

La plupart des fonctions de souris travaillent avec les registres AX, BX, CX, DX pour traiter les informations provenant du programme d'appel et lui retourner les résultats nécessaires dans ces mêmes registres. Le registre AX sert à retourner l'état de la fonction qui indique un échec de l'opération en cas d'erreur et transmis le numéro de fonction. Dans le mode de fonctionnement de la souris, l'état de ces boutons et la position du pointeur jouent un rôle important.

Voici quelques fonctions se rapportant à la souris utilisées pendant le développement du logiciel ACCEAO [6].

(i) **Fonction 00h : Réinitialiser le driver de souris**

Cette fonction initialise le driver de souris et enclenche ainsi le test de la souris. ACCEAO 4.3 utilise la version 8.20 du driver **mouse.com** du Microsoft.

Paramètre d'entrée : AX=00h

Paramètre de sortie : si AX # 0 dans ce cas, un driver de souris est installé et BX contient le nombre de bouton de la souris; dans le cas contraire c'est l'erreur, c'est-à-dire aucun driver de souris n'est installé.

(ii) **Fonction 01h Afficher le curseur**

Après appel de cette fonction, le curseur de la souris devient visible sur l'écran et suit désormais le déplacement de la souris.

Paramètre d'entrée : AX=01h

Paramètre de sortie : aucune.

(iii) **Fonction 02h Masquer le curseur**

Cette fonction élimine le curseur de la souris de l'écran.

Paramètre d'entrée : AX=02h

Paramètre de sortie : aucune

(iv) **Fonction 03h : Lire l'état de la souris**

L'appel de cette fonction fournit la position actuelle de la souris et l'état des différents boutons de la souris. Elle permet de savoir si un bouton a été enfoncé.

Paramètre d'entrée : AX=03h

Paramètres de sorties : BX = Etat des boutons de la souris

CX = Position horizontale de la souris

DX = Position verticale de la souris

(v) **Fonction 04h : Déplacer le curseur**

Déplace le curseur de la souris vers une position déterminée de l'écran, à moins qu'il n'ait été caché avec la fonction 02h.

Paramètres d'entrées : AX = 04h

CX = Position horizontale de la souris

DX = Position verticale de la souris

Paramètre de sortie : Aucune

(vi) **Fonction 07h : Définir la zone de déplacement horizontale**

Définit la zone horizontale à l'intérieur de la quelle le curseur de la souris peut se déplacer. L'utilisateur n'aura aucune possibilité de faire sortir le curseur de la souris de la zone ainsi fixée.

Paramètres d'entrées : AX = 07h

CX = Position horizontale minimale de la souris

DX = Position horizontale maximale de la souris

Paramètre de sortie : Aucune

(vii) **Fonction 08h : Définir la zone de déplacement verticale**

Définit la zone verticale à l'intérieur de laquelle le curseur de la souris peut se déplacer. L'utilisateur n'aura aucune possibilité de faire sortir le curseur de la souris de la zone ainsi fixée.

Paramètres d'entrées : AX = 08h

CX = Position verticale minimale de la souris

DX = Position verticale maximale de la souris

Paramètre de sortie : Aucune

(viii) **Fonction 09h : Définir le curseur de la souris en mode graphique**

En mode graphique, un tableau de bits définit l'apparence du curseur de la souris, en fixant de quelle manière les points sous le curseur de la souris et les points du curseur de la souris doivent être combinés. Cette fonction sert à définir ce tableau de bits, et donc l'apparence du curseur de la souris.

Paramètres d'entrées : AX = 09h

BX = Distance du point de référence au bord gauche du tableau de bits

CX = Distance du point de référence au bord supérieur du tableau de bits

ES : DX = Adresses de segment et d'offset du tableau de bits dans la mémoire

Paramètre de sortie : Aucune

Ces fonctions sont toutes utilisées pour prévoir la gestion de souris et sa mise en place dans cette dernière version 4.3 d'ACCEAO.

b- Méthodes

Devant le dysfonctionnement rencontré à la version 4.2 à cause de la dispersion des codes, il a fallu revenir à la base, notamment à la version 4.0 pour implanter la souris.

(i) Technique de conception

Dans la version 4.3, le principe est basé sur une structure. Au lieu de créer des différentes classes pour les menus, les menus déroulant et les barres de menus, il est intéressant d'exploiter les structures existantes dans la version 4.0. Les instructions concernant un objet graphique (**menus**) sont tous stockées dans cette structure. Ainsi, le pointeur de la souris peut détecter un objet et peut lancer les applications que les utilisateurs décident d'exécuter. Il est nécessaire de définir la structure existante pour bien comprendre le fonctionnement de l'interface :

- **struct str_item**

La structure **struct str_item** est représentée au tableau de la *figure 2.1*

str_item	
champs	# name[ ] : chaîne de caractère # surbr, nbitemsub, parent, adrbar : chaîne non signé # a, b, c, d : entier
variable	item [constante]

Figure 2.1 : Tableau représentant la structure **str_item**

- **name[]** est utilisé pour stocker le nom d'un menu,
- **surbr** donne le numéro d'une lettre stockée dans « name[] »pour avoir un raccourcis clavier,
- **nbitemsub** donne les nombres des sous menus,
- **parent** est utilisé pour déterminer les menus parents,
- **adrbar** indique l'ordre d'apparition des sous menus,
- **a, b, c, d** détermine les coordonnées d'un menu ainsi défini,
- **item [constante]** est une variable de type **str_item.** Elle peut être exploité par n'importe quelles fonctions qui utilisent les champs de la structure **str_item.** La constante indique les nombres des objets (**menus et sous menus**) que l'utilisateur peut manipuler dans l'interface utilisateur.

Exemple d'utilisation de cette structure :

```
if (pointer_in_box (item[1] .a, item[1] .b, item[1] .c, item[1] .d)&&left_click())
    {
        instruction;
    }
```

pointer_in_box (int, int, int, int) et left_click() : ce sont des fonctions définies dans le fichier **« souris.cpp »** et utilisent le fonction **3h**.

item[1].a (constante=1) : ceci indique l'abscisse en pixel du coin supérieur gauche du menu **« File »**.

(ii) Gestion d'événement de la souris et du clavier

Un événement est déclenché quand il y a un changement d'état de l'écran, de la souris ou du clavier.

Le déclenchement des événements dépend des instructions (en nombre entier) retournées par des fonctions. Il y a deux types d'instructions :

- le premier est obtenu lorsque l'utilisateur exécute une commande à partir

du menu (soit par la souris, soit par le clavier). Les instructions sont traitées directement par une fonction principale appelée « **entrer( )** ».

- le second est obtenu lorsque l'utilisateur exécute une commande en combinant

quelques touches du clavier. Dans ce cas, les instructions sont traitées par d'autres fonctions spéciales avant d'être exécutées par la fonction principale « **entrer()** ».

Le programme utilise des boucles imbriquées et des boucles infinies (Fig. 2.2). Ceci

est fait pour bien gérer les événements et pour qu'il y a une correspondance entre la manipulation de la souris et le clavier.

La figure.2.2 montre un organigramme qui permet de gérer la souris et le clavier dans le logiciel ACCEAO.

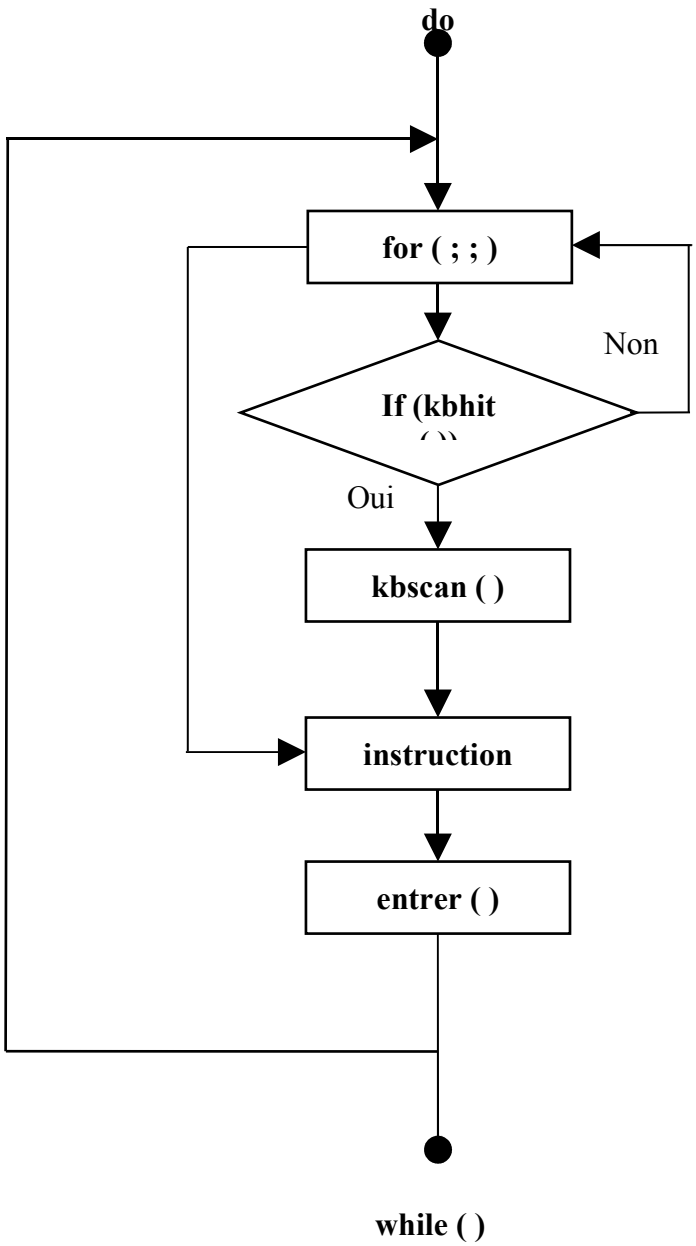

Figure 2.2 : Organigramme pour la gestion de la souris et du clavier dans ACCEAO

Explication de l'Organigramme (Fig.2.2)

do…while ( ) : c'est le boucle principal du programme.

for(; ;) : une boucle infinie qui gère toutes les fonctionnalités de la souris. Tant que l'état de la souris ou du clavier ne change pas, le programme boucle et n'effectue aucune application. Une fois qu'il y a un changement d'état, le programme sort de la boucle et donne une instruction à traiter.

- l'état de la souris peut être changé par des clics, par des mouvements, ou par l'endroit où se trouve le pointeur,… . Ces actions permettent au programme de sortir dans la boucle « **for(; ;)** ». Puis, l'instruction est traitée par la fonction « **entrer ( )** ».
- la validation d'une touche entraîne un autre événement. C'est la fonction « **kbhit ( )** » qui la détecte. Si la valeur retournée par « **kbhit ( )** » équivaut à **1** (clavier appuyé), le programme sort de la boucle « **for (; ;)** » et entre dans une autre fonction « **kbscan ( )** ». Par contre, si la valeur est équivaux à **0**, le programme reste bouclé dans la boucle « **for (; ;)** ».

kbhit ( ) : c'est une fonction définie dans le logiciel de développement. Si une touche est tapée, la fonction retourne une valeur équivaut à 1, si non la valeur de retour sera 0.

kbscan ( ) : c'est dans cette fonction qu'on traite les données du clavier. Il existe deux façons pour accéder aux menus et sous menus à l'aide du clavier :

- l'utilisation des touches de direction et la touche **Entrée** sur le clavier ;
- l'utilisation des raccourcis clavier.

Ces deux méthodes utilisent la même fonction **_scan (expression1, expression2).** Mais au lieu de manipuler les touches de direction pour entrer dans un menu ou sous menu, le raccourci clavier facilite le travail par une seule touche ou des touches combinées. Ce qui est plus rapide.

instruction : c'est une donnée à propos d'un menu. Le choix de ce menu est obtenue par l'événement d'un souris (des clics) ou du clavier (validation d'une touche).

entrer ( ) **:** c'est une fonction qui traite les instructions données. C'est aussi dans cette fonction qu'on trouve tous les liens pour effectuer les calculs.

L'utilisation de toutes ces fonctions est mise en évidence dans le programme qui suit.

Echantillon d'un programme de la boucle principale

```
void main(int countCmdline, char *lpCmdline[])

{
 ……..

do
{
 for ( ; ; )
  {
        if (pointer_in_box (item[1] .a, item[1] .b, item[1] .c, item[1] .d)&&left_click())
  /*  tester si la  souris se trouve à l'intérieure du cadre a, b, c, d et le bouton gauche est enfoncé  */
    {
          Instruction ; /* C'est une instruction effectuée par le programme*/
          entrer () ; /* Pour traiter les instructions données.*/
        }

        if ( kbhit())
          break;
  }

 kbscan (c1,c2); /*c1: code ASCII de la touche saisie; c2: code d'une touche de fonction.*/

   if (c1==0)
    {
      if (c2==61)
       {
        numitem =df1_load;   /* numéros du sous menu Load */
        c1=13; // touche entrée
        c2=0;
        gparent =1;
        }
       ……..
     }
      if ((c1==13)&&(c2==0))
       entrer();
  }

while ((c1!=0)||(c2!=16));

 ……..
}
```

CHAPITRE III :

GUIDE D'UTILISATION DU LOGICIEL ACCEAO

Ce chapitre est destiné à aider l'étudiant lors de l'utilisation du logiciel ACCEAO version 4.3. En effet, dans cette version, on a la possibilité d'utiliser la souris et/ou le clavier. Ce choix ne dépendant que l'utilisateur.

3.1 Les menus

Les menus simplifient l'utilisation du programme. Dans ce programme, on a cinq menus distincts tels que ***File, Tools, Analysis, Design, Help***. On peut consulter chaque menu en utilisant la souris et/ou les touches de direction du clavier. De même pour les sous menus.

a- Le menu *File*

Le menu ***File*** est une fonction de création, de chargement, d'enregistrement et de fermeture. Il renferme quatre sous-menus, à savoir : ***New, Load, Save, Quit***.

(i) Le sous-menu *New*

On utilise le sous-menu *New* lorsqu'on veut créer un nouveau fichier. Pour ce faire, on clique sur le menu *File* (ou appuyer sur Alt+F), puis sur *New* (ou appuyer sur F2).
Une feuille de dessin vierge apparaît sur la fenêtre et l'on peut saisir le schéma électronique (Fig. 3.1).

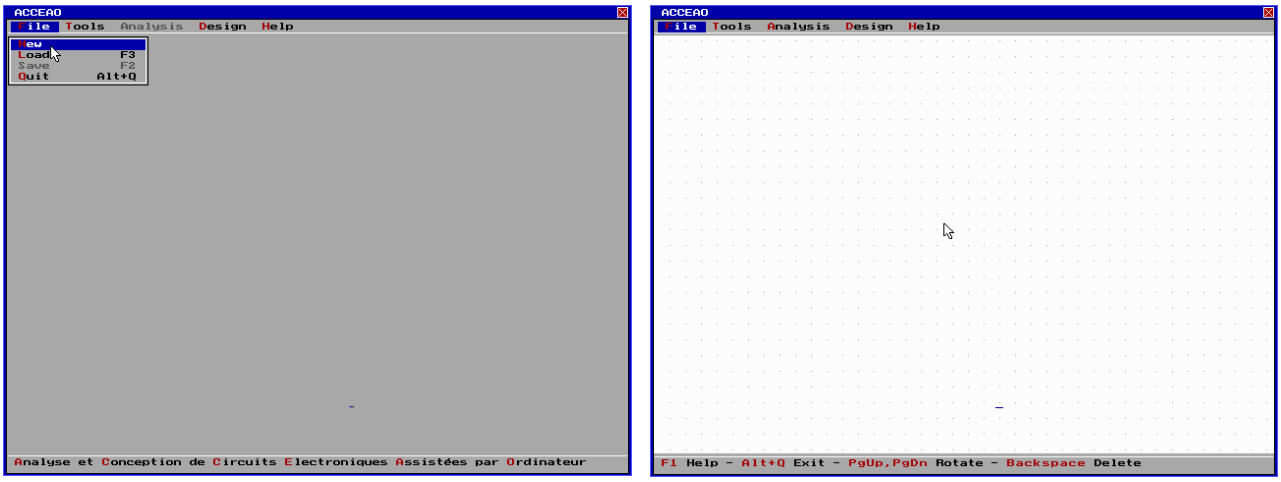

Figure 3.1 : Pour avoir une nouvelle fenêtre de saisie

(ii) **Le sous-menu *Load***

- Pour accéder au sous-menu *Load* :
  - Cliquer sur le menu *File* (ou appuyer sur Alt + f),
  - Puis cliquer sur *Load* (ou appuyer sur **l** ou **F3**),

La boite de dialogue contenant le nom des circuits qui sont déjà enregistrés dans un fichier, apparaît sur l'écran.

- Pour annuler, cliquer le bouton droit (ou appuyer sur Echap).
- Pour charger ce fichier :
  - Cliquer le bouton gauche sur un nom choisi, (ou utiliser les touches UP ARROW ou DOWN ARROW, puis appuyer sur ENTREE),
  - Le circuit choisi apparaît sur l'écran, (Fig 3.2).

Si le fichier se trouve dans un autre lecteur, appuyer sur ALT + A ou B ou C,…,

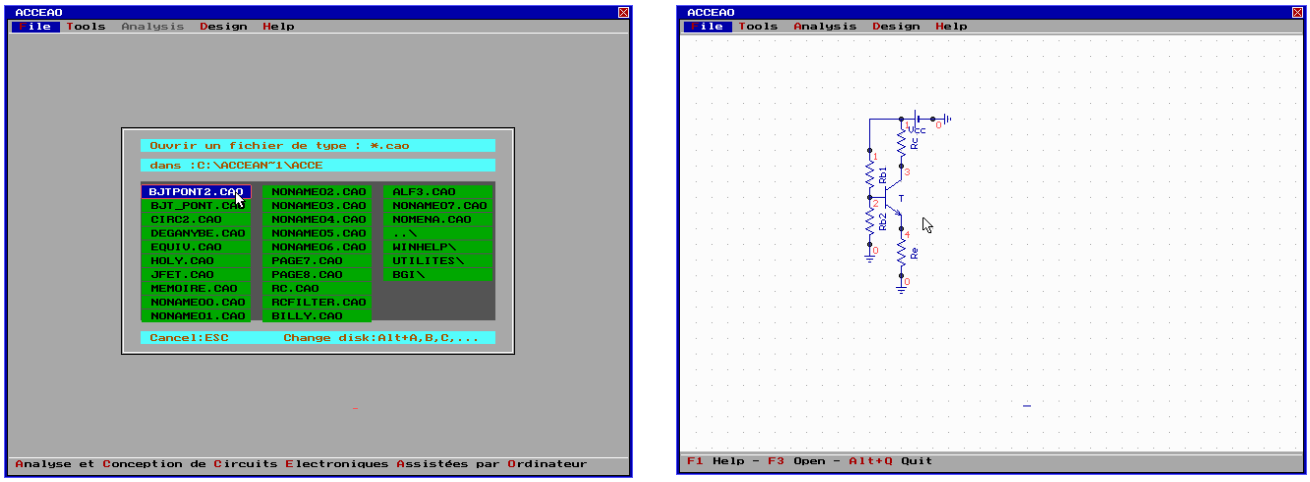

Figure 3.2 : Chargement d'un circuit sur l'écran

(iii) **Le sous-menu *Save***

Le sous-menu *Save* est utilisé pour enregistrer dans un fichier le travail effectué :

- Choisir le menu *File,*
- Cliquer à gauche sur *Save (*ou appuyer sur **F4**),

Une boite invitant l'utilisateur à donner un nom de fichier apparaît sur l'écran.

- Cliquer sur OK pour enregistrer (ou taper sur ENTRER),
- Cliquer ANNULER pour annuler (ou appuyer sur **Echap**).

(iv) **Le sous-menu *Quit***

Le sous-menu ***Quit*** est utilisé pour quitter le programme, pour ce faire :

- Choisir le menu ***File***,
- Cliquer sur *Quit* (ou appuyer sur Alt + Q)

b- **Le menu *Tools***

Le menu ***Tools*** contient toutes les fonctions se rapportant aux différentes manipulations des composants. Il dispose de six sous-menus : ***device, edit, model, library, pspice, et source.***

(i) **le sous-menu *device***

Ce sous-menu contient une liste des divers éléments du circuit tels que : la résistance, la capacité, l'inductance, les diodes, les amplificateurs opérationnels, le BJT,…, et permet aussi de les charger à l'écran (Fig 3.3) :

- Cliquer sur le menu ***Tools*** (ou appuyer sur Alt + T),
- Puis cliquer sur ***device*** (ou appuyer sur d),
- Choisir le type de composant sur la liste proposée, et cliquer (ou valider),
- Remplir le champ de saisie afficher à l'écran, et cliquer sur OK (ou taper sur ENTREE), ou bien appuyer sur le bouton droit de la souris pour l'annuler (ou taper sur Echap),

Le composant choisi est affiché sur l'écran ;

- Placer le composant à l'endroit choisi en appuyant en permanence le bouton gauche de la souris (ou utiliser les touches des directions du clavier),
- Taper sur ENTREE pour valider.

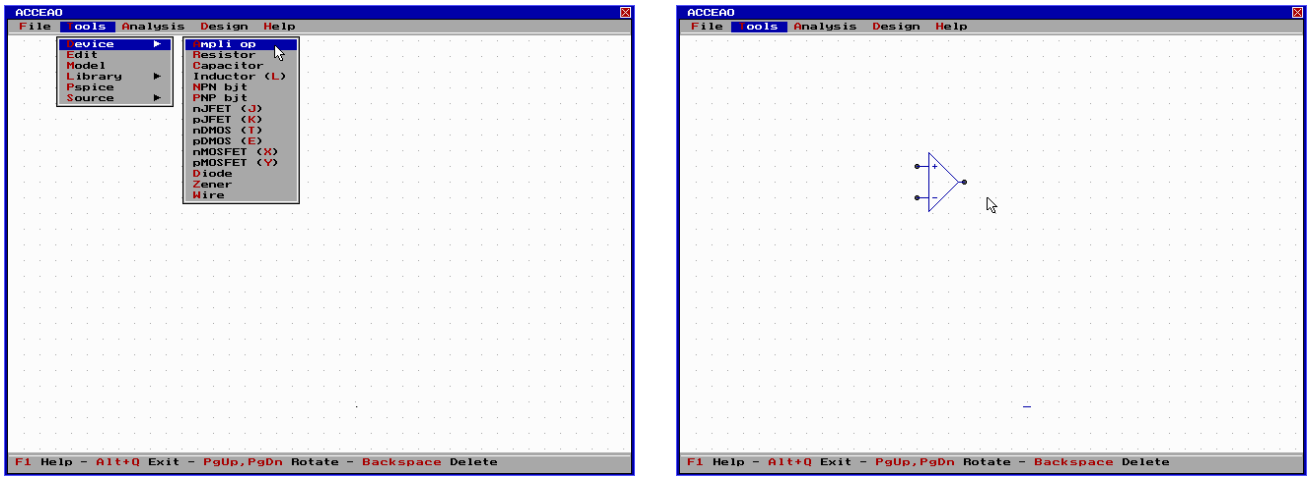

Figure 3.3 : Ajout d'un composant à partir du menu ***Device***

(ii) **le sous-menu *Edit***

Pour pouvoir changer les valeurs d'un composant :

- Choisir le menu Tools (ou appuyer sur Alt + T), puis Edit,
- Un composant est sélectionné par défaut (il clignote), et les autres composants peuvent être sélectionnés en utilisant les touches + ou -,
- La boite contenant les anciennes valeurs s'affiche sur l'écran en appuyant sur ENTRER (fig 3.4),
- On peut changer les valeurs des différents paramètres du composant,
- Appuyer sur Echap pour annuler l'action.

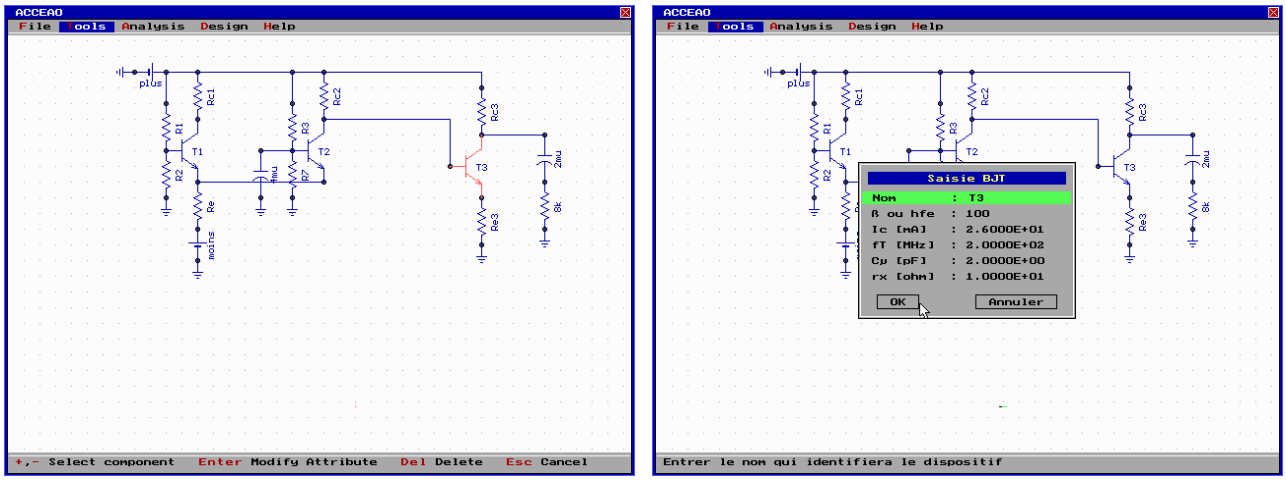

Figure 3.4 : Exemple d'utilisation du sous-menu ***Edit***

(iii) **Le sous-menu *Library***

C'est une bibliothèque contenant des types des composants comme Diodes, BJT, et FET qui permettent de charger sur l'écran un ou plusieurs composants choisis. Pour y accéder :

- Choisir le menu ***Tools***, puis ***Library,***
- En suite, choisir le type de l'opération à effectuer,
- Des opérations de consultation ou d'addition sont disponibles (fig 3.5),
- Appuyer le bouton droit de la souris pour annuler l'action (ou taper **Echap**).

La souris reste toujours disponible pour toutes autres actions, même utilisant le clavier.

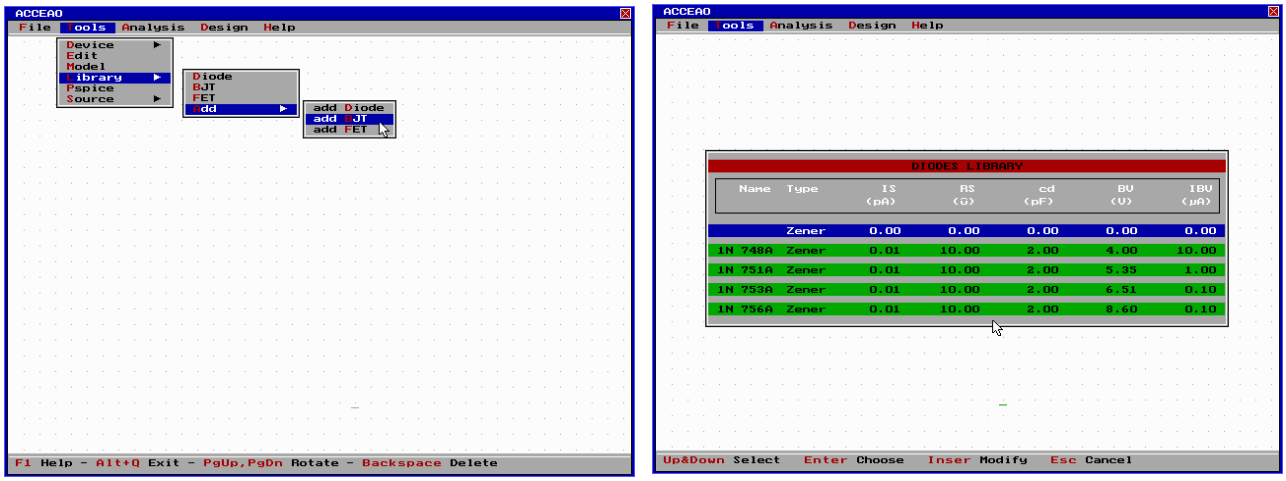

Figure 3.5: Accès dans la liste des composants dans la Bibliothèque

(iv) **Le sous-menu *source***

La source de courant, de tension, la masse,…, y sont affichées lorsqu'on choisi le sous-menu ***source*** (Fig.3.6), pour y accéder :

- Choisir le menu ***Tools,*** cliquer le bouton gauche de la souris sur ***Source*** (ou appuyer sur *S*),
- La liste des différentes sources s'affiche sur l'écran,
- Puis cliquer (ou taper sur ENTREE) convenablement sur le type de source choisie,
- Le déplacement peut se faire en appuyant les touches de direction du clavier ou en cliquant en permanence le bouton gauche de la souris.

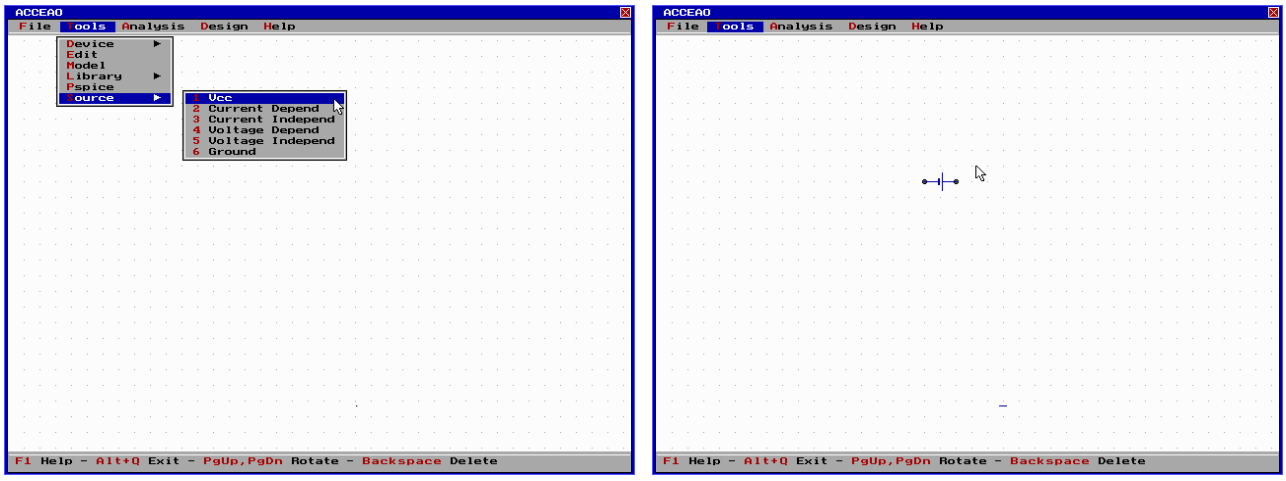

Figure 3.6 : Exemple d'une source choisie

c- Le menu *Analysis*

Ce menu contient tous les sous-menus permettant d'effectuer tous les calculs, d'une part, de donner les résultats de l'analyse d'un circuit d'autre part, enfin d'afficher le numéros de chaque nœud du circuit (Fig 3.7).

Plusieurs types d'analyses et de calculs sont disponibles :

- analyse en régime continu (***DC***),
- en régime transitoire (***Transient***),
- en régime variable (***AC***),
- calcul des fréquences d'oscillation propre d'un circuit (***Frequency***).

Pour accéder à ce sous-menu :

- D'abord charger ou tracer le circuit,
- Puis, entrer dans le menu ***Analysis***,
- Ensuite choisir le type d'analyse ou l'action à effectuer et valider.

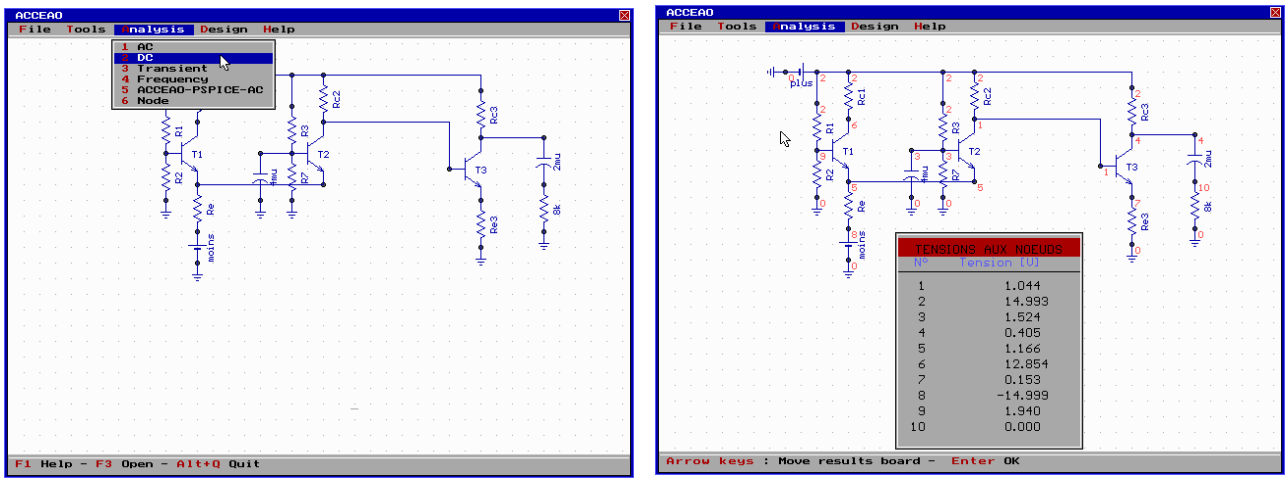

Figure 3.7 : Exemple d'utilisation d'une type d'analyse et les résultats

d- Le menu *Design*

Le menu ***Design*** contient les fonctions de conception et de traçage de diagramme de Bode. Il renferme deux sous-menus : ***Qp Triangle*** et ***Bode plots***.

(i) **le sous-menu *Qp triangle***

Le sous-menu ***Qp Triangle*** permet de concevoir un circuit simple à base de transistor

tels que BJT, MOSFET, JFET, CMOS (Fig 3.8). Pour ce faire :

- Entrer dans le menu Design, puis choisir ***Qp Triangle*** et valider en appuyant sur le bouton gauche de la souris ou taper sur ENTREE,
- Puis choisir le type de transistor et choisir le type de polarisation,
- Entrer les paramètres du circuit lorsque la boîte invitant à les saisir apparaît sur l'écran,
- Pour valider, cliquer sur **OK** ou taper sur ENTREE.

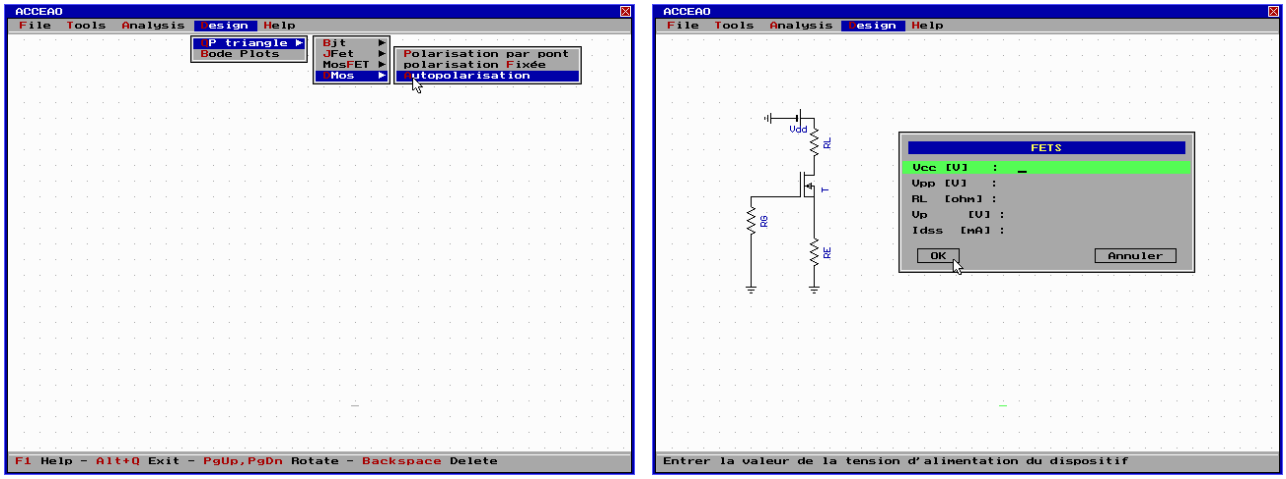

Figure 3.8 : Utilisation du sous-menu ***Qp Triangle*** et exemple d'un circuit à concevoir

(ii) **Le sous-menu *Bode Plots***

Pour pouvoir tracer le diagramme de Bode d'un circuit, on utilise le sous-menu *Blode Plots* (Fig 3.9) :

- Tout d'abord, il faut charger le circuit,
- Puis entrer dans le menu Design et choisir le sous-menu Bode Plots,
- Saisir les nœuds d'entrées et de sorties affichés dans une boite de dialogue sur l'écran, et valider,
- Remplir le boite de dialogue permettant de saisir l'intervalle de calcul, et valider aussi.

- Le schéma qui présente le diagramme de Bode est apparu dans une nouvelle fenêtre,
- Enfin, pour retourner à la fenêtre précédente, appuyer sur **Echap**.

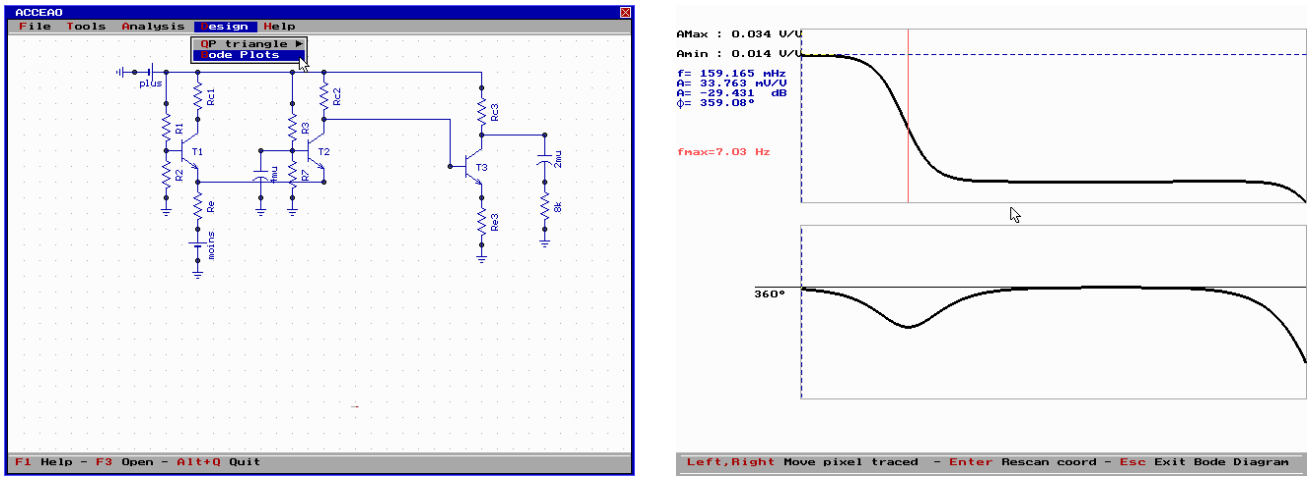

Figure 3.9 : Utilisation du sous-menu ***Bode Plots*** et un exemple d'un diagramme de Bode d'un circuit

e- Le menu *Help*

Le menu ***Help*** contient toutes les rubriques d'aides, où l'on peut trouver tous les renseignements concernant le logiciel ACCEAO et son utilisation. Il contient également deux sous-menus distincts : le sous-menu **C*ontents*** et le sous-menu ***About*** :

➢ Le sous-menu ***Contents*** contient une liste de titre d'aide, affichée dans une fenêtre. L'aide d'utilisation et quelques méthodes utilisées lors de la conception du logiciel sont affichés dans une nouvelle fenêtre en cliquant sur l'un de ces titres (Fig 3.10).

Pour entamer ce sous-menu :

- Entrer dans le menu ***Help***, puis ***Contents***,
- Choisir un titre dans la liste affichée sur la fenêtre,
- Cliquer sur le bouton gauche de la souris ou valider la touche ENTREE,
- Pour retourner à la page précédente, cliquer sur *Exit* (situé en bas – droit de la fenêtre en cour).

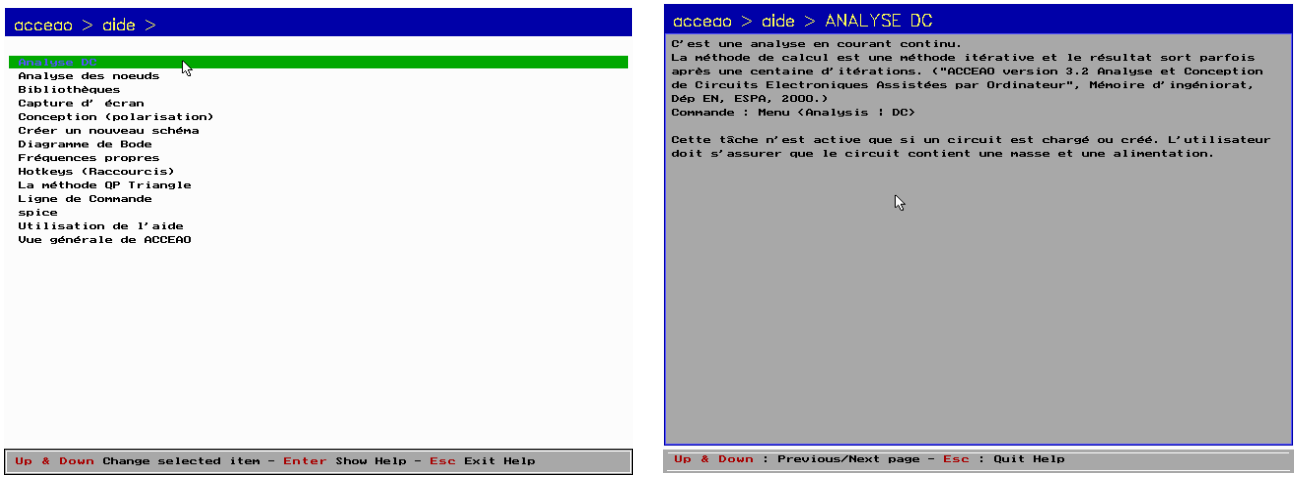

Figure3.10 : Les titres dans la liste d'aide et un exemple de ses contenus

- Les renseignements concernant le logiciel ACCEAO sont écrits dans le sous-menu ***About*** (Fig 3.11) :
  - Choisir le menu Help, puis cliquer sur ***About*** (ou valider ENTREE).

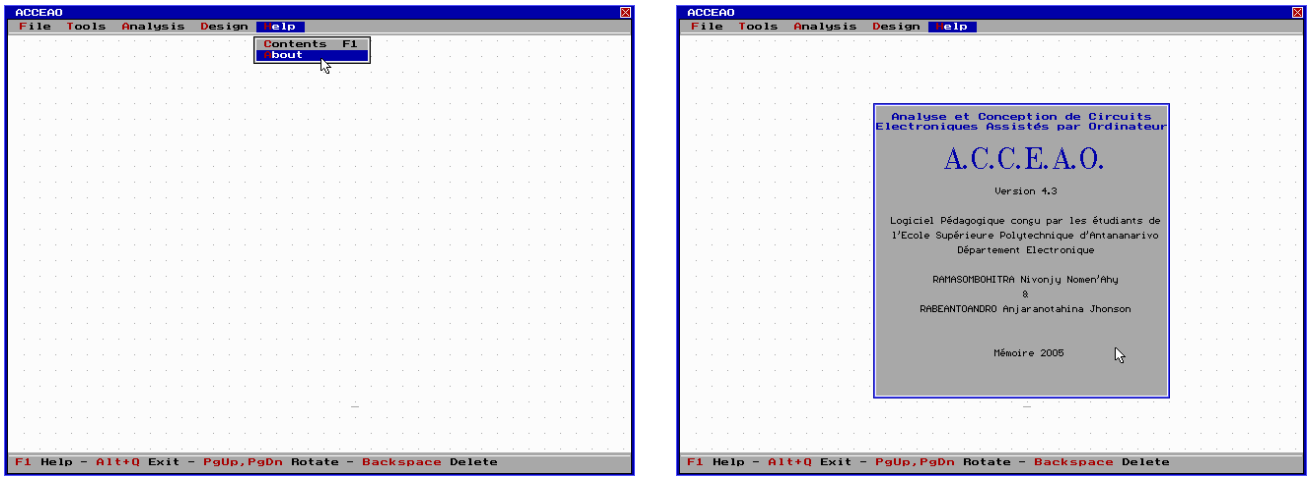

Figure 3.11 : Le sous-menu ***About*** et son contenu

3-2 Les autres fonctions

Un nouvel onglet rouge avec une croix blanche (en haut à droit) a été créé dans la barre d'état. Il s'agit de la fermeture du logiciel. Ainsi pour pouvoir quitter le logiciel ACCEAO, il suffit de cliquer le bouton gauche de la souris sur cet onglet (ou taper **Alt+Q**). Le bouton droit de la souris peut également réaliser certaines actions d'annulation comme ce qui pourrait être effectué par la touche **Echap** du clavier.

CONCLUSION

Une maintenance de ACCEAO a été assurée dans ce travail de mémoire. Dans cette dernière version 4.3 on a construit un fichier souris.cpp qui est un programme écrit en assembleur pour définir toutes les fonctions de la souris. Ensuite, ces fonctions ont été introduites dans les programmes et les structures déjà existant dans la série des versions 4.x précédentes.

L'accès aux menus, l'accès dans tous les champs de saisie du composant, le traçage des circuits, la sélection des composants et leur déplacement peuvent se faire à l'aide d'une souris sans aucun blocage du système. Ainsi, cette version plus élaborée facilite la manipulation du logiciel par l'utilisation de la souris en symbiose avec le clavier.

Les difficultés rencontrées pendant la réalisation de ce travail en ne citant seulement que la dispersion du code source nous ont permis d'apporter toutes les améliorations nécessaires pour ce logiciel. Mais des améliorations sont encore envisageables pour répondre aux besoins de l'utilisateur.

La considération des fonctions suivantes pourrait être considérée pour améliorer ACCEAO : la fonction de traitement des circuits numériques ou composants numériques, la fonction de traitement et de génération de macro modèles,…, ainsi que l'évènement « double clique »pour la souris.

ANNEXE

SOURIS.H

C'est un fichier d'entête qui déclare toutes les fonctions utilisées par le programme source **souris.cpp.**

```
#ifndef __MOUSE_H
#define __MOUSE_H
#define _ARROW     1
#define D       0
#define U       1

void initialize();
void show_pointer();
void hide_pointer();
void restrict_pointer(int x1, int y1, int x2, int y2);
void get_position(int & x, int & y);
int left_click();
int pointer_in_box(int, int, int, int);
void change_pointer(int Pointer);

#endif
```

SOURIS.CPP

Le programme ci -après développe certaines fonctions se rapportant à la souris notamment en ce qui concerne l'initialisation, le masquage, la position, les clics gauche et droit..… Il est à signalé que ce programme est développé en langage de programmation C sous MS-DOS, et est en même temps écrit en langage machine (Assembleur).

```
# include "souris.h"
# include <dos.h>
union REGS r, o;
struct SREGS s;
int cursor_arrow[32]={ ~ 0x3FFF, ~ 0x1FFF,~ 0x0FFF, ~ 0x07FF,
                       ~ 0x03FF, ~ 0x01FF, ~ 0x00FF, ~ 0x007F,
                       ~ 0x003F, ~ 0x001F, ~ 0x01FF, ~ 0x10FF,
                       ~ 0x30FF, ~ 0x787F, ~ 0xF87F, ~ 0xFC7F,
                        0x0000, 0x4000, 0x6000, 0x7000,
                        0x7800, 0x7C00, 0x7E00, 0x7F00,
                        0x7F80, 0x7C00, 0x6C00, 0x4600,
                        0x0600, 0x0300, 0x0300, 0x0000};

int cursor_full[32] ={~0xff00,~0xff00,~0xff00,~0xff00,
                      ~0xff00, ~0xff00, ~0xff00, ~0xff00,
                      ~0xff00, ~0xff00, ~0xff00, ~0xff00,
                      ~0x0000, ~0x0000, ~0x0000, ~0x0000,
                      0xff00, 0xff00, 0xff00, 0xff00,
                      0xff00, 0xff00, 0xff00, 0xff00,
                      0xff00, 0xff00, 0xff00, 0xff00,
                      0x0000, 0x0000, 0x0000, 0x0000};
void initialize()
{
        _AX=0;
        geninterrupt(0x33);
}
```

```
void show_pointer()
{
        _AX=1;
        geninterrupt(0x33);
}
void hide_pointer()
{
        _AX=2;
        geninterrupt(0x33);
}
void restrict_pointer(int x1,int y1,int x2,int y2)
{
        _AX=7;
        _CX=x1;
        _DX=x2;
        geninterrupt(0x33);
        _AX=8;
        _CX=y1;
        _DX=y2;
        geninterrupt(0x33);
}
void get_position(int & x, int &y)
{
        _AX=3;
        geninterrupt(0x33);
        x=_CX;
        y=_DX;
}
int left_click()
{
        _AX=3;
        geninterrupt(0x33);
        return(_BX & 1);
}
```

```
int pointer_in_box(int x1, int y1, int x2, int y2)
{
        int x, y;
        get_position(x,y);

        if(x1<=x && x<x2 && y1<=y && y<y2)
                return(1);
                return 0;
}
void change_pointer(int shape)
{
        int* ptr;
        r.x.ax=9;
        r.x.bx=0;
        r.x.cx=0;
        switch (shape)
        {
                case _ARROW:
                        ptr= cursor_arrow;
                        break;
                default:
                        ptr= cursor_full;
        }
        r.x.dx=(unsigned)ptr;
        segread(&s);
        s.es=s.ds;
        int86x(0x33,&r,&o,&s);
}
```

REFERENCES BIBLIOGRAPHIQUES

[1] RAZOARIHOLY Noroseheno et RANDIMBISOA Mahefa Vonjiniaina, « ACCEAO version 4.0 », Mémoire de fin d'études, Département Electronique, ESPA, 2002.

[2] ANDRIAMAMPIANINA Herinirina H et RATSIMBAZAFY Alfred C, « ACCEAO version 4.1 », Mémoire de fin d'études, Département Electronique, ESPA, 2003.

[3] RAMAROLAHY Tovohery et RADRIAMAMALARIVELO Andrison P « ACCEAO version 4.2 », Mémoire de fin d'études, Département Electronique, ESPA, 2004.

[4] Cours de « Génie Logiciel I », E 411, 4ème Année, Département Electronique, ESPA, 2003.

[5] Cours de « Programmation en langage machine », E412, 4ème Année, Département Electronique, ESPA, 2003.

[6] http://www.chez.com/antoche/souris.html.

[7] Cours de « Programmation en C », E 331 ,3ème Année, Département Electronique, ESPA, 2002.

[8] François TERRIER, « Borland C ++, volume 1 », concepts fondamentaux, SYBEX, 1992.

AUTEURS : Monsieur Nivonjy Nomen'Ahy RAMASOMBOHITRA *
Monsieur Anjaranotahina Jhonson RABEANTOANDRO**

TITRE : **« ACCEAO version 4.3 »**

Nombre de pages : 38
Nombre de figures : 17

RESUME

L'implantation de la souris a été rendue effective et efficace dans cette version 4.3 du logiciel ACCEAO ; grâce à l'intégration du fichier **souris.cpp** et à une restructuration de l'ensemble du logiciel.

Mots clés : ACCEAO, logiciel, maintenance, souris, interruption 33h, événement, structure.

Rapporteur : Monsieur Elisée RASTEFANO

Adresses des auteurs : *Lot AV 235 Ambohijanamasoandro Itaosy
Tana -102-
**Lot VT 85 NK bis A Andohanimandroseza
Tana -101-