République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Saad Dahlab, Blida
USDB.

Faculté des sciences.
Département informatique.

Mémoire pour l'obtention
d'un diplôme d'ingénieur d'état en informatique.
Option : Système d'information

Sujet :

Contrôle à distance des équipements Médicaux

Présenté par : SAAD SAOUD Hamza
BELHENNICHE Mustapha

Encadreur : Mr. ALLAM
Abdelkrim

Organisme d'accueil : Centre de Développement des Technologies Avancées (CDTA)

Soutenue le: date soutenance, devant le jury composé de :

Nom. président du jury, grade, organisme	**Président**
Nom examinateur 1, grade, organisme	**Examinateur**
Nom examinateur 2, grade, organisme	**Examinateur**

- Numéro/année promotion-

Remerciement

Tout d'abord, le vrai remerciement revient à dieu, le tout puissant qui nous a donné la force de réaliser ce travail.

Nous remercions notre promoteur M. ALLAM ABDELKRIM et M. ZERARGA LOTFY pour avoir bien voulu nous proposer ce sujet et pour leurs patience et soutient continus durant notre stage.

Nous remercions également tous les professeurs de l'USDB et CU-MEDEA qui nous ont accompagnés au long des nos études supérieures, ainsi qu'à tous nos enseignants et enseignantes d'étude primaire, moyenne et lycéenne.

Nous remercions tous ceux qui ont contribué à la réalisation de ce travail de près ou de loin.

Dédicace

Je dédie ce modeste mémoire :

A qui est très chère à mon coeur, ma merveilleuse mère pour son amour.

A qui est sacrifier sa vie pour nous, mon cher père.

A mes frères ZAKI, ALI, MOHAMED, AHMED, ABDERAHMAINE,
RACHID, YASSINE.

A toutes mes soeurs.

Et à toute ma grande famille sans exception.

A tout les gens qui m'ont aimé, à tous mes amis et collègues sans restriction.

A Mr ALLAM ABDELKRIM.

BELHENNICHE
.MUSTAPHA

Dédicace

A toute personne chère à mon cœur je dédie ce travail...

A ma merveilleuse mère pour son amour

A mon Chère père

A mes frères, Mohamed, Ahmed, Rachid

*A mes sœurs *Fatma, Merriem, Amina, Fatiha**

*A ^Hamida, fayza, Safia^*

A toute ma famille et mes amis

*A vous spécialement *Mustapha, Abderrazak,*

*Mohamed, Hamid & Aissa**

Saad Saoud hamza

Résumé

Ce travail s'insère dans le cadre du développement d'outils de contrôle des appareils médicaux sous réseau, conformément aux activités scientifique de l'équipe instrumentation virtuelle de la division (AS) Architecture des Systèmes des Centre de Développements des Technologie Avancé (CDTA).

Dans ce projet, nous nous intéressons au contrôle à distance des équipements médicaux a partir de n'importe quel poste de réseau (local ou étendu). Ce projet a pour but est de développer un système informatique sous Windows qui permet à l'utilisateur distant de contrôler des appareils médicaux, ces derniers sont connectes à la liaison série RS232 du poste serveur.

Abstract

This work has been concerned in the frame work of development of control tool of the medical apparatuses under network, which is a project of the virtual instrumentation team of division (ACE) System Architecture of the Advanced Technologies Development Centre (CDTA).

In this project, we are interested in remote control of the medical equipment has ire share of does not matter that network terminal (local or wide). The goal of this project is the development of an information processing system under Windows which makes it possible to a distant user control the medical apparatuses, that are related to serial RS232 of the PC server.

ملخص

إنّ هذا العمل يندرج في إطار تطوير وسيلة لتحكم في الآلات الطبية عبر شبكة الاعلام الآلي على مستوى مركز تطوير التكنولوجية المتقدمة.
في هذا المشروع، نهتم بالتحكم عن بعد في الآلات الطبية من أي مكان عبر الشبكة المحلية أو الأنترنات

الهدف من هذا الانجاز تطوير نظام معلوماتي على مستوى برنامج وندوس يسمح للمستعمل بمراقبة آلات الطبية هذه الأخيرة مرتبطة مع الرابطة التسلسلية للكومبيوتر.

SOMMAIRE

CHAPITRE 2 : CONTRÔLE ADISTANCE

CHAPITRE 3: CONCEPTION DE L'APLICATION LOGICIELLE

CHAPITRE 4: REALISATION DU SYSTEME

INTRODUCTION GENERALE

I. SITUATION DU PROBLEME

Aujourd'hui, l'utilisation de l'outil informatique devient de plus en plus importante dans la résolution des problèmes de nature complexe car il offre l'efficacité, la fiabilité et la rapidité dans le traitement, parmi les domaines qui exigent l'utilisation de cet outil, on trouve le domaine des architecture des systèmes et multimédia, télécommande télémédecine et communication ...etc.

Le contrôle d'une machine, d'un outil à distance rend souvent la vie de l'utilisateur facile, plus pratique, et pour des raisons de l'utilisation large, efficace de ces outils.

Les systèmes contrôlés par des réseaux informatiques sont un domaine en pleine effervescence au sien de la communauté scientifique vu l'apport de la nouvelle technologie (réseaux informatique, Internet) et les avancées en termes des performances des ordinateurs (processeurs, mémoires,...) facilitent le contrôle les équipements, telle que la télémanipulation, la télécommande...

Le travail que nous présentons dans ce mémoire s'inscrit dans le cadre de développement d'un système informatique ayant une architecture client/serveur pour le contrôle à distance (réseau) des appareils médicaux, les ordres physiques se feront à traverse la liaison série RS232 d'un ordinateur hôte, initié par l'équipe Instrumentation Virtuelle de la division (AS) Architecture des Systèmes du Centre de Développement des Technologie Avancé (CDTA).

Notre travail s'inscrit dans le cadre d'une plate-forme médicale sur réseaux, Il va falloir configurer le PC hôte en serveur en installant un serveur web de type APACHE. Dans ces conditions et ce qui suit un manipulateur lointain pourra envoyer des commandes vers les appareils médicaux à travers d'un interface (carte électronique "adaptateur") connectée au port série de cet ordinateurs. Une base de données doit être crier pour englober les utilisateurs ayant le droit de contrôler et conserver l'état des appareils contrôlés... ect.

On peut synthétiser l'objectif global de notre travail de la façon suivante :

<< Contrôle à distance des équipements médicaux sous réseaux >>

II. OBJECTIF DU TRAVAIL

Dans ce projet, nous nous intéressons au contrôle à distance des équipements médicaux liés au poste serveur à partir de n'importe quel poste de réseau (local ou étendu), tel que le poste serveur doit configurer en serveur (apache), (figure 1).

Le but de ce travail est le développement d'un système informatique sous Windows qui permet à l'utilisateur distant de contrôler (s'il a le droit) les appareils médicaux, et ces derniers sont connectés au PC serveur par la liaison RS232 a travers d'un interface (carte électronique ou adaptateur).

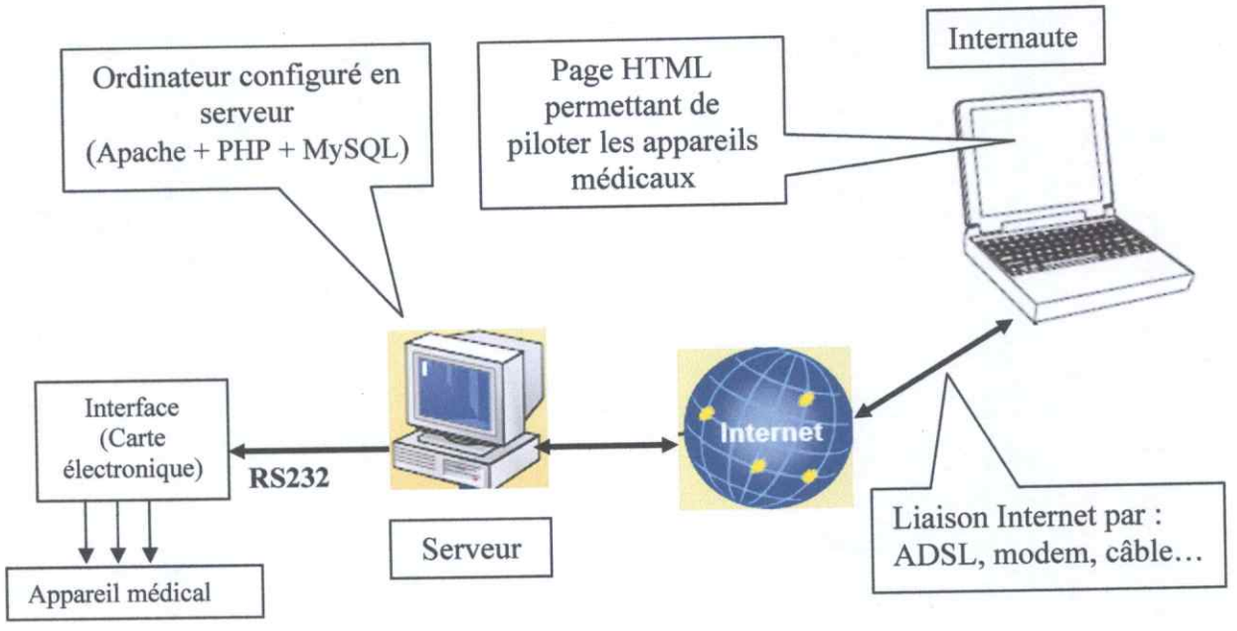


Figure 1 : *schéma générale du projet*

III. DESCRIPTION DU TRAVAIL

Le présent mémoire est composé de quatre chapitres :

Dans le premier chapitre, nous étudions l'environnement de développement, plus exactement le système d'exploitation (l'architecture, les interruption, les systèmes temps réel, la gestion des E/S (les pilotes et les contrôleurs des périphériques, l'architecture logique et physique de port série RS232).

Dans le deuxième chapitre, nous allons expliquer le principe du contrôle à distance ainsi que les différents outils informatiques se reportant à l'architecture réseau utilisée.

Dans le troisième chapitre, nous présentons la modalisation de notre application en utilisant le langage de modélisation UML.

Le chapitre quatre est réservé à l'étape implémentation et réalisation de notre application, enfin test de validation du travail réalisé.

Enfin, nous terminons ce mémoire par une conclusion générale et quelques recommandations pour la continuité de travail.

CHAPITRE 1

Gestion de RS232 sous windows

I. INTRODUCTION

Ce chapitre décrit les outils informatiques utilisés dans l'implémentation de notre application. Pour cela, nous allons présenter un aperçu du système d'exploitation (Windows) et ses entrées sorties (gestion de RS232,...).

II. LE SYSTEME D'EXPLOITATION

Un système d'exploitation (SE) réalise une sorte d'interface entre le matériel (disque, clavier, port USB, **port COM**...etc.) et les logiciels nécessitant un accès à ce matériel. Il ajoute une couche d'abstraction en proposant aux programmes de masquer les aspects techniques en les prenant lui-même en charge.

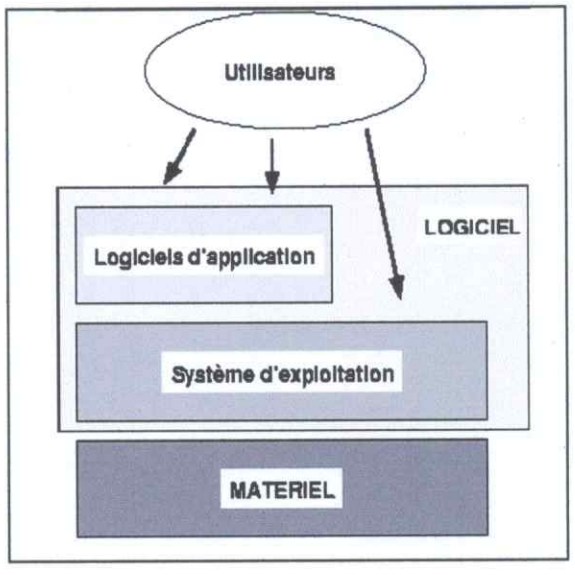


Figure 1 : *le rôle d'un système d'exploitation*

Les deux objectifs majeurs d'un SE sont :

- Transformer le matériel en une machine utilisable, c'est-à-dire fournir les outils adaptés aux besoins des utilisateurs indépendamment des caractéristiques physiques.
- Optimiser l'utilisation des ressources (matérielles et logicielles).

Les SE regroupent au moins 4 fonctionnalités essentielles :

- La gestion des processus
- La gestion de la mémoire
- Le système de fichier
- **La gestion des entrées/sorties**

Windows, UNIX, MAC-OS sons les SE les plus publiques dans le monde. Dans notre projet nous utilisons Windows comme un environnement de

développement à cause de ces services avantageux à notre projet (ses configurations et utilisations faciles...)

L'amélioration de Windows a engendré les versions suivantes:

Windows 3.1 - Windows 95 - Windows 98 et W98Se - Windows Me (Millennium) - Windows NT4 - Windows 2000 - **Windows XP** - Windows Vista.

III. INTRODUCTION AU WINDOWS XP [1]

III.1 Définition

Microsoft Windows XP : est un système d'exploitation développé par la société Microsoft. Il est essentiellement utilisé sur les plates-formes de type x86, c'est-à-dire la majorité des ordinateurs personnels communs. Nommé finalement **XP** pour *eXPérience*.

Windows XP est connu pour sa stabilité et son efficacité par rapport aux versions précédentes de Microsoft Windows. Il présente une interface graphique conviviale pour les utilisateurs. Les nouvelles possibilités de gestions de logiciels furent introduites afin d'éviter les problèmes des DLL auxquelles étaient soumises les anciennes versions grand public de Windows. Windows XP a été critiqué par des utilisateurs pour ses vulnérabilités en ce qui concerne la sécurité, l'intégration de certaines applications telles que Internet Explorer et le Lecteur Windows Media, les aspects de son interface utilisateur et les ressources importantes que XP mobilise.

Windows XP marque également un tournant important dans la stratégie de développement logiciel de la société Microsoft, qui décide ainsi d'unifier ses plates-formes sur une base commune. C'est en effet le premier système d'exploitation à utiliser un même noyau, basé sur une architecture de type Windows NT, aussi bien pour le marché professionnel que pour le marché grand public. Ainsi, il succède naturellement à Windows 2000 et marque la fin de la lignée Windows 9x avec les séries Windows 95..., 98... et Me (Millennium).

Selon Microsoft, XP fonctionne sur les machines avec un processeur minimum de 233Mhz, 128Mo de RAM et 1,5Go d'espace disque disponible. Pour être réaliste il faut au moins doubler ces spécifications. XP fonctionne en FAT32 ou en NTFS. Pour bénéficier de toutes les fonctions de sécurité, il faut utiliser NTFS.

Il existe XP familial et XP pro. XP Pro propose en plus les fonctionnalités suivantes : bureau à distance (prise du contrôle du pc à distance), le cryptage des fichiers sur le disque dur, l'architecture bi-processeur, l'intégration au réseau Windows NT/2000 (intégration aux domaines, stratégies de groupe, fonctions de sécurité Kerberos, intégration à l'Active Directory....). En réseau local, Windows XP Édition Familiale supporte 5 connexions simultanées, et Windows XP Professionnel lui peut supporter 10 connexions simultanées Pour l'intégration aux réseaux d'établissement, il faut donc impérativement utiliser XP pro (XP familial ne permet qu'un fonctionnement en réseau poste à poste).

III.2 Les différentes versions

Trois versions principales de ce système d'exploitation existent :

- Windows XP Édition Familiale (que nous appellerons « la version familiale »), destinée au grand public.
- Windows XP Professionnel (que nous appellerons « la version professionnelle »).
- Windows XP Édition Familiale/Professionnel Édition N, qui est une version de Windows sans certaines fonctions liées au multimédia.

Il existe aussi d'autres versions du système d'exploitation destinées à des usages plus précis.

Principales différences entre les versions familiale et professionnelle :

- La version familiale ne peut appartenir à un domaine, c'est-à-dire un groupe d'ordinateurs réunis en réseau et qui peuvent être gérés à distance par un ou plusieurs serveurs centraux. La plupart des professionnels utilisant Windows ont un serveur et un domaine.

- Le système de protection de fichiers est simplifié pour la version familiale. On ne peut pas, par exemple, allouer à un utilisateur donné des droits spécifiques sur un fichier.

- Certains outils (comme le bureau à distance en mode serveur) ne sont pas disponibles sous la version familiale.

- Les fichiers hors connexion qui permettent d'avoir une copie locale des fichiers qui sont sur le réseau.

- Le chiffrement des fichiers qui ne peuvent être lus par d'autres utilisateurs.

- Outils d'administration comme les stratégies de groupes, profils distants et l'installation distante.

III.3 Services pack

Microsoft propose gratuitement la mise à jour régulière de ses applications. Les corrections mineures sont faites grâce au site Windows update et lorsqu'il s'agit d'un ensemble de corrections majeures, Microsoft lui attribue le nom de « Service pack » suivi d'un numéro. La mise à jour du système d'exploitation réglant principalement des problèmes de sécurité, il est recommandé de mettre à jour le plus souvent possible (il est d'ailleurs possible d'activer la mise à jour automatique).

Actuellement, le service pack le plus récent pour Windows XP est le service pack 2.

Service Pack 1

Le Service Pack 1 est sorti le 9 septembre 2002, soit environ un an après la sortie de Windows XP.

Service Pack 2

Le Service Pack 2 est sorti le 6 août 2004. Il apporte beaucoup de nouveautés pour la sécurité de Windows XP.

En effet, il ne s'est pas contenté de mettre à jour l'OS, mais a apporté des modifications sur certains logiciels inclus dans le SP1 comme Outlook Express et Internet Explorer 6.0. Sur ce dernier, par exemple, il a ajouté une fonction de blocage des fenêtres publicitaires qui n'existent pas dans la version IE 6.0 de Windows 2000 SP4. Ce service pack introduit également le centre de sécurité qui regroupe les informations de l'état du pare-feu, de l'antivirus, des mises à jour à la fois, pour avoir une vue générale de l'état de protection de l'ordinateur, et avertit l'utilisateur lorsqu'un des ces éléments a un problème.

IV. LES E/S SOUS WINDOWS XP [3]

Le contrôle des périphériques d'entrée / sortie de l'ordinateur est une des fonctions primordiales d'un système d'exploitation. Le système d'exploitation doit envoyer les commandes aux périphériques, intercepter les interruptions et traiter les erreurs. Il doit aussi fournir une interface simple et facile d'emploi entre les périphériques et le teste de système. Cette interface doit être, dans la mesure du possible, la même pour tous les périphériques, c'est-à-dire indépendante du périphérique utilisé, le code des E/S représente une part importante de l'ensemble d'un système d'exploitation.

Et dans cette partie nous montrons comment un système d'exploitation (Windows) gère ces E/S.

Les grandes lignes de cette parie sont les suivants :

- Nous présentons rapidement les principes des E/S du point de vue matériel, puis, d'une manière plus générale, le logiciel des E/S. ce logiciel peut être structuré en couches, chaque couche s'acquittant dune tache bien définie.
- Nous nous examinerons ces différentes couches pour voir ce qu'elles font et comment elles se complètent.
- Nous étudierons les terminaux comme un exemple d'un périphérique d'E/S en détail (matériel et logiciel).

V. LES PERIPHERIQUES D'E/S [3]

Les périphériques d'E/S se répartissent en deux catégories :

Un périphérique bloc mémorise les informations dans des blocs de taille fixe, chaque bloc ayant une adresse propre. Les tailles courantes vont de 128 à

1024 octets. La propriété fondamentale de ces périphériques est qu'ils permettent de lire ou d'écrire un bloc indépendamment de tous les autres. On d'autre termes, le programme peut à tout moment lire ou écrire n'importe quel bloc. Les disques sont des périphériques bloc.

Un périphérique caractère accepte un flot de caractère sans soucier dans quelconque structure en bloc. Il ne peut pas adressé et ne possède pas de fonction de recherche. La plupart des périphériques qui ne se comportent pas comme des disques peuvent être considérés comme des périphériques caractère.

VI. LES CONTROLEURS DES PERIPHERIQUES [3]

Dans le contexte Intel, les interruptions sont gérées par un contrôleur d'interruptions (circuit intégré 8259 ou circuit inclus dans le microprocesseur lui-même).Ce composant intercepte toutes les demandes d'interruption des périphériques et les communique à l'U.C. selon leurs priorités.

Donc le système communique pratiquement toujours avec le contrôleur et non avec le périphérique.

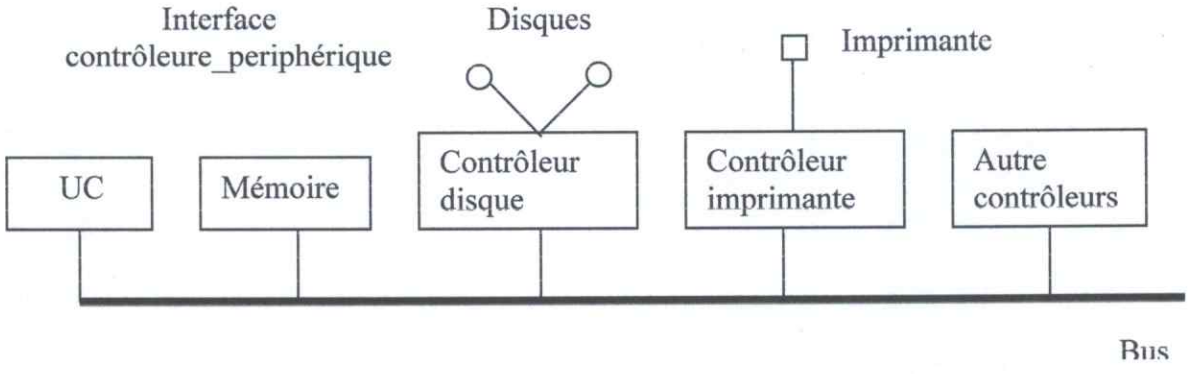


***Figure 2** : Les contrôleurs des périphériques*

L'interface entre le contrôleur et le périphérique est souvent de très bas niveau. Le contrôleur d'un terminal fonctionne comme un contrôleur série de bas niveau. Il lit dans la mémoire les octets qui contiennent les caractères à afficher et génère des signaux pour moduler le faisceau d'électrons qui dessine les caractères à l'écran.

Chaque contrôleur communique avec le processeur par l'intermédiaire de quelques registres. Ceux-ci situent dans l'espace mémoire adressable. Cette configuration est appelée E/S mappées en mémoire. D'autres ordinateurs utilisent un espace mémoire particulier pour les E/S et allouent à chaque contrôleur une partie de cette espace.

Le système d'exploitation effectue des E/S en écrivant des commandes dans les registres des contrôleurs. (Ex : le contrôleur de lecteur de disquette de IBM PC accepte 15 commandes comme « read, write, format... ») .

De nombreuses commandes ont des paramètres qu'il faut aussi charger dans les registres du contrôleur. Dès qu'une commande acceptée, le processeur peut

effectuer un autre travail, le contrôleur s'acquittant seul de la commande. Lorsque la commande est exécutée, le contrôleur envoie une interruption pour permettre au système d'exploitation de réquisitionner le processeur afin de tester les résultats de l'opération. Le processeur obtient ces résultats ainsi que l'état du périphérique en lisant un ou plusieurs octets dans les registres.

Table 1 représente un exemple des contrôleurs d'E/S dans l'architecture IBM.

Table 1 *: les adresses d'E/Set les vecteurs des interruptions de quelques contrôleurs de l'IBM PC.*

Contrôleur d'E/S	**adresse d'E/S**	**vecteur d'interruption**
Horloge	040-043	8
Clavier	060-063	9
RS232 auxiliaire	**2F8-2FF**	**11**
Disque dure	320-32F	13
Imprimante	378-37F	15
Ecran monochrome	380-3BF	-
Ecran couleur	3D0-3DF	-
Lecteur de disquette	3F0-3F7	14
RS232 principale	**3F8-3FF**	**12**

VII. LES PRINCIPES DES LOGICIELS D'E/S [3]

L'idée directrice est de décomposer le logiciel en une série de couches, les plus basse se chargeant de masquer les particularités du matériel aux yeux des couches les plus élevées. Ces dernières offrent aux utilisateurs une interface agréable, bien défini et facile d'emploi.

VII.1 Objectifs du logiciel d'E/S

L'indépendance entre ces logiciels et le matériel : on doit pouvoir écrire des programmes qui s'exécutent sans aucune modification, que leur fichiers se trouvent sur une disquette ou sur une disque dur. C'est au système d'exploitation de résoudre des problèmes engendrer par les différences qui existent entre ces périphériques qui nécessitent chacun un pilote spécifique.

La gestion des erreurs est une autre caractéristique importante du logiciel d'E/S. généralement, ces erreurs doivent être traitées à un niveau proche que possible du matériel. Un contrôleur qui constate une erreur au cours d'une lecture doit essayer de la corriger par lui-même. Si ne peut pas le faire, le pilote de périphérique doit essayer de la corriger à sont tour, peut être simplement en demandant la relecture du bloc. Les couches élevées ne doivent être prévenues que les plus basses n'arrivent pas à résoudre le problème.

Le concept des périphériques partagés par opposition aux périphériques dédiés. De nombreux périphériques comme les disques, peuvent être utilisés simultanément par plusieurs utilisateurs. D'autres périphériques, comme les imprimantes, sont dédiés à un seul utilisateur jusqu'à ce qu'il termine son travail. Ce type de périphériques induit des nombreux problèmes comme les interblocages.

VII.2 Le pilote

La fonction d'un pilote est d'isoler le matériel du noyau du SE, en offrant par voie de conséquence des applications une interface standardisée (API application programing interface). Les opérations d'E/S sont effectuées uniquement par le pilote. Celui-ci est donc responsable de l'échange. Il accède à l'interface par mémoire partagée ou par registres (espace d'E/S ou espace mémoire). Il doit répondre à temps aux demandes d'interruption et gérer l'ADM. Nous allons retrouver deux grandes familles, celle des pilotes de type caractère et celle des types bloc.

La fonction de gestion s'exécute sous interruption, à l'initiative d'un événement d'E/S ou suite à une demande de l'application.

VIII. LA GESTION D'E/S PAR INTERRUPTION [3]

- ***Problématique*** :

Comment prendre en compte un événement, comment provoquer une rupture de séquence d'exécution d'un processus dans un délai très court? Une solution les interruptions.

VIII.1. Définitions

Une interruption est un signal déclenché par un événement interne à la machine ou externe,provoquant l'arrêt d'un programme en cours d'exécution à la fin de l'opération courante, au profit d'un programme plus prioritaire appelé programme d'interruption. Ensuite, le programme interrompu reprend son exécution à l'endroit où il avait été interrompu.

Le système d'interruption est le dispositif incorporé au séquenceur qui détecte les signaux d'interruption. Ces signaux arrivent de façon asynchrone, à n'importe quel moment, mais ils ne sont pris en compte qu'à la fin de l'opération en cours.

Mécanisme général Lorsqu'une interruption survient, le processeur achève l'exécution de l'instruction en cours pour les interruptions externes (cf. plus loin), puis il se produit :

1. Sauvegarde du contexte dans une pile :

- adresse de la prochaine instruction à exécuter dans le programme interrompu.

- contenu des registres qui seront modifiés par le programme d'interruption.
- contenu du mot d'état (registre de drapeaux) rassemblant les indicateurs (tout cela forme le contexte sauvegardé).

2. Chargement du contexte du programme d'interruption (contexte actif) et passage en mode système (ou superviseur)

3. Exécution du programme d'interruption

4. Retour au programme interrompu en restaurant le contexte (commande Assembleur IRET) et en repassant en mode utilisateur.

VIII.2. Différents types d'interruptions

Les interruptions ne jouent pas seulement un rôle important dans le domaine logiciel, mais aussi pour l'exploitation de l'électronique.
Et se la que nous intéresse dans le but de notre projet là.
On distingue donc :

• **Interruptions internes** protection du système et des processus, appelées par une instruction à l'intérieur d'un programme (overflow, erreur d'adressage, code opération inexistant, problème de parité...)

• **Interruptions logiques** permettent à un processus utilisateur de faire un appel au système (software)

• **Interruptions matérielles** déclenchées par une unité électronique (lecteur, clavier, canal, contrôleur de périphérique, panne de courant,....) ou par l'horloge (hardware externes)

VIII.3. Les priorités

A chaque interruption, est associée une priorité (système d'interruptions hiérarchisées) qui permet de regrouper les interruptions en classes. Chaque classe est caractérisée par un degré d'urgence d'exécution de son programme d'interruption.

Règle : Une interruption de priorité j est plus prioritaire qu'une interruption de niveau i si j > i.

L'intérêt de ce système est la solution de problèmes tels que :

- arrivée de plusieurs signaux d'interruption pendant l'exécution d'une instruction.
- arrivée d'un signal d'interruption pendant l'exécution du signal de traitement d'une interruption précédente.

On peut utiliser un contrôleur d'interruptions pour regrouper les fils d'interruptions, gérer les priorités entre les interruptions et donner les éléments de calcul d'adresse au processeur.

VIII.4. Masquage d'interruption

Certaines interruptions présentent tellement d'importance qu'il ne doit pas être possible d'interrompre leur traitement. On masquera alors les autres interruptions pour empêcher leur prise en compte. Certaines interruptions sont non-masquables on les prend obligatoirement en compte. Une interruption masquée n'est pas ignorée : elle est prise en compte dès qu'elle est démasquée.

Au contraire, une interruption peut-être désarmée : elle sera ignorée. Par défaut, les interruptions sont évidemment armées.

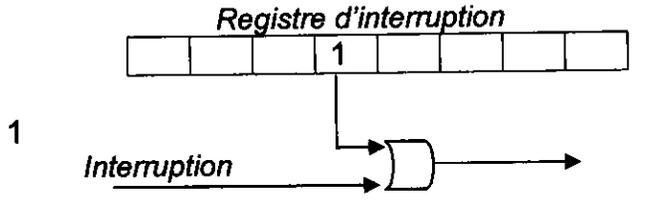


Figure 3 : *Principe de masquage d'interruption.*

Bit correspondant à un interruption masquée mis à 1 dans le registre d'interruption : sortie de ET à 1, donc masquage
Si le bit est à 0, pas de masquage.

VIII.5. Les interruptions vectorielles

Prenons l'exemple du microprocesseur 80386. Il peut comporter 256 interruptions numérotées de 0 à 255. Le lien entre chaque interruption et son programme est réalisé par une table ou vecteur d'interruptions.

Chaque élément contient l'adresse de début d'un programme d'interruption sur 4 octets (offset 2 octets, segment 2 octets). La taille de la table est donc de 4 x 256 = 1 Ko. Elle occupe les adresses 0h à 3FFh en mémoire centrale.

VIII.6. Les interruption matériels (exemple du Intel 80386 et de ses successeurs)

Elles sont déclenchées par une unité électronique. Par exemple, chaque fois qu'on presse ou relève une touche du clavier, l'IT clavier est déclenchée. Le programme d'interruption place le caractère entré dans le buffer à la suite des caractères entrés auparavant. Si le buffer est plein, il déclenche un bip ou une action convenue.

Les interruptions peuvent être masquées (exemple : clavier) grâce à l'instruction « Assembleur CLI » (CLear Interrupt flag) ou grâce à la fonction TURBO-C ® disable (). Ainsi, après CLI ou disable (), aucun caractère ne peut

être entré au clavier. Le flag IF (flag correspondant du registre d'état) est positionné à 0

De même, l'instruction Assembleur STI (SeT Interrupt flag) ou la fonction TURBO-C enable () démasque les interruptions en positionnant le flag IF à 1.

La commande Assembleur NMI (No Mask Interrupt) permet de rendre une interruption non masquable. Les interruptions qui ne peuvent être masquables sont toujours exécutables, même après CLI ou disable ().

Huit broches pour recevoir les codes de 256 interruptions
INTR : interrupt request : le processeur est informé par le contrôleur de l'arrivée d'une demande d'IT.
INTA : interrupt acknowledgement : le processeur a bien reçu par le bus de données le numéro d'IT envoyé par le contrôleur. Le mécanisme détaillé de traitement d'une interruption non masquée est le suivant :

- l'U.C. termine l'exécution de l'instruction en cours
- elle lit sur le bus le numéro de l'interruption.
- elle sauvegarde le mot d'état dans la pile et l'adresse de retour CS : IP (adresse de l'instruction suivante)
- elle lit dans la table d'interruption à l'adresse :
  [4 * n° d'IT] (Mode calculé indirect)
  Ainsi, à l'adresse [4*n°d'IT + 2] dans la table, elle trouve la partie segment CS de l'adresse du programme d'interruption et à l'adresse [4* n° d'IT] la partie offset de cette adresse.
- elle met à jour le compteur ordinal avec CS:IP
- elle exécute le programme s'achevant par IRET

(IT : InT irruption ; CS : Segment de Code de registre ; IP : Pointeur d'Instruction)

Le ROM BIOS (Read Only Memory Input/Output System) et le DOS représentent le noyau du système d'un micro-ordinateur. Les routines du ROM BIOS permettent essentiellement le traitement des opérations d'entrée/sortie et d'interface avec les périphériques.

Citons notamment les principales interruptions du ROM BIOS :
IT 8 : tic horloge (timer) appelé toutes les 55 ms (18,2 fois/s.). Elle sert par exemple à arrêter le moteur de l'unité de disquette lorsque aucun accès à la disquette n'est exécuté. Cette interruption 08H, après avoir lancé le programme correspondant, appelle l'interruption 1CH. Cette dernière ne contient qu'un retour IRET afin de permettre aux programmeurs d'implanter leurs propres programmes d'interruption (objectifs : programmation concurrente, multitâche,....)
IT 10h à 12h : variable selon le matériel connecté. Souvent, IT 10 désigne l'IT vidéo IT 11 fournit la liste du matériel disponible et IT 12 le calcul de l'espace mémoire disponible
IT 13h : gestion des disques
IT 14h : gestion de le port série
IT 15h : gestion d'un lecteur de cassettes
IT 16h : gestion du clavier
IT 17h : imprimante

IT 18h : activation du ROM BIOS
IT 19h : reset du système
IT 1Ah : gestion de la date et de l'heure

Les IT de 0 à 7 sont appelées directement par l'U.C. du microprocesseur. Il s'agit de :

IT 0 : division par 0
IT 1 : pas à pas
IT 2 : circuit RAM défectueux
IT 3 : point d'arrêt
IT 4 : débordement
IT 5 : copie d'écran
IT 6 et 7 : inutilisées

IX. LA LIAISON SERIE RS232 [4]

Le but de ce paragraphe est juste de nous faire comprendre comment fonctionne une liaison RS232 (port série de l'ordinateur) et de nous permettre de comment l'utiliser en créant vos propres interfaces.

IX.1 Définition et présentation

L'interface RS232 (RS pour *Recommended Standard*), la liaison RS-232 est issue de la norme du même nom qui permet l'envoi de données via une chaîne de niveaux logiques envoyés en série (d'où le nom du port du PC).

Elle permet de faire dialoguer deux systèmes (et seulement deux) entre eux. Les données sont envoyées par trames de 5, 6, 7 ou 8 bits soit autant de niveaux logiques.

Table 1 : *Caractéristique générale de RS232*

Nom de la connexion	Type de liaison	Type de connexion	Largeur	Débit typique	Portée typique	Fréquence
RS-232 (« série »)	Full Duplex	Point à Point	Série	115.2 kb/s max. (PC)	~10m (100m max.)	Standard

Cette liaison est de type asynchrone c'est à dire qu'elle n'envoie pas de signal d'horloge pour synchroniser les deux intervenant de la liaison, il est donc nécessaire que ces derniers soient configurés de la même manière (vitesse de transmission, nombre de bits par trame etc.). La vitesse de transmission s'exprime en bauds (bds = bits par seconde) les valeurs les plus courantes sont 2400, 4800 et 9600 bauds.

Figure 4 : *Connecteur DB 9*

IX.2 Les 9 lignes de la RS-232

La RS-232 est une liaison série asynchrone dissymétrique full duplex normalisée avec une trame d'un caractère de longueur.

__Table 2__ : les 9 lignes de l'RS232

N°	Signal	E/S	Utilisation
1	CD	E	Carrier Detect: Annonce que l'autre équipement reçoit une réponse.
2	RD	E	Received Data : Entrée de réception des données
3	**TD**	**S**	**Transmitted Data : Sortie d'émission des données**
4	**DTR**	**S**	**Data Terminal Ready : Indique à l'autre équipement que l'on souhaite communiquer.**
5	**SG**		**Signal Ground : Masse de référence des signaux (0V)**
6	DSR	E	Data Set Ready : Indique que l'équipement opposé est prêt.
7	**RTS**	**S**	**Request To Send : Demande à l'équipement opposé de se tenir prêt à recevoir.**
8	CTS	E	Clear To Send : Indique que l'équipement opposé est prêt à recevoir.
9	RI	E	Ring Indicator : Annonce que le modem reçoit un appel.

IX.3 Fonctionnement

➢ Aspect logique

Les niveaux logiques ont une grande marge d'erreur ce qui permet à la liaison RS-232 de n'être que peu sensible aux perturbations et donc de pouvoir être mise en place sur de longues distances. En effet le niveau logique " zéro " est représenté par une tension comprise entre +3 et +15V et le niveau logique " un " est représenté par une tension comprise entre -3 et -15V.

Au repos la ligne de transmission est au niveau logique 1. Lorsque l'un des systèmes veut commencer à communiquer, il prévient le système à l'autre bout de la liaison par une mise de la ligne au niveau 0, c'est le bit de Start. Viennent ensuite les bits de données au nombre de 8 (cf début du cours) ils sont soit au niveau 1 soit au niveau 0 en fonction des données.

Un dernier bit peut être ajouté, il s'agit du bit de parité qui ne joue pas du tout le même rôle. En effet le bit de parité est mis au niveau 0 par l'envoyeur si la somme des bits transmis à l'état 1 est paire et à 1 si la somme est impaire, ainsi le système qui reçoit les données peut vérifier s'il y a eu une erreur de transmission due à des interférences en comparant le nombre de bits à 1 et le bit de parité, s'il y a erreur alors le receveur peut demander à l'émetteur de renvoyer les données. Remarque : Si deux erreurs se produisent le receveur ne pourra pas les détecter puisque alors le nombre de bits au niveau 1 est en accord avec le bit de parité et si trois erreurs se produisent le receveur n'en verra qu'un etc.

Enfin après ce bit de parité viennent un ou deux bits de Stop qui signalent au receveur que la trame est terminée.

Figure 5 : *Type de trame de donnés RS232*

- **Aspect physique :**

Une liaison par définition c'est un lien entre deux éléments ou plus. Ce lien il peut-être visible (un câble) ou invisible (les ondes radio). Notre liaison RS-232 a donc besoin d'un lien physique pour fonctionner.

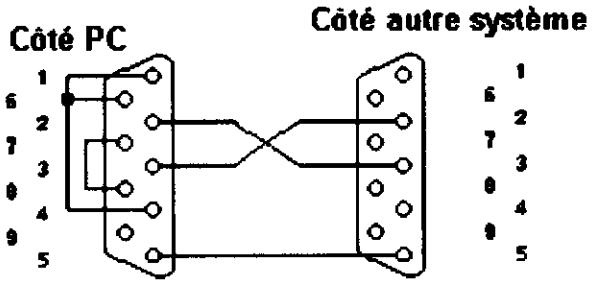


Figure 6 : *Câblage standard de fonctionnement*

Dans notre projet nous utilisons le câblage non croisé avec les trois pin : DTR (broche n° 3), TXD (broche n° 4) et RTS (broche n° 7) et le pin de masse(broche n° 5).

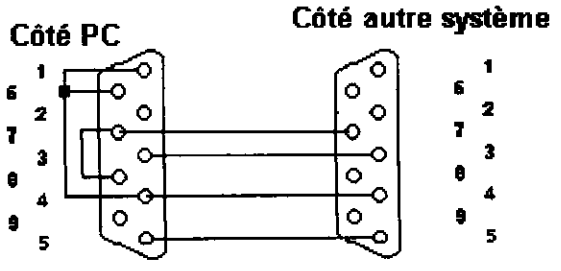


Figure 7 : *Câblage non croisé*

- **Les niveaux de tension**

Toutes les lignes de la RS-232 sont numériques. Elles transmettent donc des signaux électriques représentant symboliquement un '0' et un '1'. Ces signaux sont des tensions continues, dont la valeur est fixée par la norme. Cette norme est différente du côté émetteur et du côté récepteur. La raison de cette différentiation est la chute de tension pouvant avoir lieu dans un câble de RS-232. En effet, un câble est un cylindre qui peut, en première approximation et pour des débits pas trop élevés (on peut négliger l'effet de peau), être considéré comme une résistance de valeur R = r * L / S, où r note la résistivité du matériau conducteur utilisé, L sa longueur, et S sa section. La chute de tension à ses bornes DU est donc donnée par la loi d'Ohm : DU = R*I, où I note l'intensité traversant le fil. A l'heure actuelle, cette chute de tension est bien moins importante qu'auparavant, et la différence de normalisation entre émetteur et récepteur se justifie moins.

Il y a plusieurs raisons à cela. D'abord, il est vrai que le conducteur utilisé dans les câbles est - très légèrement - de meilleure qualité qu'il y a quelques années, et donc de résistivité plus faible. Ensuite, la longueur des câbles a grandement diminué : on ne câble plus un bâtiment entier en RS-232. Cette liaison sert surtout pour des périphériques à proximité immédiate de l'ordinateur (modem, scanner à main, caméra noir et blanc, lecteur de codes barres, etc.). Par contre, la section moyenne des câbles aurait tendance plutôt à diminuer. Mais l'endroit où le progrès le plus important a été fait, c'est dans l'intensité qui passe dans le conducteur.

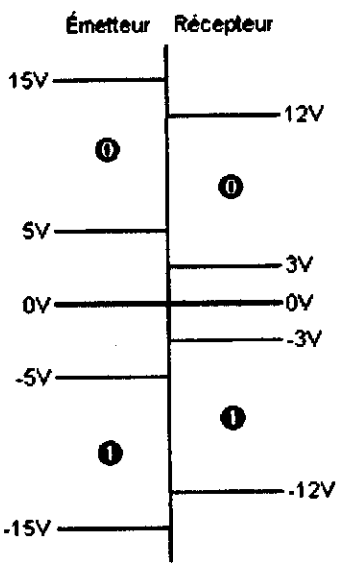


***Figure 8** : Les tensions de la RS-232*

On notera plusieurs choses : d'abord que suite à la remarque précédentes, les circuits d'interface RS-232 à l'heure actuelle émettent à +12V et -12V, car le récepteur, probablement très proche, a de bonnes chances de recevoir une tension non altérée ; il faut donc que l'émetteur reste dans la zone accessible au récepteur. De plus, ces deux tensions sont souvent présentes dans les appareils numériques, et en tous cas fournis par toutes les alimentations de PC. Notons aussi, que ce soit en émission ou en réception, un certains nombres de limites à la

tension pouvant représenter un bit. Les maxima en valeur absolu se comprennent- il ne faudrait pas qu'un hurluberlu aille émettre du ±100.000V sur la ligne ! La présence d'un « no man's land » autour de 0V a deux raisons : d'abord, cette zone inaccessible évite de confondre un 0 avec un 1, et d'autre part, le fait que le 0V soit hors limite permet une détection sommaire de pannes : un câble débranché, coupé, détérioré ou un faux contact ne peut plus être mépris pour l'émission constante de l'un des deux bits. Pour terminer, remarquons qu'il y a un équipement « bien de chez » nous, le minitel, qui possède, la plupart du temps, une liaison RS-232. Elle a cependant une bien désagréable spécificité : elle émet le '0' à 0V et le '1' à 5V. Il faut donc un câble spécial, pas excessivement coûteux mais qui comporte quand même un ou deux transistors placés sur une carte cachée dans la prise DB25 du câble, pour le raccorder à vos PC.

IX.4 Gestion de RS232

Dans ce paragraphe, nous allons considérer ce qui a trait au PC, mais uniquement dans le cadre qui nous intéresse, c'est à dire l'utilisation de la liaison RS-232.

La figure 9 décrit les principaux composant du PC entrant en jeu dans la gestion de la RS-232.

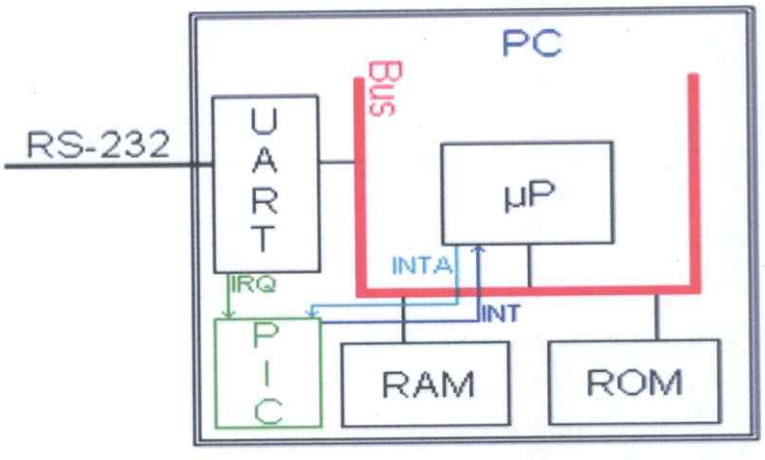


Figure 9 : *Architecture du PC pour la gestion de la RS-232*

UART : (Univesal Asynchronous Receiver Transmitter), circuit intégré contenue dans une seule puce et contrôlant généralement le port série.
PIC : (Programmable Interrupt Contrôler), contrôleur d'interruption programmable.
IRQ : (Interrupt ReQuest, numéro d'interruption matérielle - il y en a 15 différentes dans un PC)

Dans ce schéma, l'UART fait l'interface entre la **RS-232**, fondamentale série, et le bus du PC, qui est parallèle sur au moins 8 bits. Le microprocesseur gère les flux de caractères entrant et sortant. La ROM contient les procédures de gestion de l'UART (enfin, quand elles ne sont pas remplacées par des routines plus efficaces chargées depuis le disque dur...). La RAM contient les données à envoyer et celles reçues, ainsi que différentes informations sur l'état de l'UART comme l'adresse de base, le numéro de ligne d'IRQ, etc. Le PIC n'est utilisé, comme son nom l'indique, que dans la gestion par interruptions.

Il existe deux moyens principaux de contrôler la liaison **RS-232** : l'attente active et les interruptions.

IX.4.1 La gestion par attente active

Cette méthode appelée polling, consiste en une interrogation incessante du demandeur au fournisseur (de données, d'information, etc.). Dans notre cas, c'est le microprocesseur qui passe son temps à interroger l'UART. En effet, un bit dans l'UART informe le lecteur de l'arrivée d'un caractère, et un autre bit qu'un caractère a fini d'être transmis. Cette technique a l'énorme avantage d'être très simple à programmer ; par exemple, pour une simple réception de caractères, la boucle suivante suffit :

Répéter
Répéter jusqu'à'UART_recoi_quelque_chose;
lire_caractère_de_UART;
jusqu'à faux;

Cette technique a cependant un défaut énorme : dans le cas de systèmes multitâches, elle occupe perpétuellement le processeur, et pour ne produire aucun effet quand le modem n'émet rien, ce qui peut durer longtemps. De plus, si le processeur n'est pas "en attente" au moment ou des caractères arrivent (il est par exemple en train de sauver les derniers arrivés dans un fichier), ils vont s'écraser les uns les autres dans le tampon de réception de l'UART, limité à un seul caractère. Enfin, lorsque l'on doit gérer à la fois l'émission et la réception, la programmation se complique un peu, même si cela n'a rien de critique :

```
Répéter
    si UART_recoi_quelque_chose
    alors lire_charactère_de_UART;
    si (UART_fini_l'envoi et il y a_quelque_chose)
    alors envoyer_caractere_vers_UART;
jusqu'à faux;
```

Une solution plus performante en matière de gestion de la RS-232 consiste à utiliser une politique par interruptions.

IX.4.2 La gestion par interruptions

Cette méthode, appelée callback, consiste, pour le fournisseur d'information, à faire *remonter* celle-ci jusqu'à son ou ses consommateur(s). Elle est bien plus complexe à programmer, mais donne des résultats naturellement meilleurs dans un système multitâches : le processeur ne s'intéresse à la RS-232 que lorsque l'UART lui signale que c'est nécessaire. Mais il faut alors gérer tout un nombre de problèmes périphériques :

Informer le système de la manière dont l'UART va signaler l'arrivée d'informations (numéro de ligne d'IRQ et routine de traitement associée), programmer PIC pour que cette ligne d'IRQ soit validée, gérer la communication

avec les consommateurs de l'information que l'on obtient de l'UART, si ce n'est pas notre programme lui-même.

X. CONCLUSION

Dans ce chapitre nous avons défini les notions d'un SE, de périphérique et d'interface d'E/S. ensuite nous avons détaillé la notion, la composition,et fonctionnement...de l'interface RS232.La gestion de Windows XP pour l'interface RS232.à été décrite .

Nous utilisons l'interface série RS232 a cause de sa simplicité de fonctionnement et de protocole (dans le cas d'interface bas débit), l'interface RS232 est plus simple à gérer par l'utilisateur (reconnaissance et initialisation automatique par le SE, alimentation intégrée) et offre un débit bien supérieure dans la plupart des cas.

Dans la partie suivante nous expliquons le contrôle à distance (réseau, protocoles, l'architecture client/serveur...).

CHAPITRE 2

Contrôle à distance

I. INTRODUCTION

Dans ce chapitre nous allons définir quelques aspects généraux du contrôle à distance et ses outils fondamentaux. Nous nous basons surtout sur le réseau (définition, types, architecture...etc), qui est l'élément principal dans le concept du contrôle à distance.

II. LE CONCEPT DE CONTROLE A DISTANCE [6]

Dans notre cadre, il n'est pas possible de passer par un serveur Internet distant pour envoyer des ordres à un ordinateur. Il va falloir configurer notre ordinateur en serveur (ex : apache).

Le contrôle à distance selon l'explication anglaise du terme (remote control) est le contrôle et la commande d'un robot ou d'une machine quelconque à distance (les appareils médicaux dans notre projet). C'est une technologie en plein épanouissement que ce soit sur les réseaux locaux ou pour des réseaux non déterministe comme réseau Internet.

Le principe consiste à déplacer les commande distante, pour que l'opérateur puise avoir les informations à distance (image, localisation, état actuelle...) et puisse commander à distance, cette distance varie de quelque mètre à plusieurs kilomètres, la liaison peut être de type câble, micro-ondes,ou fibre optique.

III. INTRODUCTION AUX RESEAUX [5,6]

Un réseau est un ensemble d'éléments ou d'objets interconnectes entres eux, qui permettent la circulation d'information entre eux. On peut parler de plusieurs réseaux : réseaux téléphoniques, réseau hydraulique pour les canalisations d'eaux, réseau informatique...etc.

Pour nous ce qui nous intéresse ce est les réseaux informatiques. Un réseau informatique est un ensemble d'ordinateurs relies entre eux par des liens physiques (câble coaxial, paire torsadé, fibre optique...) ou par des liaisons sans fil, et qui permettent l'échange des données entre eux (sous forme d'impulsions électrique, de lumière, ou d'ondes électro-magnetique pour les réseaux sans fil).

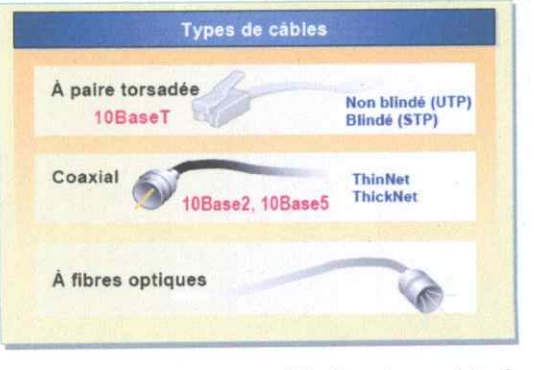


Figure 1 : *les câbles réseau utilisés dans L'infrastructure*

Les réseaux qui permettaient à leurs origine de relier des terminaux passifs aux ordinateurs centraux, permettent actuellement de relier tout type d'ordinateur

(gros serveurs, station de travail, terminaux ...). Ils sont donc indispensables pour les entreprises et les administrations (gestion, commerce, base de données, recherche), ainsi que pour les particuliers (Internet, jeux sous réseau).

On distingue plusieurs types de réseaux :

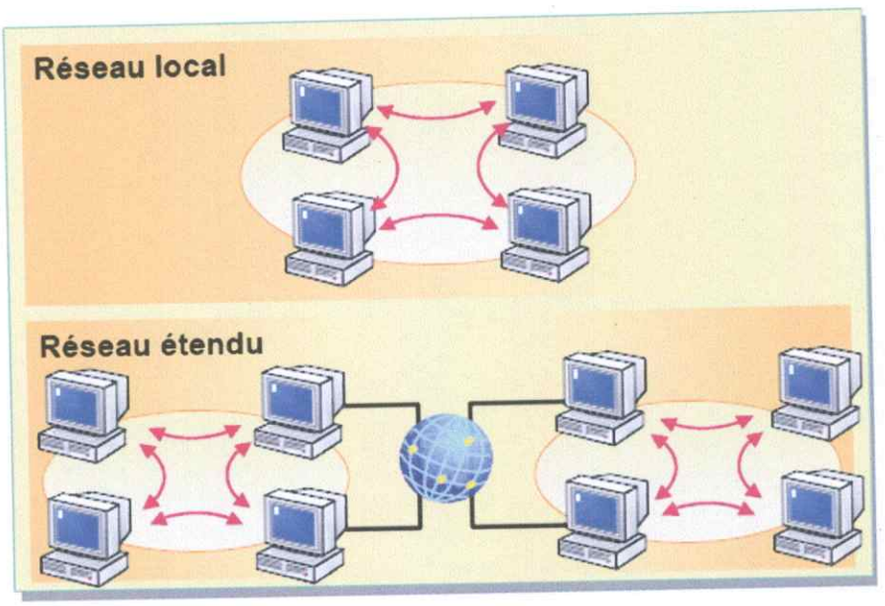


Figure 2 : *Présenter le concept d'étendue des réseaux.*

- réseau local (LAN) : qui peut s'entendre de quelques mètre a quelques kilomètres et qui satisfait les besoins internes des entreprises.
- Réseau métropolitain (MAN) : qui relier plusieurs sites situés dans la même ville, chacun possédant sont propre réseau local, par exemple les différents sites d'une administration.
- Réseau étendu (WAN) : qui permet de communiquer a l'échelle d'un pays ou du monde entier, comme Internet, les infrastructures pouvant être terrestres ou spatiales à laide des satellites de télécommunications.

On distingue aussi plusieurs topologies de réseau : les réseaux en étoiles, en anneau, en bus simple, en boucles et tout autre formes.

Il existe deux modes de réseaux : le mode point à point et le mode diffusion :

- Pour le mode point à point, le support relie l'équipement seulement, les éléments non directement connectés le font par l'intermédiaire des autres nœuds du réseau.
- Pour le mode diffusion, tous les éléments partagent le même support, de transmission. Chaque donnée envoyée par un élément est reçue par tous les autres éléments, le destinataire reconnaîtra le message qui lui est destine grâce a l'adresse qu'il véhicule, un seul élément peut envoyer une donnée a un moment donnée, c'est pour ça qu'il faut que l'émetteur écoute au préalable si la voie est libre. Dans une telle configuration, la rupture du support entraîne l'arrêt du réseau, par

contre l'arrêt d'une machine n'influe pas les autres, ce mode est très utilise dans les réseaux locaux.

On ne peut parler de réseaux informatiques sans parler de son plus grand problème qui est l'interconnexion des réseaux différent, au début des années 70, chaque constructeur apportait sa propre solution réseau (SNA d4IBM, DSA de bull, TCP/IP...), il était impossible d'interconnecter ces réseaux si une norme n'était pas faite, et c'est pour ça que l'ISO (l'**I**nternationale **S**tandard **O**rganisation) a établie la norme OSI (**O**pen **S**ystème **I**nterconnections). Cela permet à tout système ouvert que se soit un ordinateur, un terminal ou un réseau respectant cette norme d'échanger des informations avec des équipements différents issus d'autres marques respectant cette norme aussi.

L'un des rôles essentiels de la norme était de définir un modèle pour toute les architectures, basé sur un découpage en sept (7) couches, chacune ayant une fonction particulière.

7 Application
6 Présentation
5 Session
4 Transport
3 Réseau
2 Liaison
1 Physique

***Figure 3** : Les sept couches de référence de modèle OSI de l'ISO*

Chaque couche est constituée d'éléments matériels et logiciels et chacune offre un service à la couche située immédiatement au dessus d'elle en lui épargnant les détails de l'implémentation nécessaire. Chaque couche 'n' gère la communication avec la même couche 'n' de la machine distante en suivant un protocole bien spécifique.

IV. LES PROTOCOLES [6]

Pour que les ordinateurs d'un même réseau puissent communiquer entre eux, ils doivent partager le même langage, c'est-à-dire le même **protocole**. Un protocole est un ensemble de règles ou standards qui permettent aux ordinateurs d'un réseau de communiquer. Il existe aujourd'hui un certain nombre de protocoles dont chacun présente des caractéristiques et des fonctionnalités qui lui sont propres (FTP spécialisés dans l'échange de fichiers, ICPM servir à gérer simplement l'état de la transmission et des erreurs,...). Cependant, tous les protocoles ne sont pas compatibles avec tous les ordinateurs et systèmes d'exploitation.

Sur internet les protocoles utilisés font partie d'une suite de protocoles, c-a-d un ensemble de protocoles reliés entre-eux. Cette suite de protocoles s'appelle TCP/IP.
Elle contient, entre autres, les protocoles suivants :

« HTTP ; FTP ; ICMP ; TCP ; UDP ; SMTP ; MNTP ; Telnet... »

On classe généralement les protocoles en deux catégories selon le niveau de contrôle des données que l'on désire :

- les protocoles orientés connexion : il s'agit des protocoles opérant un contrôle de transmission des données pendant l'établissement d'une communication machines. Dans un tel schéma, la machine réceptrice envoie des accusés de réception lors de la communication, ainsi la machine émettrice est garantie de la validité des données qu'elle envoie. Les données sont ainsi envoyées sous forme de flot. TCP est un protocole orienté connexion.
- les protocoles non orientés connexion : il s'agit d'un mode de connexion dans lequel la machine émettrice envoie des données sans prévenir la machine réceptrice, et la machine réceptrice reçoit les données sans envoyer d'accusé de réception à la première. Les données ainsi envoyées sous forme de blocs (datagrammes). UDP est un protocole non orienté connexion.

V. LE PROTOCOLE TCP/IP [5, 6, 7]

V.1 Introduction

TCP/IP est née de la réflexion de chercheurs américains suite à un problème posé par l'armée américaine, celle-ci disposait de plusieurs bases sur le territoire, chacune disposant de sa propre logistique informatique, donc de différentes machines reliées entre elle par des réseaux locaux, étant donnée que ces bases était reliées entre elle par câbles, le problème est de trouver un moyen pour que l'information puise circuler en reconfigurant le système automatiquement en cas de ruptures de liaison pour retrouver le chemin adéquat. De là, est né le protocole IP (Internet Protocole) ou (Interconnected Network Protocol).

Le protocole TCP/IP présente les avantages suivants :

- Il s'agit d'un standard industriel. En tant que standard, c'est un protocole ouvert qui n'est pas contrôlé par une seule entreprise.
- Il contient un ensemble d'utilitaires permettant d'établir une connexion avec divers systèmes d'exploitation. La connexion entre deux ordinateurs ne dépend pas de leur système d'exploitation.
- Il utilise une architecture client-serveur multiplate-forme évolutive. Vous pouvez étendre ou restreindre le protocole TCP/IP afin de répondre aux futurs besoins d'un réseau.

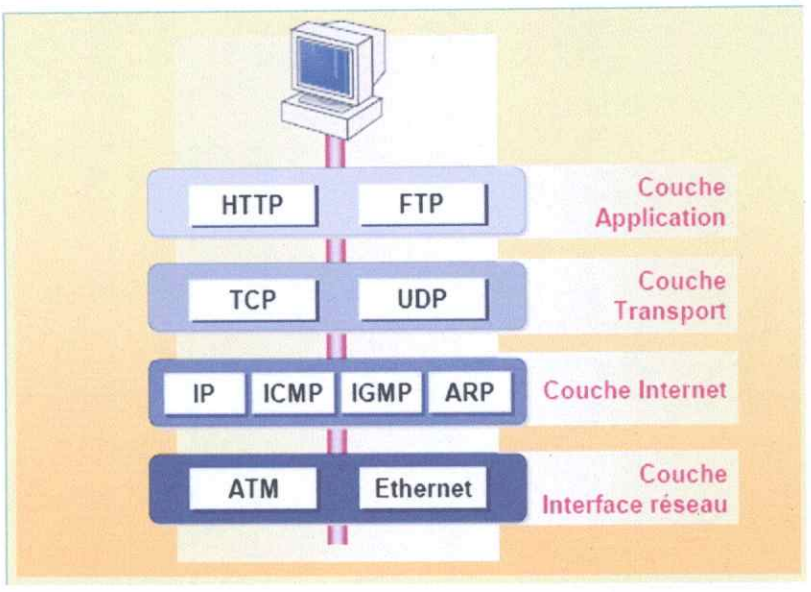


Figure 4: *Les Couches TCP/IP et leurs protocoles.*

IP est un protocole de la couche 3 du modèle OSI (la couche réseau), il assure la délivrance de datagrammes sans connexion non fiable IP, et qui a pour fonction essentielle le routage, la définition du format des datagrammes IP qui est l'unité de base des données circulant sur le réseau et la définition de la gestion de la remise non fiable des datagrammes.

Le problème de ce protocole est qu'il envoi l'information d'une machine à l'autre alors que l'information s'échange d'une application a l'autre, et c'est pour ça qu'on a crée le protocole TCP.

TCP (Transport Control Protocol) est l'un des principaux protocoles de la couche transport (couche 6) du modèle OSI, il gère les données provenant ou à destination de la couche inférieure (le protocole IP) au niveau des applications. C'est un protocole orionté connexion qui permet à deux machines de communiquer en contrôlant l'état de la transmission, il assure un service fiable. Grâce à ce protocole la couche Internet ne se préoccupe que de l'envoie de données sous forme de datagramme sans se préoccuper du contrôle de données qui assuré par le protocole TCP.

Donc le nom TCP/IP à été choisi en référence à ces deux principaux protocoles qui le caractérisent. Aujourd'hui TCP/IP intègre beaucoup d'autre protocole (ICMP, IGP, FTP, HTTP, SMTP,...).

V.2 Protocoles et couches TCP/IP

Le protocole TCP/IP organise le processus de communication décrit ici en affectant ces activités à différents protocoles de la pile TCP/IP. Pour accroître l'efficacité du processus de communication, les protocoles sont organisés en couches. Les informations sur l'adressage sont placées en dernier afin que les ordinateurs d'un réseau puissent rapidement vérifier si les données leur sont destinées. Seul l'ordinateur de destination ouvre et traite toutes les données.

Le protocole TCP/IP utilise un modèle de communication à quatre couches pour transmettre des données entre deux sites. Les quatre couches de ce modèle sont les couches Application, Transport, Internet et Interface réseau. Tous les protocoles appartenant à la pile de protocoles TCP/IP se trouvent dans les couches de ce modèle.

- **Couche Application**

La couche Application correspond à la couche supérieure de la pile TCP/IP. L'ensemble des applications et des utilitaires sont contenus dans cette couche et l'utilisent pour accéder au réseau. Les protocoles de cette couche permettent la mise en forme et l'échange des informations utilisateur. Ces protocoles sont les suivants :

- Protocole HTTP (*Hypertext Transfer Protocol*)

Ce protocole permet de transférer les fichiers qui composent les pages Web d'Internet.

- Protocole FTP (*File Transfer Protocol*)

Ce protocole permet le transfert interactif de fichiers.

- **Couche Transport**

La couche Transport permet d'organiser et d'assurer la communication entre des ordinateurs, et transmet les données à la couche Application supérieure ou à la couche Internet inférieure. La couche Transport indique également l'identificateur unique de l'application à laquelle les données doivent être remises.

La couche Transport est constituée de deux protocoles principaux qui contrôlent la méthode de remise des données. Ces protocoles sont les suivants :

- Protocole TCP (*Transmission Control Protocol*)

Ce protocole garantit la remise des données par le biais d'un accusé de réception.

- Protocole UDP (*User Datagram Protocol*)

Ce protocole fournit une remise rapide des données mais ne la garantit pas.

- **Couche Internet**

La couche Internet est chargée de l'adressage, du conditionnement et du routage des données à transmettre. Cette couche contient quatre protocoles principaux :

- Protocole IP (*Internet Protocol*)

Ce protocole est chargé de l'adressage des données à transmettre et de leur acheminement jusqu'à la destination.

- Protocole ARP (*Address Resolution Protocol*)

Ce protocole est chargé d'identifier l'adresse MAC (*Media Access Control*) de la carte réseau de l'ordinateur de destination.

- Protocole ICMP (*Internet Control Message Protocol*)

Ce protocole fournit des fonctions de diagnostic et de signalement d'erreurs dues à un échec de remise des données.

➢ Protocole IGMP (*Internet Group Management Protocol*)

Ce protocole est chargé de la gestion de la multidiffusion dans le protocole TCP/IP.

- **Couche Interface réseau**

La couche Interface réseau est chargée du placement des données sur le support réseau et de leur retrait du support à réception. Cette couche contient des dispositifs physiques tels que des câbles réseau et des cartes réseau. La carte réseau présente un numéro hexadécimal unique à 12 caractères, comme B5-50-04-22-D4-65, qui correspond à l'adresse MAC. La couche Interface réseau ne contient pas le type de protocoles logiciels inclus dans les trois autres couches, mais renferme des protocoles tels qu'Ethernet et ATM (*Asynchronous Transfer Mode*), qui définissent la transmission des données sur le réseau.

V.3 Le protocole TCP

TCP est un protocole de la couche 6 (application) du modèle OSI. Qui permet la gestion des données au niveau application. C'est un protocole orienté connexion. C'est–à-dire que l'état de la transmission est contrôlé. Les applications dialoguant lors d'une connexion sont considérées l'une comme un serveur et l'autre comme un client. Qui doivent établir la connexion avant de pouvoir dialoguer. Après l'établissement de la connexion, les deux machines s'échanges des messages spécifiques. Cette connexion est bidirectionnelle simultanée (full duplex) composée de deux flots de données indépendants et de sens contraire les données sont encapsulées c'est-à-dire qu'on ajoute aux paquets de données un entête qui va permettre la synchronisation de la transmission et d'assurer leurs réception.

TCP échange entre les deux machines un flux d'octet non interprété par TCP c'est aux applications des deux extrémités que revient ce travail.

Dans le cas d'informations trop volumineuses , elles sont fractionnées en données de taille optimale par TCP .De même TCP peut regrouper des données pour former qu'un seul datagramme de taille convenable pour décharger le réseau, cette unité est appelée segment.

La Figure (5) montre le format des données TCP

<table>
<tr><td colspan="32">0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31</td></tr>
<tr><td colspan="16">Port source</td><td colspan="16">Port distinction</td></tr>
<tr><td colspan="32">Numéro d'ordre</td></tr>
<tr><td colspan="32">Numéro d'accusé de réception</td></tr>
<tr><td colspan="4">Décalage donné</td><td colspan="6">réservé</td><td>URG</td><td>ACK</td><td>PSH</td><td>RST</td><td>SYN</td><td>FIN</td><td colspan="16">Fenêtre</td></tr>
<tr><td colspan="16">Somme de contrôle</td><td colspan="16">Pointeur d'urgence</td></tr>
<tr><td colspan="24">Options</td><td colspan="8">Remplissage</td></tr>
<tr><td colspan="32">Données</td></tr>
</table>

Figure 5 : *Format des données TCP.*

Tableau 1 : *explication des différents champs d'une donnée TCP.*

Port source	16 bits	Port relatif à l'application en cours sur la machine source
Port destination	16 bits	Port relatif à l'application encours sur la machine destination
Numéro d'ordre	16 bits	Lorsque le drapeau SYN est à 0, le numéro d'ordre est celui de premier mot de segment en cours. Lorsque SYN est à1, le numéro de séquence est le numéro de séquence initial utilisé pour synchroniser les numéros de séquence (ISN).
Numéro d'accusé de réception	32 bits	Dernier segment reçu par le récepteur.
Décalage des données.	4 bits	Il permet de repérer le début des données dans le paquet. Le décalage est ici essentiel car le champ d'option est de taille variable.
Réservé	6 bits	Champ inutilisé actuellement mais prévu pour l'avenir.
Drapeaux (oflags)	6×1 bits	Les drapeaux représentent des informations supplémentaires. URG : si ce drapeau est à 1 le paquet doit être traité de façon urgente. ACK : si ce drapeau à 1, le paquet est un accusé de réception. PSH (PUSH) : si ce drapeau est à 1, le paquet fonctionne suivent la méthode PUSH. RST : si ce drapeau est à1, la connexion est réinitialisée. SYN : si ce drapeau est à 1, les numéro d'ordre sont synchronisés.

		FIN : si ce drapeau est à 1, la connexion s'interrompt.
Fenêtre	16 bits	Champ permettant de connaître le nombre d'octets que le récepteur souhaite recevoir sans accusé de réception.
Somme de contrôle	Checksum ou CRC	La somme de contrôle est réalisée en faisant la somme des champs de données de l'en-tête, afin de pouvoir vérifier l'intégrité de l'en-tête.
Pointeur d'urgence	16 bits	Indique le numéro d'ordre à partir duquel l'information devient urgent.
Options	Taille variable	Des options diverses.
Remplissage		On remplit l'espace restant après les options avec des zéros pour avoir une langueur de 32 bits.

Les principales caractéristiques du protocole TCP sont :

- permet de remettre en ordre les datagrammes en provenance du protocole.
- permet de vérifier le flot de données afin d'éviter une saturation de réseau.
- permet l'initialisation et la fin d'une communication de manière courtoise.
- permet de formater les données en segments longueur variable afin de les remettre au protocole IP

Il a aussi deux très importantes caractéristiques qui sont :

- **La fonction de multiplexage :**

C'est faire transiter sur une même ligne des données provenant d'applications diverses.
Ces opérations sont réalisées grâce au concept de ports (ou sockets); c'est –à-dire un numéro associé à un type d'application, qui, combiné à une adresse IP, permet de déterminer de façon unique une application qui tourne sur une machine donnée.

- **fiabilité des transferts :**

TCP est un protocole fiable .Il possède un système d'accusé de réception qui assure bonne réception des données .Lors de l'émission d'un segment ,un numéro de séquence lui est associe .A la réception de ce segment ,la machine réceptrice va renvoyer un accusé de réception (un segment de données dont le drapeau ACK est à1) munis d'un numéro égal au numéro de la séquence précédant .

Aussi à chaque envoi d'un segment, la machine émettrice enclenche une minuterie, qui si elle expire et que l'accusé de réception n'arrive pas ,alors dans ce cas la machine émettrice considère que le segment est perdu et l'envoi de nouveaux.

Toutefois, si le segment perdu arrive à destination en retard, alors la machine réceptrice saura que c'est un doublon grâce à son numéro de séquence et l'éliminera.

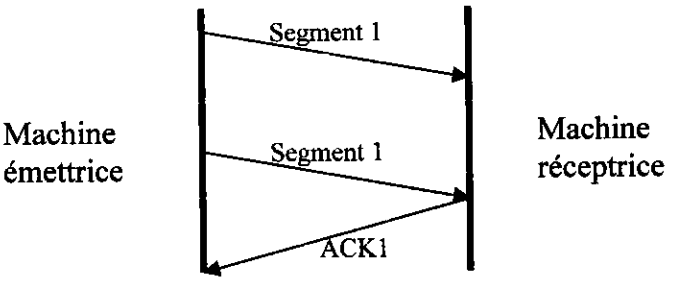


***Figure 6** : Fiabilité des transferts.*

V.4 La connexion TCP

- Etablissement d'une connexion :

Etant donné que ce processus de communication, qui se fait grâce à une émission de données et d'un accusé de réception, est basé sur un numéro d'ordre (appelé généralement numéro de séquence), I faut que les machine émettrices et réceptrices (client et serveur) connaissent le numéro de séquence initial de l'autre machine.

L'établissement de la connexion entre deux applications se fait souvent selon le schéma suivant :

- les ports TCP doivent être ouverts ;
- l'application sur le serveur est passive, c'est –à-dire que l'application est à l'écoute, en attente d'une connexion ;
- l'application sur le client envoie une requête de connexion au serveur dont l'application est en ouverture passive .L'application du client est dite « en ouverture passive.
- Les deux machines doivent donc synchroniser leurs séquences grâce à un mécanisme commun appelé three ways handshake (poignée demain en trois temp),que l'on retrouve aussi lors de la clôture de session .

Ce dialogue permet d'initier la communication, il se déroule en trois temps, comme sa dénomination l'indique :

- Dans un premier temps, la machine émettrice (le client) transmet un segment dont le drapeau SYN est à 1(pour signaler qu'il s'agit d'un

segment de synchronisation), avec un numéro d'ordre N, que l'on appelle numéro d'ordre initial du client.

- Dans un second temps ,la machine (le serveur) reçoit le segment initial provenant du client ,puis lui envoie un accusé de réception ,c'est –à-dire un segment dont le drapeau ACK est à 1est le drapeau SYN est à1 (car il s'agit là encore d'une synchronisation).Ce segment contient le numéro d'ordre de cette machine (de serveur) qui est le numéro d'ordre initial du client .Le champ le plus important de ce segment est le champ accusé de réception qui contient le numéro d'ordre initial du client ,incrémenté de 1.
- Enfin, le client transmet au serveur un accusé de réception, c'est-à-dire un segment dont le drapeau ACK est à 1 et le drapeau SYN est à zéro (il ne s'agit plus d'un segment de synchronisation) .Son numéro d'ordre est incrémenté et le numéro d'accusé de réception le numéro de séquence initial du serveur incrémenté de 1.

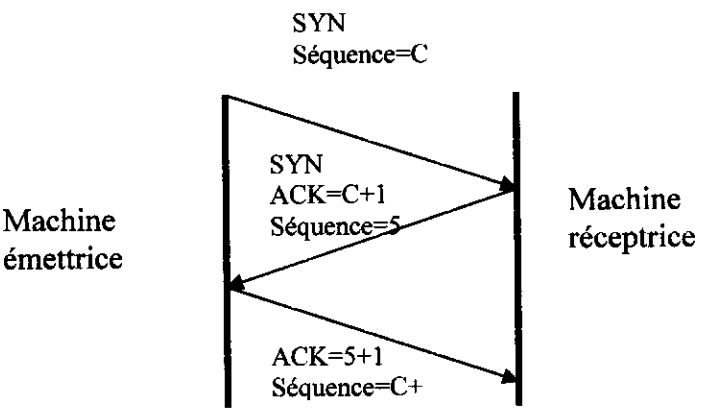


Figure 7 : *Synchronisation de deux machines.*

Suite à cette séquence comportant trois échanges, les deux machines sont synchronisées et la communication

Fin d'une connexion : le client peut demander à une connexion au même titre que le serveur.

La fin de la connexion se fait comme suivante :

- Une des machines envoie un segment avec le drapeau FIN à 1, et l'application se met en état d'attente de fin, c'est-à-dire quelle finit de recevoir le segment en cours et ignore les suivants.
- Après réception de ce segment, l'autre machine envoie un accusé de réception avec le drapeau FIN à1continue d'expédier les segments en cours.
- Suite à cela, la machine informe l'application qu'un segment FIN a été reçu, puis envoie un segment FIN à l'autre machiné , ce qui clôture la connexion ...

V.5 Le protocole IP

Le protocole IP permet d'identifier l'emplacement de l'ordinateur de destination dans une communication réseau. Le protocole IP est un protocole sans connexion non fiable qui est essentiellement chargé de l'adressage des paquets et de leur routage entre des ordinateurs mis en réseau. Bien que le protocole IP tente systématiquement de remettre un paquet, un paquet peut être perdu, endommagé, remis hors séquence, dupliqué ou retardé. Toutefois, le protocole IP n'essaie pas de résoudre ces types d'erreurs en demandant une nouvelle transmission des données. L'accusé de la remise des paquets et la récupération des paquets perdus relèvent d'un protocole de couche supérieure, comme le protocole TCP, ou de l'application proprement dite.

- **Activités effectuées par le protocole IP**

Vous pouvez considérer le protocole IP comme la salle de traitement du courrier de la pile TCP/IP, dans laquelle se déroulent les opérations de tri et de remise des paquets. Les paquets sont transmis au protocole IP par le protocole UDP ou TCP à partir de la couche Transport supérieure ou transmis à partir de la couche Interface réseau inférieure. La fonction principale du protocole IP consiste à router les paquets jusqu'à leur destination.

V.6 Adressage

Chaque ordinateur d'un réseau Internet possède une adresse IP unique codée sur 32 bits. Une adresse est représenté de 4 nombre séparés par des points dans une « notation décimale pointe » avec un octet pour chaque nombre, chaque nombre est compris entre 0et 255.Une adresse IP est constitue d'un paire (id de réseau ,id de machine) et appartient a une certaine classe (A B C D et E) selon la valeur de son premier octet .

Donc on a pour le premier octet de :

- 0 a 127 : classe A
- 128 a 191 : classe B
- 192 a 223 : classe C
- 224 a 239 : classe D
- 240 a 247 : classe E

Les adresses de classe A sont utilisées pour les grands réseaux qui comporte plus de 65536 ordinateur comme celui du réseau MIT, le nombre de réseaux de classe A est limite a127 le monde.

Les adresse de classe B sont réservées aux ayant entre 256 et 65536 ordinateurs.

Pour la classe C on ne peut dépasser les 256 machine, le nombre de post de ce type peut dépasser 2 millions.

Certaines adresse IP ont une signification particulières on peut citer :

- 0.0.0.0 est utilise par une machine pour connaître sa propre adresse IP.
- « id de réseau » « id de machine nul » permet de désigne le réseau lui-même.
- « id de réseau » « id de machine avec tous ses bits a 1 est une adresse de diffusion (broadcasting)
- Les adresse de classe a de 10.0.0.0 a 10.255.255.255, de classe b de 172.16.00 à la constitution de réseaux intranet.
- 127 .0.0.1 adresse local (localhost), l'adresse d'u PC par défaut.

Le système des adresses IP permet aussi la création de sous –réseaux en découpant la partie réservée à l'adresse machine sur un réseau en deux parties dont la première sera un identificateur de sous –réseaux.

VI. LE MODELE CLIENT/SERVEUR [8,9]

Le modèle client serveur est apparu dans les année 90 de l'aboutissement d'un ensemble d'évolutions technologique (capacité mémoire, performance des processeurs et des réseaux, évolution des logiciel :interface graphique, multimédia, interface de communication),pour que tout utilisateur d'une entreprise ou autre puisse accéder à toute information autorisée par les règles de confidentialité et de sécurité de manière instantanée depuis n'importe quel poste en utilisant une interface aussi simple que possible.

Une architecteur client /serveur signifie que des machine clients (des machines faisant partie du réseau) contactent un serveur , une machine généralement très puisante , qui leur fournit des services sont des programmes fournissant des données telles que l'heure, des fichiers, une connexion.....

Cette architecture comporte plusieurs caractéristiques, on peut citer les plus importantes:

- Le serveur est fournisseur de service et client est consommateur.
- C'est toujours le client qui déclenche la demande de service. le serveur attend passivement la requête du client.
- Un serveur traite plusieurs clients en même temps et contrôle leur accès aux ressources.
- Il est possible d'ajouter et de retirer des stations clientes. Il est possible de faire évoluer les serveurs.
- Le modèle client/serveur est indépendant des plates-formes matérielles et logicielles.
- On peut modifier le serveur sans toucher au client. La réciproque est vraie.

Cette architecture est utilisée dans plusieurs domaines tels que : la gestion de base de données, système transactionnels, système de messagerie, web, Internet....etc. vu tous les avantages qu'elle comporte (des ressources centralisées :étant donnée que le serveur gère des ressource communes à tous les utilisateurs, une meilleure sécurité : car le nombre de points d'entrée

permettant l'accès aux données est moins important, une administration au niveau serveur, un réseau évolutif ou l'on peut supprimer ou ajouter des clients sans perturber le réseau).

Néanmoins, cette architecture a quelques inconvénients tels que le coût élevé du à la technicité du serveur, le maillon faible que constitue le serveur étant donné que toute l'architecteur tourne autour de lui et est donc dépendante de lui.
La figure montre le fonctionnement d'une architecture client/serveur

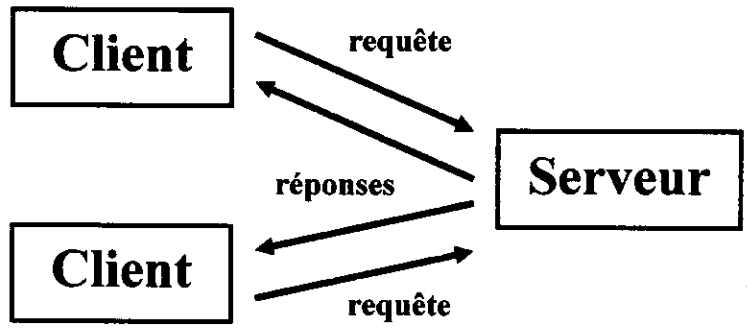


Figure 8 : *Fonctionnement d'une architecture client/serveur*

VI.1 Les socket

Comment plusieurs applications communiquent-elles simultanément sur un réseau ?

Sur un réseau, de nombreuses applications communiquent simultanément.
Lorsque plusieurs applications sont actives sur un même ordinateur, le protocole TCP/IP nécessite une méthode permettant de les distinguer. À cet effet, le protocole TCP/IP utilise un socket, également appelé point terminal dans une communication réseau, pour identifier une application particulière.

- **Définition**

Un socket est la combinaison d'une adresse IP et du port TCP ou UDP. Une application crée un socket en spécifiant l'adresse IP de l'ordinateur, le type de service (TCP pour garantir la remise des données, sinon UDP) et le port surveillé par l'application. Le composant adresse IP du socket permet d'identifier et de repérer l'ordinateur de destination, tandis que le port détermine l'application particulière à laquelle les données doivent être envoyées.

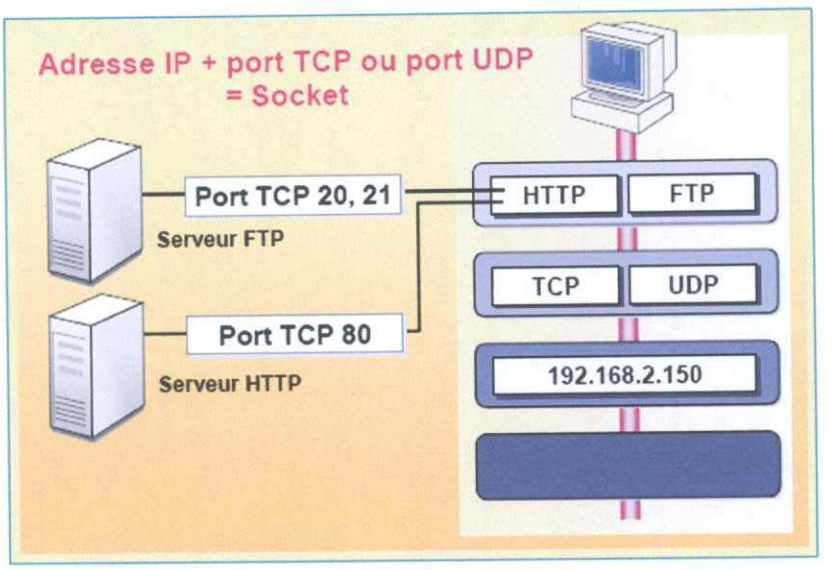


***Figure 9** : le rôle des sockets et des ports dans la communication réseau.*

- **Le port**

Un *port* est l'identificateur d'une application dans un ordinateur. Un port est associé aux protocoles TCP ou UDP de la couche Transport ; il est désigné par un port TCP ou un port UDP. Un port peut être pourvu d'un numéro quelconque situé entre 0 et 65 535. Les ports des applications TCP/IP côté serveur standard, appelés numéros de port connus, sont réservés aux numéros inférieurs à 1 024 afin d'éviter une confusion avec les autres applications. Par exemple, l'application FTP Server utilise les numéros de port TCP 20 et 21.

On va parler maintenant d'un mécanisme important pour l'implémentation d'une architecture client/serveur qui est le socket.

Dans une connexion client/serveur chaque extrémité utilise une socket, on a une socket client et une autre socket serveur.

La socket est identifié grace à son adresse IP et à son numéro de port.

La socket client demande une connexion à une socket serveur.

La socket serveur attend une demande de connexion puis dialogue avec le ou les sockets clientes dont la requête est acceptée.

Quand un processus serveur gère chaque requête de façon indépendante, il est qualifié de serveur itératif. Il gère les requêtes une par une dans leurs ordre d'arrivée mais il n'y a pas de chevauchement entre les traitements de requêtes.

A l'inverse, quand on ne peut pas prévoir le temps nécessaire à un serveur pour réponde, on dit qu'il s'agit d'un serveur concurrent. Ce serveur crée un processus distinct pour chaque requête de service. Typiquement, à chaque fois qu'une connexion est établie, le serveur lance un nouveau processus qui reste en

écoute d'éventuelles autres demandes de connexion. Cela nécessite un environnement multi-tâches.

Dans le cas du protocole TCP, ou l'on travaille en mode connecté, la figure montre une socket en mode connecte.

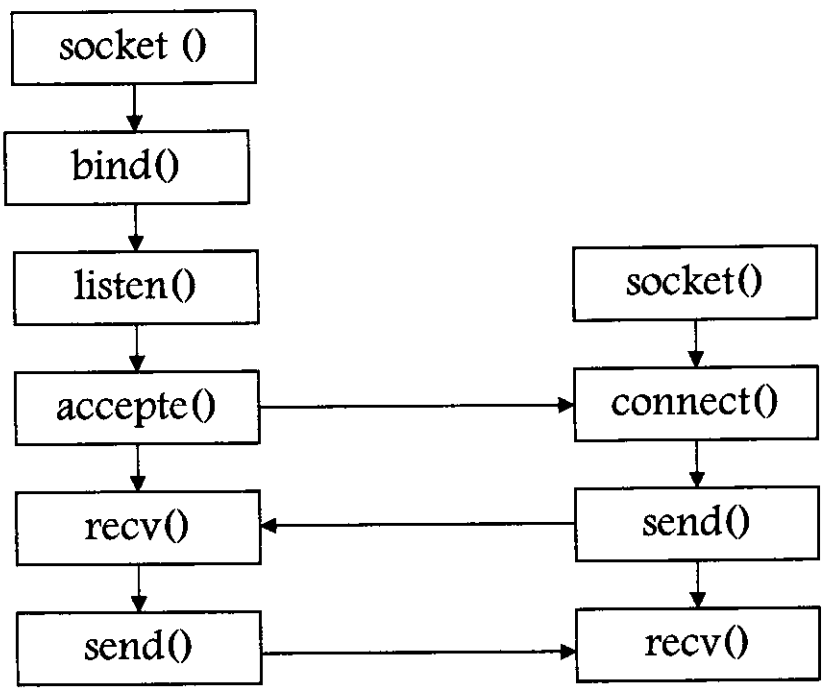


***Figure 10** : Socket en mode connecté.*

Dans le cas du protocole UDP, où l'on travail en mode deconnecté, la figure (10) montre une socket en mode déconnecté.

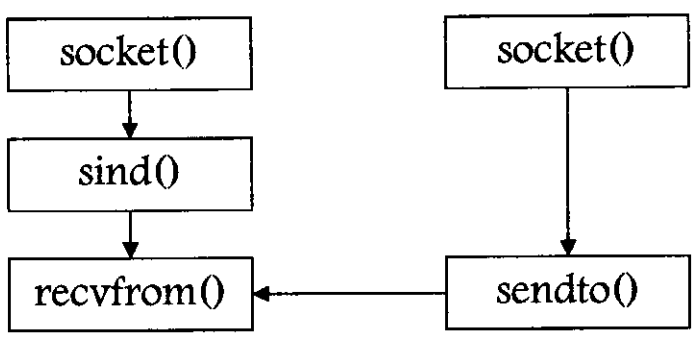


Figure 11 : Socket en mode déconnecté.

Voici les définitions des fonction citées la-dessus :

- Socket ( ): pour la création de la socket d'écoute (constricteur).
- Bind ( ) : pour l'attribution du port d'écoute.

- Listen ( ) : pour l'écoute du port, et l'attente d'une éventuelle demande de connexion de la part d'un client.
- Accep( ) : pour l'acception du client et la création d'un socket de communication.
- Recv( ) : pour la réception de données
- Send( ) : pour l'émission de données.
- Connect ( ) : pour la tentative de connexion au serveur de la part du client.

VII. CONCLUSION

Dans ce chapitre « contrôle a distance » nous avons abordé la présentation du concept de contrôle a distance, puis nous avons définis quelques notions principales qui concerne st le réseau informatique en différents aspects (physique, logique, méthodologique...).

CHAPITRE 3

Conception de l'application logicielle

I. INTRODUCTION

Dans ce chapitre nous allons présenter la conception de notre application et suivent cette conception nous allons faire la modélisation.

Pour la modélisation en utilisant le langage de modélisation UML avec cinq diagrammes de l'ensemble des diagrammes de ce langage ; en détaillant le diagramme de classe, diagramme de cas d'utilisations, diagramme de séquence, diagramme de collaboration et en finissant par le diagramme d'activité.

II. PRESENTATION ET CONCEPTION DE L'APPLICATION

Notre projet s'insère dans le cadre du développement des système informatiques qui assurées le contrôle a distance (réseau local ou Internet) des appareils électroniques (industrielles, financières, **médicaux**...etc).

Donc, nous allons configurer le PC hôte en serveur en installant un serveur web APACHE et on crier un petit site web dynamique et un système de gestion et de contrôle. Dans ces conditions, un manipulateur lointain pourra piloter les actionneurs d'une carte électronique (Exemple : 3sorties) connectée sur le port série de cet ordinateur pour manipuler les appareils médicaux voulus, et cela nécessite de développer un pilote pour l'échange des données entre eux. Une base de données sera criée pour la gestion des utilisateurs distants afin de définir les droits d'accès de ces derniers et aussi, pou garder l'état des appareils connectés...ect.

III. MODELISATION DE L'APPLICATION EN UML

Nous allons faire la modélisation en utilisant le langage de modélisation UML qu'est un langage visuel et adapté à toutes les phases du développement et compatible avec toutes les techniques de réalisation de notre l'application et indépendant du langage de programmation utilisé et qui permet différentes vues (statique, dynamique et fonctionnelle) pour représenter notre système.

Donc nous avons utilisés cinq diagrammes des l'ensemble de diagrammes du langage UML.

III.1 Diagramme de classes

Les diagrammes de classes représentent un ensemble de classes, d'interfaces et de collaborations, ainsi que des relations. Ce sont les diagrammes les plus fréquents dans la modélisation des systèmes oriente objet. Ils représentent la vue de conception statique d'un système.

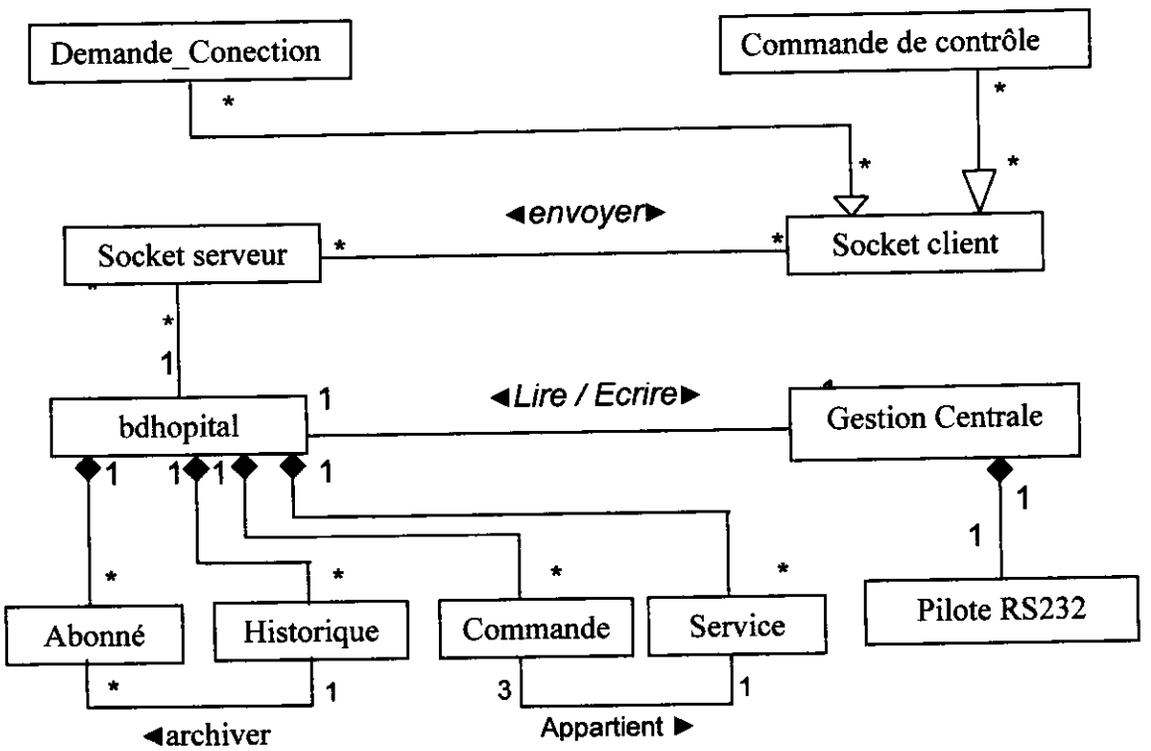


Figure 01 : *Diagramme de classes*

La figure 01 représente le diagramme de classe, de notre système qui comporte toues les classes que nous avons implémentées avec ses relations.

Nous avons représentés les classes sans ses attributs et ses fonctions pour réduire la taille du diagramme. Voici la description détaillée de chaque classe.

- **La classe socket_Serveur**

Cette classe est spécialement destinée à l'échange des données (émission / réception, coté serveur) avec le client. Elle comporte des attributs et des fonctions qui aident à cette génération (Figure 02).

Socket_Serveur	
-Nom	: String
-N° port	: Int
-Adresse	: String
+Socket()	
+Bind()	
+Listen()	
+Accept()	

Figure 02 : *La classe socket_Serveur*

Socket() : pour la création de la socket d'écoute.
Bind() : pour l'attribution du port d'écoute.
Listen() : pour l'écoute du port, et l'attente d'un éventuelle demande connexion de la part d'un client.

- **La classe socket_Client**

Cette classe est spécialement destinée à l'échange des données (émission / réception, coté client) avec le serveur. Elle comporte des attributs et des fonctions qui aident à cette génération (Figure 03).

Connect() : pour la tentative de connexion au serveur de la part du client.

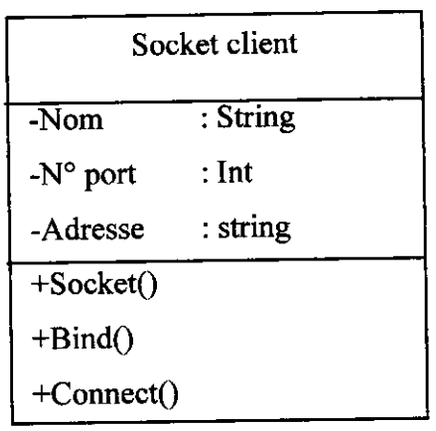


Figure 03 : *La classe socket_Client*

- **La classe Demande_conection**

Cette classe est une classe générée de la classe Socket_Client, elle permet au client de demander la connexion au serveur. Elle comporte les attributs et les fonctions suivants :

Demande_Conection() : constricteur pour la création et l'initialisation d'un nouveau demande de connexion au serveur, il retourne vrais en cas de succès.

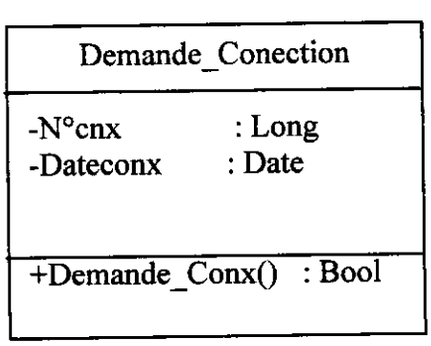


Figure 04 : *Classe Demande_Conection*

- **La classe Commande de Contrôle**

Cette classe est une classe générée de la classe Socket_Client, elle permet au client d'envoyer des commandes au serveur (la commande des appareils médicaux). Elle comporte les attributs et les fonctions suivants :

Commande de Contrôle
-N°cmd : Int -Datecmd : Date
+Commande ()

Figure 05 : *La classe Commande de contrôle*

Commande () : constructeur pour la création d'un nouvelle commande de au serveur, il retourne vrais en cas de succès.

- **La classe bdhopital**

Cette classe est destinée pour le stockage des données, elle permet au serveur de traiter les sockets des clients efficacement (elle fournit les données nécessaires au serveur pour rendre les services au client).

La classe de bdHopital comporte deux sous classes, ces sous classes sont montrées dans la figure 06.

NomBdd : nom de la base de données qui l'identifier.
MotPasseBdd : mot de passe de la base de donnes qui autorise l'accès a ce dernier.
creerTable() / modifTable() : permets au serveur de creer/modifier un table respectivement.
changeMotPasse() : permet au serveur de changer le mot de passe de la BDD.

bdhopital
-NomBDD :String -MotPasseBDD : String -AliasName :String
+crierTable() :Bool +modifTable() +suppTable() +changeMotPasse()

Figure 06 : *La classe bdhopital*

- **La classe Commande**

Est une sous classe de la classe bdhopital, elle permet au serveur de visualiser et de modifier l'état des appareils de manière administrative (prioritaire).

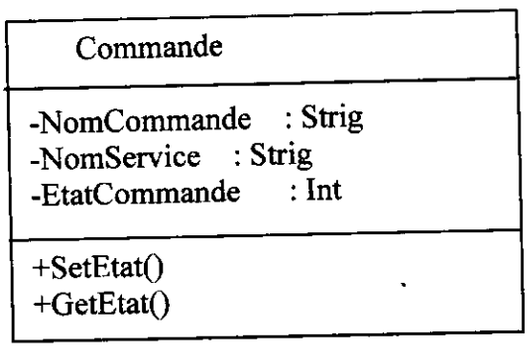


Figure 07 : *La classe commande*

NomCommande : nom de commande qui l'identifier.
NomService : nom de service qui l'appartient.
EtatCommande : l'état actuelle de commande (1 : Activée, 0 : Désactivée).
SetEtat () : permet au serveur de modifier l'état d'un commande.
GetEtat () : permet le de lire l'état d'un commande.

- **La classe Abonné**

Cette classe est une sous classe aussi de la classe bdhopital, elle permet au serveur de insérer, modifier...un client (autorisation de contrôle a fin d'identification).

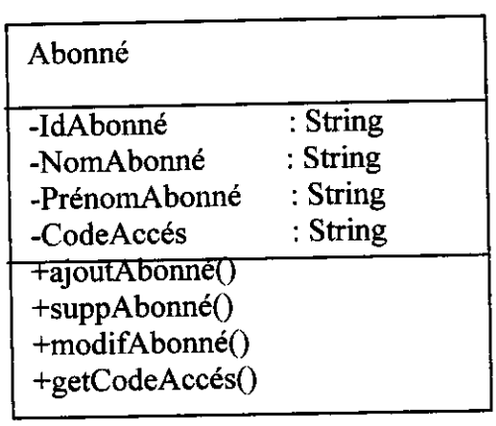


Figure 08 : *La classe Abonné*

IdAbonné : identification personnelle d'un abonné.
CodeAccés : code d'autorisation pour modifie les états des appareils.
ajoutAbonné() : permet au serveur d'ajouter un nouveau client.
suppAbonné() : permet au serveur de supprimer un abonné.
modifAbonné() : permet au serveur modifier les attributs d'un abonné.

- **La classe service**

Cette classe est une sous classe aussi de la classe bdhopital, elle permet au serveur de insérer, modifier...service correspond à un appareil médical.

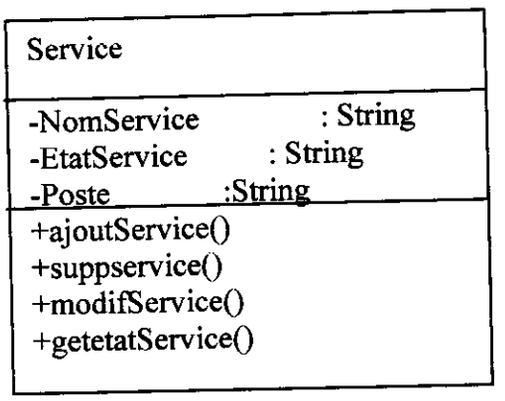


Figure 09 : *La classe Service*

NomService : identification d'un Service.
EtatService : état de service (activée ou désactivée).
ajoutService() : permet au serveur d'ajouter un nouveau service.
suppService() : permet au serveur de supprimer un service.
modifService() : permet au serveur modifier les attributs d'un service.

- **La classe Historique**

Cette classe est une sous classe aussi de la classe bdhopital, elle permet au serveur de garder un historique d'accès des abonnés.

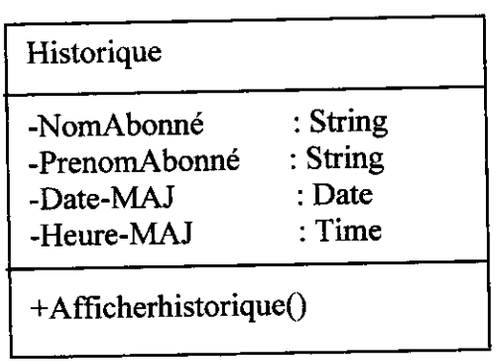


Figure 10 : *la classe historique*

Date-MAJ : date de fait du contrôle.
Heure-MAJ : heure de fait du contrôle

- **La classe gestion Centrale**

Cette classe corresponde te a la forme principale de l'application serveur, elle est destinée pour la lecture des données à partir de la base de données

bdhopital » et l'émette vers les appareils médicaux à travers du carte électronique sou forme des signaux et en fonction de la sous classe PiloteRS232. Elle à également pour rôle, la gestion standard de la base de données bdhopital. Elle comporte des attributs et des fonctions qui aident à cette génération

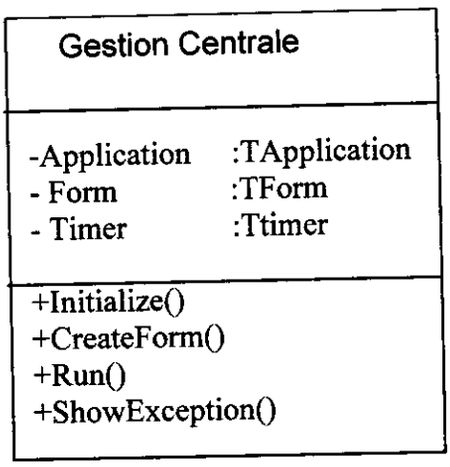


Figure 11 : *La classe Gestion Centrale*

Initialize(): initialise l'application
CreateForm(): crier la forme de l'application
Run(): lance l'exécution de l'application
ShowException() : gère les exceptions.

- **La classe Pilote RS232**

Cette classe est une sous classe de la classe Gestion Centrale, elle permette au dernier de réagir les commandes vers le port série COM1.

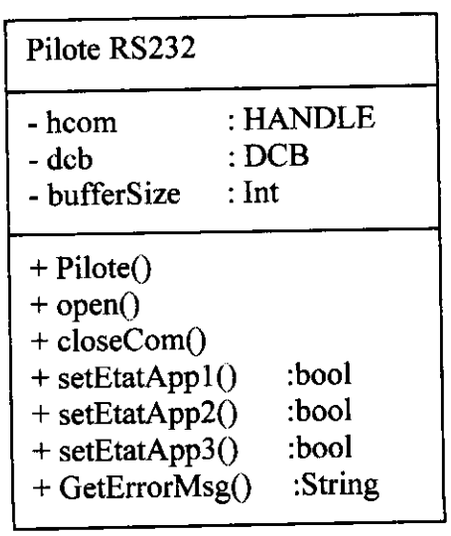


Figure 12 : *La classe Pilote RS232*

hcom : Fichier de sortie sur le port COM.
dcb : Configuration du Port.

bufferSize: la taille du buffer de port
PiloteRS232() : constructeur de la classe permet l'initialisation de ses paramètres.
Open () : ouvrir le port Com "numPort" de vitesse "vitesse".
closeCom() : fermer le port série.
setEtatCmd1 () : Activée le commande n°1. Return: true if succès.
setEtatCmd2 () : Activée le commande n°2. Return: true if success.
setEtatCmd3 () : Activée le commande n°3. Return: true if success.
GetErrorMsg() : Retourne un message d'erreur généré par ces fonctions.

III.2 Diagramme de cas d'utilisation

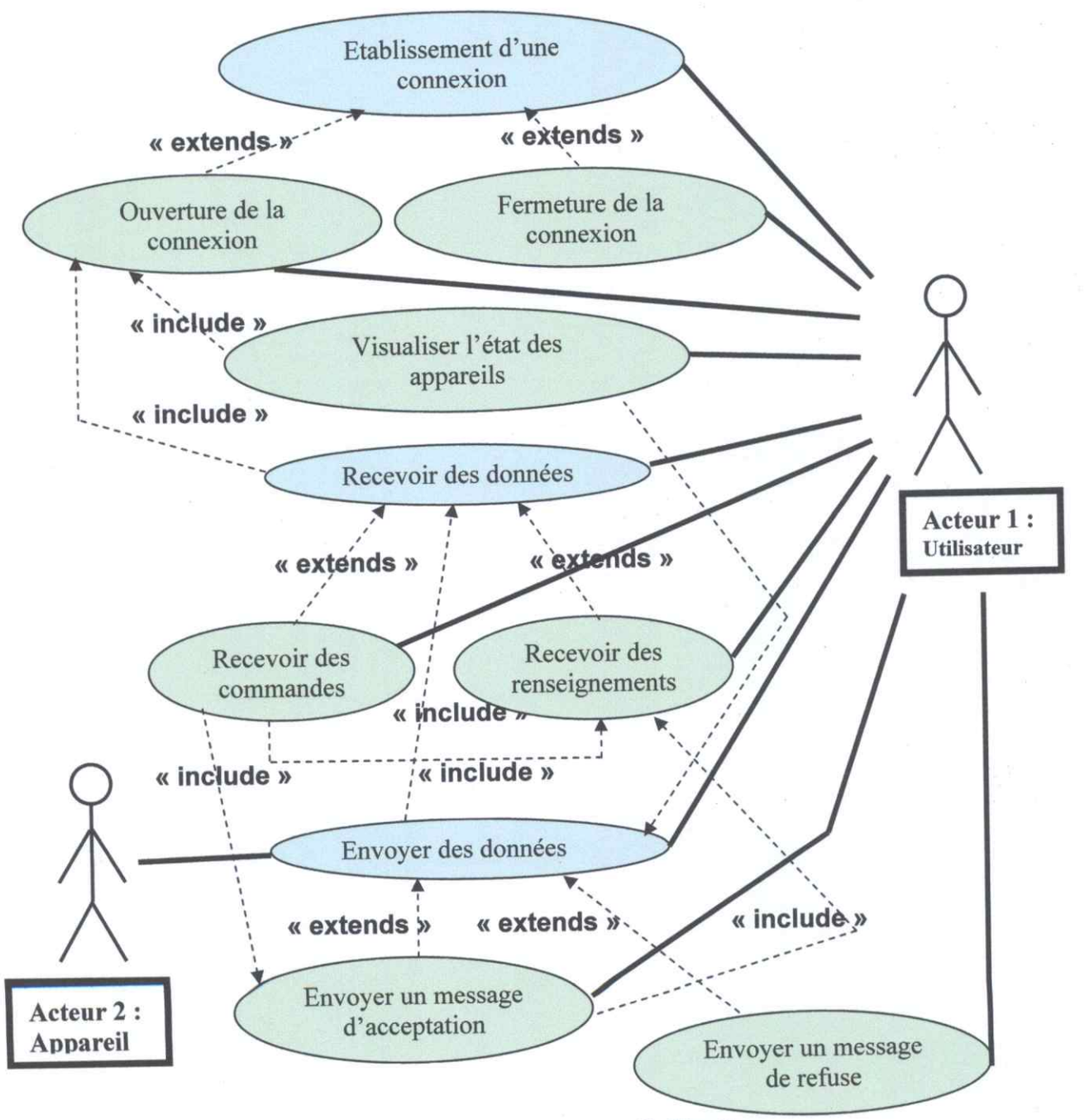


Figure 13 : *Diagramme de cas d'utilisation*

Les diagrammes de cas d'utilisation représentent un ensemble de cas d'utilisation et d'acteur (sorte de classe particulière) et leurs relations. Ils sont particulièrement importants dans l'organisation et la modélisation des comportements d'un système.

Dans notre étude conceptuelle nous définissons deux acteurs (utilisateurs externes) :

- L'utilisateur (l'opérateur humain) : il a la possibilité de ouvrir/fermer la connexion, visualiser l'état des appareils médicaux liés au machine serveur (a partir du port série) en temps réel et d'envoyer des commandes a des appareils a condition d'avoir l'acceptation du serveur (vérification de la base de donnée « bdhopital») et cela afin d'envoyer ses renseignements (Nom, prénom, code d'accès).

- Les appareils médicaux : les appareils s'interagits par la machine serveur de notre système, et à travers de l'interface connectée au port série, et a partir de la base de donnée « bdhopial », elle peut réserver ces états.

III.3 Diagrammes d'interaction

Le diagramme d'interaction représente une interaction, c'est-à-dire un ensemble d'objets et leurs relations, ainsi que les messages qu'ils peuvent s'échanger. Ils représentent la vue dynamique du système.

On a deux types de diagramme d'interaction :

III.3.1 Diagramme de séquence

Les diagrammes de séquences sont des diagrammes d'interactions qui mettent l'accent sur le classement chronologique des messages.

RMQ : App_med : les appareil médical.

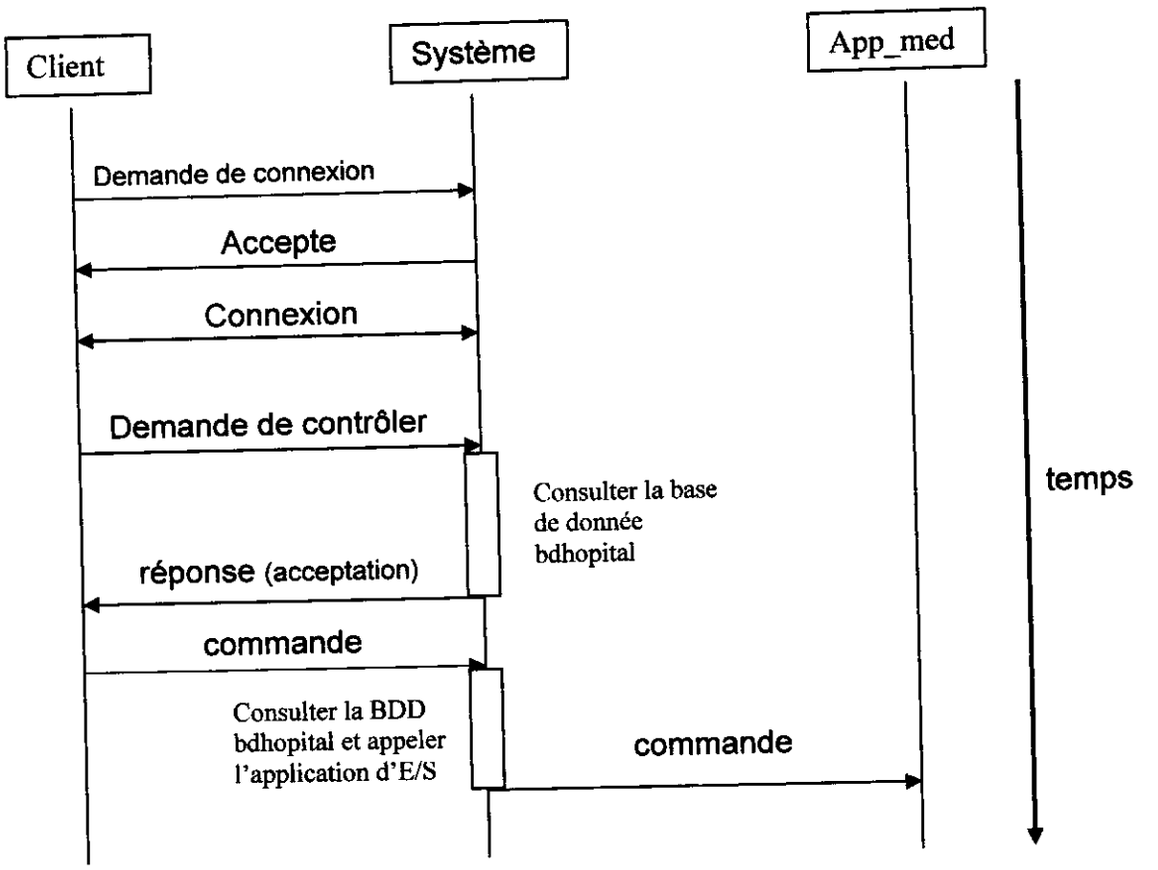


Figure 14 : *Diagramme de séquence (cas d'acceptation)*

Au lancement du système, le client établie une connexion avec le serveur, sans laquelle aucune autre séquence d'utilisation n'est possible.

Après cette séquence, il existe une séquence conditionnelle c'est à dire ce qui suite .dépend de celle-ci, tel que le client demande d'envoyer des commendes au app_med(Appareil médical), le serveur le rendre une réponse (acceptation ou refuse) qui dépend de la BDD (la consultation de la base de donnée 'abonné') :

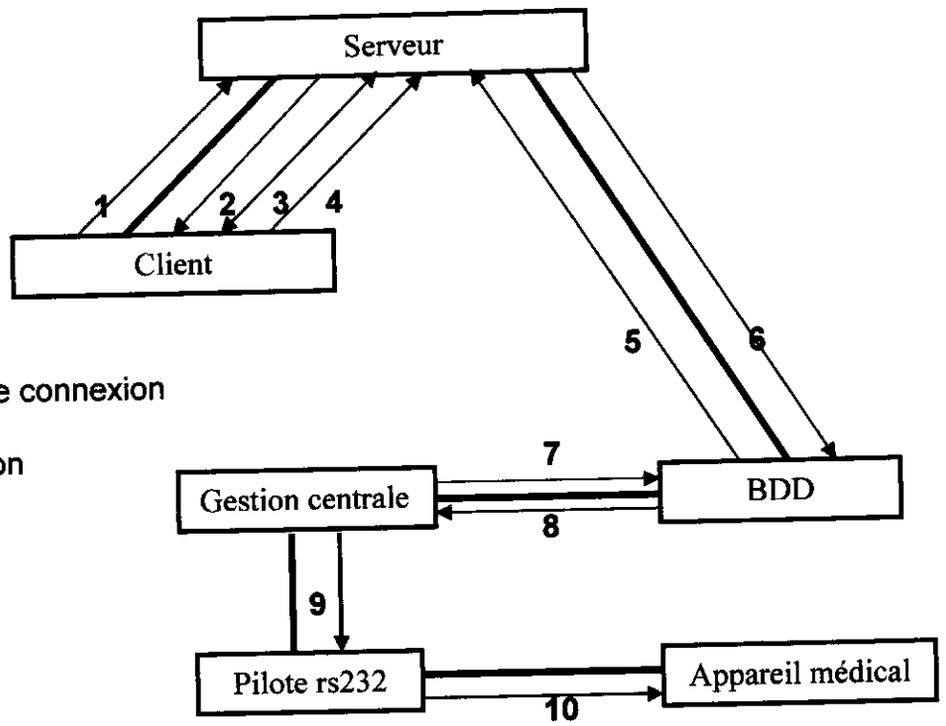
Cas d'acceptation : il existe trois séquences possibles :

- Envoi des commendes
- Envoi de requête
- Réception des données

Cas de refus : dans ce cas le rôle du client dans ce système consiste a voir l'état des appareils médicaux.

III.3.2 Diagramme de collaboration

Les diagrammes de collaborations sont des diagrammes qui mettent l'accent sur l'organisation structurelle des objets qui envoient des messages.



1- demande connexion
2- accepte
3- connexion
4- requête

5- donnée
6- requête

7- requête
8- donnée
9-10- commande

Figure 15 : *Diagramme de collaboration*

III.3.3 Diagramme d'activité

Les diagrammes d'activités sont un type particulier du diagramme d'états-transitions qui décrit la succession d'activités aux seins d'un système. Il représente une vue dynamique du système.

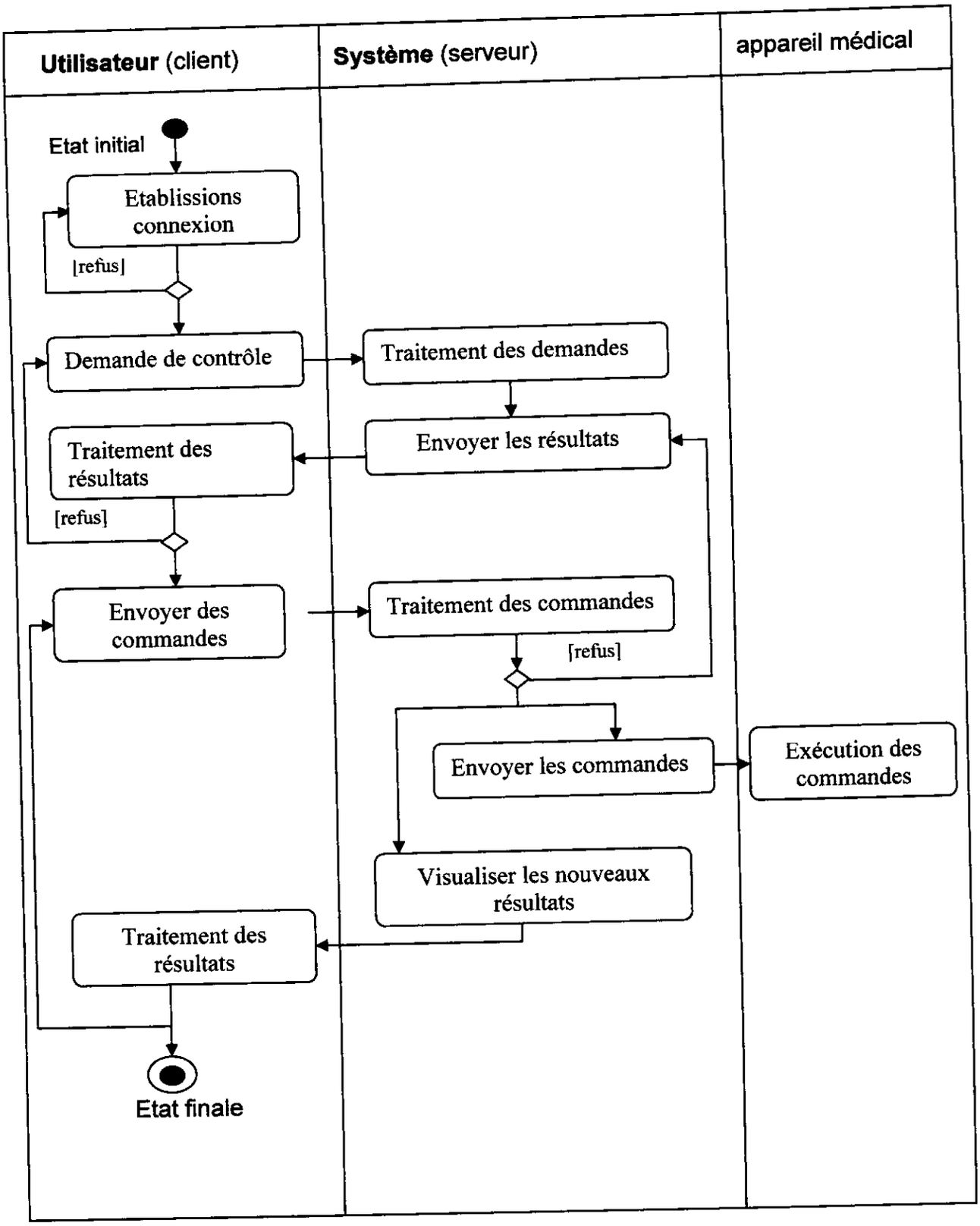


Figure 16 : *Diagramme d'activités pour le système*

IV. CONCLUSION

Dans ce chapitre, nous avons commencés par la présentation de la problématique et les objectifs à atteindre avant la réalisation de notre application. Par la suite, nous sommes passés à la solution proposée avec les défirent diagrammes de modélisation du langage UML.

Dans le chapitre suivant, nous allons présenter la réalisation de notre système suivent la conception proposée, avec des testes réels pour la validation.

CHAPITRE 4

Réalisation du système

I. INTRODUCTION

Après la description de la conception et des solutions proposées dans le chapitre précédent, l'étape suivante est de passer à la présentation de l'application logicielle et la réalisation de ces solutions avec la prise en compte des différents besoins des utilisateurs.

Pour ce la, dans ce chapitre nous allons présenter le travail réalisé et l'environnement de travail en premier nous survolerons les différents outils informatique utilisés dans l'implémentation de notre application.

Nous présenterons un aperçu du système d'exploitation Windows et les langages de programmation C++, et autres outils de développements qui nous intéresse : le serveur Web APACHE MySql et PHP.

II. PRESENTATION GENERALE DE L'APPLICATION

Notre application est un logiciel au sens d'un site web dynamique, qui comporte des fonctionnalités de contrôle a distance via un serveur web, c'est le coté « serveur –client », la machine serveur utilise une carte d'interfaces (carte électronique) qui doit la piloté pour l'envoi des données au appareils médicaux, et cela c'est le coté « serveur- appareils médicaux ».

Ce travail est un ensemble de fenêtres Windows permettant un affichage graphique de l'état des appareil médicaux en temps réel avec possibilité de les commandés à distance (fenêtre de type html). Toutes ces fonctionnalités sont très utiles dans les ateliers et les entreprises et dans divers domaines.

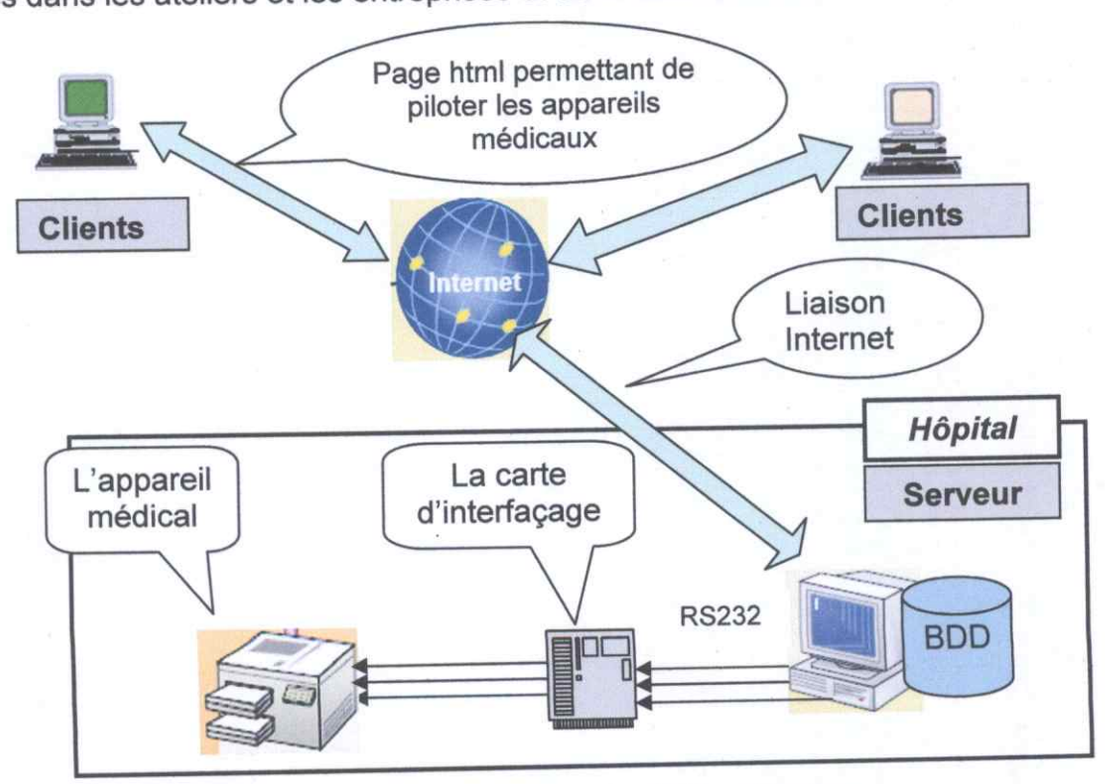


Figure 1 : *schéma générale de l'application*

III. ENVIRONNEMENT DE DEVELOPPEMENT

Nous avons implémenté notre application sur un micro-ordinateur doté d'un processeur de 3 GHZ de fréquence, 256 MOct de RAM.

L'application est réalisée sur une plate forme Windows XP avec l'utilisation du langage C++Builder6 pour la pilotage de la carte électronique car il accepte l'orienté objet et il est doté d'une interface simple permette l'interaction de l'utilisateur avec la machine, et du langage PHP pour la réalisation des pages web car c'est le langage le plus efficace et le plus utilisé pour la création des sites web, et facile de connexion avec d'autre outils de développements, plutôt MySql.

Notre application est une application interactive, puisque l'utilisateur a la possibilité de visualiser les états des appareils en temps réel et peut les contrôler, nous étions obligé d'utiliser un serveur Web et un serveur de base de donnes.

Le premier c'est pour la gestion de communication et rendre ce service de contrôle fiable et efficace. Dans notre cas, nous avons utilisé le serveur Web APACHE et le deuxième, c'est pour la création et la gestion de la base de données qui stock les données nécessaires au serveur Web. dans notre cas, nous avons utilisé MySql.

Dans la suite de ce chapitre nous allons détaillés ces outils et leur manipulations :

III.1 Le système d'exploitation [14]

Ces avantages et autres (notre manipulation suffisante de ce système d'exploitation), qui nous permettent de choisir **Windows XP Professionnel** comme une plate forme de base.

III.2 Le serveur EasyPHP [16]

"**EasyPHP 1.8**" permet d'installer tout le nécessaire pour commencer à programmer en PHP avec Apache et Mysql. Il simplifie toute l'installation grâce à un setup automatisé et évite toute la configuration fastidieuse du serveur. La version 1.8 est composée de : Apache 1.3.33 - Php 4.3.10 - Mysql 4.1.9 - Phpmyadmin 2.6.1

EasyPHP installe et configure automatiquement un environnement de travail complet permettant de mettre en oeuvre toute la puissance et la souplesse qu'offre le langage dynamique PHP et son support efficace des bases de données. EasyPHP regroupe un serveur Apache, une base de donnée MySQL, le langage PHP ainsi que des outils facilitant le développement des sites ou des applications.

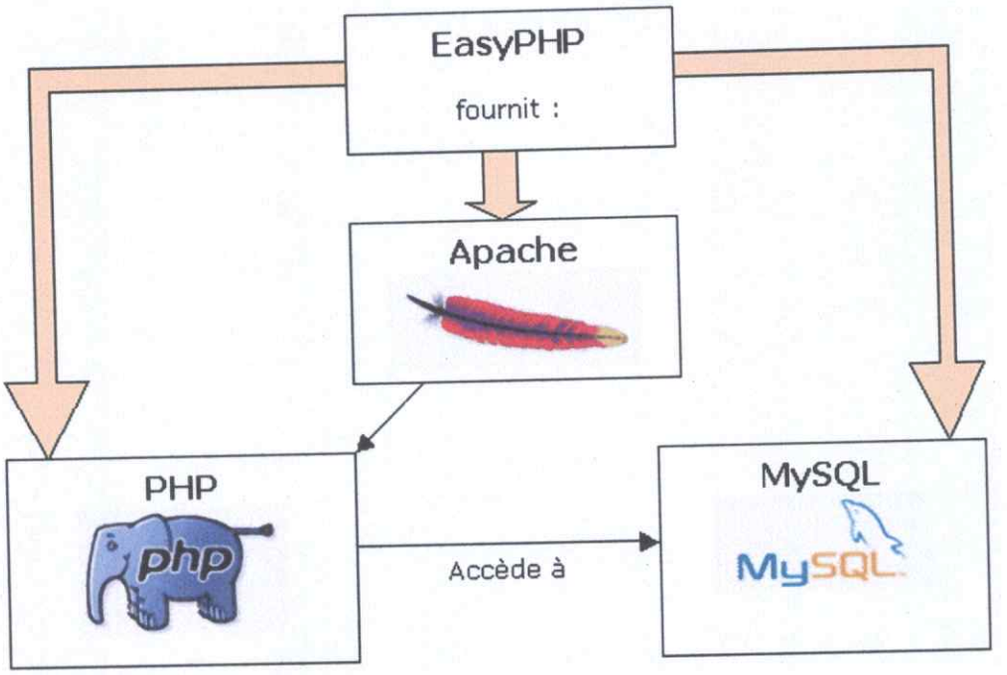


Figure 2 : EasyPHP (composition et fonctionnement)

EasyPHP est un donc paquetage contenant à la fois Apache, PHP et MYSQL.

➢ **Les fonctionnalités d'EasyPHP**

Chose la plus importante, EasyPHP propose le téléchargement en une fois et l'installation en un assistant des trois programmes précédemment cités, Apache, PHP et MySQL.

Cela permet d'installer automatiquement ceux-ci, en se libérant des problèmes liés à la configuration manuelle qui est souvent nécessaire lorsqu'on les installe séparément

Lorsqu'EasyPHP est lancé, les serveurs Apache et MySQL sont automatiquement lancés (il est même possible de le faire automatiquement au démarrage de Windows). Une petite icône s'installe dans la barre des tâches, à côté de l'horloge, permettant un accès rapide aux fonctions proposés par EasyPHP :

- Arrêter et Redémarrer les serveurs Apache et MySQL.
- Accéder au "Web local", c'est-à-dire la racine des sites webs.
- Un panneau d'administration en PHP
- Un outil de configuration d'EasyPHP
- L'accès aux logs
- L'aide

Dans notre projet nous allons utiliser EsyPHP1.8 (Apache 1.1.33 + PHP 4.3.10 + PHPMyAdmin 6.2.1 + Mysql 4.1.9).

➢ **Le serveur Apache**

Un serveur Web est un logiciel servant des requêtes d'un navigateur respectant le protocole de communication client-serveur HyperText Transfer Protocol (HTTP), et écoute en permanence sur un port donné (en général 80). Apache (existe sur pratiquement toutes les plates-formes, y compris Windows, Unix, Solaris...).

Apache est le serveur Web le plus utilisé sur le marché (75% du marché des serveurs Web).

➢ **Le serveur MySql**

MySQL est un véritable serveur de base de données SQL multi-utilisateur et multi-threaded. SQL est le plus populaire langage de base de données dans le monde. MySQL est une configuration client/serveur ce qui consiste en un serveur démon mysqld, différents programmes clients et des librairies.

SQL est un langage standardisé qui rend facile le stockage, la mise à jour et l'accès à l'information. Par exemple, nous pouvons utiliser les requêtes SQL pour récupérer des informations sur un produit ou stocker des informations client sur un site web. MySQL est suffisamment rapide et flexible pour gérer des historiques et des images.

Les principaux objectifs de MySQL sont la rapidité, la robustesse et la facilité d'utilisation. MySQL a été originellement développé parce que nous avions besoin d'un serveur SQL qui puisse gérer des grandes bases de données de manière plus rapide que ce que pouvaient offrir les distributeurs de bases de données.

La base sur laquelle MySQL est construite est un ensemble de routines qui ont été largement éprouvées pendant des années dans un environnement de production exigeant. Même si MySQL est encore en développement, il propose déjà un ensemble de fonctionnalités riches et extrêmement utiles.

Donc ,nous allons utiliser le plus populaire des serveurs de bases de données relationnelles Open Source, bien que notre serveur d'une version 4.1.9, puisque ces dernières versions dispose généralement de meilleures performances et une meilleure stabilité.

➢ **Le langage PHP**

Le langage PHP est un langage extrêmement puissant : il permet de créer des pages web, au travers desquelles l'utilisateur peut échanger des informations avec le serveur ; c'est ce qu'on appelle des pages web dynamiques. Programmer en PHP est assez simple. En revanche, PHP n'est pas un langage compilé, c'est un langage interprété par le serveur : le serveur lit le code PHP, le transforme et génère la page HTML.

III.3 Le langage C++ [17]

C++ est un langage de programmation. Ce langage permet la programmation sous de multiples paradigmes comme, par exemple, la programmation procédurale, la programmation orientée objet et la programmation générique. Au

cours des années 1990, C++ est devenu l'un des langages de programmation les plus populaires dans l'industrie informatique. Le langage C++ n'appartient à personne et par conséquent n'importe qui peut l'utiliser sans payer de droits.

Notre maîtrise suffisante avec ces avantages, qui nous allons diriger d'utiliser le C++ (Builder C++ 6) pour la création d'une application d'E/S (pilotage) de trois sorties avec le port série (RS232).

IV. PRESENTATION DU TRAVAIL REALISE

IV.1 Architecture Client/Serveur

La première étape dans le contrôle à distance des appareils médicaux est l'établissement de la connexion entre les PC utilisateurs externes (client) et le PC hôte (serveur), notre application basée sur l'architecture Client/Serveur qui utilise le mécanisme des sockets. La figure 2 montre le flux des données dans notre application.

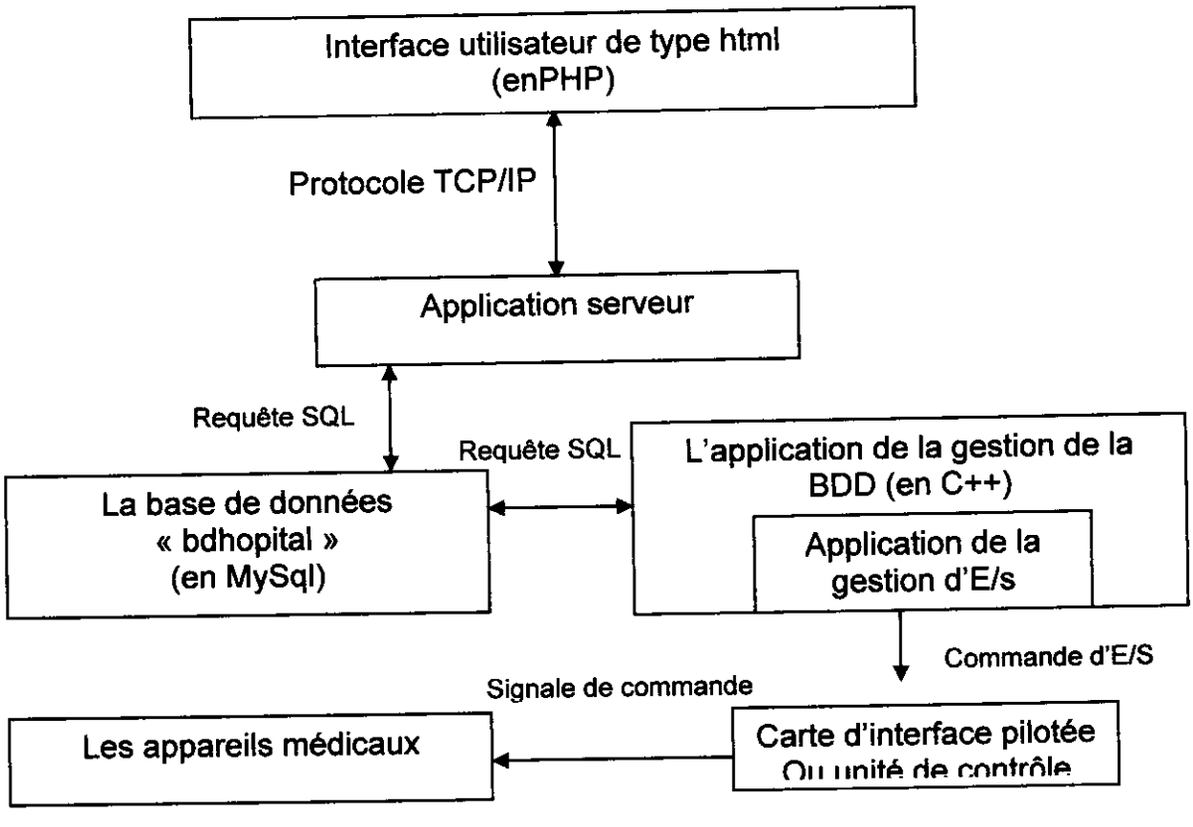


Figure 3 : *Flux des données entre les différentes parties fonctionnelles du système*

Après l'établissement de la connexion entre le client et le serveur le client pourra échanger les données avec le serveur et pourra avoir accès à ses ressources (les appareils médicaux dans notre cas) pour les contrôler.

IV.1.1 Les interfaces Client

Notre application possède une interface utilisateur, qui permet aux utilisateurs distants de communiquer avec le serveur. Cette interface est de type html et est généré par une scripte PHP au niveau de l'application serveur.

- **Algorithmes**

Les utilisateurs peuvent suivre l'algorithme suivant pour contrôler les appareils :

1. *demande de connexion (valider le url) ;*
2. *si refus la demande aller à 1 ;*
3. *sinon la demande (affichage du page web) ;*
4. *connexion a la base de donnée ;*
5. *visualiser état des appareils ;*
6. *demande de contrôler les appareils*

- saisir le nom et prénom et code d'accès ;
- si le nom ou prénom ou code d'accès incorrects aller a 6;
- sinon peut contrôler les appareils ;

7. *fin de connexion ;*

- **L'interface Utilisateur**

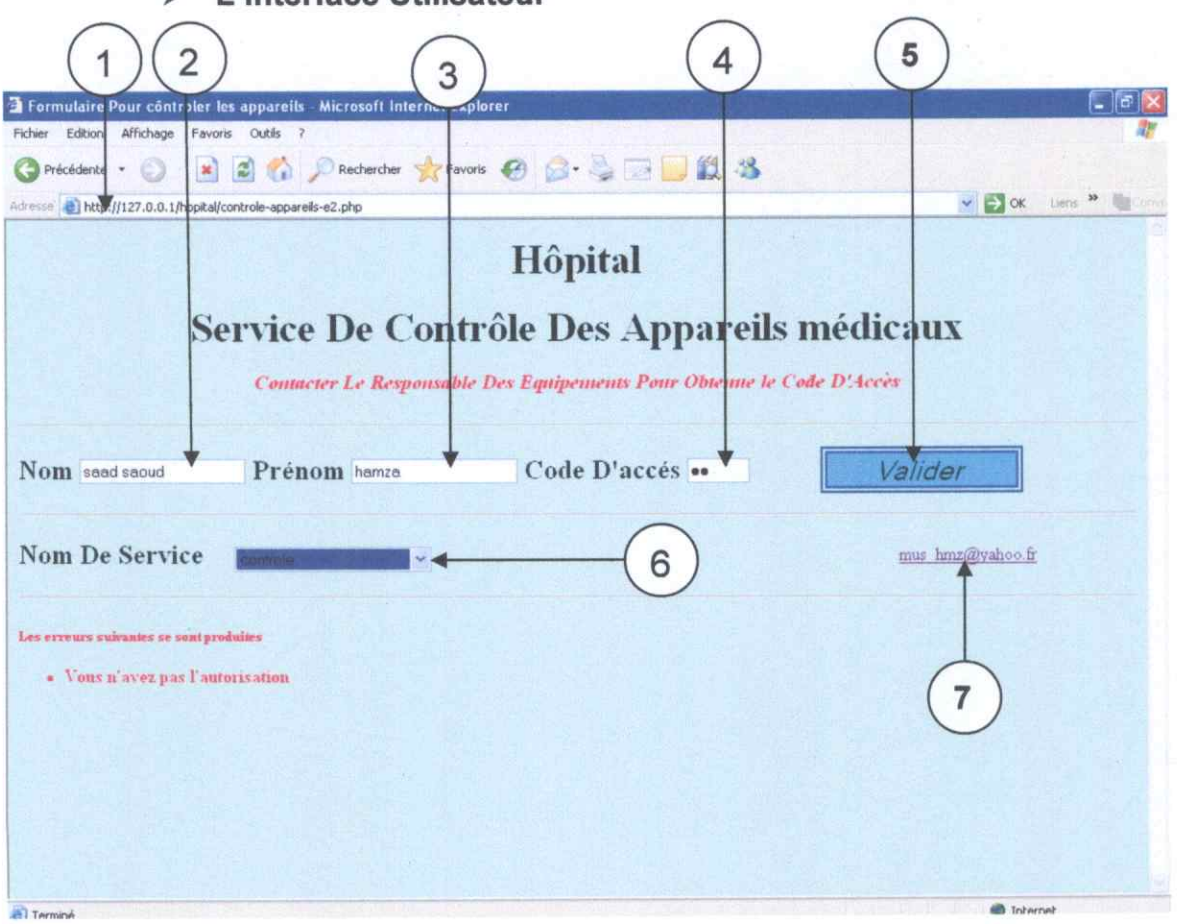


Figure 4 : *l'interface utilisateur*

1 : barre d'adresse : champ éditable permet la connexion au site de l'application.
2 : champ de saisie pour le Nom de l'utilisateur.
3 : champ de saisie pour le Prénom de l'utilisateur.
4 : champ de saisie pour le Code d'accès de l'utilisateur.
5 : bouton « Valider » valider les information saisir dans les champs (2, 3,4).
6 : liste permettent de sélectionné le nom de service voulez contrôler.
7 : l'adresse E-mail des réalisateur de l'application, s'il y des remarques.

- **L'utilisation de l'interface d'utilisateur**

Cette interface est apparue des que l'utilisateur valide l'adresse complète de l'application http://mustapha/hopital/controle-appareils.php , (cas d'un réseau local avec le serveur Apache).

Cette interface permet à un utilisateur (abonné) de envoyer ces renseignements au serveur pour lui donne la permission de contrôle. Un abonné est un utilisateur qui a le droit de contrôler les appareils médicaux (ex : le chef service), a partir de leurs nom, prénom et code d'accès qui doit le saisir dans les champs (2, 3 et 4) respectivement. Après la validation, une autre page sera affichée (figure 5).

➢ **L'interface abonné**

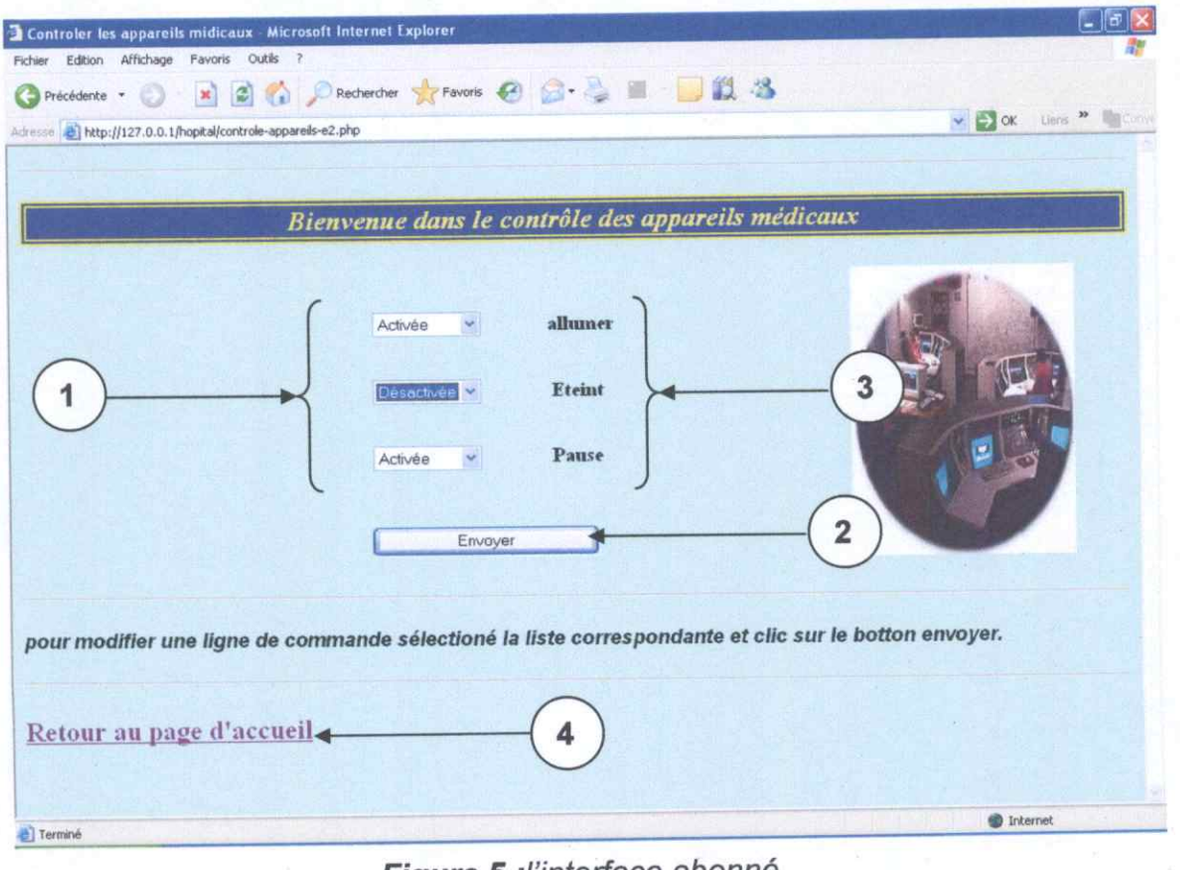


***Figure 5** :l'interface abonné*

1 : trois listes de menu permettent de sélectionner les états des commandes.
2 : bouton 'Envoyer' permet de valider les choix sélectionnés.
3 : trois labels permettent de visualiser les noms des commande qui envoyer vers les appareils médicaux.
4 : réafficher la page d'accueil.

- **Utilisation de l'interface d'abonné**

Cette interface permet à l'abonné de contrôler les appareils médicaux et en temps réel, il suffit de sélectionner la commande correspondante à l'appareil qu'on veut contrôler, puis, cliquer sur le bouton 'envoyer' pour appliquer ces choix, pour les envoyer au serveur.

Après la validation des modification une autre page sera affichée indique les changements effectués.

➢ **L'interface confirmation**

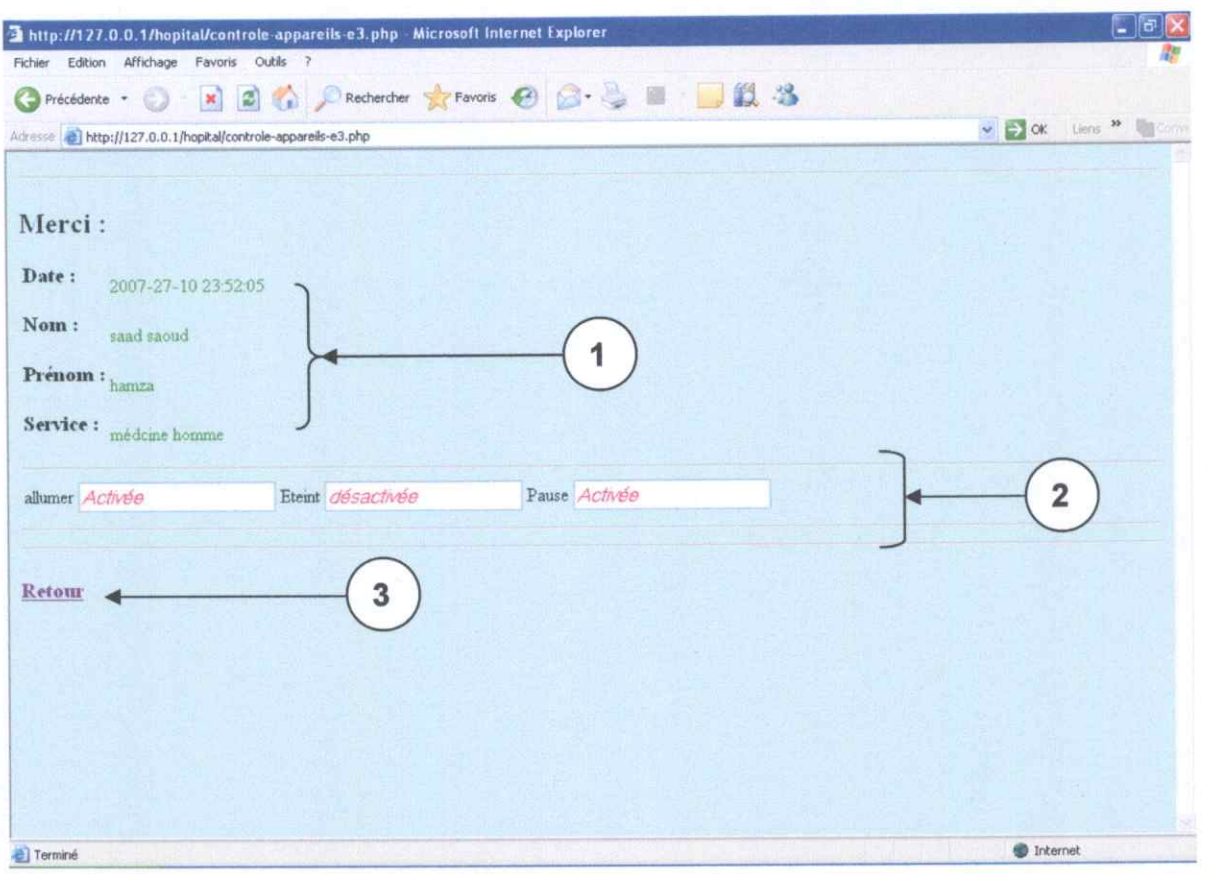


Figure 6 : *l'interface confirmation*

1 : les informations du client.
2 : les modifications effectuées.
3 : réafficher la page précèdent.

IV.1.2 Les interfaces de Serveur

Les interfaces du serveur permettent à un utilisateur administrateur de gérer les services du serveur graphiquement et de manière facile tel que la gestion de connexion, les clients et abonnés, les appareils...etc.

- **Algorithmes**

L'application qu'il existe dans le PC hôte (Serveur) suive la procédure suivante :

1. *demande de convexion avec mysql ;*
2. *saisir nom d'utilisateur et le mot passe ;*
3. *ouvrir la connexion avec serveur mysql et ouvrir le port COM ;*
4. *répéter*
   Consulter la basse de donnée ;
   Afficher l'état de chaque appareil ;

   Jusqu'à faux ;
5. *MAJ la table abonné ;*
6. *MAJ la table appareil ;*
7. *envoyer les commandes sur le port ;*
8. *sauvegarder l'historique d'accès ;*
9. *fermer la connexion et fermer le port;*

- **L'interface Gestion Centrale**

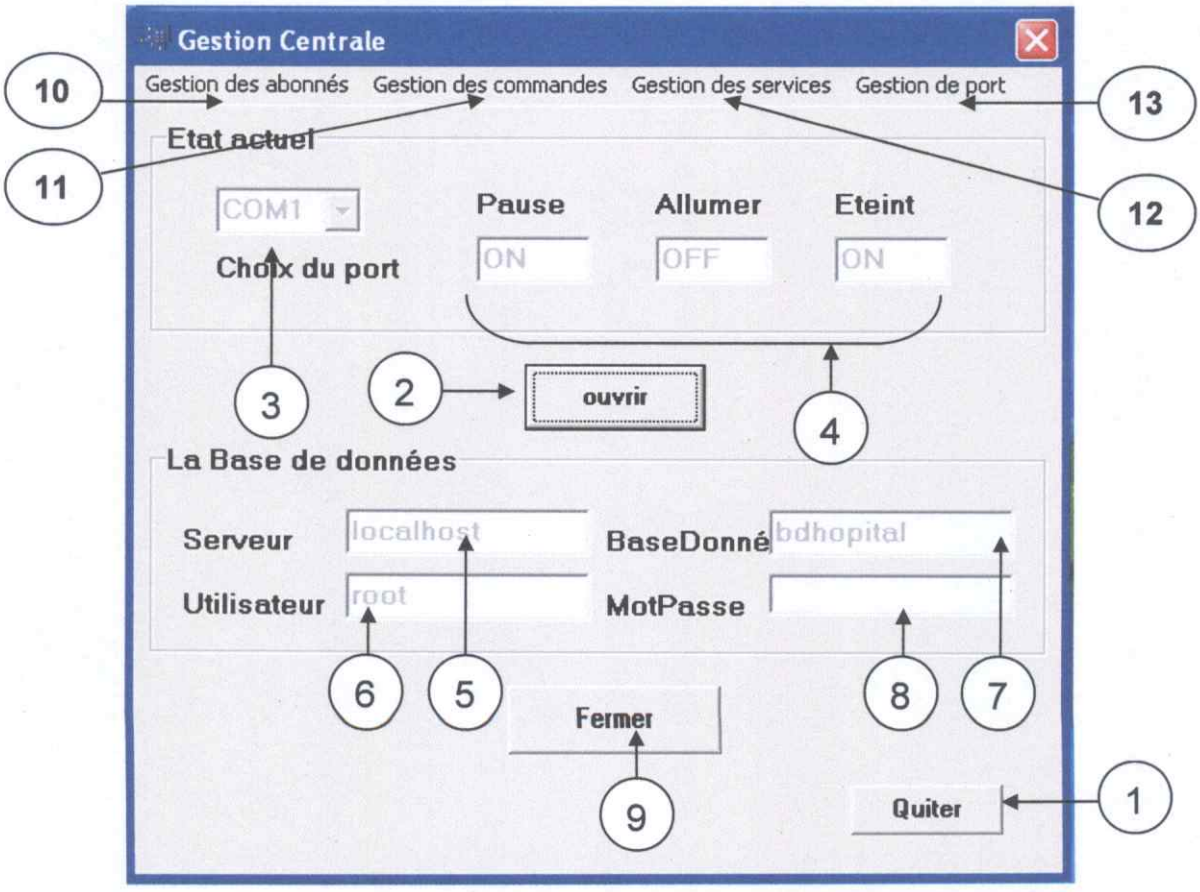


Figure 7:*L'interface Gestion Centrale*

Cette interface est l'interface principale de l'application serveur tel que touts les fonctionnements du projet sera commencés à partir de cette interface.

Cette interface permet de ouvrir/fermer la connexion avec la base de données 'bdhopital', et de choisir le numéro de port Com à utilisé c'est à dire permettent aux abonnés de contrôler les appareils médicaux en temps réel, et autres, avec quatre menus qui permettent la gestion des appareils, des abonnés et le port.

1 : le bouton 'Quitter' permet de fermer l'application

2 : le bouton 'ouvrir' permet d'ouvrir la connexion au serveur EasyPHP et d'initialiser la lecture/écriture des données avec la base de données.

3 : champ de sélection permet de choisir le groupement des appareils pilotés au port série correspondant.

4 : trois champs non éditables visualisant l'état actuel des commandes.

5, 6, 7, 8 : champs non éditables permettant respectivement la visualisation de : nom de serveur, nom d'utilisateur, nom de la base de données et le mot de passe d'utilisateur.

9 : le bouton 'Fermer' permet de fermer la connexion au serveur et de fermer le port série.

10 : menu permet d'exécuter tous les fonctions qui consternent les abonnés.

11 : menu permet d'exécuter tous les fonctions qui consternent les commandes de chaque service.

12: menu qui faire des appelles à tous les fonctions qui consternent les commandes.

13 : menu pour la gestion de port.

- **Utilisation de l'interface Gestion Centrale**

L'utilisateur administrateur qui veut démarrer ce système doit sélectionner le numéro de port série correspondant au l'appareil à contrôler, puis, il va cliquer sur le bouton 'ouvrir' pour appliquer et voir les états (ses changements) actuels des appareils pilotés au port sélectionnés qui sont affichés dans le champ 4 (ON : Activée et OFF : Désactivée). La déconnection du système avec la BDD se fait par le bouton 'Fermer'.

1. Le menu 'gestion des abonnés' (champ 10) fournit des fenêtres supplémentaires permettent au l'administrateur de ajouter et supprimer, modifier les abonnés ainsi voir tous les abonnés avec l'historique de d'accès.

➢ **L'interface Modification abonné**

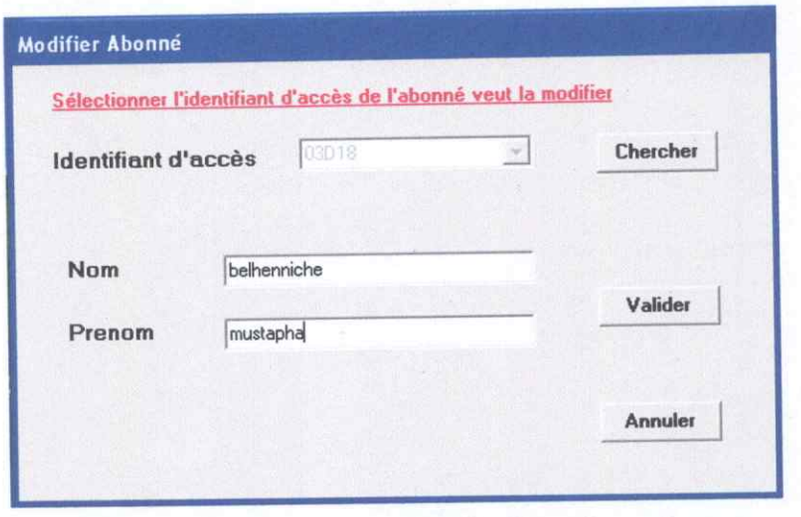


Figure 8 : *l'interface Modification abonné.*

L'interface représenté dans la figure 8 est générée a partir de la sélection du menu 'Gestion des abonnés' de la barre des menus de l'interface 'gestion Centrale' par le chemin suivant (Gestion des abonnés -> Modifier abonné), elle permettent de modifie le nom et prénom de l'abonné.

➢ **L'interface *Liste des abonnés***

Cette interface est générée a partir de la sélection du menu 'Gestion des abonnés de la barre des menus de l'interface 'Gestion Centrale' par le chemin suivant (Gestion des abonnés -> Afficher tout).

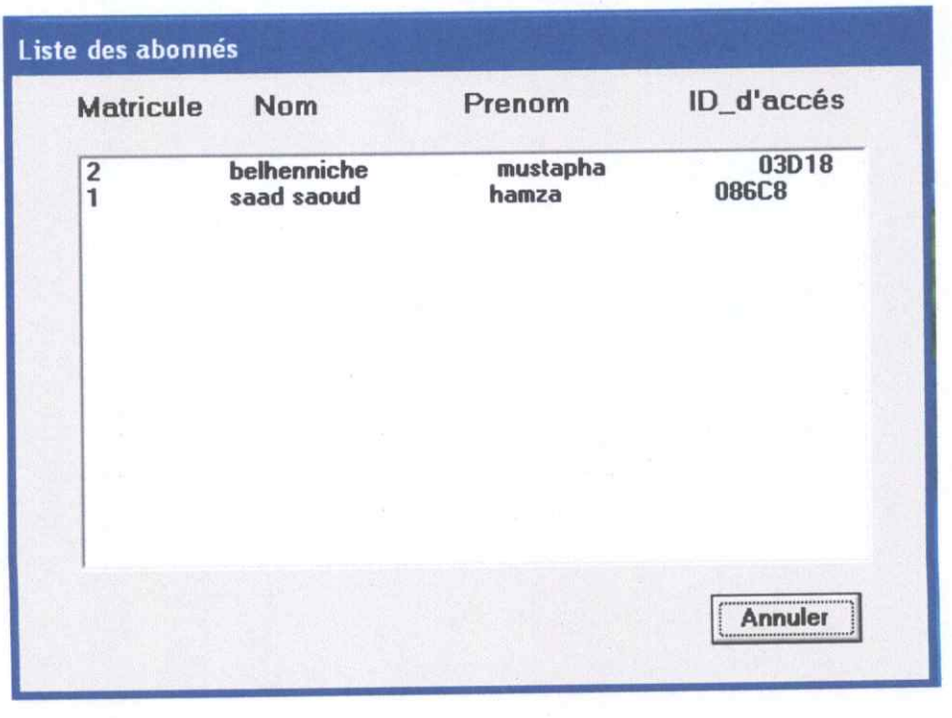


Figure 9 : *L'interface Liste des abonnés*

➢ L'interface Suppression Abonné

Cette interface est générée à partir de la sélection du menu 'Gestion des abonnés ' de la barre des menus de l'interface 'Gestion Centrale' par le chemin suivant : (Gestion des abonnés -> Supprimer abonné)

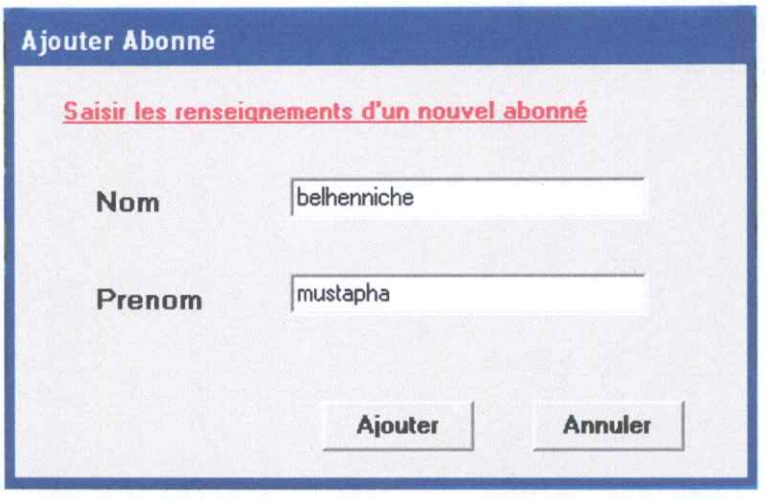


Figure 10 : *L'interface Suppression Abonné*

➢ L'interface Ajout commande

Cette interface est générée à partir de la sélection du menu 'Gestion des commandes' de la barre des menus de l'interface 'Gestion Centrale' par le chemin suivant : (Gestion Centrale->Ajout commande).

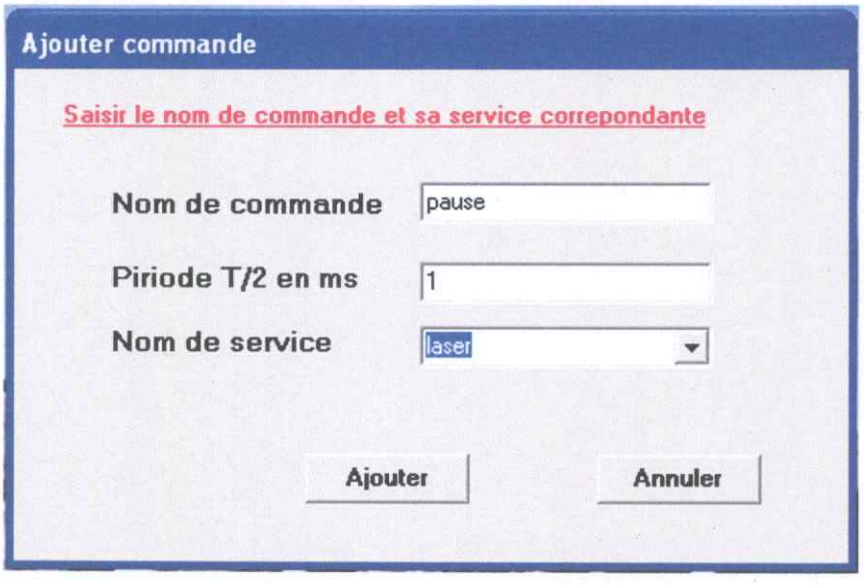


Figure 11 : L'interface Ajout Commande

➢ L'interface modification commande

Cette interface est générée à partir de la sélection du menu 'Gestion des commande' de la barre des menus de l'interface 'Gestion Centrale' par le chemin suivant : (Gestion des commander->modifier commande)

Cette interface permet de modifier les noms et les états des commandes des chaque port.

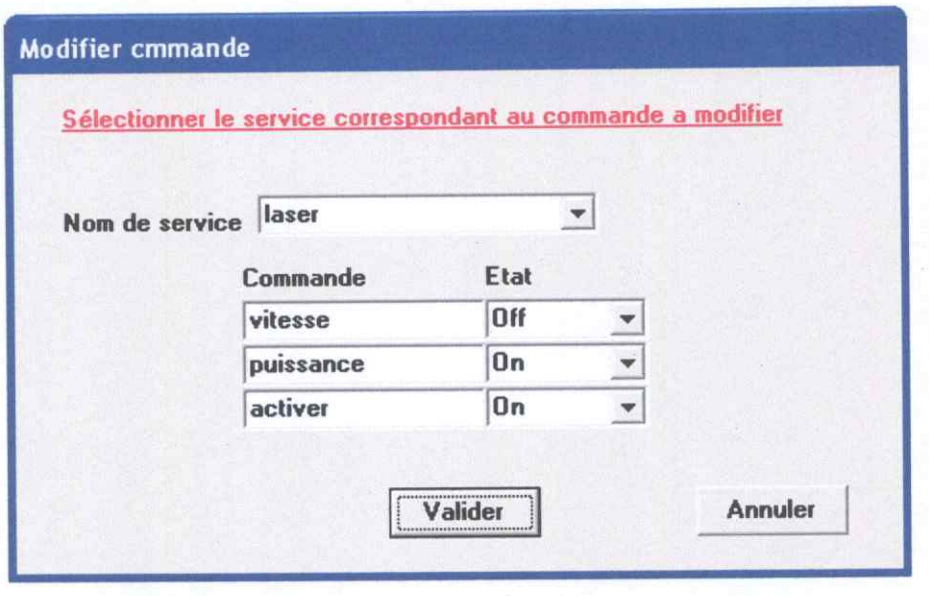


Figure 12 : *L'interface modification commande*

➢ L'interface Ajout service

Cette interface est générée a partir de la sélection du menu 'Gestions des services' de la barre des menus de l'interface 'Gestion Centrale' par le chemin suivant (Gestion services->Ajout service).

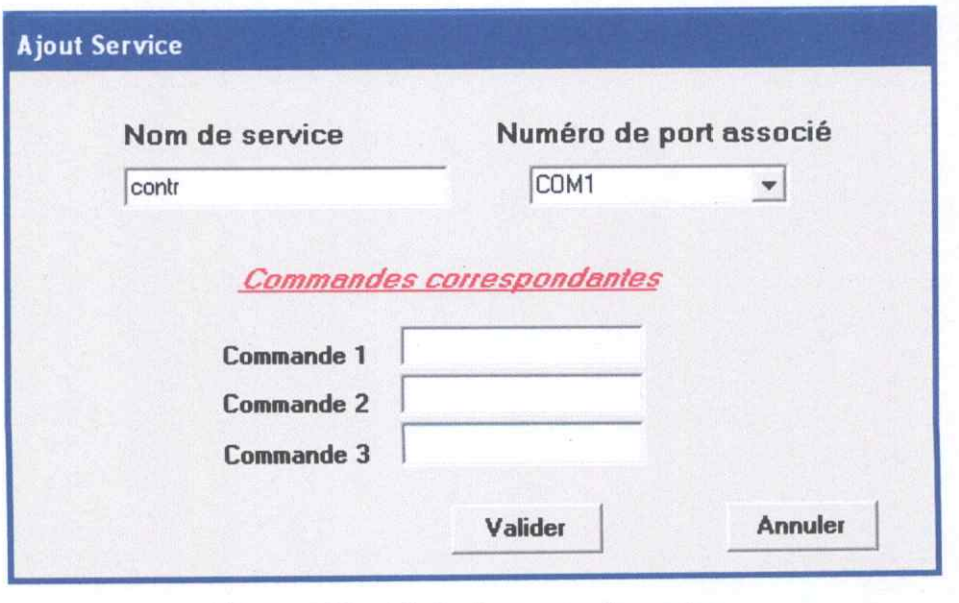


Figure 13 : *L'interface ajout service*

Cette interface permet d'ajouter un service avec trois actions et de sélection son port correspondante.

➢ **L'interface Suppression service**

Cette interface est générée a partir de la sélection du menu 'Gestion des service' de la barre des menus de l'interface 'Gestion Centrale' par le chemin suivant (Gestion service->Supprimer service).qui permettent de supprimer un service avec ses commandes.

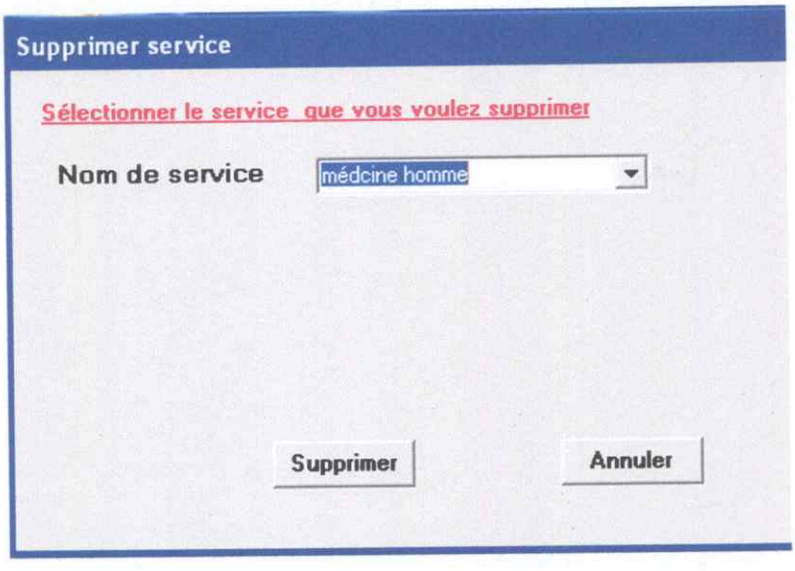


Figure 14 : *L'interface Suppression service*

➢ **L'interface Modification service**

Cette interface est générée a partir de la sélection du menu 'Gestion des service' de la barre des menus de l'interface 'Gestion Centrale' par le chemin suivant (Gestion service->Modifier service).qui permettent de changer l'état de service.

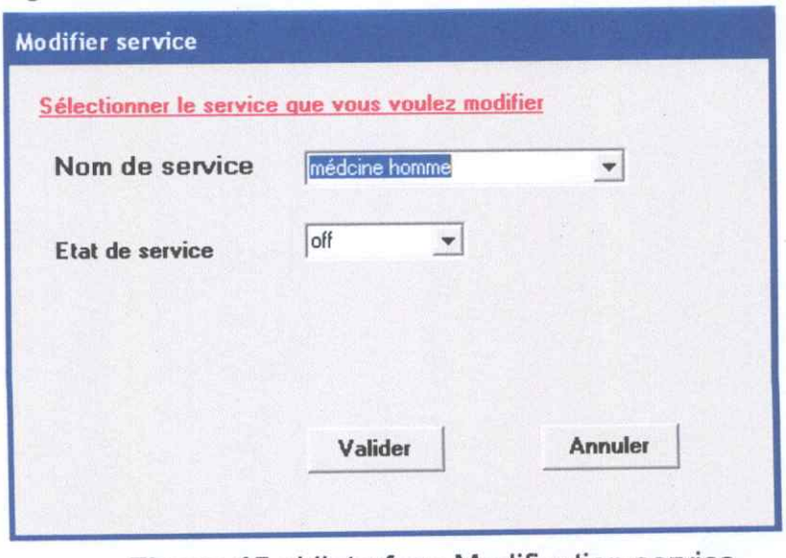


Figure 15 : L'interface Modification service

V. TESTE ET VALIDATION

V.1 Les outils de teste

- ✓ Un PC de bureau de performance moyenne muni : d'un système d'exploitation Windows, d'un serveur Web Apache et d'un serveur de base de donnés MySql (EasyPHP) qui sont installés et configurés, des langages de développements C++ et PHP qu'ils sont installés et configurés et d'un navigateur Web Internet Explorer pour l'échange des données entre les clients et le serveur.
- ✓ La carte électronique de pilotage « carte 3st » a réalisée.
- ✓ Un câble RR232 en bon état et non croisé.
- ✓ Trois appareils médicaux (remplaçants par des lampes juste pour la vérification que les commandes « signaux » s'arrivent) de type électrique d'alimentation alternative 220V et sont en bonne état.

V.2 Préparation de test

Dans cette partie nous avons un PC doté d'un système Windows XP Professionnel, ce dernier est muni d'un serveur EasyPHP 1.8, dont ses serveurs Apache et MySql et le langage PHP sont pré configurés entre eu, et d'un langage Builder C++ 6 installé complètement. Donc dans cette étape il reste de suivre ce qui suit :

- ✓ Copie de l'application PHP dans le répertoire "www" d'EasyPhp.
- ✓ Réalisation de l'alias entre C++ et la base de donnée (MySql).
- ✓ Réalisation du câblage entre les différents terminaux

V.3 Réalisation du test

Nous avons testé notre application dans le centre de développement des technologies avancé pour le laser (avec l'unité de commande de laser), et pour un groupe des lampes (avec la carte d'interfaçage 3st) et cala juste pour tester l'arriver des commandes de l'utilisateur.

➢ Au niveau du poste serveur (administrateur)

- on lance le serveur EasyPHP (ie : Apache, MySql)
- on lance l'exécutable de l'application serveur et en sélectionner le numéro de port COM de la liste du menu de l'interface centrale (le port qui pilote l'appareils médical que voulez vous contrôler), puis on clique sur le button <ouvrir> (ouvrir la connexion avec le serveur Mysql et ouvrir le port COM).

➢ **Au niveau du poste client (utilisateur)**

- on lance Internet explorer et on tape l'adresse de l'application serveur (http://mustapha/hopital/controle-appareils.php) dans le champ d'adresse, puis on la valider, l'interface d'accueil va s'apparaître.
- Si l'utilisateur a le droit de contrôle, il sufi de saisir ces renseignements (nom, prénom et code d'accès) et les valider, un interface de contrôle s'appâitre.
- L'étape suivant est le plus nécessaire tel que l'utilisateur peut envoyer leurs commandes de contrôle aux appareils médicaux et peut voir les résultats.

- **Test 1 : avec carte 3st**

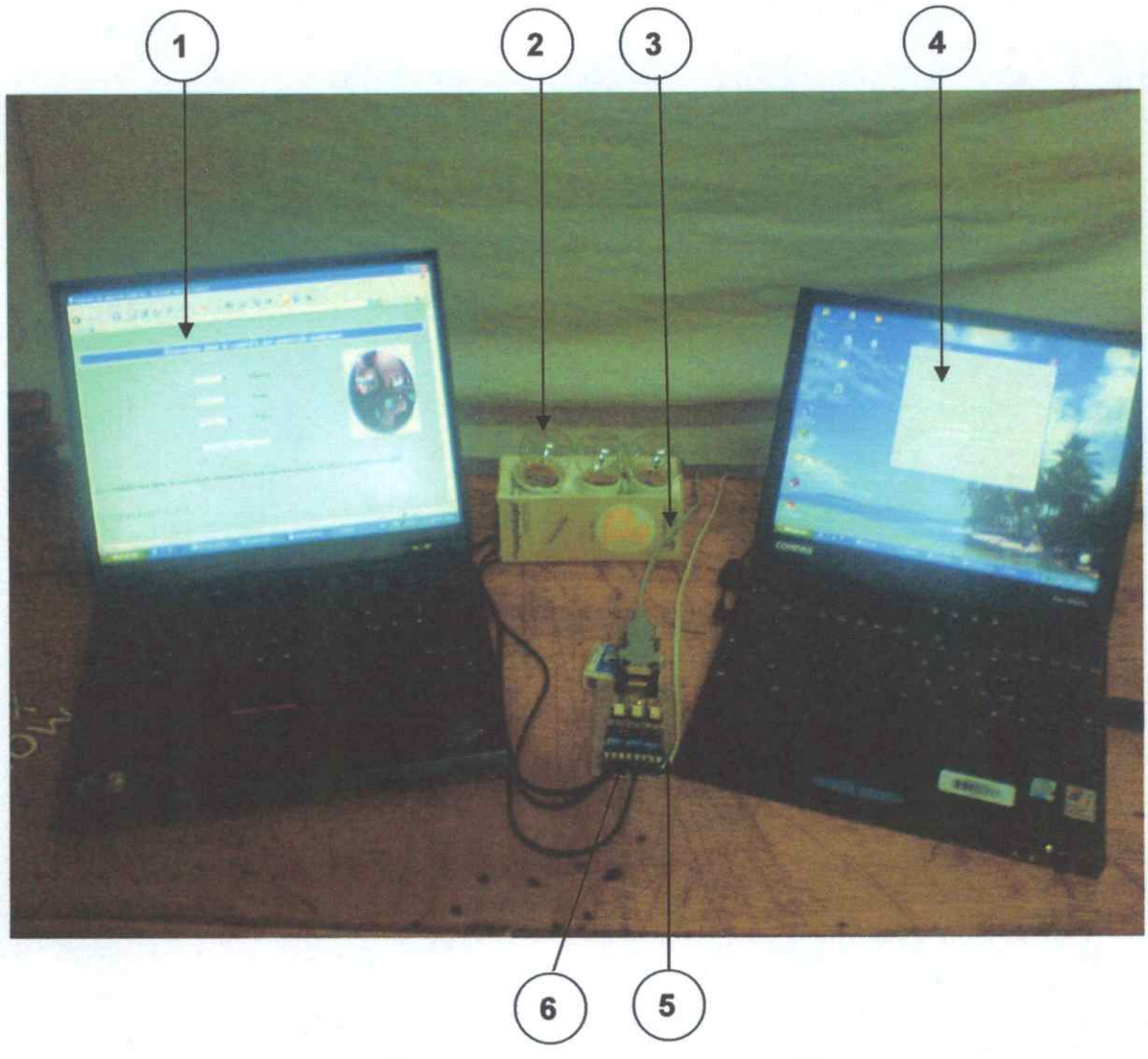


Figure 16 : Exemple de test réel

1 : la page web qui s'affiche aux pc client.
2 : trois lampes branchées aux la carte trois sorties, cette dernière connecté avec la liaison rs232 au pc serveur.
3 : câble rs232 non croisé branché sur le pc hôte (serveur).ce dernier contient l'application gestion central est scrute en permanence.

4 : application gestion centrale dans le pc hôte (serveur)
5 : source d'alimentation électrique de la carte trois sorties.
6 : les trois sorties de la carte de pilotage.

- **Test 2 : avec laser**

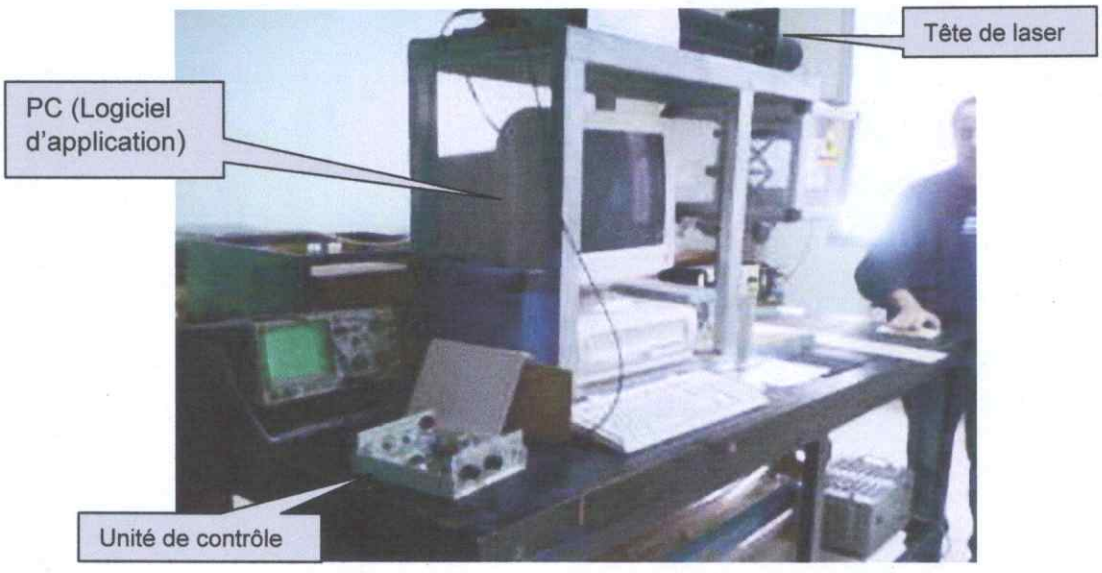


Figure 17 : *laser avec commande extérieur*

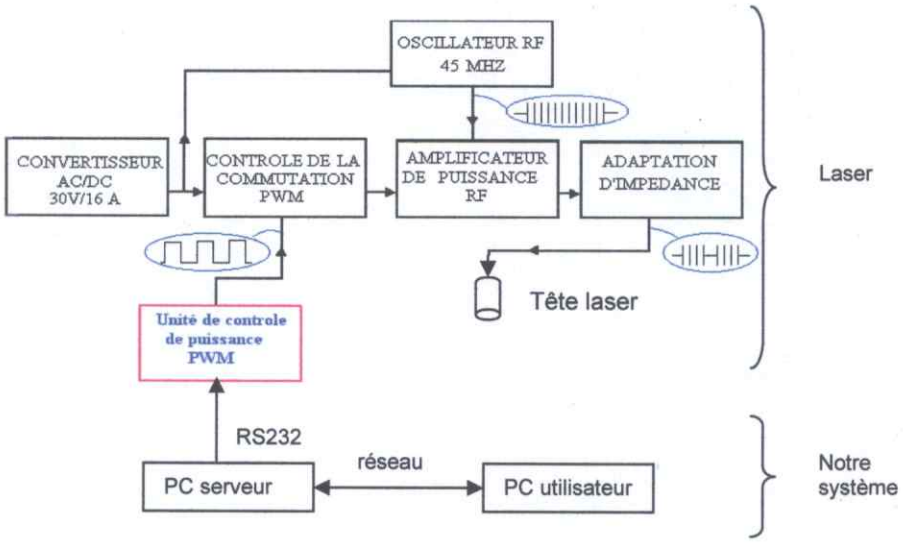


Figure 18 : *le contrôle extérieur de laser par notre application*

Ce qui nous intéressons dans notre test c'est l'unité de contrôle de laser, qui se connectée avec notre PC serveur par la liaison RS232 en mode synchronisation extérieure.

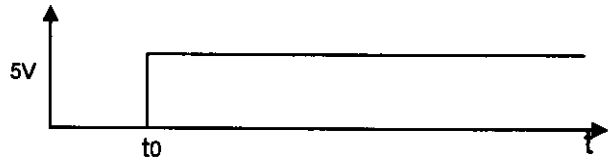
Cette unité inclus trois modes de fonctionnement, à savoir : manuel, automatique et **synchronisation extérieure**.

Cette unité admette les trois commandes (activation, synchronisation et puissance).

Les types des signaux d'entrée sont :

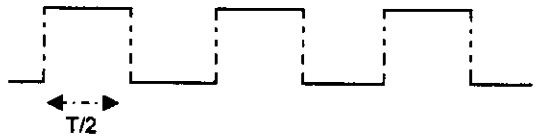
T : période. f : fréquence.
P : puissance. ↑ : augmente.

- activation :



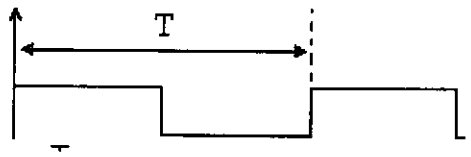
- synchronisation :

T constante ⇒ f constante



- puissance :
  (puissance de marquage)

T variable tel que
T ↑ ⇒ P ↑ et l'inverse.



Règle : $f = 1/T$

La puissance de laser sera maximale (100%) à la fréquence d'entré 5khz,

L'action de marquage se faite à fin d'un mécanisme de produit des trois signaux.
(activation de marquage a une certain puissance).

VI. CONCLUSION

Dans ce chapitre, nous avons présentés notre application de vue implémentation, tel que nous avons présentés l'environnement, les outils de développement les interfaces essentielles de l'application. Et nous avons finis par des tests réels avec notre groupe de développement au niveau de CDTA, et cela pour la validation.

CONCLUSION GENERALE

Le travail présenté dans ce mémoire entre dans le cadre du développement d'outils de contrôle des appareils médicaux sous réseau, et qui fait partie des activités scientifiques des projets de l'équipe instrumentation virtuelle de la division Architecture des Systèmes (AS) du Centre de Développements des Technologie Avancé (CDTA).

Le but de notre travail consiste à développer un système informatique bâtis sur une architecture client/serveur permettant de Contrôler des appareils médicaux sur réseau.

Dans ce travail, mémoire, nous avons présenté, en premier lieu, l'architecture physique et logique de la liaison RS232 et ses fonctionnalités au système Windows. Ensuite, nous avons présenté quelques notions dans le domaine de réseau (ce que nous intéresse). En fin, nous avons monté les différentes fonctions offertes par l'application logicielle développée suivi d'un test de validation.

Le résultat de notre travail est l'enrichissement de domaine de contrôle par des fonctions qui permettent de :

- ✓ Piloter les actionneurs d'une interface (carte électronique) connectées au port série (RS232) du PC hôte, ainsi que les appareils médicaux liés à cette interface.
- ✓ La gestion administrative des utilisateurs (permission, authentification, ...).
- ✓ Fournir aux utilisateurs distants une interface graphique de format Web, permettant à ces derniers de visualiser et de contrôler « selon les Authentification des utilisateurs » les appareils médicaux liés au serveur en temps réel
- ✓ La gestion administrative de la base de données qui comporte les abonnés, les appareils...à partir d'une interface facile à utiliser.
- ✓ Transmission des commandes des utilisateurs vers les appareils médicaux en temps réel.

Notre application tient compte un contrainte majeure :

- L'utilisation d'un PC hôte comme un serveur qui contient une seul port série (DB9), se qui implique d'envoyer que trois commandes.

ANNEXE A

UML

I. HISTORIQUE D'UML [9, 10]

L'approche objet est depuis longtemps une réalité, les concepts de base sont stables est largement approuvés. L'approche objet a commence à apparaître vers la fin des années 60, avec simula qui a été le premier langage de programmation à implémenter le concept de type abstrait a l'aide des classes, et en 1976 smaltalk implémente les concepts fondateurs de la méthode objet : encapsulation, agrégation et héritage. Les premiers compilateurs C++ font leur apparition au début des années 80 antique et de nombreux langages orientes objet le jour (Eiffel, loops.....)

Actuellement l'approche objet est devenue incontournable, des iros qu'on cherche à concevoir des logiciels complexes qui doivent résister a des évolutions incessantes, la programmation objet bénéfice d'une panoplie d'outils st de langage performants.

Mais l'approche comporte quelques inconvénients, qui peuvent induirent leurs utilisateur dans l'erreur et mettre en péril.

De ces difficultés on peut citer :

- L'approche objet n'est pas très intuitive, car il est difficile pour le cerveau humain de décomposer un problème informatique en termes d'objet et d'interaction entre ces objets, et rien dans les concepts de base de l'approche objet ne dicte comment modéliser la structure objet d'un système de manières pertinentes.

- L'application des concepts objet nécessite beaucoup de rigueurs, car il y a beaucoup de développeur pensent objet qu'a travers les langage de programmation.

C'est pour cela qu'il faut disposer d'un outil qui sera un guide dans l'utilisation des concepts objets

Et pour cela il nous faut :

1) Un langage (pour s'exprimer clairement à l'aide des concepts objets), qui doit permettre de :

   - Représenter des concepts abstraits (graphiquement par exemple)
   - Limiter les ambiguïtés (parler un langage commun, au vocabulaire précis, indépendant des langages de programmations orientés objet)
   - Faciliter l'analyse (simplifier la comparaison et l'évaluation de solutions).

2) Une démarche d'analyse et de conception objet pour :

- Ne pas effectuer une analyse fonctionnelle et se contenter d'une implémentation objet, mais penser objet dès le départ.
- Définir les vues qui permettent de décrire tous les aspects d'un système avec des concepts objets.

C'est pour ça que les langages de modélisation orientés objets ont fait leur apparition, au milieu des années 70, quand les spécialistes ont commencé à éxprimenter de nouvelles approches d'analyse et de conception. Entre1989 et 1994 le nombre de méthodes objet est passé de 10 à 50, sans qu'elles répondent vraiment aux besoins des utilisateurs.

Basées sur l'expérience acquise, de nouvelles méthodes sont apparues, les plus importantes sont : booch, OOSE (Object Oriented Software Engineering) et OMT (Object Modeling technique), elles sont été reconnues au niveau mondial comme étant incontournable. Chacune de ses méthodes constituait une méthode complète mais présenter encore des inconvénients, en résumé chacune des méthodes avait un créneau dans le quel elle excellait.

Au milieu des années 90 les principaux auteurs des méthodes, Booch, OOSE et OMT qui sont respectivement Grady Booch (Rational Software Corporation), Ivar Jacobson (objectory) et James Rumbaugh (General Electric) ont commencé à adopter les idées des deux autres, puis se sont décidés à travailler ensemble pour la création d'un langage de modélisation unifié, et cela pour les trois raisons :

- Poursuivre cette évolution ensemble pour éliminer les différences inutiles qui aurait embrouillé les idées des utilisateurs.
- Apporter une stabilité au marche orienté objet.
- Apporter des améliorations aux trois méthodes et répondre à des problèmes qu'aucune d'elle n'a traités de manière satisfaisante.

Et pour atteindre les objectifs suivants :

1- La modélisation des systèmes au moyens de techniques orienté objet, depuis leur conception jusqu'a leur artéfact exécutable.

2- La résolution les problèmes d'échelles inhérentes aux systèmes complexes et essentiels.

3- La création d'un langage de modélisation utilisable à la fois par les humains et les machines.

De là, donc est née l'UML (Unified Modiling Language) traduit (langage de modélisation objet unifié) de la fusion des trois méthodes (Booch, OMT et OOSE), qui est le fruit d'un large consensus. très nombreux acteurs industriels de renoms (HP, IBM, ORACLE, MICROSOFT...) l'ont adopté et participe à son développement en peu de temps,UML est devenus un standard incontournable.

Nous résinons ci-dessous le récapitulatif des évolutions de ce langage de modélisation :

- En 1995 : méthode unifiée 0.8 (intégrant les méthodes Booch'93 et OMT)
- En 1995 : UML 0.9 (intégrant la méthode OOSE)
- En 1996 : UML 1.0 (proposée à l'OMG)
- En 1997 : UML 1.1 (standardisée par l'OMG)
- En 1998 : UML 1.2
- En 1999 : UML 1.3
- En 2000 : UML 1.4
- En 2003 : UML1.5

II. DESCRIPTION D'UML

UML (Unified modeling Language) est un langage standard conçu pour l'écriture des plans d'élaboration de logiciel. Il peut être utilisé pour visualiser, spécifier, construire et documenter les artefacts d'un système à fortes composantes logicielles.

UML est utilisé pour visualiser dans le sens où il permet d'utiliser au mieux les représentations graphiques ou textuelles en fonction des besoins, où chaque symbole UML possède une sémantique bien définie, et chaque développeur ou outil peut interpréter ce modèle sans ambiguïté.

UML est un langage pour spécifier dans le sens où il construit des modèles précis, sans ambiguïté et complets. Et cela en spécifiant toutes les décisions importantes en termes d'analyse, de conception et d'implémentation.

UML est un langage pour construire dans le sens où l'on peut construire ou traduire les modèles d'UML dans un langage de programmation tel que JAVA, C++... et cela s'appelle l'application de l'igenierie vers l'aval. L'inverse est aussi possible c'est-à-dire passer du code vers le modèle ou plus précisément la reconstruction du modèle à partir du code. Cela s'appelle la retro-ingenierie, mais il nécessite le support d'un outil pour cause de perte d'information.

UML est un langage pour la documentation dans le sens où il permet de documenter l'architecture d'un système dans ses moindres détails.

UML est utilisé dans beaucoup de domaine tel :

- Les services informatiques d'entreprise.
- Les services bancaires et financiers.
- Les télécommunications.
- Les transports.
- La défense et aéro-spaciale.
- Le commerce de détail.
- L'électronique de détail.
- Les sciences.
- Les services distribués basé sur le web mais aussi à des systèmes qui n'appartiennent pas à cette catégorie, comme le workflow d'un système judiciaire, ou la conception de matériel informatique.

II.1 Les points forts d'UML

- UML est un langage formel et normalisé : gain de précision, gage de stabilité, encourage l'utilisation d'outils.

- UML est un support de communication performant :

  - Il cadre l'analyse.
  - Il facilite la compréhension de représentations abstraites complexes.
  - Son caractère polyvalent et sa souplesse en font un langage universel.

II.2 Les points faibles d'UML

- La mise en pratique d'UML nécessite un apprentissage et passe par une période d'adaptation.
- Le processus qui est non couvert par UML est une autre clé de la réussite d'un projet, or, l'intégration d'UML dans un processus n'est pas triviale et l'amélioration d'un processus est une tache complexe et longue.

III. COMPOSITION D'UML

Il se compose de trois éléments essentielles : les briques de base, les règles qui déterminent la manière d'assembler les briques de base et quelques mécanisme généraux.

III.1 Les briques de base

Il existe trois sortes de brique :(Les éléments, les relations et les diagrammes)

III.1.1 Les éléments :

➢ Eléments structurels :

C'est les parties les plus statiques du modèle, il représente des éléments conceptuels ou physiques et il en existe sept :

- **La classe :** elle représente un ensemble d'éléments qui partagent les mêmes attributs, les mêmes opérations, les mêmes relations et la même sémantique, elle implémente une ou plusieurs interfaces.
  - Attribut : un attribut de classe définit une propriété commune aux objets d'une classe, chaque objet (instance d'une classe), asa propre identité indépendante des valeurs de ses attributs.
  - Opération : une opération définit une fonction appliquée à des objets d'une classe.
  - Accessibilité : des attributs et opérations. Il existe trois niveaux de protection :
    a) Publique (+) : accès à partir de toute entité interne ou externe à la classe.

b) Protégé (#) : accès à partir de la classe ou des sous-classe
c) Privé (-) : accès à partir des opérations de la classe.

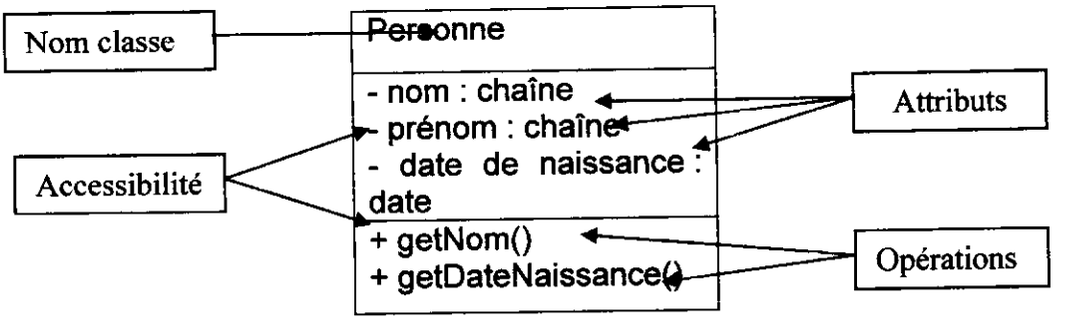


Figure 1 : *Structure d'une classe (Personne)*

- **Interface :** c'est un ensemble d'opérations qui définissent la fonction d'une classe ou d'un composant, et qui décrit le comportement apparent de cet élément.

Figure 2 : *L'interface*

- **Collaboration :** elle définit une interaction et constitue une société de rôle et de divers éléments qui travaillent ensemble pour fournir un comportement coopératif qui présente une utilité supérieur à la somme de toutes les parties.

Figure 3 : *Collaboration*

- **Cas d'utilisation :** c'est la description d'une séquence d'actions exécutées par un système, pour produire un résultat qui peut être constaté par un acteur particulier.

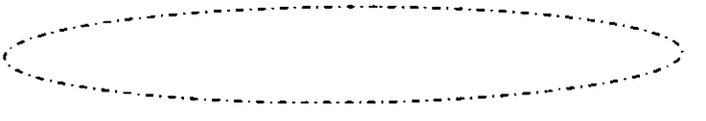


Figure 4 : *Cas d'utilisation*

- **Classe active :** c'est une classe dont les objets possèdent un ou plusieurs processus ou threads et qui peut lancer une activité de commande. Elle est différente d'une classe dans le sens où ses objets représentent des éléments dont le comportement est différent de celui des autres éléments. Elle à la même forme qu'une classe mais avec un trait épais.

- **Composant :** c'est une partie physique remplaçable d'un système, qui se conforme à un ensemble d'interfaces et permet la réalisation. Dans un système, il existe différents types de composants de déploiement, tel que le composant COM+ou java Beans, ainsi que des composants qui constituent les interfacts du processus de développement, tel les fichiers du code source. Généralement, un composant représente l'enveloppe physique d'élément de nature logique, comme des classes, des interfaces ou des collaborations.

- **Un nœud :** c'est un élément physique qui intervient lors de la phase d'exécution, il représente une ressource de calcul et dispose généralement au moins d'un peu de mémoire et souvent d'une capacité de traitement.

Ces sept éléments constituent les éléments structurels de bases qui peuvent êtres inclus dans un modèle UML.

Il existe également des variations de ces sept élément, comme les acteurs,les signaux, les utilitaires (sortes de classes), les processus et les threads (sorte de classe active), les applications, les documents, les fichiers, les bibeloteurs, les pages et les tables (sortes de composants).

➢ **Eléments comportementaux :**

Les éléments comportementaux représentent les parties dynamiques des modèles UML. Ce sont les verbes du modèle et ils représentent son comportement dans le temps et dans l'espace.

Il existe deux types d'éléments comportementaux :

- une interaction : c'est un comportement qui comprend un ensemble de message échangés au sein d'un groupe d'élément, dans un contexte particulier pour atteindre un but bien défini.

- Un automate à état finis : c'est un comportement qui précise les séquences d'états d'un élément ou d'une interaction au cours de leurs durées de vie, en réponse à des événements, ainsi qu'à leurs réactions à ces événements.

Ces deux éléments constituent les éléments comportementaux de base qui peuvent être inclus dans un modèle UML. Sur le plan sémantique, ces éléments sont lie à divers éléments structurels, essentiellement des classes, des collaborations et des objets.

➢ **Eléments de regroupement :**

Il représente les parties organisationnelles des modèles UML. Ce sont des boites dans lesquelles un modèle peut être décomposé. Il existe un seul type fondamental d'élément de regroupement : le paquetages.

Un paquetage est un mécanisme général qui permet de regrouper des éléments, des éléments structurels, des éléments comportementaux, et même d'autre éléments de regroupement peuvent être range dans un paquetage. A l'inverse des composants (il existe lors de la phase d'exécution), c'est un élément purement conceptuel (il existe seulement lors de la phase de développement).

Ajouter au paquetages qui sont des éléments fondamentaux, il existe des variations, tels que les frameworks, les modèles et les sous-systèmes (sortes de paquetages).

➢ **Eléments d'annotation :**

Les éléments d'annotation représentent les partis explicatifs des modèles UML. Ce sont des commentaires qui peuvent accompagner tout élément dans un modèle, à des fins d'explications, de description, et de remarque. Il existe un seul type, appelé "`note". Une "note" est simplement un symbole utilisé pour représenter les contraintes et les commentaires rattachés à un élément où a un ensemble d'éléments.

III.1.2 Les relations d'UML:

Il existe quatre types de relation dans UML :

➢ **Dépendance :** c'est une relation sémantique entre deux éléments selon laquelle un changement apporté à l'un (élément indépendant) peut affecter la sémantique de l'autre (élément dépendant). La dépendance est représenté par une ligne en pointilles qui peut être flèche, elle comprend une étiquette.

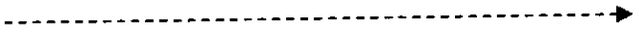


Figure 5 : *Relation de dépendance*

➢ **Association :** c'est une relation structurelle qui décrit un ensemble de lien, un lien constituent une relation entre différent objets. Il comporte :

- Rôle : définit la manière dont une classe intervient dans une relation.
- Multiplicité : est une information portée par le rôle, qui quantifie le nombre de fois où un objet participe à une instance de relation comme 1(un et un seul), 1..N (de un à N), 1..* (de un à plusieurs).
- Contrainte : porte sur une relation ou sur un groupe de relations pour indiqué une relation d'ordre (notée {contrainte}).

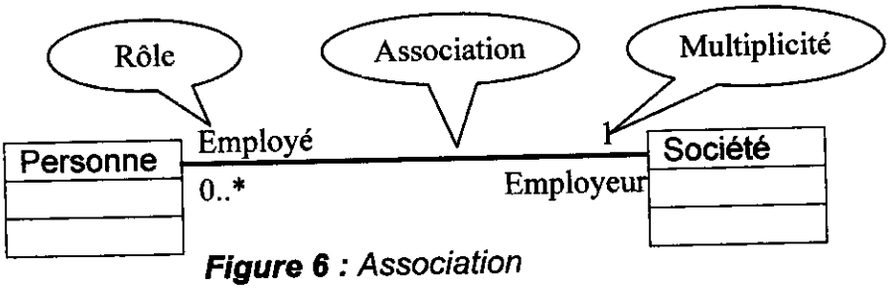


Figure 6 : *Association*

- **Agrégation :** est une association non symétrique une des extrémités joue un rôle prédominant par rapport à l'autre, il se justifier dans les cas suivants :

  - Une classe B « fait partie » intégrante d'une classe A.
  - Les valeurs d'attributs de B se propagent dans celle de A.
  - Une action sur A implique une action sur B.
  - Les objets de B sont subordonnés aux objets de B.

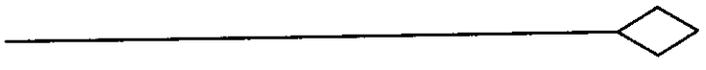


Figure 7 : *Relation d'agrégation*

- **Généralisation :** c'est une relation sémantique entre classificateurs, selon laquelle les attributs de l'élément parent.

  Une relation de généralisation est représenté par une flèche dont le trait est plein et dont la pointe creuse est dirigés vers le parent.

Figure 8 : *Relation de généralisation*

- **la réalisation :** c'est une relation sémantique entre classificateurs, selon laquelle un classificateur spécifie un contrat dont l'exécution est garantie par un autre classificateur. Les relations de réalisation apparaissent à deux occasions : entre les interfaces et les classes ou les composants qui les réalisent, et entre les cas d'utilisation et les collaborations qui les réalisent.

  Elle est représentée par un mélange d'une agrégation et d'une relation de dépendance.

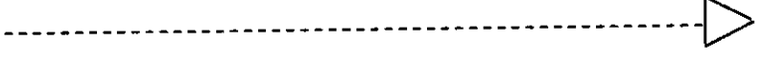


Figure 9 : *Relation de réalisation*

III.1.3 les diagrammes

L'UML comporte 9 diagrammes pour la modélisation des systèmes,. dans une modélisation, il n'est pas nécessaire d'utiliser les 9 diagrammes.

(Diagramme de classes, diagramme d'objets, diagramme de cas d'utilisations, diagramme de séquences, diagramme de collaborations, diagramme état transition, diagramme de composants, diagramme de déploiement, diagramme d'activité).

III.2 les règles d'UML

Comme tout langage, UML possède un certain nombre de règles qui permettent de produise des modèles correctement mis en forme.

Les règles sémantiques d'UML

- **Les noms :** la manière de designer les éléments, les relations et les diagrammes.
- **Le contexte :** l'environnement qui donne une signification bien précise à un nom.
- **La visibilité :** la manière dont ces noms peuvent être vus et utilisés par d'autres.
- **L'intégrité :** la manière dont les objets établissent des relations correctes et cohérent entre eux.
- **L'exécution :** les conséquences de l'éxécution ou de la simulation d'un modèle dynamique.

III. 3 Les mécanismes généraux

Quatre mécanismes permettent et rendent UML plus simple à utiliser :

a) **Spécification :** dans UML, derrière chaque notation graphique d'un élément, il existe une spécification en fournissant un énoncé textuel de la syntaxe et de la sémantique pour cet élément. Par exemple pour une icône d'une classe on trouve spécification qui petite les attributs, les opérations...

b) **Décoration :** d'autres informations (visibilité de ces attributs et de ces opérations, ou encore si elle est abstraite ou non) peuvent être dans la notation d'une classe, ceci en les représentant par des décorations graphiques ou un textuelle ajoutée.

c) **Distinctions communes :** en UML il existe deux distinctions fondamentales :

- Entre une classe et un objet : une classe est une abstraction alors que l'objet est la manifestation concrète de cette abstraction.

- Entre interface et implémentation : une interface définie un contrat et l'implémentation représente la réalisation concrète de ce contrat respectant la sémantique complète de l'interface.

d) **Mécanisme d'extensibilité :** UML étant un langage ouvert qui peut être élargi pour suffire à exprimer toutes les nuances possibles

-Stéréotype : il permet de crier de nouvelles briques de base qui dérivent de celle d'un vrais problème qui existe mais qui sont adapte un problème donne. Par exemple en C++, en travaille avec des exceptions qui ne sont que des classes à traitées de manière spéciale.

-Etiquette : elle permet la création de nouvelles informations spécifiant un élément en élargissant ces propriétés.

-Contrainte : élargit la sémantique d'une brique de base, permettant l'introduction de nouvelles règles ou en modifiant celle existante.

e) **l'architecture :**

L'architecture est la clé de voûte du succès d'un développement car elle décrit des choix stratégiques qui déterminent les qualités du logiciel.

La figure est une vue qui a beaucoup inspire UML dans son modèle d'architecture.

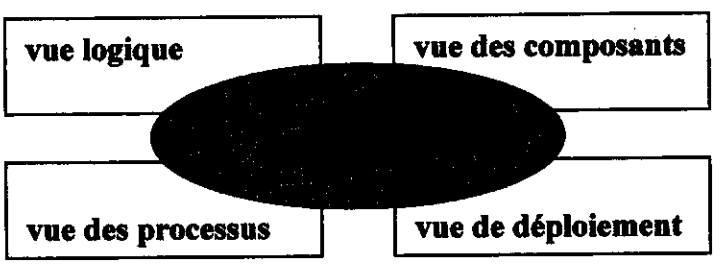


Figure 10 : *Modèle d'architecture*

- **la vue logique :** c'est une vue de haut niveau qui modélise les éléments et mécanismes principaux du système,elle se base sur l'abstraction et l'encapsulation. Elle identifie les éléments du domaine, ainsi que les relations et interactions entre ces éléments, et organisent les éléments en catégories.
- **La vue des composants :** c'est une vue de bas niveau qui montre l'allocation des éléments de modélisation dans des modules (fichiers sources, bibliothèques dynamiques, bases de données, exécutables,...etc.), en identifiant les modules qui réalisent (physiquement) les classes de la vue logique, elle montre aussi les contraintes de développement (bibliothèques externes...) et l'organisation des composants, elle montre aussi l'organisation des modules en sous systèmes.
- **La vue des processus :** elle est très importante dans les environnements multitâches, elle montre la décomposition des

système en terme des processus (tâche), la communication entre ces processus, la synchronisation et la communication des activités parallèles (threads).

- **La vue de déploiements :** elle est très importante dans les environnements distribués, décrit les ressources matérielles et la répartition du logiciel dans ces ressources.
- **La vue des besoins des utilisateurs (vue des cas d'utilisation) :** elle guide et unifie toutes les autre, elle justifie l'architecture d'un système informatique, et définit les besoins du client et centre l'architecture du système sur la satisfaction de ces besoins, et conduit à la définition d'un modèle d'architecture pertinente et cohérent.

ANNEXE B

Présentation de la carte 3st

I. INTRODUCTION

La carte électronique « 3st » est un outil qui ouvre de nouvelles respectives dans le domaine de contrôle à distance des équipements.

Dans ce chapitre, nous allons présenter la carte électronique et tous les composants électroniques utilisés.

II. SCHEMA ELECTRIQUE

Le pilotage de notre carte s'effectue par trois sorties du port RS232 : DTR (broche n° 3), TXD (broche n° 4) et RTS (broche n° 7).

L'utilisation de la tension du secteur nous conduit à réaliser une isolation galvanique afin de protéger correctement l'ordinateur .cette isolation est effectue à l'aide d'un optocoupleur (du type MOC3041), un tel circuit se compose de deux paries distincts.

La première partie est constituée d'un diode infrarouge qui va venir mettre en conduction le triac contenu dans la deuxième partie,il est également muni d'un dispositif qui détecte le passage à zéro de la tension du secteur afin d'éviter de générer des parasites lors de l'alimentation de la charge . le courant de l'ordre de 10 mA nécessaire à l'activation de la diode infrarouge est généré par la sortie du port série .la limitation de l'intensité est assurée par résistance de 1kΩ.

La diode LED mis en série permet de valider visuellement l'état de la diode interne au circuit par précaution, la diode 1N4148 permet d'annuler le courant, outre la sécurité offerte par les optocoupleurs leurs l'utilisation nous permet de nous passer d'une alimentation en courant contenue, en effet la diode de commutation est alimentée par le port série et le triac par la tension secteur.

Donc nul besoin de transformateur et autre régulateur de tension, la faible puissance du triac interne à l'optocoupleur (max = 100 mA) ne permet pas une alimentation directe d'une charge importante.

Un deuxième triac mis en cascade permet de disposer d'une puissance beaucoup plus importante, celui utiliser ici est un BTA 08-700B en boiter T0 220. Il peut fonctionner sous une tension alternative max, de 700 V et débiter un courant de 40 A !

Si nous utilisons le montage pour piloter des charges inférieures à 100 w, on peut se passer d'un dissipateur. Dans le cas contraire celui-ci devient obligatoire afin d'éviter la destruction du triac.

Chaque triac possédé une varistance monté en parallèle, ce composant voit son impédance chuter très fortement en présence d'une surtension (tension > tension nominale 220 V), protégent ainsi le circuit placé en aval.

La figure 1 représenter le schéma électrique de notre carte.

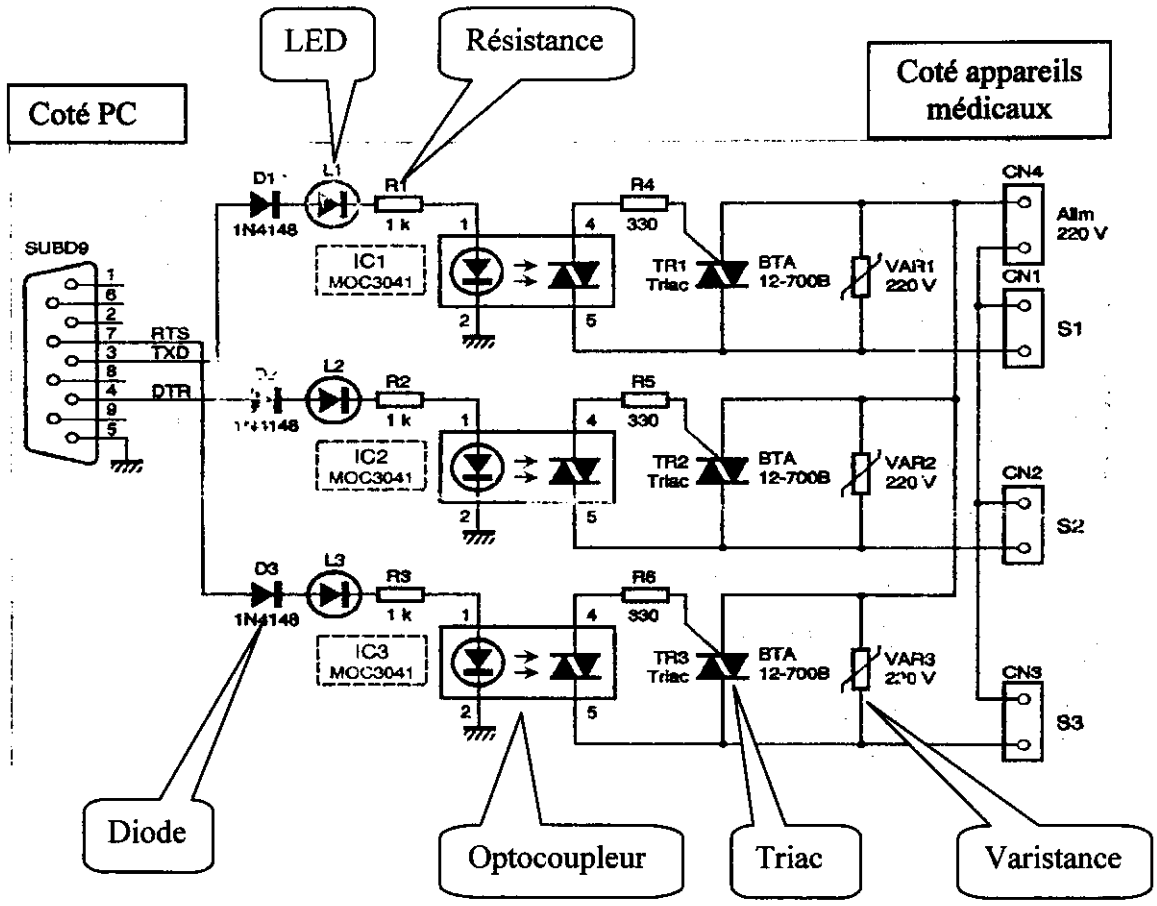


Figure 1 : *schéma générale de la carte électronique « 3st » a réalisé*

III. LES COMPOSANTS DE LA CARTE D'INTERFAÇAGE « 3ST » [1]

La carte électronique utilisée est constituée des composants suivants :

- R1 à R3 : Résistances 1KΩ 1/4 W
- R4 à R6 : Résistances 47 Ω 1/4 W
- D1 à D3 : Diode 1N4148
- L1 à L3 : LED ∅ 3mm
- IC1 à IC3 : Opto-Coupleurs MOC341
- TR1 à TR2 : Triacs BTA 12-700B
- VAR1 à VAR3 : Varistances 220 v
- CN1 à CN4 bornier à vis 2 plots (pour facilité le changement des composants)
- SUBD0 : connecteur SUBD9 broches / femelle /coudé 90°

Chaque composant est caractérisé par ses propriétés de fonctionnement et sa composition, nous allons détaillons la Description de chaque composant par la suite:

III.1 Diode

La diode est un composant électronique qui ne laisse passer le courant électrique que dans un seul sens.

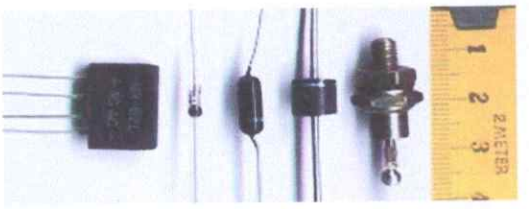


Figure 2 : *Les différents Type de diode*

➢ **Applications usuelles**

Les diodes sont utilisées dans des montages redresseurs et écréteurs principalement.

Comme composant discret (composant utilisé isolément), elles peuvent servir de détrompeur dans un circuit où la polarité est indispensable au bon fonctionnement en empêchant la circulation du courant dans le mauvais sens.

Elles sont fréquemment utilisées dans le redressement du courant alternatif.

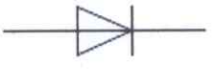


Figure 3 : *Représentation symbolique d'une diode dans un circuit*

➢ **Principe de fonctionnement**

Lors de l'aboutement des deux cristaux, les électrons surabondants de la partie N ont tendance à migrer vers la partie P pour y boucher les « trous ». Il se crée une zone sans porteur de charge, isolante, appelée zone de déplétion. Il existe donc, à l'équilibre thermodynamique, une différence de potentiel entre la partie N et la partie P (dite potentiel *de* jonction).

III.2 Varistance

➢ **Description**

Une varistance est une résistance électrique très fortement non linéaire, utilisée aujourd'hui principalement pour faire des parafoudres.

Varistance signifie simplement « résistance variable ».

Les varistances sont composées d'oxydes métalliques (oxydes de zinc ZnO pour les dernières générations, le carbure de silicium (SiC) ayant été utilisé antérieurement).

Les poudres d'oxyde métallique sont assemblées par frittage sous la forme de bloc de céramique, le plus souvent sous forme de cylindre ou de disque.

➢ **Principe**

Ces blocs de varistance ont une caractéristique tension/courant extrêmement non linéaire, Au delà d'un certain seuil de tension l'impédance de la varistance chute pour permettre l'évacuation du courant créant la surtension, quand la tension revient à son niveau normal l'impédance de la varistance reprend sa valeur à l'état de veille (donc pas de courant). Pour les fortes amplitudes de courant dévié la tension aux bornes de la varistance augmente.

Ce type de composant est utilisé pour protéger le réseau de distribution et également les équipements terminaux sous forme d'appareillage électrique modulaire mais pas pour les équipements de télécommunication en haut débit car la capacité parasite induit un affaiblissement du signal dans les hautes fréquences.

III.3 Opto-coupleur

Un Opto-coupleur (ou optocoupleur) est un dispositif simple d'optoélectronique servant à assurer une isolation galvanique entre deux circuits électriques.

➢ **Composition**

Le principe de base de l'Opto-coupleur est très simple : Il se compose de :

- une boîte étanche à la lumière avec à l'intérieur et face à face
- une diode électroluminescente (DEL)
- un phototransistor

➢ **Fonctionnement**

Sur le schéma de figure 4, la diode est représentée entre les broches 1 et 2 tandis que le phototransistor est situé entre les broches 3 (collecteur) et 4 (émetteur).

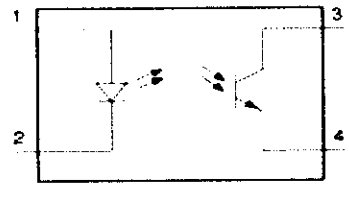


Figure 4 : *Schéma d'un Opto-coupleur*

➢ **Utilisations**

Les Opto-coupleurs sont utilisés, là où une séparation galvanique de deux circuits électriques est nécessaire: par protection ou pour relier deux circuits qui n'ont pas les mêmes masses. Les Opto-coupleurs ne peuvent commander que des courants continus, mais se distinguent par des vitesses de commande très élevées et un domaine de tension relativement large.

III.4 Triac

Le tria est muni d'un dispositif qui détecte le passage à zéro de la tension du secteur afin d'éviter de générer des parasites lors de l'alimentation de la charge.

Le triac est un composant actif de plus en plus utilisé dans les montages d'aujourd'hui. permet en effet de commuter des charges importantes à partir de circuits beaucoup plus modestes.

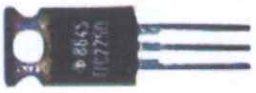


Figure 5 : *Triac*

➢ **Fonctionnement**

Ce composant dont le symbole est rappelé figure 6 fonctionne théoriquement selon quatre modes, appelés aussi quadrants.

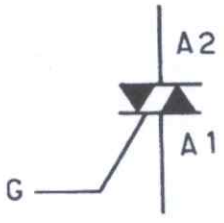


Figure 6 : *schéma d'un triac*

Ces quatre quadrants dépendent en fait de la polarité de l'anode A2 par rapport à A1 et de la gâchette du triac.

III.5 Led

Une diode électroluminescente (abrégée en **DEL**), également appelée **LED** de l'anglais pour light-emitting diode est un composant électronique capable d'émettre de la lumière lorsqu'il est parcouru par un courant électrique.

Une DEL produit un rayonnement monochromatique incohérent à partir d'une transformation d'énergie. Elle a un spectre d'émission continu et fait partie de la famille des composants optoélectroniques.

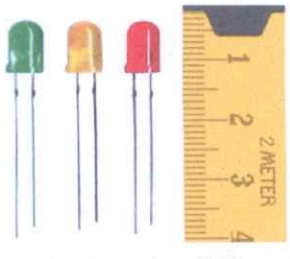


Figure 7 : *Diodes de différentes couleurs*

➢ **Utilisations :**

L'amélioration du rendement des DEL permet de les employer en remplacement de lampes à incandescence ou fluorescence classiques, à condition de les monter en nombre suffisant :

- noyées dans le bitume pour la matérialisation des pistes la nuit ou par temps de brouillard.
- Signalisation portative individuelle (piéton, cycliste).
- Feux de signalisation automobile ou motocycliste (clignotant, veilleuses, feux de position).
- Lampes de poche à piles ou à générateur incorporé
- Lampes alimentées par panneau solaire de balisage des jardins

IV. PRINCIPE D'UTILISATION DE LA CARTE ELECTRONIQUE

La réalisation ne doit pas poser de problème cependant il faut garder à l'esprit que la tension du secteur est présente sur plusieurs pistes de la carte, aussi il peut être prudent lors des manipulations une fois le montage mis sous tension. Un cordon de 3 conducteurs permettant alors d'effectuer la liaison au montage. La carte pourra directement être connectée au port série du pc.

La carte électronique réalisée est illustrée par des photos. Les figures 8 et 9 illustrent le tracé du circuit imprimé et le schéma d'implémentation des composants électronique.

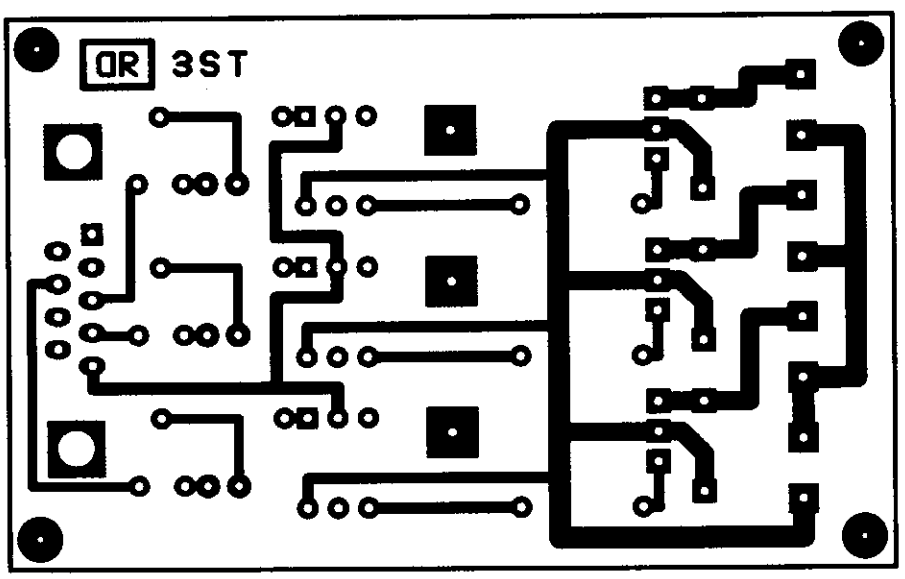


Figure 8 : *Tracé du circuit imprimé*

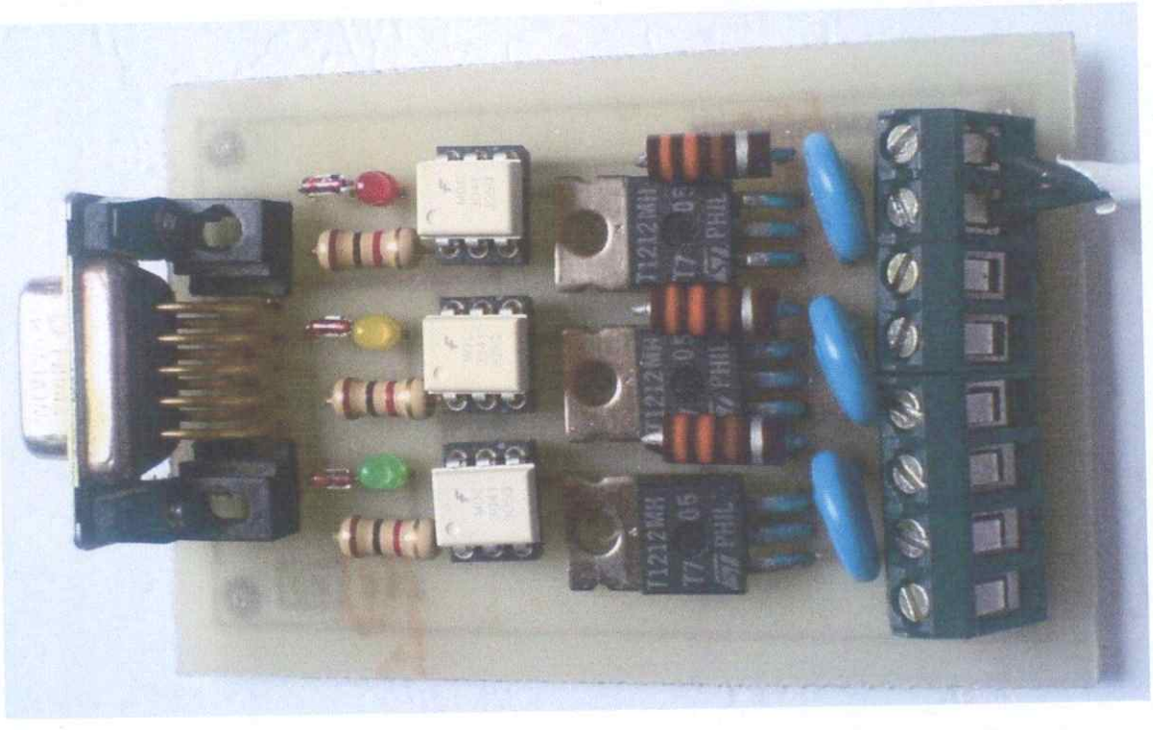


Figure 9 : *carte électronique réalisée*

V. CONCLUSION

Dans ce chapitre, nous avons présentés la description de la carte électronique et ses composants, nous avons expliqués fonctionnement de chaque composant et son rôle dans la carte.

La carte électronique est illustrée par des photos le, circuit imprimé à l'échelle 1(typon) représentant la connexion des composants électroniques a été réalisé au niveau du CDTA (Centre de Développement des Technologie Avancé).

Dans allons utilisée la 3st seulement pour la démonstration le trois commande peuvent être changer l'état d'une appareil.

ANNEXE C

Pilote trois sorties

I. INTRODUCTIN

Le pilotage s'effectue avec les trois sorties du port RS232 : DTR (broche n° 3), TXD (broche n° 4) et RTS (broche n° 7), Alors avec ce pilote crée peut être d'envoyer trois signaux (commande) vers l'appareil médical.

I.1 Définition

Le pilote 3 sorties est une interface entre l'unité de commande (appareil médicaux) et notre application qui permettant d'envoyer 3 signaux à travers le port série.

Nous avons utilisons le langage C++ pour créer la class Pilote.

I.2 La classe Pilote :

❖ Déclaration de la classe

```
#ifndef PILOTE_H
#define PILOTE_H

#include <string>

using namespace std;

class Pilote
{
public:
    //------ CONSTRUCTOR ------

    Pilote();
    virtual ~Pilote();

    //------ OUVERTURE ET CONFIGURATIO ------

    bool ouvrir_port(int numPort, long vitesse);        //ouvrire le port Com "numPort" de vitesse de transmission
                                                        // par deffaut : 8 bits / 1 stop bit / no parity).
                                                        // Return: true si succés.
    bool ouvrir_port(int numPort, long vitesse,         //ouvrire le port Com "numPort" de vitesse "speedInBaud".
              int nbBit, int parity, float nbStopBit);  // suivé par : "nbBit" bit / "nbStopBit" stop bit / "parit
                                                        // Return: true si succés.
    void closeCom();                                    //fermer le port série.

    //------ ENVOI DES COMMANDE PAR LES LIGNES DE COMMANDES ------

    bool comande1(bool val);            // Envoyer la comande n°1. Return: true if success.
    bool comande2(bool val);            // Envoyer la comande n°2. Return: true if success.
    bool comande3(bool val);            // Envoyer la comande n°3. Return: true if success.

    string getErrorMsg();           // Return: un message d'error  generé par ces fonctions.

private:
     std::string Nom_Port;

    HANDLE  hcom;               //Fichier de sortie sur le port COM

    DCB  dcb;               //Configuration du Port
    int  bufferSize;

);

#endif
```

❖ Définition de la classe

```
#include <windows.h>
#include <stdio.h>

#include "Pilote.h"


////////////////////////////////////////////////////////////////////////
// Construction/Destruction                                           //
////////////////////////////////////////////////////////////////////////

/// Constructeur de la classe Pilote
/// les paramètre d'une communication série sur COM1 à 9600 Bauds, 8 bit et pas de parté
Pilote::Pilote()
{
    hcom = NULL;    //Fichier de sortie sur le port COM
    bufferSize = 8192;  // Buffer d'entrée sortie.
}
Pilote::~Pilote()
{
    if(hcom != NULL)
        closeCom();
}

bool Pilote::ouvrir_port(int numPort, long vitesse)
{
    return openCom(numPort, vitesse, 8, 0, 1);
}


/*****************************************************************************
**** Fonction: ouvrir_port
**** Rôle: Ouvre le port RS232 choisi.
**** Renvoie: Faux si l'ouverture du port a échoué, sinon Vrai.
*****************************************************************************/

bool Pilote::ouvrir_port(int numPort, long speedInBaud, int nbBit, int parity, float nbStopBit)
{   char buf[] = "COM1";

    //--- Vérification des paramètres passés à la fonction:
    if(numPort<1 || numPort>9)      // Vérification que le numéro de port est compri entre 1 et 9.
        return false;

    if(speedInBaud<1)       // Vérification de la vitesse de communication
        return false;

    if(nbBit<5 || nbBit > 9)
        return false;

    if(parity<0 || parity > 2)
        return false;

    if(nbStopBit<1 || nbStopBit > 2)
        return false;

    //--- Ouverture du port série en lecture / écriture.
    hcom=CreateFile(buf, GENERIC_READ | GENERIC_WRITE, 0, 0, OPEN_EXISTING , 0, 0);
    if (hcom==0 || hcom==INVALID_HANDLE_VALUE)
        return false;
```

```
    //--- Dimensionne les buffers d'entrée et sortie.
    if ( !SetupComm(hcom, bufferSize, bufferSize) )
        return false;

    //--- Récupère l'objet de configuration du format des bytes.
    if ( !GetCommState(hcom, &dcb))
        return false;


    //--- Stocke l'objet de configuration du format des bytes.
      if (!SetCommState(hcom, &dcb))
        return false;
      else
        return true;

}

/*****************************************************************************
**** Fonction: CloseCom
**** Rôle: Ferme le port RS232 actuellement ouvert.
*****************************************************************************/

void Pilote::closeCom()
{
    CloseHandle(hcom);
    hcom = NULL;
}
/*************************** comande1(val) ************************************************
          if( EscapeCommFunction(hcom, CLRBREAK) == TRUE )
              return true;
      }
   return false;
}



/*************************** comande3(val) ************************************************
**** Positionne DTR au niveau val (0 ou 1)   */
bool Pilote::comande3(bool val)
{
    if(val)
      {
          if( EscapeCommFunction(hcom, SETDTR) == TRUE )
              return true;
      }
    else
      {
          if( EscapeCommFunction(hcom, CLRDTR) == TRUE )
              return false;
      }
    return false;
}

//*******************************La fct d'erreur*************************
string Pilote::getErrorMsg()
{
    LPVOID lpMsgBuf;
```

II. CONCLUSION

Dans ce annexe nous avons présentés le code source de la classe Pilote réalisée (déclaration et définition), afin de savoir les caractéristiques, le fonctionnement...du port série (chapitre 1).

LISTE DES FIGURES

CHAPITRE 1 : GESTION DE RS232 SOUS WINDOWS

CHAPITRE 2 : CONTRÔLE À DISTANCE

CHAPITRE 3 : CONCEPTION DE L'APPLICATION LOGICIELLE

CHAPITRE 4 : REALISASION DU SYSTEME

ANNEXE A : UML

ANNEXE B : PRESENTATION DE LA CARTE 3ST

Bibliographie

[1] Les informations se trouvent sur le site web www.Wikipédia.com

[3] « Architecture des systèmes d'exploitation »

[4] les informations se trouvent sur le site web www.royale.zerezo.com.

[5] « cours de réseau informatique » Abdelhalim Akka.

[6] « son image Réseau : transmission d'une séquence vidéo » Rémi BARLAND et Vicent
RAMPAL. Ecole Supérieur en Science Informatique

[7] « Cours de réseau Maîtrise d'informatique » Pascal Niclas.

[8] « réseau informatique »
Les informations se trouvent sur le site web www.commentcamarche.fr

[9] « Architecture client/serveur »
Les informations se trouvent sur le site web www.centralweb.fr

[10] « le guide de l'utilisateur UML » Grady booch, James Rumbaugh etIvar Jacobson.
Edition EYROLLES.

[11] « UML en français »
Les informations se trouvent sur le site www.UMLenfrancais.fr

[12] « méthode s formelles avec UML, Modélisation, Validation et généralisation de
testes1 » Aliain le guennec , IRISA / Université de Rennes.

[13] « UML Unified Modeling Longuage –langage unifié pour la modélisation objet -»
Frédéric Juliard, Université de bretagne.

[14] « Microsoft Windows XP »
Les informations se trouvent sur le site www.Microsoft.com

[15] « les serveurs Web »
Les informations se trouvent sur le site www.Alianwebserver.com

[16] « présentation d'EasyPHP »
Les informations se trouvent sur le site www.Developpez.com

[17] « C++ ». Les informations se trouvent sur le site web www.Wikipédia.com