

Chapitre 4

Conception et Réalisation

CHAPITRE 4 : CONCEPTION ET REALISATION

1. Introduction

Le développement des systèmes de communication et de transmission des données, a donné naissance à des nouvelles technologies telles que la téléphonie sur IP. Ça se qui nous pousse à réaliser un projet de téléphonie IP. Donc Qu'est ce qu'un projet de téléphonie IP ?

Un projet de téléphonie sur IP vise à rendre obsolète la présence d'un PABX. Dans un tel projet, les téléphones traditionnels doivent être remplacés par des téléphones IP (IPPHONE ou SoftPhone). Le PABX, quant à lui, est remplacé par un logiciel s'exécutant sur un système d'exploitation (Windows, Linux, Unix, etc.) appelé "soft switch". Néanmoins, en pratique, la migration se fait le plus fréquemment via la mise en place d'une infrastructure hybride où des téléphones traditionnels coexistent avec des téléphones IP. On parle alors d'IP PABX.

Afin de réaliser un projet de téléphonie IP on contient la Conception et Réalisation d'une Application TOIP (Talk Over IP) sous windows de téléphonie sur IP de type pc-to-pc (SoftPhone).

2. La conception

Comme nous avons vu dans le deuxième chapitre un SoftPhone est un téléphone (logiciel) s'installant sur les PC des utilisateurs, qui, via une sortie son (casque ou haut-parleur) et un micro, permet de communiquer avec un autre utilisateur via le réseau voix sur IP. Cette solution offre des services multimédia encore plus vastes qu'un IPPhone matérielle. Il est géré par le serveur de TOIP "Asterisk" (PABX-IP), Donc tout les softphone sont gérés par le serveur par une architecture client/serveur mais les communications entre clients sont des communications peer to peer (communication entre abonnés).

2.1. Architecture de application

La conception consistait à réaliser les fonctions communication vocal.

Le client se connecte au serveur et le serveur est transféré au client que vous voulez appeler qui est enregistré sur un serveur.

2.1.1. Communication vocale

- **Un module audio** : pour la lecture et l'acquisition vocale.
- **Un module de découpage** : pour décomposer (coder et compresser) le flux audio sur des unités transférables sur le réseau et pouvoir les reconstituer à la réception.
- **Un module de communication et transmission** : pour connecter les correspondants et transférer la voix sous forme de paquets IP.

Le principe général de cette fonction est comme suit :

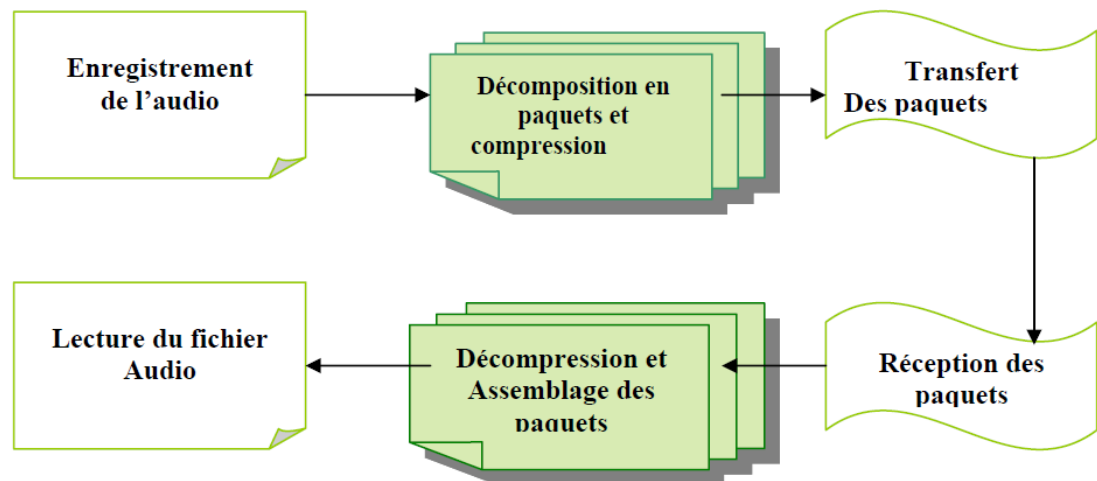


Figure 4.1: Principe de la communication vocal

Dans ce cas l'application va fournir les services suivants :

- ✓ Enregistrement de la voix : acquies la parole par la carte son de l'ordinateur est les enregistrer dans ce ordinateur sous le format .wav.
- ✓ Et la taille de fichier doit être choisie de sorte que la perte de données doit être annulée.
- ✓ Décomposition des fichiers audio en paquet : Il faut que la taille des paquets doive être fixée par l'utilisateur pour contrôler le transfert.
- ✓ Transfert des paquets : transformation des paquets sur IP
- ✓ Réception des paquets : les paquets transférés par l'émetteur sont arrivé au récepteur (ordinateur destinataire).
- ✓ Assemblage des paquets : création d'un fichier pour stocker les paquets reçus. A la fin de transmission l'émetteur envoie un signal indiquant la fin du transfert des paquets et le récepteur ferme le fichier.
- ✓ Lecteur du fichier audio : le signal Donnés obtenu, est converti en signal analogique pour sa restitution sonore.

2.1.2. Softphone

Un **softphone** est un type de logiciel utilisé pour faire de la téléphonie par Internet depuis un ordinateur plutôt qu'un téléphone. Les communications peuvent se faire au moyen d'un microphone et d'un casque ou de haut-parleurs reliés à la carte son, mais il existe aussi un type de périphérique dédié à cette tâche, semblable à un téléphone et se branchant sur un port USB.

Les interfaces des **softphones** sont souvent intuitives et de la forme d'un téléphone. Les fonctionnalités des **softphones** sont les mêmes que celles des téléphones classiques. En plus des fonctionnalités de téléphonie classique, les softphones incorporent souvent des services supplémentaires comme la video sur IP, la présence, permettant de connaître la disponibilité de ses contacts et de nombreux autres services. Par ailleurs, l'application peut également être intégrée avec d'autres applications installées sur l'ordinateur : avec la messagerie électronique par exemple, afin de pouvoir appeler directement un numéro de téléphone dans un courriel (*click-to-call*) ou d'accéder au gestionnaire de contacts pour générer des appels.

3. Réalisation

3.1. Environnement logiciel de développement

3.1.1. Langage de programmation

Nous avons utilisé le C# comme langage de programmation orienté objet (sous le Visual Studio 2012). Nous avons choisi l'orienté objet pour la facilité de développement, de maintien et de la réutilisation des différents modules de l'application.

Nous avons utilisé serveur de TOIP "Asterisk" (PABX-IP) pour coûté serveur.

3.1.2. Les bibliothèques utilisées

Nous avons utilisé bibliothèque :

VoIPSDK.dll : Cette bibliothèque regroupe toutes les classes nécessaires pour l'utilisation du technique " VOIP ". Cette bibliothèque constitue le noyau de notre application, donc il faut qu'elle soit présente tout le temps dans la mémoire.

3.2. Utilisation de l'application

Cette application est compatible seulement avec lui-même, pour l'utiliser il faut vérifier les étapes suivantes :

- ✓ L'application doit être installée sur les deux postes.
- ✓ Connaître l'adresse IP du serveur de TOIP "Asterisk" (PABX-IP).
- ✓ Configuration du microphone.
- ✓ L'application active sur les postes.

3.3. Serveur

Asterisk est un autocommutateur téléphonique privée (PABX) open source pour les systèmes d'exploitation UNIX, il est publié sous licence GPL.

Asterisk comprend un nombre très élevé de fonctions, tel que les appels téléphoniques, la messagerie vocale, les fils d'attentes, les conférences, etc. Il implémente plusieurs protocoles H.320, H.323, SIP et IAX. Durant ce chapitre, on montrera les étapes d'installation et de configuration d'Asterisk sous le système d'exploitation Linux, ainsi que l'installation et la configuration de Blink qui est un téléphone VoIP softphone, freeware.

3.3.1. Installation d'Asterisk 1.4

Avant d'installer Asterisk, il faut préparer le système sous lequel on installera notre serveur. Pour cela, il faut installer tout d'abord les pré-requis nécessaires

❖ Détermination des pré-requis

Les pré-requis nécessaires pour que l'installation du serveur Asterisk s'accomplit avec succès, sont classés dans un tableau ci-dessous :

Nom du paquetage	Commande d'installation	Note
GCC 3.x	yum install gcc	Nécessaire pour compiler zaptel, libpri, et asterisk
Ncurses-devel	yum install ncurses-devel	Nécessaire pour menuselect
libtermcap-devel	yum install libtermcap-devel	Nécessaire pour asterisk

Kernel Development Headers	yum install kernel-devel	Nécessaire pour compiler zaptel
Kernel Development Headers (SMP)	yum install kernel-smp-devel	Nécessaire pour compiler zaptel
GCC C++ 3.x	yum install gcc-c++	Nécessaire pour asterisk
OpenSSL (optionnel)	yum install openssl-devel	Dépendance de OSP, IAX2 encryption, res_crypto (RSA key support) Nécessaire pour asterisk
zlib-devel (optionnel)	yum install zlib-devel	Dépendance de DUNDi Nécessaire pour asterisk
unixODBC; unixODBC-devel (optionnel)	yum install unixODBC-devel	Dépendance de func_odbc, cdr_odbc, res_config_odbc, res_odbc, ODBC_STORAGE
Libtool (optionnel; recommandé)	yum install libtool	Dependence de ODBC-related modules
GNU make (version 3.80 ou plus)	yum install make	Nécessaire pour compiler zaptel et asterisk

Tableau 4.1 : Liste de paquetages nécessaires pour compiler asterisk, libpri et zaptel

❖ Téléchargement des codes sources

Voilà les lignes de commandes nécessaires pour le téléchargement d'Asterisk et libpri identifie l'url. Après on télécharge via la commande *wget*

```
# cd /usr/src/

# wget http://downloads.asterisk.org/pub/telephony/asterisk/releases/asterisk-1.4.tar.gz

# wget http://downloads.asterisk.org/pub/telephony/libpri/old/libpri-1.4.3.tar.gz

# wget http://downloads.asterisk.org/pub/telephony/zaptel/releases/zaptel-1.4.8.tar.gz
```

❖ Extraction des paquetages

Les paquetages téléchargés sont des archives compressés qui contiennent le code source, on aura besoin de les extraire, en utilisant la commande *tar*, avant de les compiler.

```
# tar zxvf zaptel-1.4.8.tar
# tar zxvf libpri-1.4.3.tar
# tar zxvf asterisk-1.4.0.tar.gz
```

❖ Compilation et installation

Le Zaptel est un noyau chargeable qui présente une couche d'abstraction entre le matériel et les pilotes de Zapata dans le module Asterisk.

# cd /usr/src/zaptel-1.4.8	=accès au dossier du Zaptel
# make clean	=supprime les fichiers inutiles après installation.
# ./configure	=construction d'un nouveau fichier <i>makefile</i> .
#make menuselect	=execution de la partie menuselect dans le fichier make file
#make	=compilation du code source
# make install	=execution de la partie install dans makefile.

Makefile est un fichier qui contient les instructions à exécuter à partir des commandes, *./configure*, *make*, *make install*, *make config*, etc. chacune de ces commandes exécute le code approprié à elle dans ce fichier.

Libpri est utilisé par les décideurs du multiplexage temporel (TDM) des appareils VoIP, mais même s'il n'y a pas le matériel installé, il est conseillé de compiler et installer cette bibliothèque. Elle doit être compilé et installé avant Asterisk, car elle sera détecté et utilisé lorsqu'Asterisk est compilé.

```
# cd /usr/src/libpri-1.4.3
# make clean
# make
# make install
```

Asterisk est un serveur de téléphonie open-source permettant de disposer sur un simple PC les fonctions réservées aux PABX professionnel

```
# cd /usr/src/asterisk-1.4
# make clean
# ./configure
# make menuselect
# make install
# make samples
```

Dans le cas où on voudrait bien lancer zaptel et le serveur asterisk au démarrage du système, il faut exécuter après la compilation et l'installation des paquets la commande suivante :

```
# make config           =cette commande charge le serveur Asterisk au démarrage du système
```

Ainsi Asterisk est installé il suffit maintenant de lancer le serveur et de se connecter à la console CLI (Command Line Interface) via la commande :

```
# asterisk -r
```

3.3.2. Configuration d'Asterisk

❖ Identification des fichiers de configuration

Une fois l'installation d'Asterisk est effectuée, plusieurs fichiers sont créés :

- /usr/sbin/ : Contient le fichier binaire d'Asterisk (programme principal).
- /usr/lib/asterisk/ : Contient les fichiers binaires qu'Asterisk utilise pour fonctionner.
- /usr/lib/asterisk/modules/ : Contient les modules pour les applications, les codecs, et les drivers.
- /var/lib/asterisk/sounds/ : Contient les fichiers audio utilisés par Asterisk, par exemple pour les invites de la boîte vocale.
- /var/run/asterisk.pid : Fichier contenant le numéro du processus Asterisk en cours.
- /var/spool/asterisk/outgoing/ : Contient les appels sortants d'Asterisk.
- /etc/asterisk/ : Contient tous les fichiers de configuration.

Le dernier dossier nous intéresse vu qu'il contient les fichiers de configuration du serveur Asterisk, parmi ces fichiers on trouve :

- **agents.conf**: Contient la configuration de l'utilisation des agents, comme dans le cas d'un centre d'appel. Ceci nous permet de définir les agents et de leur assigner des ID et des mots de passe.

- **asterisk.conf:** Définit certaines variables pour l'utilisation d'Asterisk. Il sert essentiellement à indiquer à Asterisk où chercher certains fichiers et certains programmes exécutables.
- **extensions.conf:** Configure le comportement d'Asterisk. C'est le fichier qui nous intéresse le plus dans ce travail.
- **iax.conf:** Configure les conversations VoIP en utilisant le protocole Inter-Asterisk Exchange (IAX).
- **rtp.conf:** Ce fichier de configuration définit les ports à utiliser pour le protocole RTP (Real-Time Protocol). Il faut noter que les numéros listés sont des ports UDP.
- **sip.conf:** Définit les utilisateurs du protocole SIP et leurs options. On peut aussi définir d'autres options globales pour SIP telles que, quels ports utiliser et les timeout qu'on va imposer. Nous focalisons sur ce fichier puisque notre solution est basée sur le protocole SIP.
- **zapata.conf:** Configure les paramètres de l'interface téléphonique Zapata.

3.3.3. Configuration des comptes users

Les deux fichiers à configurer sont *sip.conf* et *extensions.conf*. Dans le fichier *sip.conf*, on créera des utilisateurs utilisant le protocole sip pour l'établissement de la connexion, voilà les deux clients que nous avons créés au niveau du fichier :

```
[100]
type=friend          (spécifie le type d'utilisateur)
context=sip          (spécifie le type de routage à utiliser)
host=dynamic         (spécifie une adresse IP par laquelle l'utilisateur peut accéder à son
compte)
dtmfmode=rfc2833
secret=100            (mot de passe)
callerid= "100"<1111> (identifiant d'utilisateur)

[200]
type=friend          (spécifie le type d'utilisateur)
context=sip          (spécifie le type de routage à utiliser)
host=dynamic         (spécifie une adresse IP par laquelle l'utilisateur peut accéder à son
compte)
```

```
dtmfmode=rfc2833
secret=200          (mot de passe)
callerid="200"<1113> (identifiant d'utilisateur)
```

3.3.4. Configuration des extensions

Passant maintenant à la configuration du fichier *extensions.conf*

```
[sip]          (il faut saisir le nom du context entre crochet)
exten=>1111,1,Dial (SIP/100,20,tr)
               (20 est la durée en seconde de l'attente avant la décrochage si pas de réponse)
exten=>1113,1,Dial (SIP/200,20,tr)
```

Si l'appelant compose le numéro 1111, il est mit en relation avec le poste dont le numéro est 1111 qui utilise le protocole SIP.

Il existe d'autres options qu'on peut ajouter dans le fichier *extensions.conf*, telles que la boîte vocale et le renvoi d'appel. La syntaxe du fichier est sous le format suivant :

exten=> extension, priorité, commande (paramètre)

- Extension: C'est généralement le numéro de téléphone ou le nom du client.
- Priorité: C'est un numéro qui indique la priorité de la commande, le serveur prend en considération la priorité de la commande en utilisant le numéro inscrit dans la syntaxe.
- Commande: C'est la commande qui peut exister, comme la commande dial (appel), voicemail (boîte vocale), etc.

On peut utiliser plusieurs options pour un seul numéro d'appel, on peut mettre par exemple un transfert d'appel vers un autre numéro ou vers la boîte vocale selon des priorités.

```
exten => 123,1,Answer
exten => 123,2,Playback      (répondeur)
exten => 123,3,Voicemail(9)  (9 est le numéro de la boîte vocale)
exten => 123,4,Hangup
```

Dans chaque ajout ou modification d'un client, il faut mettre à jour le serveur Asterisk en utilisant les commandes suivantes :

```
Localhost*CLI> sip reload
Localhost*CLI> dialplan reload
Localhost*CLI> reload
```

Ouvert à tous, gratuit et simple d'utilisation. Asterisk a de quoi s'imposer. Ces vrais concurrents sont plutôt les PBX Hardware. Qui sont chers mais performant et fiable.

Les solutions libres peuvent fournir les outils les plus performants et les mieux documentés sans procurer un même service relationnel.

3.4. Client

Un client est un type de logiciel utilisé pour faire de la téléphonie par Internet depuis un ordinateur plutôt qu'un téléphone. Les communications peuvent se faire au moyen d'un microphone et d'un casque ou de haut-parleurs reliés à la carte son.

3.4.1. Interface principal de l'application (SoftPhone)

C'est la fenêtre principale de l'application qui permet la communication vocal



Figure 4.2: Interface principal de l'application.

3.4.2. Fenêtre Register client

Cette fenêtre permet de registration de client dans serveur de TOIP "Asterisk" (PABX-IP).

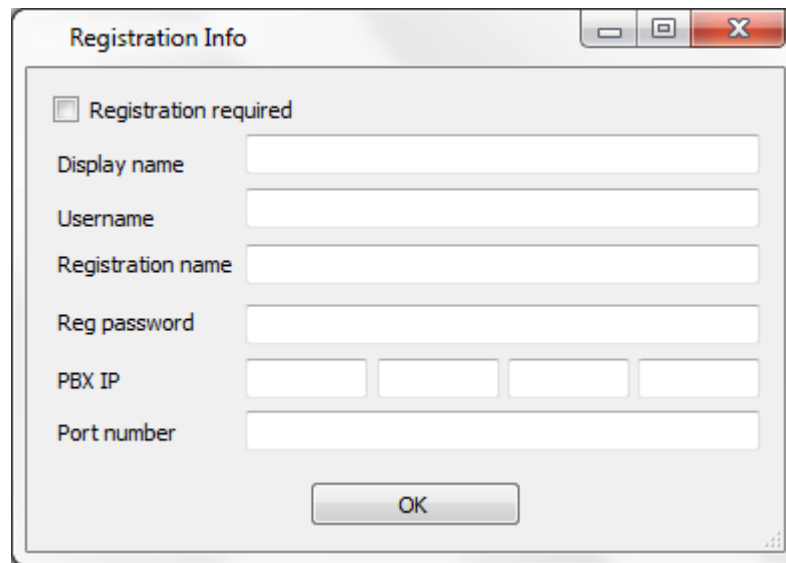
The image shows a Windows-style dialog box titled "Registration Info". It contains a checkbox labeled "Registration required" which is currently unchecked. Below this are several input fields: "Display name", "Username", "Registration name", and "Reg password", each followed by a single-line text box. The "PBX IP" field is followed by four separate single-character input boxes. The "Port number" field is followed by a single-line text box. At the bottom center of the dialog is an "OK" button. The dialog has standard Windows window controls (minimize, maximize, close) in the top right corner.

Figure 4.3: Fenêtre register client dans serveur de TOIP "Asterisk" (PABX-IP).

4. Conclusion

Dans ce chapitre j'ai présenté une conception et une réalisation d'une application Windows de téléphonie IP qui est un SoftPhone dont communication vocale. Bien que notre système paraisse primitif quant à la qualité de transmission, nous pensons que le présent travail servira de support à d'autres extensions telles que l'intégration d'autre services (le transfère des fichiers sur IP, le fax, double appel ...) ou par développer cette même application pour fonctionner sous linux en se basant sur le protocole SCTP pour améliorer la qualité de service