CHAPITRE 3 : PRESENTATION DES PROCEDURES DE TRAITEMENT D'APPEL LORS D'UNE COMMUNICATION ENTRE DEUX ABONNES DU RESEAU TELEPHONIQUE FIXE

D'après ce qu'on a vu, le SS7 intervient dans beaucoup de domaines. Si on ne mentionne que les plus courants, il est utilisé par le réseau téléphonique fixe et le réseau mobile.

Cette partie du travail, qui est réservée à la simulation se portera sur un cas particulier, que l'on a choisi pour évoquer clairement le principe du SS7. Ceci consiste en une présentation des procédures de traitement d'appel lors d'une communication entre deux abonnés du réseau téléphonique fixe, en s'échangeant des messages SS7. Pour la présente étude, on s'intéressera donc au fonctionnement au niveau de l'application, plus précisément au protocole de niveau4.

1 But de la simulation

Le but est d'une part, de fournir un outil permettant de comprendre le SS7, que ce soit à travers son architecture, ou ce soit à travers même de son principe de fonctionnement. D'autre part, un objectif à atteindre est de pouvoir, par la même occasion, faire comprendre la simulation en apportant les documentations nécessaires concernant le réseau sémaphore et le réseau téléphonique dans la réalisation même, afin que quiconque se servira de l'outil réalisé puisse assimiler le fonctionnement.

2 Choix

Comme déjà introduit ci-dessus, cette simulation concerne les procédures de traitement d'appel entre deux abonnés du réseau téléphonique fixe. Le travail à réaliser est de présenter les messages SS7 échangés entre les entités du réseau pendant cet appel.

2.1 Choix de l'animation

Ce qu'on désire pour ce travail c'est de faire comprendre le fonctionnement de la signalisation SS7. La meilleure façon d'atteindre ce but c'est de présenter graphiquement les événements et de montrer ce qui se passe pour les cas de figure qui puissent se présenter. C'est la raison pour laquelle on a intégré l'animation pour la réalisation de la simulation.

2.2 Choix du langage de programmation

Le choix d'un langage de programmation dépend du domaine d'application. Le travail à réaliser nécessite ici un langage permettant de dessiner les graphiques nécessaires à la présentation, d'insérer des boutons pour servir d'outil pour l'utilisateur, de faire les tests suivant les manipulations effectuées, de jouer un son et surtout de faire des animations pour présenter chacun des événements.

Il existe quelques outils offrant ces fonctionnalités. En ce qui concerne le présent travail, on a opté pour le langage « Java ». En effet, outre les nombreux avantages que présente ce langage de programmation, comme le fait d'être sûr, sécurisé, robuste, haute performance, portable, indépendant des architectures, le choix s'est basé sur les quelques caractéristiques suivantes :

- Java est un langage orienté objet : un objet est un programme avec ses caractéristiques et comportements propres. Chaque fichier source contient la définition d'une ou plusieurs classes qui sont utilisées les unes avec les autres pour former une application. Un développeur peut utiliser des objets déjà conçus et il n'est plus nécessaire de réécrire un objet qui existe déjà.

- Java permet de réaliser des applications graphiques. Il possède des classes fournissant les composants graphiques.

- Java permet de jouer du son au format quelconque voulu.

- Java est multithreadé et dynamique : il permet l'utilisation de threads qui sont des unités d'exécution isolées. Ce qui permet de réaliser des animations sous java et donc adaptée à la réalisation voulue.

D'autant plus que c'est un langage enseigné au sein du Département Télécommunication, on estime qu'il serait judicieux d'appliquer et d'approfondir les notions reçues.

3 Présentation du travail réalisé [1] [14] [24] [31]

3.1 Présentation des interfaces

Dès que le programme est lancé, on est directement mené à la page d'accueil présentée ci-après :

Figure 3.01: Présentation de la page d'accueil

L'icône « flèche » situé au coin inférieur droit permet de parcourir la page suivante par un simple clic.

La nouvelle page qui s'ouvre après *[Figure 3.02]* présente à première vue la partie simulation, qui n'est autre que la présentation des messages SS7 échangés entre les entités du réseau, lors de l'appel se passant entre les deux abonnés du réseau téléphonique fixe.

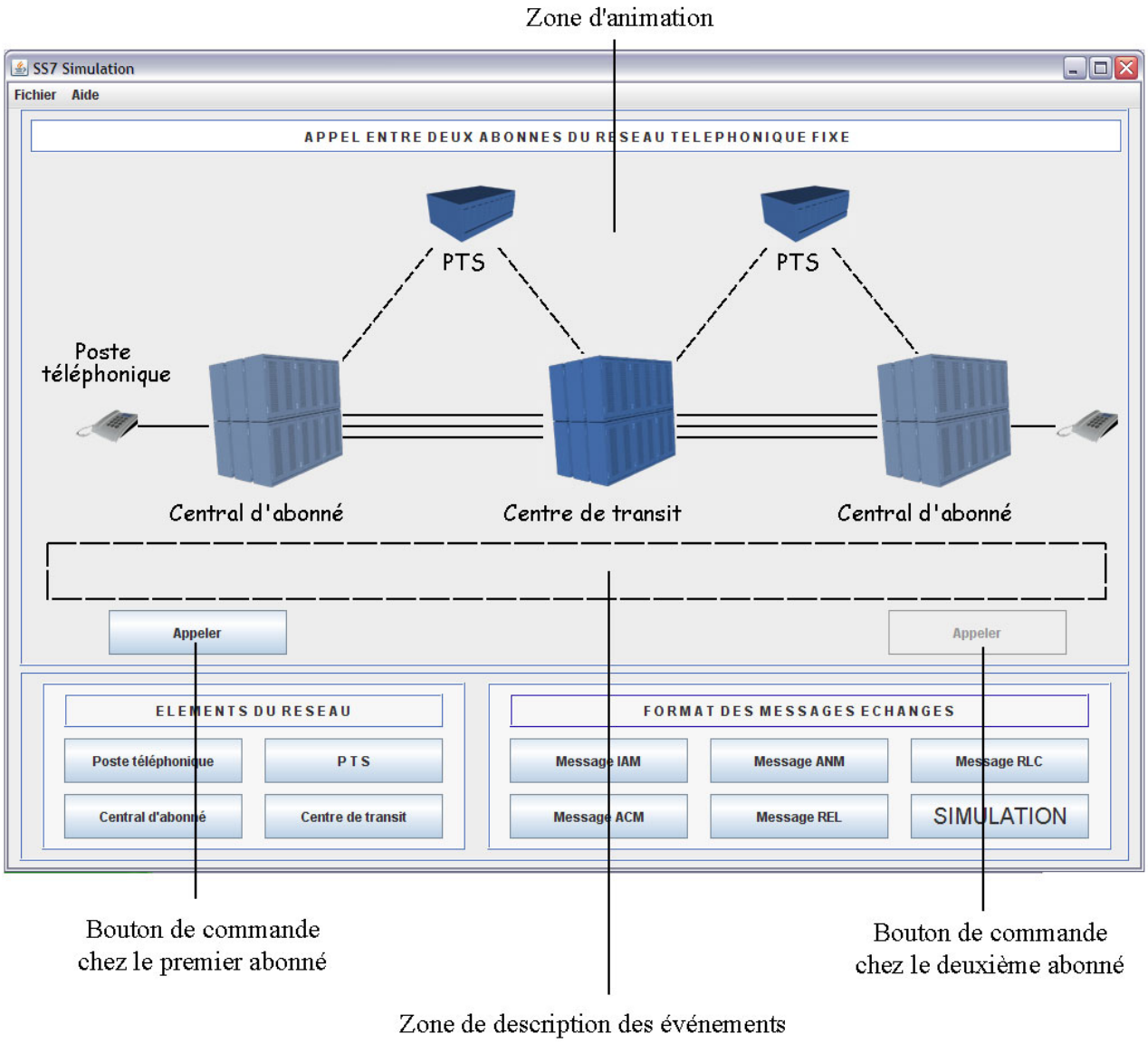

Figure 3.02 : Présentation de la fenêtre de la simulation

3.2 Présentation des éléments de la simulation

3.2.1 Architecture du réseau

Le réseau présenté pour la simulation est formé des éléments suivants :

- pour chacun des abonnés, un poste téléphonique et un central auquel celui-ci est rattaché, les deux reliés par des lignes d'abonné,

- un centre de transit pour mettre en exergue son existence dans le réseau téléphonique,
- deux PTS, chacun reliant le central d'abonné au centre de transit,
- des faisceaux de circuits mettant en relation les autocommutateurs entre eux,
- des canaux sémaphores pour véhiculer les messages de signalisation SS7.

Remarque : Cette architecture n'est qu'un choix arbitraire. Elle peut sembler différente du réseau réel qui pourrait être plus compliqué, mais le but ici c'est de pouvoir comprendre le principe.

3.2.2 Outils de commande

Pour la manipulation, différents boutons sont mis à la disposition de l'utilisateur dont un bouton poussoir de chaque côté des abonnés. Les opérations effectuées par l'abonné se traduisent donc par l'appui sur ces boutons.

3.3 Simulation

1. L'appel commence par appui sur le bouton « Appeler » du côté de l'abonné demandeur. Ce qui équivaut à l'envoi des chiffres de la numérotation vers son central de rattachement.

2. Après, le central d'abonné analyse le numéro reçu. Lorsqu'il connaît la direction de l'appel, il envoie un message d'initiation nommé « IAM », ou Initial Address Message, sur un canal de signalisation vers le centre de transit. Pour ce faire, le message est routé par le PTS afin d'atteindre ce centre de transit.

   Lorsque le message IAM émis par le commutateur d'origine est reçu par le commutateur intermédiaire c'est-à-dire le centre de transit, ce dernier sélectionne un circuit libre à partir du numéro du destinataire présent dans le message IAM reçu. Il émet alors à son tour un message IAM au commutateur suivant, c'est-à-dire au central auquel est rattaché le demandé, tout en passant toujours par le PTS.

Remarque : Du fait que le SS7 ne concerne que la signalisation entre les commutateurs c'est-à-dire dans le cœur du réseau, on s'attache de ne pas présenter dans la simulation les signaux échangés pour l'accès de l'usager au réseau.

En terme d'illustration, une capture du déroulement de la simulation est donnée ci-après. En fait, une flèche montrant la direction et la nature de chaque message échangé est représentée.

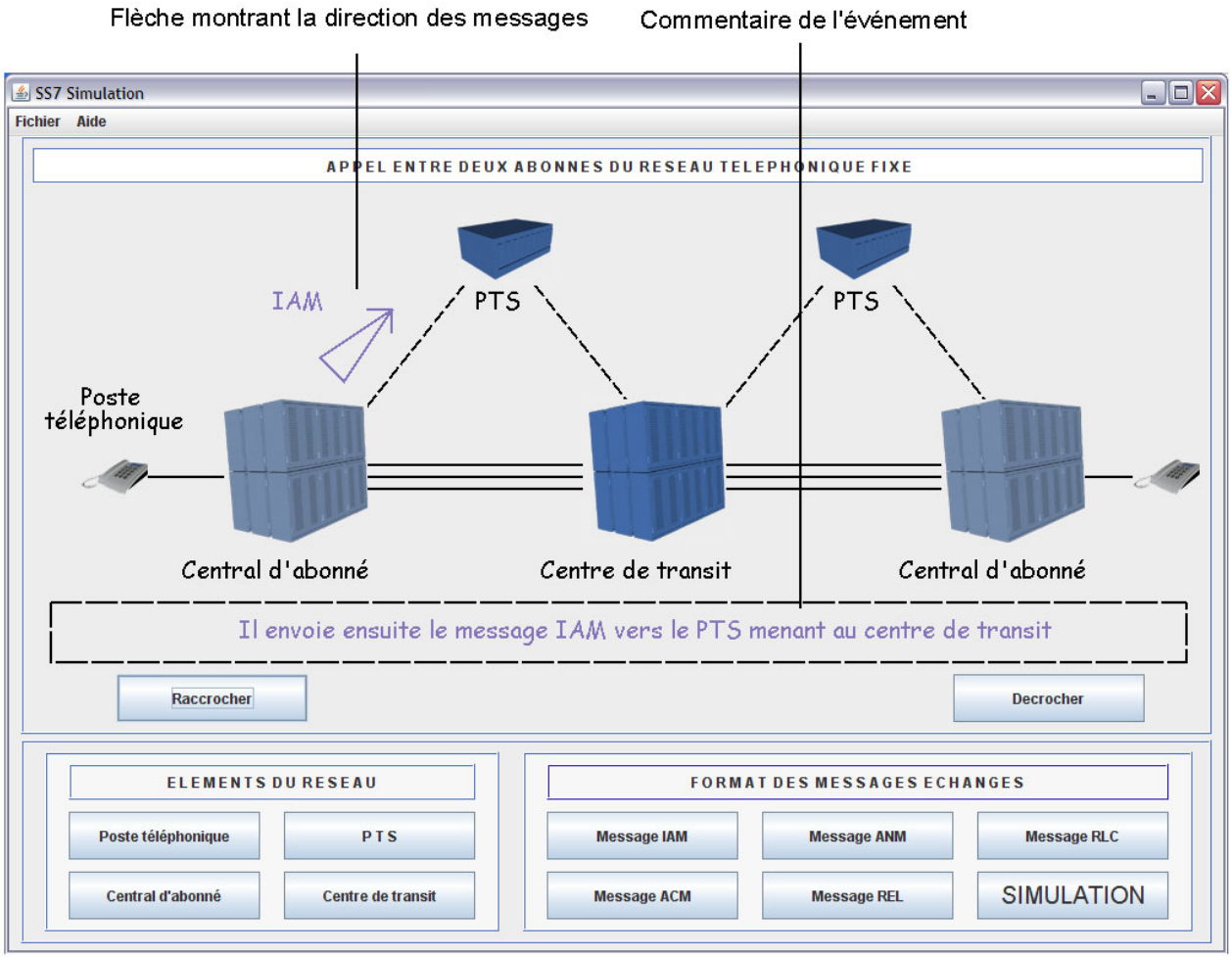

Figure 3.03 : Capture d'une animation

Si la taille de l'information à envoyer est supérieure à 272 octets (ce qui représente la taille maximum de l'information pouvant être encapsulée dans le champ information d'une trame TSM), le message IAM est segmenté par le commutateur origine grâce à l'utilisation du message de segmentation (SGM, Segmentation message). Deux messages sont alors émis à la suite, le premier étant un message IAM et le second, un message de segmentation. Un paramètre optionnel « indicateur d'appel facultatif émis vers l'avant » dans le premier message contient ainsi un bit qui permet d'indiquer au commutateur suivant que le message IAM est segmenté en deux messages.

Le message IAM est donc le premier message SS7 émis dans le réseau pour l'établissement d'un appel. Ce message contient le numéro du demandé ainsi qu'un certain nombre d'informations concernant le demandeur, l'acheminement et l'appel.

Pour voir plus en détail, voici le format de ce message :

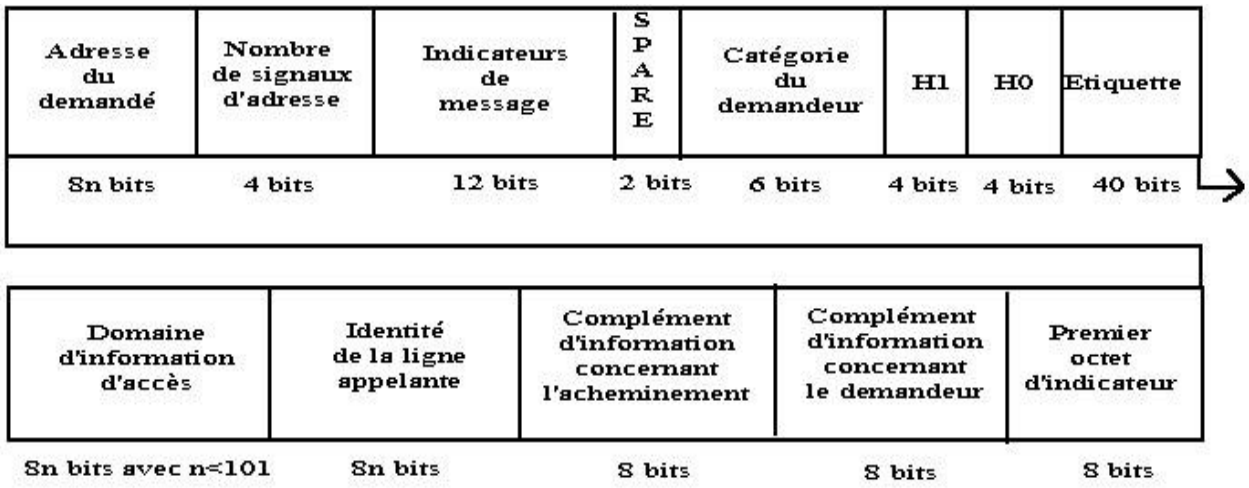

Figure 3.04 : Format du message IAM

3. Lorsque le message IAM est reçu par le commutateur d'arrivée (commutateur d'accès du destinataire), ce dernier analyse le numéro demandé pour déterminer à quel correspondant l'appel doit être connecté. Il vérifie aussi l'état de la ligne du demandé et procède à diverses vérifications pour déterminer si la connexion est autorisée ou non.

   Le choix s'est porté sur un état qui admet que l'abonné désiré est dans un état libre, le central affirme que la demande d'appel peut aboutir.

   Pour cela, il renvoie en arrière un message « ACM » ou Address Complete Message qui est acheminé de proche en proche jusqu'au commutateur d'origine. Le message d'adresse complète doit provoquer l'établissement de la connexion dans tous les centres qu'il traverse. C'est à ce moment que retentit la sonnerie du poste du demandé et de l'autre côté, l'appelant entend dans son combiné la tonalité d'appel, sauf dans le cas d'un terminal à réponse automatique.

Le message ACM quant à lui a le format suivant :

Domaine d'informations d'accès	Indicateur de présence d'information à recevoir sur le circuit	En réserve	Nature de l'accès du demandé	Indicateurs de message	H1	H0	Etiquette
8n bits avec n<=51	1 bit	4 bits	3 bits	8 bits	4 bits	4 bits	40 bits

Figure 3.05 : Format du message ACM

La phase pendant laquelle la sonnerie retentit chez le demandé a été représentée comme suit :

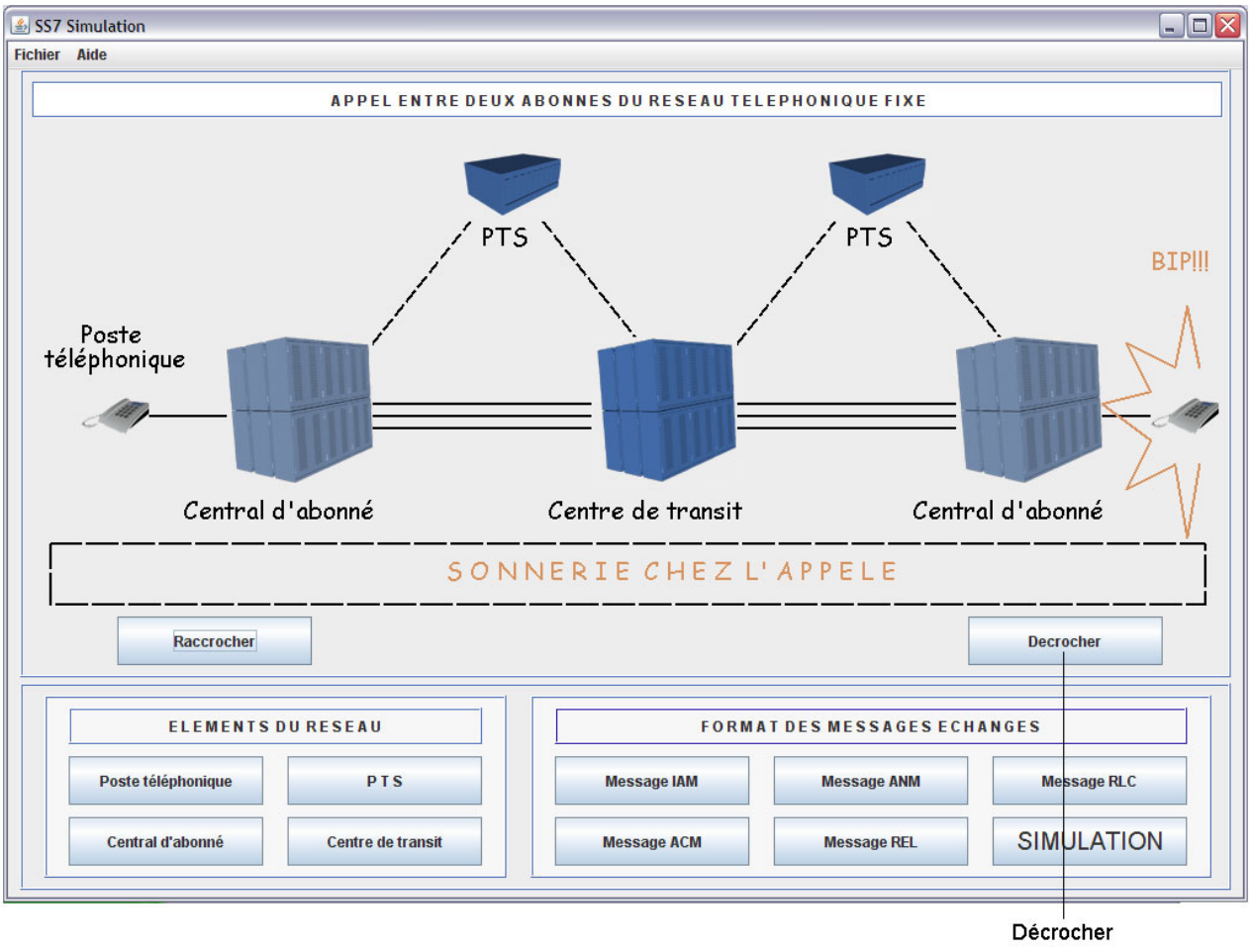

Figure 3.06 : Capture lors de la phase de sonnerie

Et du fait que java offre la fonctionnalité de jouer du son, on a également intégré dans cette phase de sonnerie un programme permettant d'entendre la sonnerie de l'abonné demandé, jusqu'à

ce que ce dernier décroche son combiné, sinon, lorsque le demandeur décide de raccrocher son téléphone avant que le correspondant désiré décroche le sien.

4. Dès que l'appelé décroche son combiné, ce qui est présenté pour cette simulation par l'appui du bouton « Décrocher » du coté de l'appelé, un nouveau message de réponse nommé « ANM », ou Answer Message, est envoyé depuis le central de rattachement du demandé vers celui de l'appelant, tout en passant par le centre de transit. C'est à partir de cet instant que peut commencer la communication entre les deux correspondants et il est à noter que les signaux de conversation sont transmis sur le circuit préalablement établi. Ceci fait démarrer généralement la taxation au centre de départ.

On montre par la suite le format général du message ANM.

Figure 3.07 : Format du message ANM

Les échanges précédemment effectués se résument par le chronogramme qui suit :

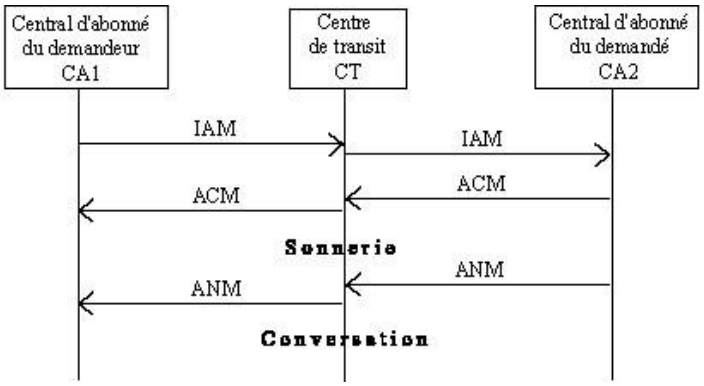

Figure 3.09 : Chronogramme des échanges

Voici comment on avait illustré la phase de conversation entre les deux abonnés.

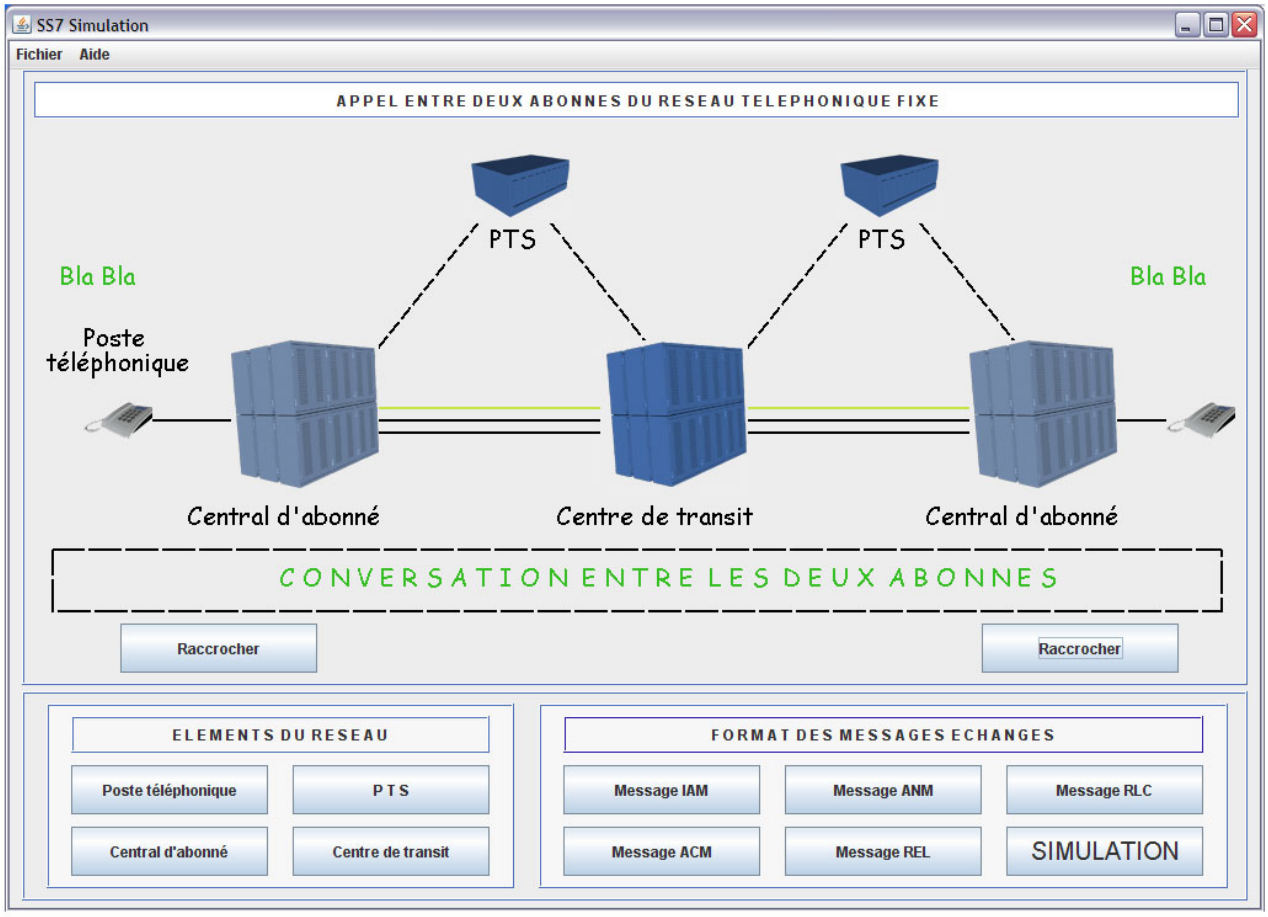

Figure 3.08 : Capture lors de la phase de conversation

Ici, un autre son a été introduit pour la simulation d'une petite conversation entre les deux correspondants.

La prochaine étape de la simulation concerne la terminaison de l'appel. Là, deux cas de figures peuvent se présenter :

- soit c'est l'appelant qui raccroche en premier (bouton « Raccrocher » du côté de l'appelé)
- soit c'est l'appelé qui l'effectue en premier. (bouton « Raccrocher » du côté de l'appelant).

Cas 1 : L'appelé raccroche en premier

5. Lorsque l'abonné demandeur raccroche en premier, le signal de fin « REL » ou Realese est émis depuis son central de rattachement vers le centre de transit sur le canal de

signalisation. Le centre de transit réagit en émettant vers l'amont un signal de libération de garde « RLC » ou Realese Complete qui a pour effet de libérer ce même circuit dans le centre amont et puis vers l'aval un autre signal de fin REL qui est relayé jusqu'au central de l'appelé. C'est après qu'un message RLC est transmis vers le centre de transit pour libérer cette deuxième partie du circuit.

Pour mieux comprendre ce premier cas de libération, on va donner son chronogramme *[Figure 3.10]*.

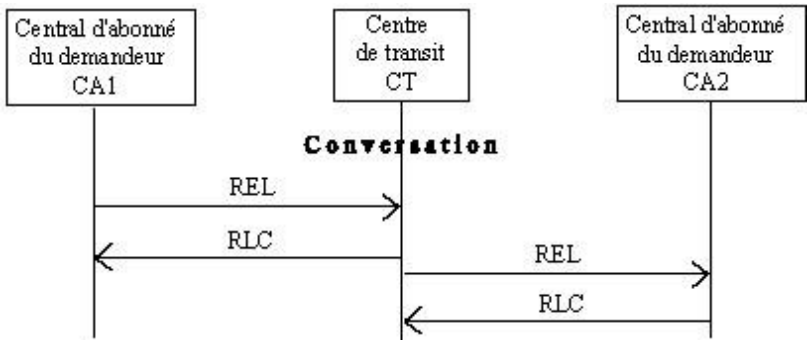

Figure 3.10 : Premier cas de libération

Cas 2 : L'appelant raccroche en premier

6. Lorsque l'abonné demandé raccroche en premier, la libération se fait autrement. Un signal de raccrochage du demandé « RAU » (Raccrochage) est d'abord émis depuis son central de rattachement. Ce message de raccrochage du demandé ne doit pas provoquer la rupture du trajet de conversation, mais doit provoquer le démarrage d'une temporisation d'attente de raccrochage du demandeur.

   Le message de fin REL est ensuite envoyé depuis le central de l'appelant après que le demandeur a raccroché ou que la durée de la temporisation est écoulée. C'est seulement par la suite donc que la libération s'effectue pas à pas, la libération de la partie située entre le central d'abonné et le centre de transit, puis la libération de la partie située entre ce dernier et le central du demandé, par le message RLC.

Ce deuxième cas de libération se résume par le chronogramme suivant :

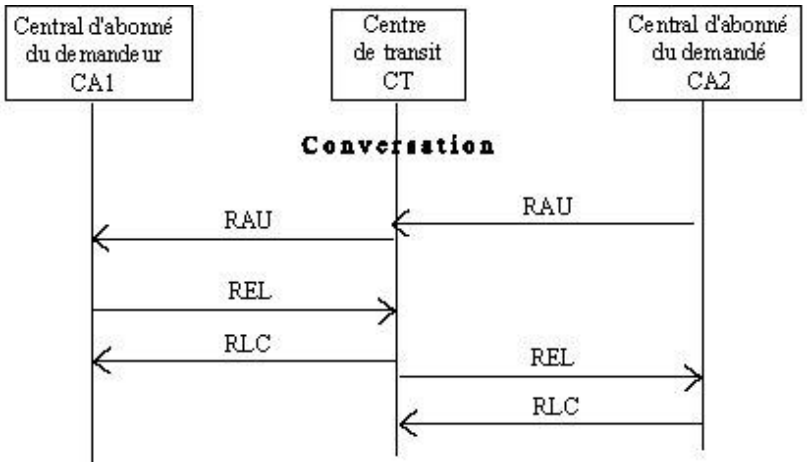

Figure 3.11 : Deuxième cas de libération

Dans cette partie, on va présenter le format général des messages de libération REL et RAU.

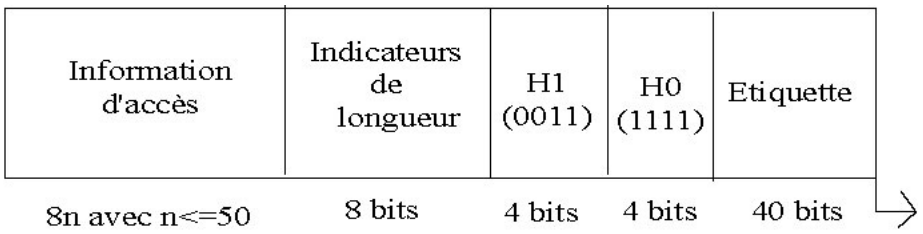

Figure 3.12 : Format du message REL

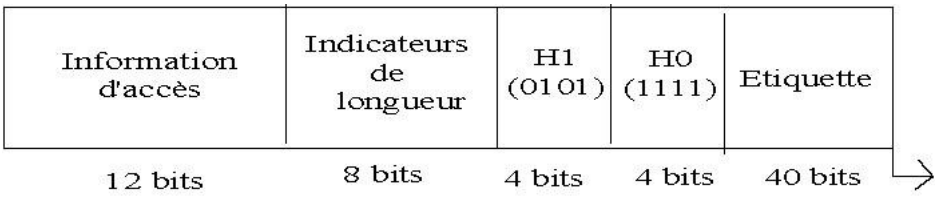

Figure 3.13 : Format du message RAU

La fin de l'appel est illustré comme suit :

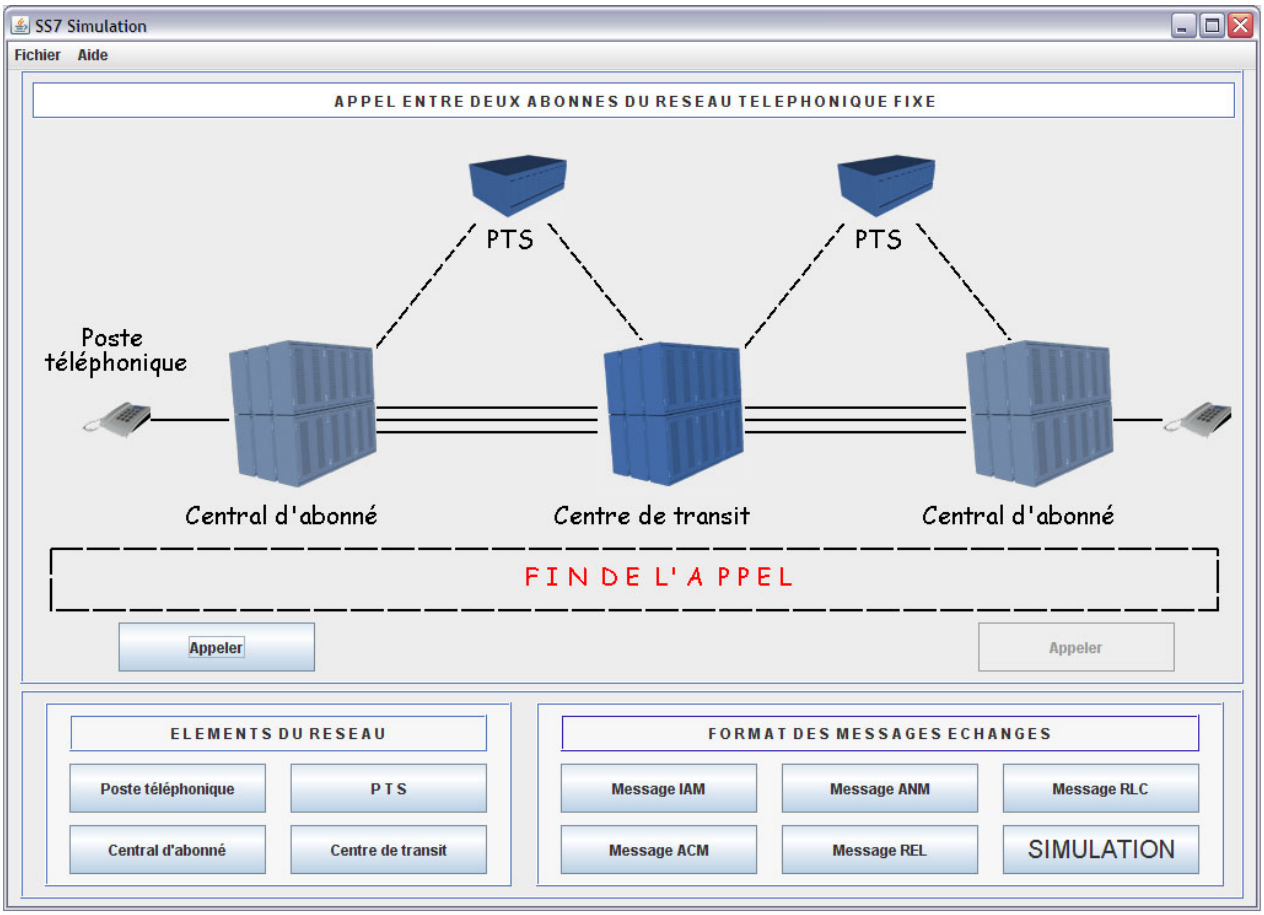

Figure 3.14 : Représentation de la fin de l'appel

Etant donné que le travail est fait pour faciliter la compréhension des échanges lors de l'appel, une chose voulue est que l'utilisateur puisse bien suivre les étapes de la communication. C'est la raison pour laquelle on a intégré la zone de commentaire, réservée à la description des événements au fur et à mesure que la simulation avance.

Ce que l'on a mis en exergue et qu'il faut remarquer lors de cette simulation est :

- le fait que le SS7 concerne les échanges de signaux entre les commutateurs du réseau dans le but d'établir et de rompre une communication
- le passage des messages de signalisation par les PTS qui les dirige vers les centraux destinataires

- la dissociation entre le canal qui transporte la signalisation et le canal véhiculant par la suite la conversation
- la différence entre la procédure de libération dans le cas où c'est l'appelé qui raccroche en premier et celui où c'est l'appelant qui le fait en premier
- et les différents messages SS7 échangés.

3.4 Complément de la présentation

Etant donné que cette simulation a été élaborée dans un but de fournir un élément d'apprentissage, on a également produit des accès auxquels l'utilisateur pourra recueillir toutes les informations nécessaires à la compréhension de la simulation, à savoir les informations sur les différents éléments du réseau mis en œuvre ici. C'est la nécessité des différents boutons que l'on trouve dans la moitié inférieure de la page *[Figure 3.15]*.

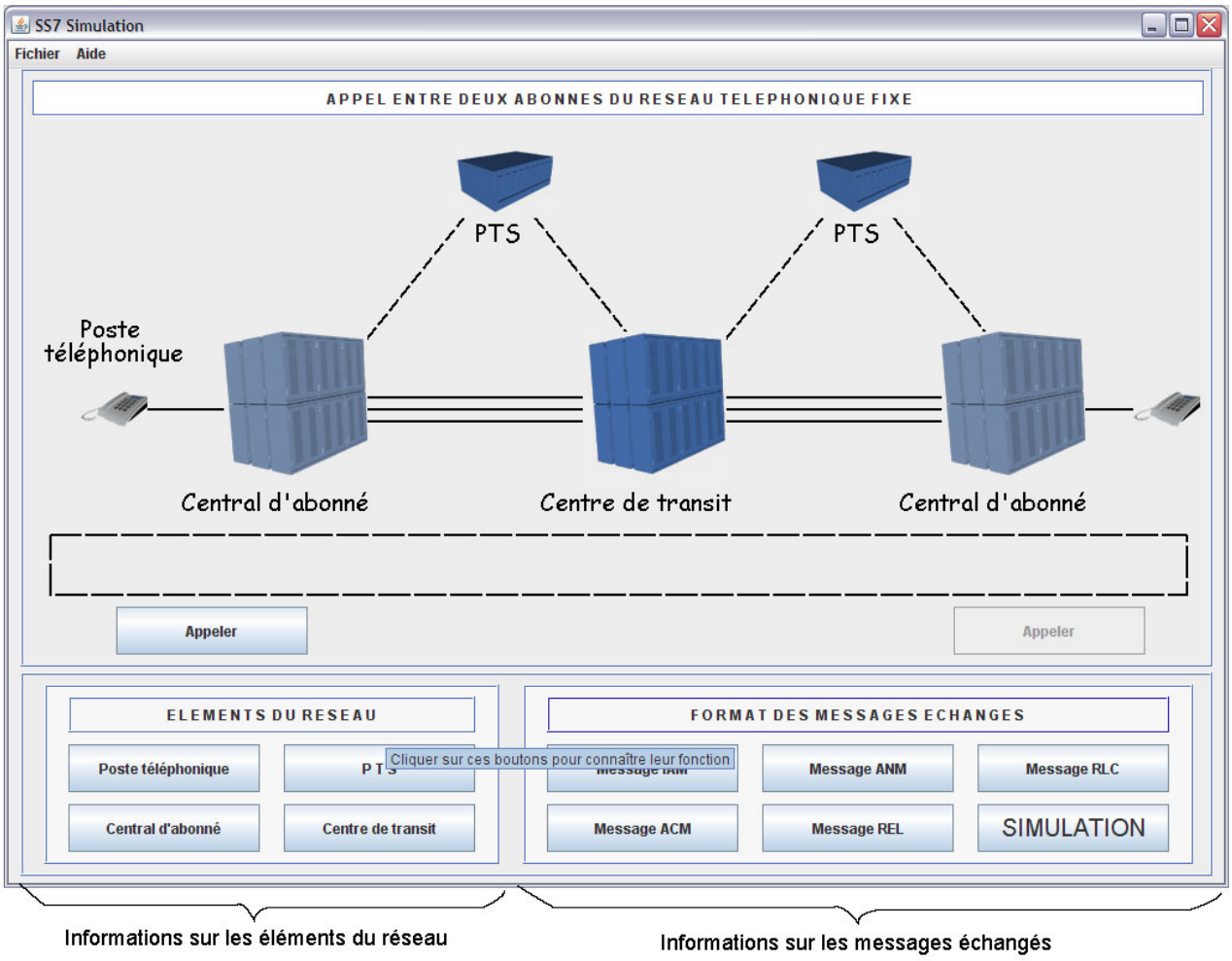

Figure 3.15 : Outils d'acquisition d'information

En effet, on y trouve des informations nécessaires sur les éléments vus lors de la simulation à savoir :

- le poste téléphonique
- le central d'abonné
- le centre de transit
- et le PTS.

Les informations s'obtiennent par appui sur ces boutons. Mais pour plus de simplicité de manipulation, un simple clic sur le graphique situé dans la zone d'animation permet également d'avoir les informations correspondantes.

En guise d'exemple, voici l'information disposée à propos du poste téléphonique.

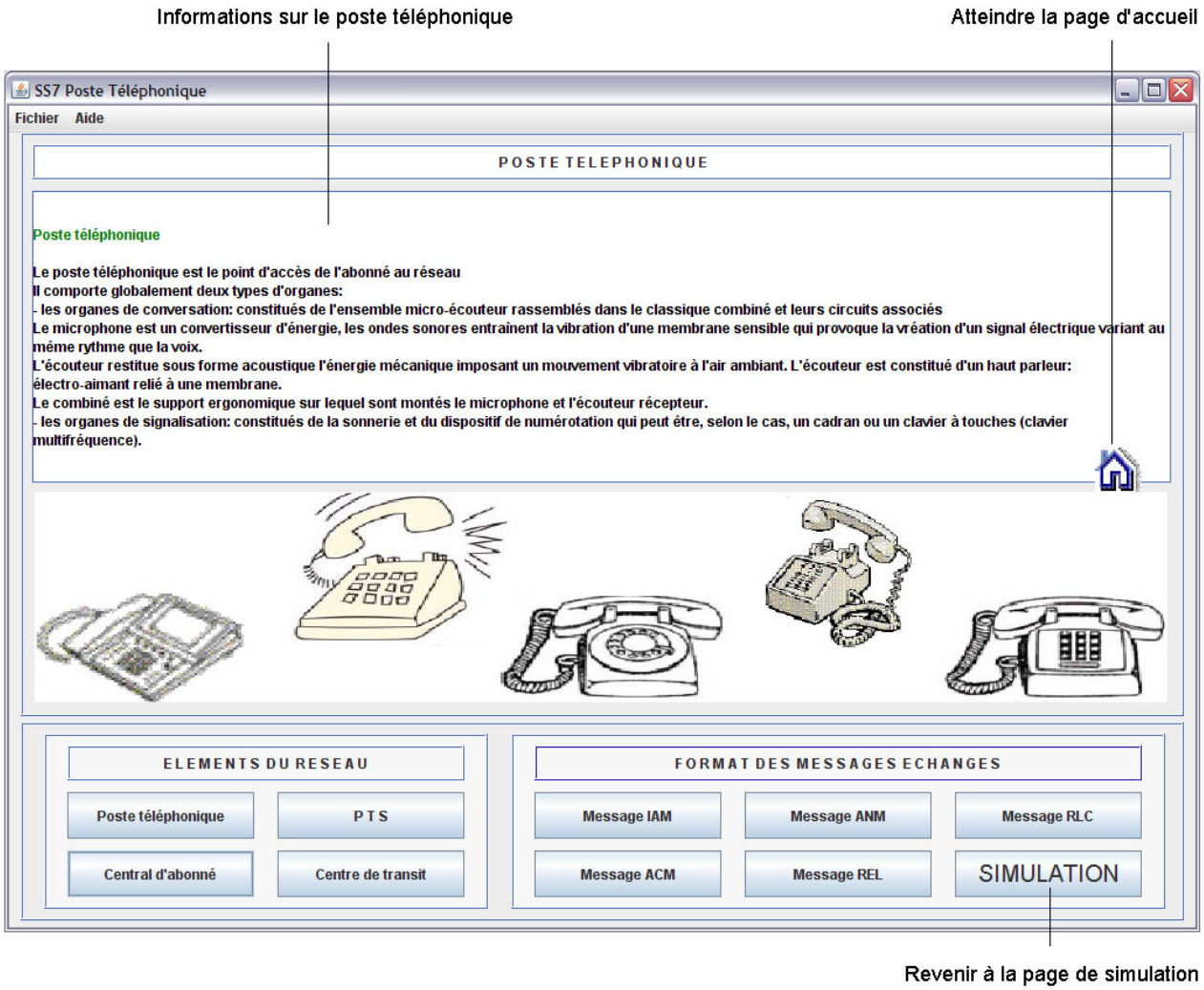

Figure 3.16 : Présentation du volet d'information sur le poste téléphonique

Par ailleurs, des informations concernant les différents messages échangés sont également disponibles sur cette partie, intégrant le format de chacun de ces messages sur un menu déroulant.

À titre indicatif, on montrera ci-après ce qu'on obtient sur appui du bouton « Message IAM ».

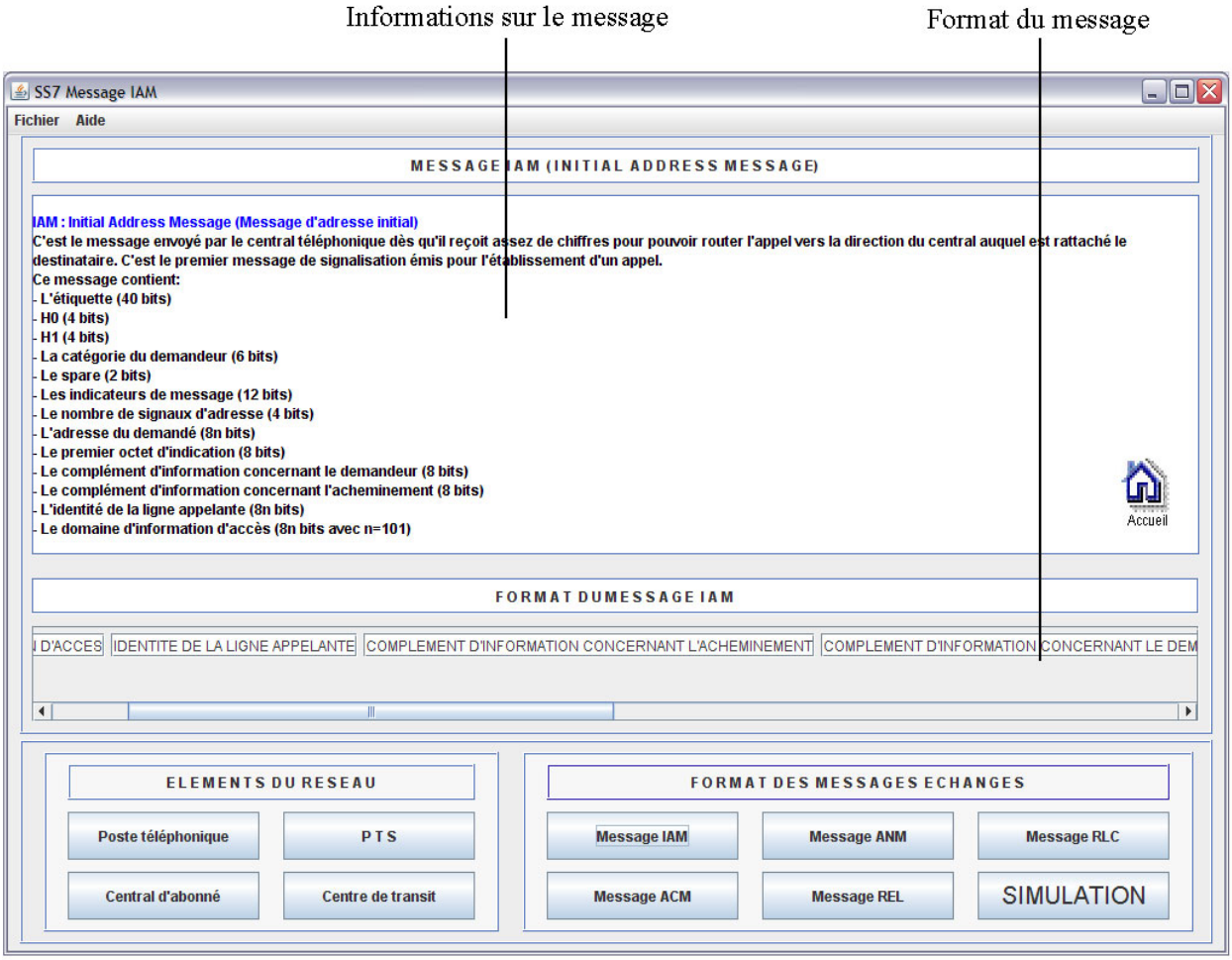

Figure 3.17 : Présentation du volet d'information du message IAM

Il est à noter que l'on peut aussi acquérir plus d'informations c'est-à-dire voir plus de détails sur les champs du format des messages par un clic sur l'élément désiré. Une nouvelle fenêtre s'ouvre pour cela, comme on peut le voir sur la capture d'écran suivante :

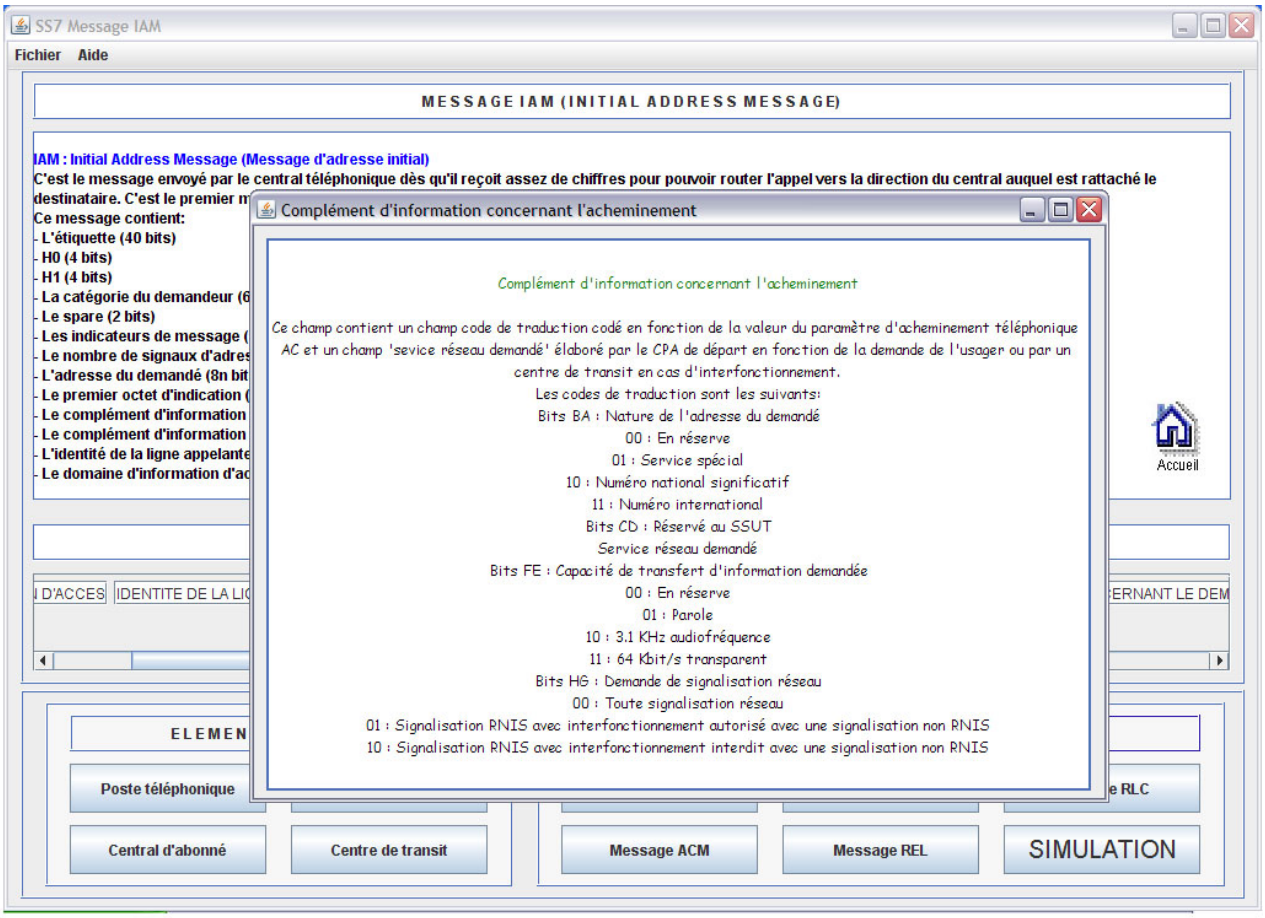

Figure 3.18 : Fenêtre d'information des champs

Remarque : Pour revenir à la partie simulation, il fallait appuyer sur le bouton « SIMULATION » situé au côté inférieur droit de chaque page. Pour aller plutôt à la page d'accueil, on clique sur l'icône « home page ». D'ailleurs, un petit manuel d'utilisation est mis à la disposition des utilisateurs dans la barre de menu « Aide » pour les guider.

4 Programme [32] [33] [34]

Les lignes de programme permettant d'écrire la réalisation sont assez longues. C'est pourquoi on ne donne ci-après qu'une partie essentielle de la manière dont on l'a réalisée. C'est celle de la classe Dessin, une des classes créées, que l'on va montrer.

Pour dessiner sous java, on dérive l'une des classes suivantes :

- `java.applet.Applet` pour l'écriture d'applets simples
- `java.awt.Canvas` pour l'écriture d'applets complexes et d'applications.

Les classes permettant de créer une forme géométrique quelconque sont définies dans `java.awt.geom.`

Pour dessiner dans un `Canvas`:

- on définit une sous classe de `Canvas`
- on surcharge la méthode public `void paint (Graphics g)` dans cette sous-classe
- ensuite on utilise les fonctions graphiques pour dessiner.

La classe Graphics2D est utilisée pour créer et manipuler les images. Cette classe dérive de la classe `graphics`. Le principe de son utilisation est le suivant :

- on crée une instance d'un objet Graphics2D, par casting sur l'objet Graphics de l'entête de méthode classique
- on utilise cet objet pour les fonctions d'affichage.

Pour les tracés, ils sont définis par l'attribut `Stroke`. Pour définir un tracé, on construit un objet `BasicStroke` et on le passe en argument à la méthode `setStroke`. Pour afficher une image enregistrée sur le disque dur, on utilise la méthode `drawImage`. Pour afficher un texte aux coordonnées précisées, on se sert de la méthode `drawString(String texte)`.

Pour adapter la police et la taille des caractères à un style de son choix, on utilise la classe `Font`. Le constructeur de la classe Font est `Font(String, int, int).` C'est la méthode `setFont()` de la classe `Graphics` qui permet de changer la police d'affichage des textes.

Pour réaliser une animation sous java, on utilise un `Thread`. Le `Thread` pourrait se définir comme l'espace de temps alloué à une tâche. Un objet `Thread` ne peut être défini qu'à l'intérieur d'une classe qui implémente `Runnable`. Dans ce contexte, les procédures de rafraîchissement sont encapsulées dans une boucle, éventuellement infinie. Cette boucle est implémentée dans la méthode `run()` de l'objet et mise en veille pendant un certain laps de temps. L'objet `Thread` utilise la méthode `start()` pour démarrer effectivement le `Thread` sur lequel elle est invoquée. Ce qui va provoquer l'appel de la méthode `run()` du `Thread`.

```
import java.awt.*;
import java.awt.geom.*;
```

```
class Dessin extends Canvas implements Runnable
{
        …
        public void paint(Graphics g)
        {
                Graphics2D g2 = (Graphics2D) g;
                float dash[] = {1.0f,1.0f};
                BasicStroke bs=new BasicStroke (2.0f, 2, 2,0.0f, dash, 0.0f);
                g2.setStroke(bs);
                g2.setColor(new Color(148,150,138));
                Font fonte;
                fonte=new Font("Comic Sans MS",Font.PLAIN,20);
                g2.setFont(fonte);
                g2.drawString("Central d'abonné",125,325);
                …
                for(int x8=280, y8=460, i=0; i<2; i++)
                {
                        g2.drawLine(x8,240,y8,240);
                        x8+=300;
                        y8+=300;
                }
                …
                Rectangle2D r1=new Rectangle2D.Double(15,345,950,50);
                g44.draw(r1);
                …
                int temp=2000;
                Thread t2;
                public void decrocher()
                {
                        t2=new Thread(this);
                        t2.start();
                        k1=100;
```

```
        }
        public void run()
        {
            while(k1==100)
            {
                try
                {
                    …
                    repaint();
                    t.sleep(temp);
                    …
                }
                catch(Exception erre)
                {
                    System.out.println("Erreur");
                }
            }
        }
        …
    }
}
```

Cette partie simulation a permis non seulement de comprendre le fonctionnement du SS7, à travers les messages SS7 échangés lors de l'appel entre deux abonnés du réseau téléphonique fixe, mais permet également de savoir plus sur les éléments même de ce réseau. Ce qui fait que la réalisation est bien adaptée au domaine de l'apprentissage pour tout ce qui est intéressé par le sujet. C'est un exemple simple de communication mais le principe de base reste le même dès qu'il s'agit d'une communication utilisant le protocole ISUP, notamment pour l'établissement et la rupture d'une communication. Il permet d'avoir une petite idée sur les procédures effectuées au sein du réseau lors des simples gestes que l'on fait pour un appel.

Par ailleurs, la réalisation de ce travail a permis d'approfondir la connaissance du langage Java.