LA VOIX SUR GPRS

P. de Frino[1], S. Robert [2], G. Cecchin[3]

Résumé

Cette étude a pour objectif de réaliser une application qui fonctionne sur PDA et qui permette d'envoyer des fichiers de différents types (voix, texte, vidéo) à un destinataire mobile à travers le réseau GPRS. Les données sont envoyées sur un serveur qui génère un message SMS pour avertir le destinataire. Ce dernier peut alors soit stocker, transférer, effacer ou charger les données depuis le serveur. La solution d'envoyer le message sur un serveur plutôt qu'au destinataire directement permet de ne pas se préoccuper s'il a son PDA sur lui pour lire ses messages.

1. Introduction

Les réseaux GSM sont principalement utilisés pour établir des communications vocales. Ils peuvent cependant offrir des services de transmission de données en mode circuit ou pour des échanges de messages courts. En mode circuit, l'échange de données se fait suite à un établissement d'une communications et monopolise le canal durant toute la communication. Ce mode de fonctionnement est loin d'être idéal pour le transfert de données, notament à cause du trafic sporadique engendré. D'autre part la largeur de bande à disposition est relativement faible, 9.6 kb/s. [1,2] alors qu'avec GPRS, *General Packet Radio Service*, on dispose d'un réseau de données avec des terminaux mobiles radio et des vitesses d'accès allant jusqu'à 170 kb/s. Dans ce cas, il n'y a pas de réservation de largeur de bande comme dans le cas de la commutation par circuits. On peut faire usage de la largeur de bande « disponible » du réseau GSM.

Quand GPRS offre un service Internet, le terminal dispose d'une adresse IP dont le champ « réseau » est spécifique au réseau GPRS. Il peut donc être vu comme un réseau de données à part entière, qui utilise une partie du réseau GSM et qui est pourvu d'un accès radio. Les débits prévus permettent d'envisager des applications telles que la consultation du Web, le transfert de fichiers, la transmission de vidéo compressée. Par rapport à GSM, de nouveaux sites ont été introduits :

- Le nœud de service GPRS (**SGSN**, *Serving GPRS Support Node*): routeur qui gère les stations présentes dans une zone donnée.
- Le noeud passerelle GPRS (**GGSN**, *Gateway GPRS Support Node*) est relié à un ou plusieurs réseaux de données: routeur qui permet aux paquets venant des réseaux de données externes d'être acheminés vers le SGSN du destinataire.

De plus, on dispose d'un enregistreur de localisation: GPRS Register (GR) qui contient les informations de routage.

[1] Ingénieur HES en télécommunications, LanExpert, Lausanne

[2] Professeur à l'EIVD-Institut des télécommunications, Yverdon

[3] Ingénieur ETS en électricité, Swisscom Mobile, Berne

2. Service d'envoi de fichiers à travers GPRS

2.1 Architecture matérielle

Le service développé [3] a pour but d'utiliser le réseau GPRS et de permettre l'accès aux ayant-droit uniquement. Il faudrait pour ceci disposer d'un serveur directement relié au réseau de Swisscom Mobile. Or, ceci est impossible pour des raisons évidentes de sécurité, fiabilité et confidentialité. La solution choisie a été de relier le serveur à Internet. La connectivité est assurée puisque de son côté GPRS est aussi relié à Internet. Le serveur (appelé VoGPRS) se trouve en zone démilitarisée à l'adresse IP suivante : 193.134.216.180.

Le client est composé de deux éléments qui permettent, ensemble, de générer, d'envoyer et de récupérer des messages vocaux :

- Ipac de Compaq avec enregistreur vocal pour la création de messages vocaux.
- T39m d'Ericsson supportant GPRS, muni d'un port de communication infrarouge compatible avec la norme IrDA.

Il y a deux sortes de liaisons qui vont être établies : D'une part entre le Ipac et le téléphone mobile par liaison IrDA et d'autre part entre le téléphone mobile et le serveur VoGPRS.

Le choix du langage de programmation s'est porté sur eMbedded Visual Basic 3.0, notamment à cause de ses fonctions évoluées permettant de se connecter à des bases de données et de ses nombreux modules déstinés au monde Internet.

La gestion des messages se fait à l'aide d'une base de données. Cette solution présente l'avantage de résoudre un bon nombre de problèmes liés aux accès concurrents, verrous mortels et autres, qui se rencontrent dans les applications client-serveur. Il est à noter ici que la seule resource qui est accédée de manière concurrente est la base de donnée. Toutes les opérations de gestion de messages se résument à des commandes du langage SQL.

2.2 Structure de la base de données

La base de données doit être capable de gérer tous les utilisateurs ainsi que tous les messages échangés entre eux. Il y a deux entités à considérer, d'une part les utilisateurs et d'autre part les messages. L'association entre les deux est définie par le nom « communication ». Il faut cependant remarquer qu'un utilisateur peut être soit expéditeur d'un message, soit destinataire.

La façon la plus évidente d'étiquetter un utilisateur est d'utiliser son numéro MSISDN, *Mobile Station ISDN Number*, qui est son numéro de téléphone (e.g. +41 79 567 98 35), unique dans le monde. Un autre attribut a été ajouté : le mot de passe «Password ». Ainsi on s'assure bien que l'utilisateur utilisant un numéro MSISDN est bien le propriétaire du numéro. Il faut noter que ce choix est naturel car c'est la seule entité qui permette de faire usage des SMS sans avoir à utiliser les resources des bases de données du réseau GPRS (HLR, *Home Location Register*).

Les attributs pour les messages sont les suivants :

- Nom du message (contenu)
- Taille du message
- Date d'arrivée sur le serveur
- Numéro de message

L'association «communication » doit contenir l'expéditeur, le destinataire et le numéro du message échangé entre les deux utilisateurs. Un utilisateur a la possibilité de transférer un message qu'il reçoit vers un autre destinataire. Il faut donc définir un nouvel attribut, la date d'envoi qui est déterminée par le serveur. Il y a encore deux autres attributs à définir : 1. Indication si un SMS a été envoyé au destinataire. 2. Indication pour savoir si le message a été consulté par le destinataire.

Graphiquement les différentes entités peuvent être représentées de la manière suivante (Figure 1) :

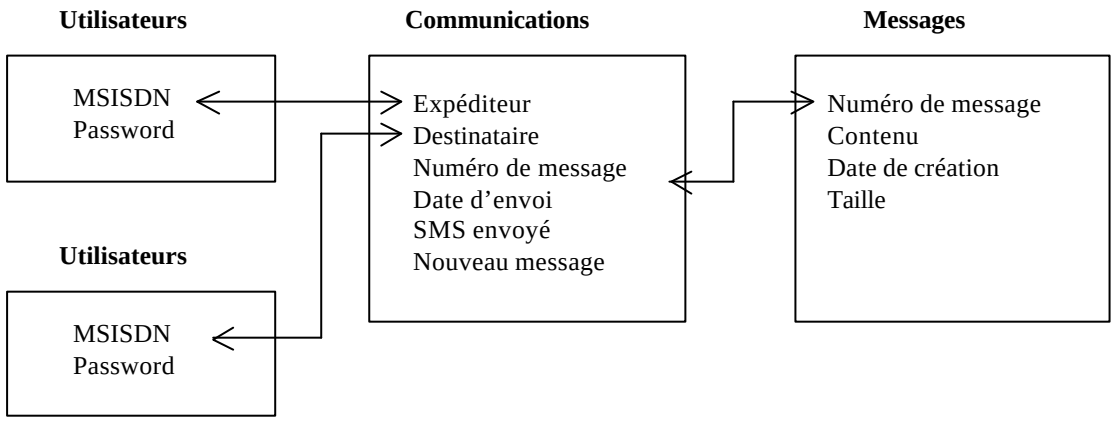

Figure 1. Schéma de la base de données

Le nombre d'attributs a été limité au strict nécessaire pour éviter d'alourdir les procédures de mise à jour des données (effacements, modifications, insertions).

La base de données utilisée est une base de données Access fonctionnant sur Windows NT. Il est évident que pour une utilisation professionnelle et intensive du service il faudrait passer à une base de données du type Oracle ou Sybase. Néanmoins, pour le développement de prototypes, la base de donnée utilisée est satisfaisante. Toutes les opérations s'effectuant sur la base de données sont standards, de type SQL. Une migration éventuelle vers un autre type de bases de données en est ainsi grandement facilité. Le langage Java par exemple offre comme passerelle l'interface *JDBC Java Database Connectivity* qui permet de faire le lien entre des applications développées en Java et différents types de bases de données.

2.4 Accès à la base de données

La mise à jour de la base de données (insertion ou effacement de messages) peut être faite soit par le client, soit par l'application fonctionnant sur le serveur.

La première méthode consiste à faire les mises à jour en envoyant des requêtes SQL depuis le client (Ipac). Le système d'exploitation WinCE fonctionnant sur Ipac permet l'utilisation de la base de données « Microsoft SQL Server 2000 WinCE » et fournit les outils nécessaires pour se connecter à des bases de données distantes. La base de données Access va donc être accédée par l'application du client à travers une passerelle et le réseau GPRS. A cause d'un certain nombre de problèmes, notamment à cause de la sécurité qui ne peut être garantie, cette solution est à proscrire [3].

La deuxième solution consiste à faire toutes les mises à jour à partir du serveur. Cette solution offre de bons critères de sécurité :

1. Une seule application a un accès sans restrictions à la base de données.
2. Toutes les opérations qu'on peut effectuer depuis l'application sont programmées. Il est donc possible de restreindre le nombre des opérations au minimum (insertion et effacement de message uniquement).

D'autre part, une tentative de piratage se trouve plus difficile à effectuer si l'accès à la base de données n'est autorisé que par une application fonctionnant sur le serveur. En plus, le choix des bases de données est relativement important (WinCE n'a qu'un choix très limité).

2.5 Le serveur

La principale fonction du serveur est de mettre à jour la base de données en fonction des requêtes reçues. Lors de la réception d'une demande d'envoi de message, le serveur doit envoyer un SMS permettant d'avertir le destinataire qu'il a reçu un nouveau message. Le choix du langage de programmation s'est porté sur Java, essentiellement pour les mêmes raisons que celles citées pour le choix du langage de l'application client, à savoir la portabilité et la rapidité. Java, en plus d'être portable, offre de nombreuses fonctionnalités (« packages ») lui permettant de se connecter à des bases de données, de communiquer à travers le réseau IP, de définir des interfaces graphiques.

Quand un utilisateur désire modifier l'état de la base de données (envoi ou effacement de message), il va envoyer un fichier de commande indiquant au serveur les opérations à effectuer. Trois fichiers de commande ont été définis : *Delete*, *Insert* et *Forward*. Ces fichiers de commande sont des fichiers texte qui portent le nom de FileCmdMSISDN.cmd. Le fait d'insérer le numéro MSISDN permet de différencier les fichiers de commande entre eux.

L'envoi des fichiers entre le client et le serveur équivaut à envoyer des fichiers entre deux ordinateurs reliés par un réseau IP. Le choix s'est porté sur FTP, *File Transfert Protocol*, qui, contrairement à HTTP, *Hyper Text Transfer Protocol*, ne demande pas de gérer les tempons (*buffers*) pour savoir si un fichier a été complétement transféré ou non. Le protocole FTP exige que les communications soient générées par le client. Il faut donc que l'Ipac initie la communication. L'application fonctionnant sur les clients ne fait usage que de deux fonctions :

- Mettre un message sur le serveur : FTPPutFile
- Récupérer un message du serveur : FTPGetFile

2.6 La gestion des fichiers

Le serveur FTP reçoit deux types de fichiers, les fichiers de commande et les messages vocaux. La différenciation est faite au niveau du nom. Le nom des fichiers de commande est défini (FileCmdMSISDN.cmd) alors que le nom des messages vocaux ne l'est pas puisque c'est l'utilisateur qui va le choisir. Cependant les messages vocaux se différencient des fichiers de commande par leur extension : *.wav. Les messages vocaux générés par l'Ipac (Voicie Recorder) ont cette extension par défault. Il n'est cependant pas impossible que deux messages vocaux portent le même nom, ce qui conduirait à une confusion. Il faut donc s'assurer que les messages émis par deux clients différents portent des noms différents. Dès que le message arrive sur le serveur, on ajoutera le numéro de téléphone de celui qui l'a généré dans son nom. Ainsi un message portant le nom « enr1.wav » sera renommé «MSISDNenr1.wav » dès qu'il arrive sur le serveur. Notons ici que cette solution est extensible à d'autres types de fichiers (vidéo, texte,…).

Les fichiers de commande arrivent dans le répertoire où a été installé le serveur FTP. Dès qu'un tel fichier (*.cmd) est reçu, il doit être traité immédiatement. Deux « threads » ont été écrits pour surveiller le répertoire dans lequel arrivent les fichiers de commande (D:\VoGprs\Commande). Le premier surveille le répertoire et avertit le second thread dès qu'un fichier arrive. Le second thread a pour tâche de traiter les données du fichier de commande. Si plusieurs fichiers de commande doivent être traités, la discipline de FIFO, *First In First Out*, est appliquée.

2.7 Avertir les utilisateurs

Il y a plusieurs façon de procéder pour avertir un destinataire qu'il a reçu un message : e-mail, SMS,… La solution de l'e-mail est trop restrictive car tous les téléphones portables ne peuvent pas nécessairement lire des e-mails. Par contre, tous les téléphones portables ont la possibilité d'envoyer et recevoir des SMS. GSM prend en charge l'itinérance, ce qui signifie que le client pourra être atteint même s'il se trouve momentanément sur un autre réseau que celui qui l'héberge. D'autre part, les SMS sont très populaires en téléphonie mobile. Il a été choisi d'avertir l'utilisateur avec un SMS sous GSM et non sous GPRS, notamment à cause de la fiabilité et la disponibilité de ce service.

L'envoi d'un SMS peut s'effectuer de plusieurs manières, soit par email, soit à l'aide du téléphone mobile. La solution par messagerie électronique n'est pas aisée à mettre en œuvre, surtout pour des raisons de sécurité [3]. L'autre solution, par SMS, consiste à relier un téléphone GSM (nous avons utilisé un Ericsson R520m à cet effet) qu'on connecte à un port de communication du serveur à l'aide d'un câble. Le téléphone mobile doit supporter les commandes AT et pouvoir être utilisé en mode modem. La commande qui permet d'envoyer des messages SMS est « AT+CMGS » dans notre cas (il suffit de consulter le mode d'emploi du téléphone portable). Il suffit donc d'envoyer cette commande avec les paramètres indiquant le contenu du SMS, le destinataire sur le port série et le SMS sera envoyé.

Il existe deux types de SMS : 1. Texte, 2. PDU, *Protocol Data Unit*. Le premier type permet d'envoyer des SMS sans codage alors que le second en utilise un. Le téléphone mobile utilisé ne permet que l'envoi du second type de SMS. Les formats des trames PDU sont différents suivant que le SMS est envoyé ou reçu. Il s'est donc agit dans un premier temps de décoder les messages des trames PDU. Ensuite un petit programme a été écrit pour envoyer des messages SMS du type « You have got a new VoGPRS mail ».

2.8 Configuration du service

Pour pouvoir utiliser l'application, il faut que le client s'identifie. Comme déjà écrit plus haut, la manière la plus simple est d'utiliser le numéro de téléphone MSISDN qui est unique. Cette identification est obligatoire car il faut s'assurer que le client qui utilise l'application le fasse en déclinant son identité et non celle d'un autre.

La configuration du service sur Ipac doit idéalement être effectuée par le réseau GPRS et non par l'utilisateur. Une solution consiste à effectuer les opérations suivantes :

- Le client envoie un SMS contenant le message : « Configure 1234 » (SMS de configuration avec un mot de passe, 1234).
- Le serveur récupère le message et le numéro de téléphone de celui qui a envoyé le message.
- Le serveur crée un fichier portant le nom MSISDNPassword.txt. Ce fichier contient le mot de passe dans le répertoire « Configuration » sur le serveur FTP.
- Quand le SMS est envoyé, l'utilisateur entre son numéro de téléphone et son mot de passe
- L'utilisateur effectue une requête FTP dans le but de récupérer son fichier de configuration, le télécharge. Le Ipac va se configurer automatiquement.
- Il faut que l'Ipac soit configuré pour pouvoir envoyer des messages et en recevoir.

Le décodage d'une trame SMS-PDU n'est pas immédiat. La réception se fait à l'aide de commandes AT. Voici un exemple de trame :

07911497949900F0040B911497160973F0000010117201801000 0EC3A7D3983C56A545100C068301

Nous reconnaissons le numéro de téléphone de SMSC après décodage (1497949900F0=41794999000), le numéro de téléphone de l'expéditeur après décodage (1497160973F0=41796190370). Le message contenu dans le SMS, codé sur 7 bits est : C3A7D3983C56A545100C068301. Il a donc fallu réaliser un programme pour récupérer le numéro de téléphone de l'utilisateur ainsi que les données qu'il a envoyé.

4. Conclusion

La réalisation de telles applications permet certainement à la téléphonie mobile de s'ouvrir à de nouveaux espaces, bénéfiques à son développement. Au moment de la mise sur pied de ce projet, il n'y avait pas encore des appareils regroupant les fonctionnalités d'un PDA et d'un téléphone mobile GPRS ensemble. En disposant de tels appareils nous pourions donc imaginer beaucoup d'applications dérivées avec un confort d'utilisation nettement amélioré.

Références

[1] Jian Cai, David J. Goodman, "General Packet Radio Service in GSM", IEEE Communication Magazine, October 1997.

[2] Xavier Lagrange, Philippe Godlewski, Sami Tabbane, « Réseaux GSM », 4ème ed., Hermès Science publications, 1999.

[3] Pasquale de Frino, « Voix sur GPRS », Travail de diplôme EIVD 2001. Prof : S. Robert.

[4] Nick Grattan, Marshall Brain, «Windows CE 3.0, Application Programming », Microsoft Technologies Series, 2000.