Remerciements

Tout d’abord je tiens à remercier Allah le tout puissant et miséricordieux de m’avoir donner la force et le courage pour terminer ce modeste travail.

Mes premières pensées vont à ma famille, tout particulièrement à mes chers parents à qui je dois tout ce succès et pour qui le sens du sacrifice au cours de ces longues années d'études a été le plus précieux de leurs cadeaux. Grace à leur soutien et aide ainsi que leurs encouragements subliment dosés, ils m'ont permis de vivre mon cursus d'études avec une extrême volonté et efficacité. Grand merci

J'adresse ensuite une profonde gratitude et un grand remerciement à mon encadreur Dr BENADDA Belkacem pour sa patience et son aide significatif au long de ma carrière universitaire en générale et spécialement pendant la réalisation de ce travail et que sans cette aide, ce travail n'aurais jamais connu de début ni de fin.

J'exprime aussi mon respect et remerciement pour monsieur BELDJILALI Bilal de m'avoir aidé à terminer ce travail et que sa position de Co-encadreur m'a permis de vivre une très grande expérience sur le plan pratique et théorique.

J'exprime aussi ma profonde reconnaissance à Dr Bahri Sidi Mohammed et Dr Djennas Sidahmed entant qu'enseignants qui nous ont permis d'évoluer sur un plan scientifique et aussi d'avoir eu l'amabilité de présider et examiner ce travail.

Mon profond remerciement je l'adresse à tous mes amis et collègues qui ont contribué de près ou de loin à ce travail spécialement ceux qui m'ont aidé sur le plan pratique ainsi que la révision et correction de ce mémoire.

Mes remerciements les plus distinguées je les adresse aussi à toute personnes ayant fait de moi ce que je suis et cela est destiné pour mes enseignants ainsi que certains amis au sein de Microsoft Algérie qui m'ont beaucoup appris et ont contribué à l'enrichissement de mes connaissances pédagogiques ou informatique et de m'avoir toujours encouragé à s'améliorer et à apprendre.

Préface

Les technologies de l'information et de la communication ne cessent d'influencer d'avantage la vie quotidienne : téléphonie mobile en Visio conférence, internet mobile, ebook, commerce électronique, payement par terminal mobile pour citer quelques exemples. Les entreprises actuellement pensent sérieusement au budget allouer aux infrastructures et services de télécommunications. L'idée de combiner avantageusement le réseau intranet d'une entreprise au réseau téléphonique ou mobile peut aboutir à des gains important en performances et coûts des communications. Une combinaison qui ne peut être réalisée sans l'utilisation d'une passerelle ou Gateway GSM.

Les Gateway GSM apportent des avantages multiples elles permettent particulièrement d'acheminer des messages court standard ou multimédia par le biais de simple portails web, d'assurer une meilleures qualité de communication avec des faibles coûts, profiter des promotions offertes par les opérateurs de la téléphonie. Toutefois, ces équipements restent relativement coûteux en investissement initial. Une chose qui n'est pas appréciée par les petites et moyennes entreprises Algériennes. Il faut noter que l'utilisation des Gateway GSM est quasiment inexistante actuellement en Algérie.

Dans ce travail on propose de réaliser un Gateway GSM faible coût simple d'utilisation et adaptée au contexte Algérien. Cette Gateway est un système embarqué réalisé autour d'une Raspberry PI combiné avec un puissant software qu'on a développé localement avec la technologie .NET. L'organisation du travail est structurée en cinq chapitres, le premier détail les principes, protocoles et utilisations associés avec différents type de Gateway GSM. Un second chapitre consacré à une conception globale est déploiement de la solution sur une carte Raspberry PI. Le troisième chapitre est consacré à une description des outils software de la technologie .NET, une technologie qu'on a utilisé pour développer la solution software de la Gateway proposée une conception qui serra détaillée dans le quatrième chapitre. Le cinquième chapitre est consacré à l'aspect sécurité, en effet, vu les incidences économiques, il est important d'assurer une confidentialité des données transmise vers la Gateway.

I) Chapitre 1 : Etat de l’art du projet

Introduction

Les opérateurs téléphoniques et leurs offres se multiplient, surtout quand on voit récemment, la grande firme Google s'est lancée dans le marché du GSM en annonçant la création de l'opérateur téléphonique Google, cela donne une image commerciale de la grandeur de ce marché ainsi que de la grandeur de la demande de la part des consommateurs.

En se lançant dans une analyse, les plus grands consommateurs sont généralement les entreprises peu importe leurs tailles. Maintenant avec la révolution des TIC, toutes ces entreprises consomment d'une façon non modérée les différents services de la téléphonie mobile et tout ce qui va avec. En plus de ça, l'usage des réseaux informatiques est aussi intense que même dans une supérette ou une petite administration on trouve au moins deux ordinateurs dont la liaison à internet est Quasi-impérative. Pour ces raisons les spécialistes de la télécommunication ont pensé à un concept qui permet de basculer entre un réseau informatique et un réseau GSM. Ce Moyen est la Gateway GSM.

2) Les Gateways GSM sur le marché

Maintenant que l'usage des Gateways GSM s'est démocratisé, la Gateway la plus simple est commercialisée sous forme d'un Dongle USB avec carte SIM pour assurer un accès internet. Les grandes entreprises de fabrication d'équipements électroniques comme Cisco, DLINK et ALPHATECH se sont lancées dans cette industrie qui reste très lucrative, et les modèles proposés sont de différentes tailles et différentes capacités, et donc le choix de l'équipement est déterminé par le besoin mais en contrepartie cela veut dire que ce genre de système reste limité du point de vue extensibilité.

Les utilisateurs des Gateways utilisent ces dernières de différentes manières mais l'usage reste toujours lié d'une façon ou d'une autre à un aspect financier. L'exemple le plus simple est une petite entreprise dont le besoin de communication est modéré. Selon une petite enquête qu'on a menée avant de commencer la conception de cette Gateway, les factures liées aux services téléphoniques constituent en moyenne 10% de la totalité des charges. Donc cela reste quand même significatif et peut s'alourdir si la PME passe par des périodes difficiles.

L'usage de la Gateway peut réduire ces factures jusqu'à 80%, en terme de pourcentage globale cela implique la réduction de cette charge jusqu'à un seuil pouvant aller jusqu'à 2%.

En prenant toujours l'échantillon d'une PME en général les services sont subdivisés en 3 parties, la plus grande partie est réservée aux messages courts (SMS) avec un pourcentage approximatif de 40%, puis 30%est réservé aux appels téléphoniques vocaux, ensuite 20% pour les USSD et 10% pour les services liés aux réseaux étendus comme les réseaux de 3ème génération (3G).

Cette analyse varie considérablement d'une entreprise à une autre et ces résultats restent approximatifs Par exemple un point qui s'impose dans cette brève analyse est l'usage faible de réseau 3G alors que cela est illogique d'un point de vue global mais techniquement parlant, cela est juste car les réseaux de 3ème génération sont très récents en Algérie et les opérateurs téléphoniques ne couvrent pas la totalité du territoire national. Donc la balance offre / demande reste limitée par un manque technique qui changera forcément par le temps.

Pour que l'image de la consommation des services téléphonique s'éclaircisse, voici une représentation graphique illustrant cette situation :

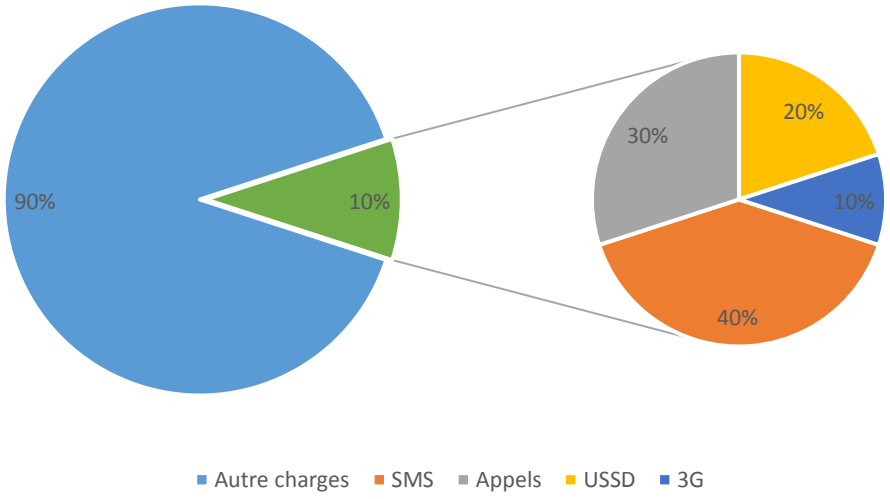

Figure 1 L'usage des services téléphonique au sein d'une PME exemplaire

On remarque dans le schéma précédent la répartition de l'usage des services téléphoniques et que l'usage des SMS et appels téléphonique constitue la partie majeure des charges liées aux services téléphoniques.

En Algérie les entreprises quel que soit leurs taille cherchent à éliminer les charges le plus possible mais pensent rarement à la réduction de la partie « services téléphoniques », par manque d'outils,

bien que les trois opérateurs téléphoniques ne cessent de mettre sur les marchés des offres attrayantes ainsi que des facilités et pour réduire les dépenses en communications.

Si on cherche profondément quel est le meilleur moyen pour payer le moins cher possible des sms Ou des appels téléphoniques cela nous mène forcément vers une étude de prix dans le marché local. Mais en gros la tarification des services reste toujours favorable dans un même réseau et les prix augmentent si le correspondant est dans un réseau autre que le réseau source (le réseau dans lequel nous sommes abonnées) cela est logique et c'est pareil dans tous les pays du monde qui possèdent plus qu'un seul opérateur téléphonique.

Par exemple si on veut faire passer un SMS de l'opérateur 1 vers un opérateur 2 le tarif est plus cher que de faire passer un SMS d'un abonné de l'opérateur 1 vers l'opérateur 1. C'est logique car l'opérateur ajoute le tarif de passage entre opérateurs qui sera détaillé plus techniquement dans les parties suivantes de ce mémoire.

3) Usage des Gateways GSM

Les firmes assurant l'installation des passerelles GSM ont été Fondées il y a plus de 20 ans, la norme GSM est la norme la plus populaire pour les téléphones mobiles dans le monde. Il y a plus de trois milliards de connexions mondiales vers les réseaux GSM, permettant aux abonnés d'utiliser leurs téléphones dans le monde entier. De nouvelles connexions sont ajoutées au taux de 15 ajouts par seconde par les 700 opérateurs mobiles ainsi que dans 218 pays et territoires à travers le monde. L'organisation GSM, qui est responsable de plus de 86% des connexions mobiles mondiaux informe que «la croissance des communications mobiles ne cesse d'augmenter ».

Cela implique qu'on ne peut pas sous-estimer l'importance des téléphones mobiles comme un pilier dans le monde de la communication.

Les téléphones mobiles, loin d'être un luxe coûteux, sont devenus une nécessité abordable. Dans les pays à travers le monde le nombre de numéros mobiles dépasse maintenant celle des numéros de lignes fixes. La culture mobile mondiale qui a émergé a été un facteur clé derrière l'installation de plus de 12 000 passerelles GSM à travers le Royaume-Uni et près de 50 000 aux USA.

Mais dans les pays en voie de développement comme l'Algérie aucune statistique n'est fournie pour 2 raisons, la 1ère est le manque de statistiques dans le secteur des IT et la 2ème est l'absence de l'usage des Gateways GSM dans le secteur de la téléphonie mobile.

En raison de la croissance de l'utilisation des communications mobiles, une proportion de plus en plus élevé d'appels sont faits pour mobiles, par opposition aux lignes fixes.

4) Avantages d'une passerelle GSM

En plus de la réduction des coûts des appels mobiles de 40-80%, les passerelles GSM ont plusieurs avantages :

4.a - Taux d'appel moins cher

Les Gateways GSM sont des commutateurs sophistiqués utilisés pour assurer que chaque appel est acheminé dans le circuit le moins cher possible.

4.b - Appel optimal de la qualité sonore

Les antennes assurent un circuit qui permet de recevoir un signal optimal de façon qu'aucun compromis ne dégrade la qualité de l'appel. En outre, les passerelles GSM disposent d'une qualité supérieure pour assurer la clarté de l'appel.

4.c - Les appels internationaux vers plusieurs pays, sans frais supplémentaires

Certains opérateurs comprennent plusieurs pays internationaux sur leur plan d'appel. Cela signifie que les mobiles dans des endroits tels que l'Union européenne, l'Amérique et l'Asie peuvent être appelés sans coût exorbitants.

4.d - Fonctionner sur les taux de communications alloués

En utilisant la possibilité de gestion dynamique des cartes SIM, vous pouvez définir quand et comment les cartes SIM sont utilisées. Surtout, cela signifie que vous pouvez éteindre automatiquement les cartes SIM qui ont utilisé la totalité de leurs minutes incluses. Cela garantit que chaque SIM est utilisé à sa capacité maximale et qu'aucun frais supplémentaire n'est fait.

4.e - Convient pour les petites, moyennes et grandes organisations d'entreprise

Avec une capacité de gérer simultanément un nombre quelconque d'appels, les réseaux GSM sont en mesure de répondre aux demandes des organisations avec plusieurs numéros d'appels entrants et sortants.

4.f - Seule une passerelle GSM requise

Les opérateurs dans de multiples endroits ou des pays peuvent profiter de la centralisation des Gateways GSM.

La centralisation signifie qu'une seule passerelle placée à un endroit donné est nécessaire. Tous les appels vers les mobiles seront ensuite acheminés sur le réseau interne ou externe à partir de cet emplacement. La centralisation des Gateways GSM en utilisant une seule passerelle coûte moins chère que l'utilisation de deux ou plusieurs passerelles.. Elle élimine également la nécessité d'investir dans plus d'une passerelle, éliminant les dépenses supplémentaires sur l'équipement.

4.g - Éliminer les risques de défaillance de la ligne

Les réseaux GSM offrent des lignes à sécurité intégrée pour assurer la continuité de l'entreprise. Cela veut dire que si un appel échoue, les appels peuvent encore être reportés.

4.h - Compatibilité avec la plupart des types de lignes téléphoniques

Les passerelles GSM commerciales ont le Q931 et QSIG comme protocole leur permettant d'être installées dans la plupart des organisations et d'être compatible avec plusieurs technologies.

4.i - Une performance approuvée

Les Gateways GSM présentes sur le marché sont des appareils très performants et leurs taux d'usage dans les pays occidentaux confirment leurs utilités et qualités. Par exemple il y a eu plus de 12 000 installations de passerelles GSM à travers le Royaume-Uni. Ce nombre ne cesse d'augmenter.

5) Installation d'une passerelle GSM

Avant l'installation d'une passerelle GSM, il est important de prendre en compte :

5.a - L’emplacement

Une étude initiale du site de la passerelle est nécessaire pour déterminer si la structure de l’entreprise est capable de supporter une passerelle sans compromis sur la performance de celle-ci. Particulièrement la qualité du signal et la connexion au réseau local.

5.b - Un prix tout compris sans coûts d'investissements initiaux ou de frais cachés

La solution la plus simple de la passerelle est une location mensuelle unique qui comprend tout, de la gestion des comptes à l'entretien de l'équipement. Cela évite des coûts supplémentaires imprévus survenant à une date ultérieure.

5.c - Identification des appelants

Se rappeler que l'identité de la ligne appelante (CLI) est généralement retenue lorsque les appels sont effectués via une passerelle GSM. Si la CLI est représentée ce sera le numéro de la carte SIM. Sinon, pour un supplément un autre CLI, local (LIN) peut être affiché.

Certains opérateurs de téléphonie mobile n’aiment pas l'utilisation de passerelles GSM, car elle réduit leur chiffre d'affaires et la rentabilité. Votre fournisseur de passerelle devra donc vérifier avant l'installation que vous ne violez pas les termes et conditions.

6) Le fonctionnement du réseau GSM

En travaillant sur la Gateway GSM il est impératif de détailler la partie « Réseau GSM » parce que le rôle des Gateways comme on l’a cité au préalable est de faire circuler l’information entre réseau informatique et réseau GSM.

Le Système mondial de communications mobiles (GSM) est une norme mondialement acceptée pour la communication cellulaire numérique. GSM est le nom d'un groupe de normalisation créée en 1982 pour créer un mobile standard européen commun de téléphone qui serait chargé de formuler des spécifications pour un système mobile cellulaire paneuropéen radio fonctionnant à 900 MHz. Il est estimé que de nombreux pays en dehors de l'Europe vont rejoindre le partenariat du GSM.

Tout au long de l'évolution des télécommunications cellulaires, différents systèmes ont été développés sans le bénéfice de spécifications normalisées. Cela présentait de nombreux problèmes directement liés à la compatibilité, notamment avec le développement de la technologie de radio numérique.

La norme GSM est destinée à résoudre ces problèmes.

De 1982 à 1985 des discussions ont eu lieu pour décider entre la construction d'un système analogique ou numérique. Après de multiples essais sur le terrain, un système numérique a été adopté pour le GSM. La tâche suivante consistait à choisir entre une solution étroite ou à large bande. En mai 1987, la solution division de temps étroite accès multiple (TDMA) a été choisie.

GSM fournit des recommandations, non des exigences. Les spécifications GSM définissent les fonctions et les exigences de l'interface en détail, mais ne traitent pas le matériel. Il s'agit de limiter les designers aussi peu que possible, mais encore pour rendre possible pour les opérateurs d'acheter des équipements de différents fournisseurs. Le réseau GSM est divisé en trois grands systèmes : le système de commutation (SS), le système de station de base (BSS), et le système d'exploitation et de support (OSS).

6.a - Le système de commutation

Le système de commutation (SS) est responsable de l'exécution du traitement d'appel et des fonctions relatives aux abonnés. Le système de commutation comprend les unités fonctionnelles suivantes :

6.b - L'enregistreur de localisation nominal (HLR)

Le HLR est une base de données utilisée pour le stockage et la gestion des abonnements. Le HLR est considéré comme la base de données la plus importante, car elle stocke des données permanentes sur les abonnés, y compris le profil d'un abonné du service, des informations de localisation, et l'état de l'activité. Quand une personne achète un abonnement auprès de l'un des opérateurs de PCS, il ou elle est inscrit dans le HLR de cet opérateur.

6.c - Le centre de commutation de services mobiles (MSC)

Le MSC effectue la fonction de téléphonie du système de commutation. Il contrôle les appels vers et à partir d'autres systèmes téléphoniques et de données. Il effectue également des fonctions telles que frais de billetterie, interfaçage réseau, signalisation de la voie commune, et autres.

Le registre de localisation des visiteurs (VLR) -Le VLR est une base de données qui contient des informations temporaires sur les abonnés qui sont nécessaires pour le MSC afin de desservir les abonnés de visite. Le VLR est toujours intégré avec le MSC.

Quand une station mobile se déplace dans une nouvelle zone MSC, VLR relié à MSC va demander des données au sujet de la station mobile à partir du HLR. Plus tard, si la station mobile effectue un appel, le VLR aura les informations nécessaires à l'établissement de l'appel sans avoir à interroger le HLR à chaque fois.

6.d - Le centre d'authentification (AUC) unité

Communément appelé l'ASC,il fournit l'authentification et le cryptage des paramètres qui permettent de vérifier l'identité de l'utilisateur et d'assurer la confidentialité de chaque appel. L'ASC protège les opérateurs réseau de différents types de fraude trouvés dans le monde cellulaire d'aujourd'hui.

6.e - Le registre d'identité d'équipement (EIR)

L'EIR est une base de données qui contient des informations sur l'identité de l'équipement mobile qui empêche les appels de stations mobiles volées, non autorisées, ou défectueuses.

L'ASC et l'EIR sont mises en œuvre comme des nœuds autonomes ou comme un nœud combiné AUC / EIR.

6.f - Le système de station de base (BSS)

Toutes les fonctions liées à la radio sont effectuées dans le BSS, qui se compose de contrôleurs de station de base (BSC) et des stations de base (BTS).

Le BSC fournit toutes les fonctions de contrôle et des liens physiques entre le MSC et le BTS. Il est un commutateur de grande capacité qui fournit des fonctions telles que le transfert intercellulaire, transfert des données de configuration de cellule, et le contrôle de la fréquence radio (RF) des

niveaux de puissance dans les stations d'émetteur-récepteur de base. Un certain nombre de BSC sont servis par un MSC.

La BTS gère l'interface radio de la station mobile. Le BTS est l'équipement radio (émetteurs-récepteurs et antennes) nécessaires pour desservir chaque cellule dans le réseau. Un groupe de BTS sont commandés par un BSC.

6.g - Le fonctionnement et l'appui du système

L'exploitation et l'entretien du centre (MOC) est connecté à tous les équipements du système de commutation et au BSC. La mise en œuvre du MOC est appelé Système d'exploitation et de Soutient (OSS). L'OSS est l'entité fonctionnelle à partir de laquelle les moniteurs de l'opérateur de réseau contrôlent le système. Le but de l'OSS est d'offrir le soutien rentable client pour les activités opérationnelles et d'entretien centralisés, régionales et locales qui sont nécessaires pour un réseau GSM. Une fonction importante de l'OSS est de fournir une vue d'ensemble du réseau et de soutenir les activités de maintenance des différentes organisations d'exploitation et d'entretien.

6.h - Éléments fonctionnels supplémentaires

D'autres éléments fonctionnels représentés sur la figure 2 sont les suivants :

6.i - Le Centre de messages (MXE)

Le MXE est un nœud qui fournit des services voix, fax et messagerie de données intégrées. Plus précisément, le MXE gère le service de messages courts, la diffusion cellulaire, messagerie vocale, messagerie fax, e-mail, et de notification. Le nœud de service mobile (MSN) -Le MSN est le nœud qui gère les services mobiles de réseau intelligent (IN).

6.j - Le service mobile passerelle de commutation passerelle centre (GMSC)

C'est un nœud utilisé pour interconnecter deux réseaux. La passerelle est souvent mise en œuvre dans un MSC. Le MSC est alors désigné comme le GMSC.

6.k - GSM unité d'interfonctionnement (GIWU)

Le GIWU compose à la fois matériel et logiciel qui fournit une interface à divers réseaux de communication de données. Grâce à la GIWU, les utilisateurs peuvent alterner entre la parole et les données au cours du même appel. Le matériel informatique GIWU est physiquement situé au MSC/ VLR. La meilleure façon pour comprendre le réseau GSM est de le schématiser. Pour cela le schéma suivant représente un réseau GSM :

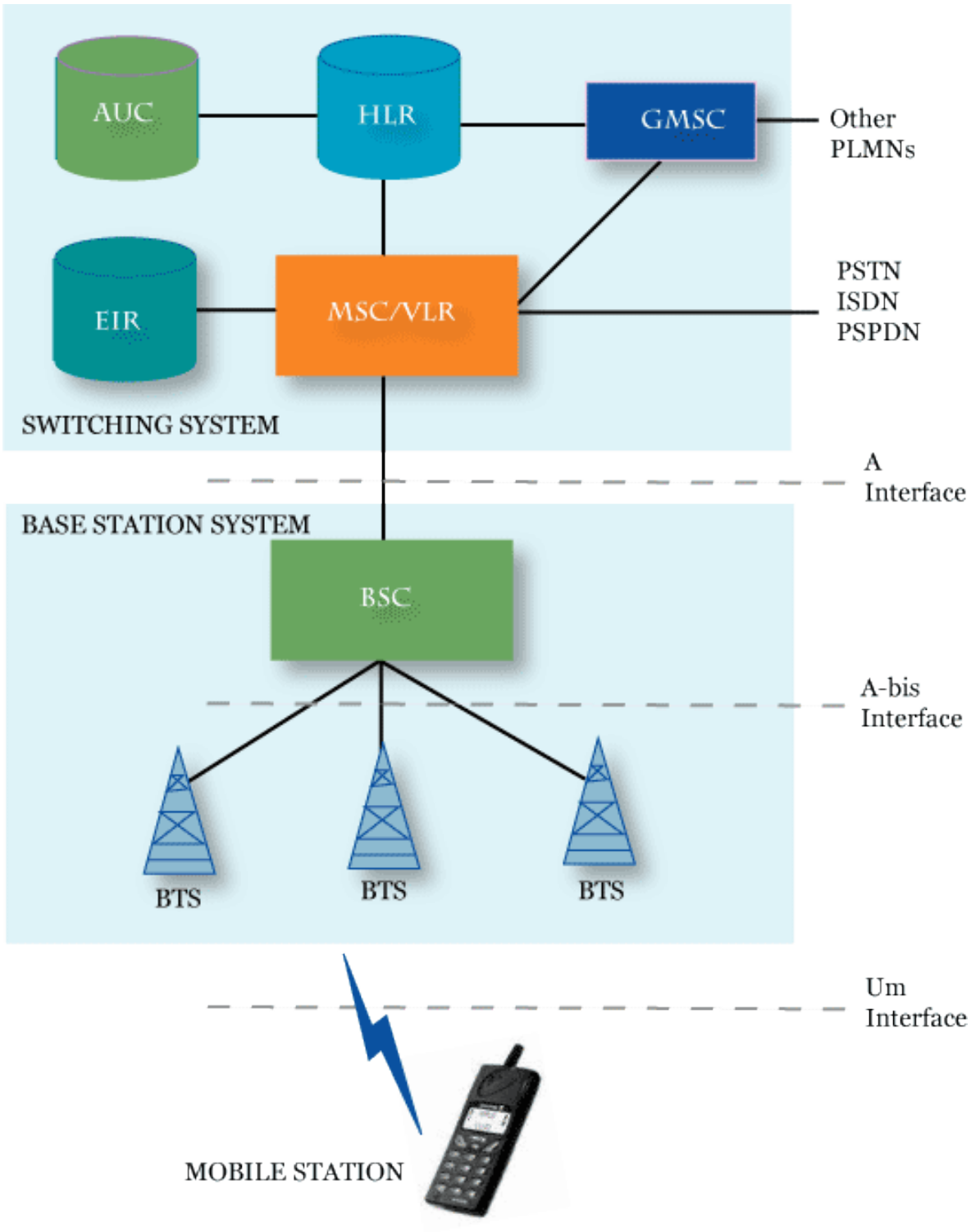

Figure 2 Vue global sur le réseau GSM

7) Exemple d'utilisation des Gateways GSM

Les Gateways GSM sont très utilisés dans plusieurs domaines d'application. De nos jours, avec la démocratisation des smartphones, les applications mobiles de tchat ont connu une révolution et les sociétés qui développent ce genre d'applications pensent à créer l'addiction chez les utilisateurs en voulant les rendre tout le temps connectés même s'ils ne possèdent pas un accès à internet.

Pour ces raisons ils ont pensé à ajouter des fonctionnalités qui laissent l'utilisateur connecté à ce qui se passe par SMS soit par Tchat soit par l'envoie des nouvelles des réseaux sociaux toujours par SMS. Cela lui permet d'interagir (commenter, répondre, ….) Par SMS d'une façon indépendante de la connexion à internet.

On prend par exemple Facebook, Skype et Vibrer, le 1èr le premier réseau social au monde car il contient le nombre le plus élevé d'utilisateurs actifs, et cela grâce à sa politique qui facilite aux utilisateurs inscrits d'interagir soit en mode connecté ou en mode déconnecté. Le réseau envoi des sms à l'utilisateur dès qu'une nouvelle chose se passe dans son réseau proche (quelqu'un publie un contenu, ou commente une publication …) et donc l'utilisateur sera obligé d'interagir instantanément. Pour ce qui concerne Skype et Viber ce sont 2 applications de tchat très connues de nos jours, ces dernières permettent à l'utilisateur de contacter son correspondant en temps réel soit en se connectant à internet et tchatant en mode client-serveur (classique) soit en envoyant des SMS vers le téléphone directement via l'application.

Ces 3 exemples comme d'autres, utilisent tout simplement des Gateways GSM pour l'envoie d'SMS.

Les Gateways sont connectés aux réseaux de ces entreprises et des serveurs dédiés gère l'envoie.

La seule condition pour l'envoie est la possession des numéros de téléphones des utilisateurs, quand à la tarification, les prix sont vraiment bas, par exemple Facebook envoie des SMS gratuitement aux utilisateurs, et si les utilisateurs veulent interagir avec Facebook c'est

uniquement l'opérateur qui tarifiera l'envoie d'SMS, ni plus ni moins. Quand à Skype c'est 9 centime d'euro par message.

Ci-dessous se trouve 2 figurent qui expliquent comment ces réseaux et applications effectuent l'envoie des SMS :

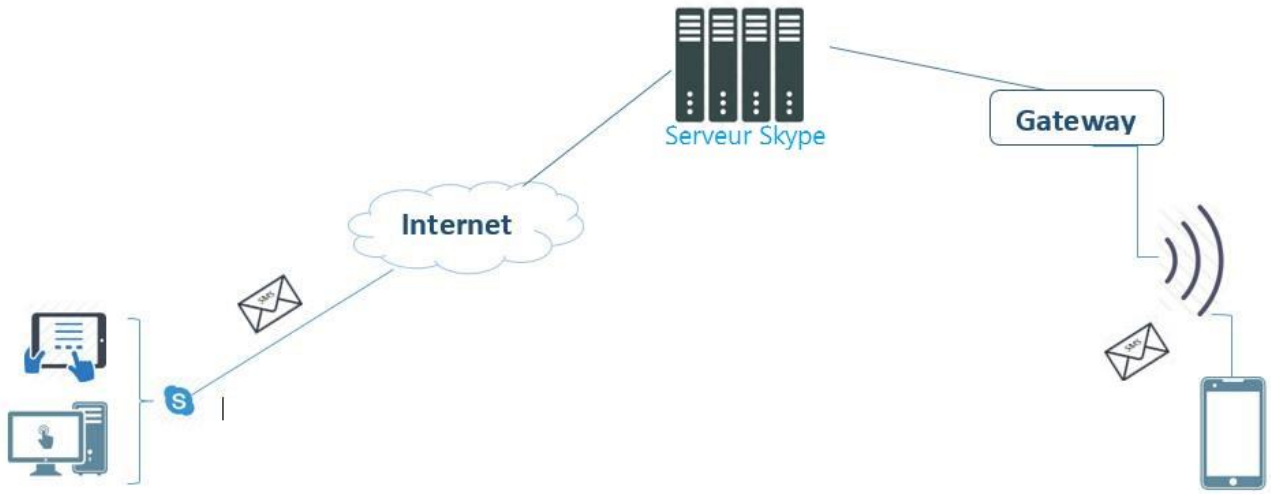

Figure 3 Schéma synoptique d'envoi d'SMS de Skype vers un téléphone mobile.

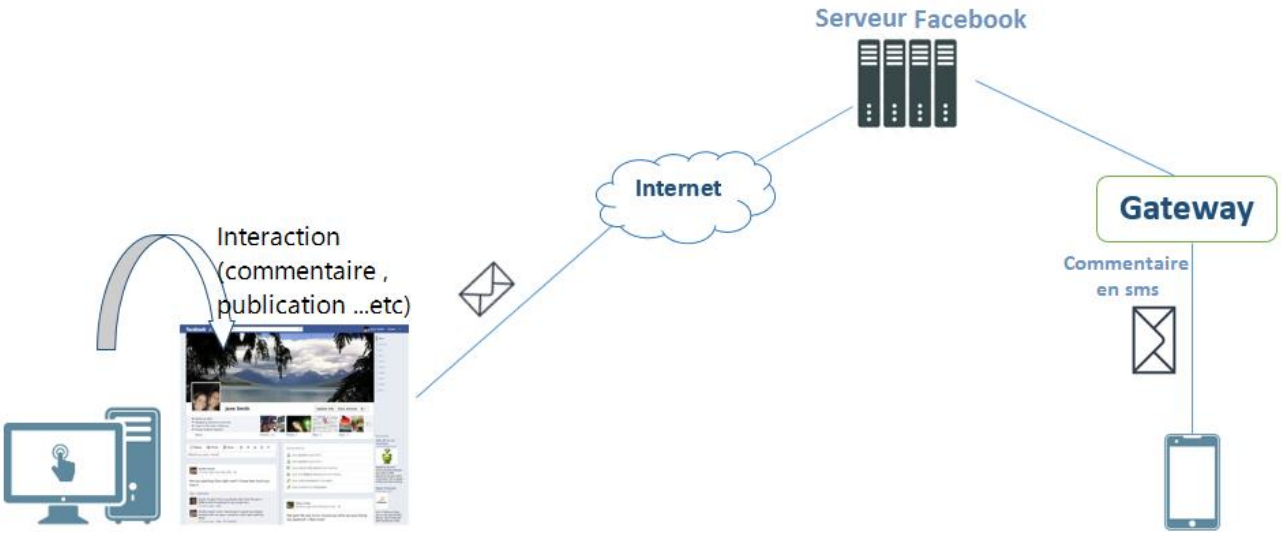

Figure 4 Schéma synoptique de notification SMS du réseau social Facebook.

8) Conclusion

Dans ce chapitre on a vu l'importance et la facilité d'usage des Gateways GSM surtout que le monde du GSM qui ne cesse de se développer jour après jour et que plusieurs projets que nous utilisons dans nos vie quotidiennes sont basées sur les Gateways GSM. Il est donc très important de les étudier en détail et penser réellement à leur production et utilisation.

II) Chapitre 2 : Implémentation du hardware de la Gateway

1) Introduction

La Gateway GSM offre beaucoup de possibilités économiques : nouveaux services en télécommunications, réduction des coûts des infrastructures, développement et déploiement de nouvelles applications et services. Nous nous sommes fixé l'objectif de réaliser une Gateway GSM adapté au contexte Algérien avec un prix de revient abordable. Dans ce chapitre nous allons décrire les composantes matérielles que nous avons sélectionnées : une carte mère Raspberry Pi avec des modules GSM et comment nous les avons regroupé pour constituer notre Gateway GSM.

2) Présentation de la Raspberry pi

La Raspberry pi est un ordinateur de taille d'une vieille K7 audio, qui démarre depuis une carte SD, elle a été créé avec des objectif purement éducatif précisément pour apprendre à développer facilement des solutions embarqués : programmer toute en exploitant des périphériques informatiques à l'image d'un ordinateur ordinaire. Puis ce principe s'est développé et a permis aux développeurs d'innover des systèmes hardware plus avancé, car les soucis des circuits électroniques a été simplifié par cette nouvelle génération de cartes (Raspberry PI, Beagl Bone, Arduino…).

Souvent les spécialistes en programmation sont limités en conception électronique. En effet, les systèmes d'exploitations offrent l'encapsulation software qui leurs permet d'éviter l'étude des circuits complexes.

La Raspberry es dotée d'un ensemble de broches appelées GPIO comme figure dans la figure suivante :

Figure 5 Interface GPIO de la Raspberry pi

Les GPIO sont présentés dans cette partie car la Gateway GSM doit être réalisé à l'aide des modules GSM qui se branchent dans les pins de GPIO. Toutefois, pour des raisons de disponibilité ce genre de modules n'a pas été utilisé dans ce projet. La présentation de cette partie représente une porte d'une future extension de ce Projet.

Les GPIO sont généralement constitués de PIN en sorties et d'autres en entrées numériques (cela dépend de la configuration utilisateur). D'autres PINs fournissent la masse (GND), il y a aussi des pins qui donnent un signal d'horloge en cas de besoin de synchronisation. Et aussi il y a des pins qui fournissent un signal sinusoïdal destiné à l'usage analogique.

Tout cela implique le besoin de numérotation de ports.

Dans la figure suivante les GPIO sont présentés avec leur numérotation :

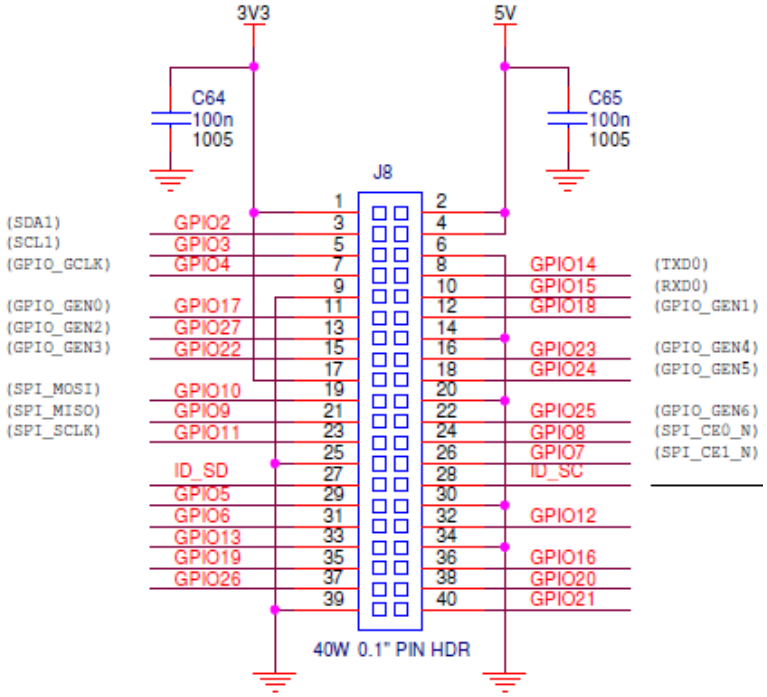

Figure 6 Schéma des pins de GPIO numérotés.

À présent la Raspberry pi peut se connecter à un téléviseur ou à un moniteur à l'aide de la HDMI, ou bien DVI (avec un adaptateur HDMI vers DVI) ou encore un composite (connecteur rond de couleur jaune couramment appelée KINCH).

Le Raspberry Pi de base est dotée d'un processeur ARM de fréquence d'horloge de 700 MHz.

Le processeur ARM est un processeur de faible puissance conçu à la fin des années 80 par une société basée au Royaume-Uni dite Acorn Computers, à l'origine comme un coprocesseur pour la BBC Micro.

Aujourd'hui vous pouvez trouver le processeur ARM dans de nombreux dispositifs, on peut se permettre de citer les appareils d'Apple (iPod, iPhone, et iPad), ou la majorité des appareils Android, smartphones, les routeurs haut débit à domicile et les modems, et même les boxes de télévisions numérique, et maintenant, bien sûr, dans la Raspberry Pi.

Comme moyen d'affichage la Raspberry Pi a une sortie HDMI pour se connecter à un téléviseur ou moniteur, cela peut aussi être utilisé avec une fonctionnalité appelée CEC, qui permet de contrôler l'appareil avec une télécommande de télévision.

Le modèle B Raspberry Pi contient deux ports USB, Le Modèle B+ contient 4 ports USB. Ces ports ne sont adaptés qu'aux appareils de faible puissance. Tous périphérique connecté au port USB qui nécessite la 500mA doit être interfacé par le biais d'un concentrateur avec alimentation externe, également sur le modèle A, il est obligatoire d'utiliser un concentrateur car il dispose d'un seul port USB.

Lors des premiers tests avec les modems GSM on a rencontré un problème d'exploitation. Particulièrement lors d'e l'usage de multiples modem GSM, Raspberry bloque puis le système redémarre sans message d'erreur ni indication dans le fichier journal du système. Cela été principalement liée à l'alimentation, car l'alimentation de base ne suffisait pas pour alimenter tous les modems GSM ainsi que la carte. Pour cette raison on a utilisé une alimentation externe, cela va être détaillé au cours de la présentation du prototype.

En ce qui concerne la connectivité le modèle B Raspberry Pi possède un port Ethernet 10/100, les LED près de la sortie audio et ports USB indiquent l'état du réseau (sur le modèle B), y compris la vitesse (10 / 100Mbps).

Le Raspberry Pi a également une sortie composite pour l'affichage, Cette affichage est limité à 640 × 480 à fin qu'il soit plus pratique d'utiliser le connecteur HDMI pour le connecter à un écran, sur les modèles A et B cela est un standard jaune. La Raspberry Pi a une prise de sortie audio analogique sur la carte.

Tous ces détails sont illustrés dans la figure suivante :

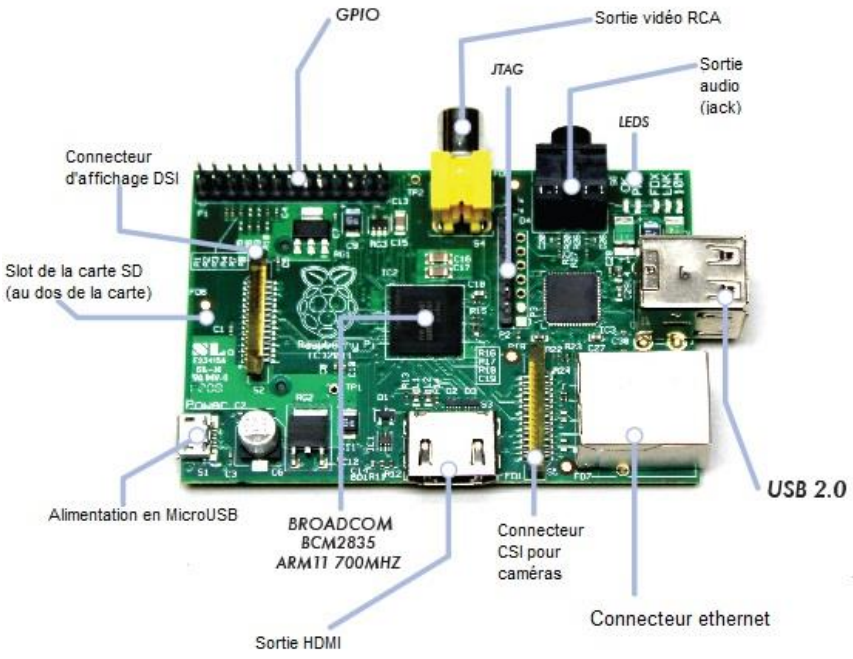

Figure 7 Les différentes interfaces et composants de la Raspberry Pi

La figure ci-dessous présente une illustration du port de carte mémoire de type SD qui n'est pas visible dans le schéma précédent.

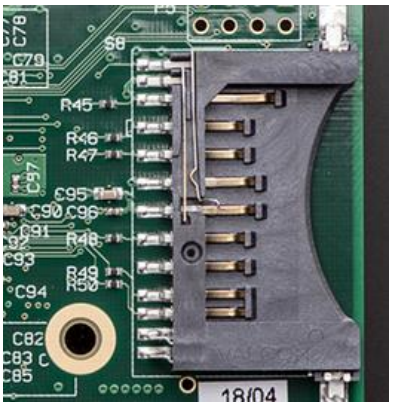

Figure 8 Slot de la carte SD au dos de la Raspberry Pi

3) Les systèmes d'exploitation utilisés

Les systèmes d'exploitation qui fonctionneront sur le Raspberry Pi comprennent la plupart des distributions Linux à base de Debian ou Redhat, ou même RISC OS et Android. Il existe des distributions Android adaptés pour les Raspberry pi car de nombreuses chaines de productions de tablettes numérique ou smart TV sont faites à base de Raspberry pi, cette dernière constitue donc une technologie prometteuse à la fois pour l'industrie ainsi que pour la science et développement.

La Raspberry Pi est dotée de 256 Mo de mémoire vive dans le Modèle A, 512 Mo dans les modèles actuels du modèle B, ou 1 Go pour le modèle B+.

Pour des raisons de disponibilité des cartes on a choisis le modèle B, la taille de la mémoire est plus au moins suffisante pour notre projet.

4) Déploiement du système d'exploitation

Comme tout système embarqué, Raspberry Pi a besoin d'un firmware ou système d'exploitation pour pouvoir démarrer et exploiter le hardware.

Dans notre projet on a besoin d'un système d'exploitation pour pouvoir développer l'ensemble des softwares permettant la GATEWAY GSM de fonctionner, et pour cette raison on va détailler dans cette partie la préparation du système d'exploitation et l'utiliser dans la Raspberry.

Dans notre cas on installera « Raspbian » comme système d'exploitation. Lancé en juin 2012, RASPBIAN version Wheezy est un système d'exploitation très optimisé et riche en terme de packages, il est doté de plus de 35 milles packages disponible dans les repository officiels.

RASBIAN est système d'exploitation gratuit basé sur la distribution Linux DEBIAN et optimisé pour le hardware de la rapsberry pi. L'ensemble des programmes contenus dans Raspbian permettent aux utilisateurs de la Raspberry d'effectuer de multiples taches et manipulations souvent utilisées sur des simples ordinateurs. Ce qui nous intéresse c'est la possibilité de développer sur ce système d'exploitation en utilisant des modems GSM.

L'installation de Raspbian doit s'effectuer sur une carte mémoire SD de classe 10 au minimum pour pouvoir fonctionner correctement.

En premier plan on commence par le téléchargement de Raspbian depuis son site officiel, il est disponible en image d'extension IMG et de taille de 3.05 Go pour préserver l'ensemble de fichier et permettre d'installer un système bootable. Après le téléchargement de l'image de l'OS, on formate la carte SD vierge de taille de 8 Giga et plus de préférence en FAT32 de préférence. Puis on télécharge un logiciel permettant de graver des images sur des disques amovibles, dans notre cas on peut utiliser Linux pour graver cette image sur la carte SD en tapant la commande suivante :

dd bs=512k *if*=2014-09-09-wheezy-*raspbian*.img of=/dev/sdb

bs est la taille des blocks lors de la gravure et **sdb** est le nom du disque lors de la détection dans le système avant de le monter.

Sinon sous Windows on peut utiliser un logiciel recommandé par la plupart des communautés de Raspberry et Raspbian qui s'appelle « Win32 Disk Imager ».

L'interface du logiciel est très simple, on introduit le chemin de l'image de l'OS et le disque amovible et on lance l'écriture en cliquant sur Write. L'opération prend quelques minutes avant qu'elle se termine en affichant une fenêtre indiquant le succès de l'opération.

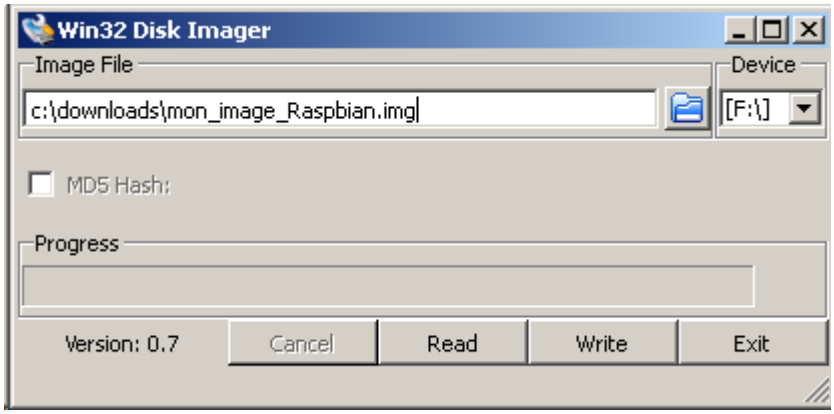

Figure 9 Imprimé d'écran du logiciel de gravure

Après la gravure de Raspbian dans la carte SD on l'insère dans le slot de la carte mémoire et on alimente la Raspberry, un 1er écran apparaitra quelque secondes avant de booter correctement.

Si on branche la Raspberry à un écran via la fiche Kinch ou HDMI un bureau contenant le logo de Raspbian apparaitra avec quelques icones.

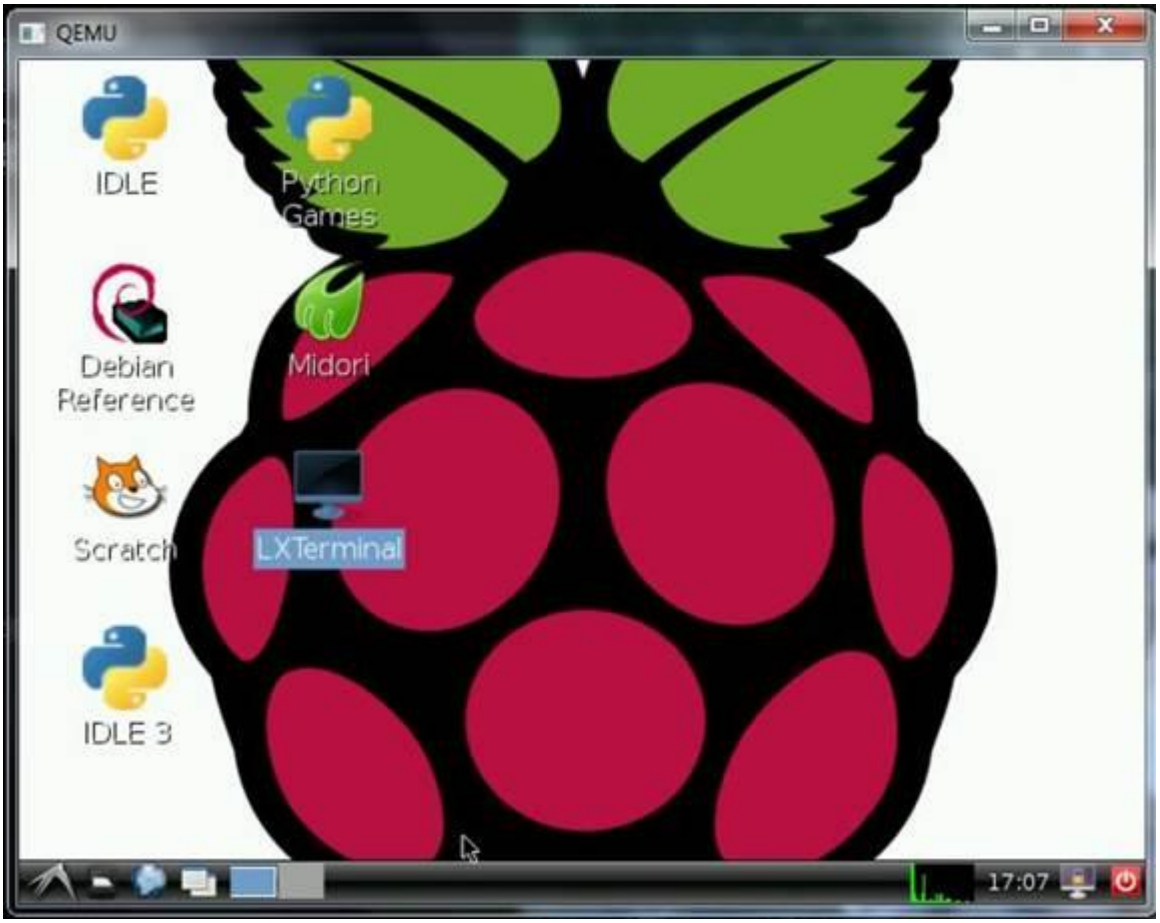

Figure 10 Bureau de Raspbian

Si cela s'affiche ça veut dire que la Raspberry Pi fonctionne correctement. On peut la contrôler via un clavier et une souris sinon un protocole dédié appelé SSH permet un accès à distance, ce protocole sera détaillé dans les chapitres suivants.

5) Installation des modules & modems GSM

Ce projet doit normalement se réaliser à l'aide des modules GSM pour Raspberry Pi qui se branchent dans les GPIO mais faute de disponibilité on a dû utiliser des modems GSM de 3ème génération (Souvent connus sous le nom des Clés 3G) qui se branchent à l'aide d'un connecteur USB. Cela implique l'usage d'un driver spécial pour la communication entre machine et modem. Pour utiliser plusieurs Clés 3G avec des connecteurs USB on peut les brancher dans la Raspberry pi mais puisque cette dernière est alimentée avec un courant de 322mA et un voltage de 5.09V cette alimentation ne suffira pas pour alimenter plusieurs modules USB 3G connectés et plus c'est pour cette raison on doit utiliser un concentrateur USB (HUB USB) doté d'une alimentation externe puis le brancher au port USB de la Raspberry Pi.

Nous allons donner une brève explication sur son mode de fonctionnement, particulièrement côté hardware et software, puisque sans hub USB la passerelle GSM ne fonctionnera pas correctement.

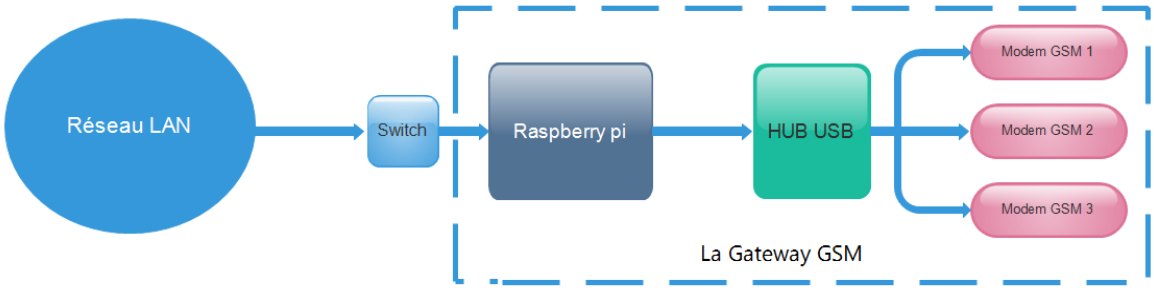

Figure 11 Schéma synoptique de la Gateway GSM

Un modem GSM est un type spécialisé de modem qui accepte une carte SIM, et fonctionne sur un abonnement à un opérateur de téléphonie mobile, comme un téléphone mobile. Du point de vue de l'opérateur mobile, un modem GSM ressemble à un téléphone mobile.

Quand un modem GSM est connecté à un ordinateur, L'ordinateur peut utiliser le modem GSM pour communiquer via le réseau mobile. Bien que ces modems GSM soient le plus souvent utilisés pour fournir une connectivité Internet mobile, beaucoup d'entre eux peuvent également être utilisés pour envoyer et recevoir des messages SMS et MMS voire même des appels téléphoniques.

Un modem GSM peut être un dispositif de modem dédié avec une série, USB ou Bluetooth, ou bien un téléphone mobile qui peut fournir les capacités d'un modem GSM.

Dans ce document, le terme modem GSM est utilisé comme un terme générique pour désigner n'importe quel modem qui prend en charge un ou plusieurs des protocoles dans la famille évolutive GSM, y compris les technologies 2.5G GPRS et EDGE, ainsi que la technologie 3G WCDMA, UMTS, HSDPA et HSUPA.

Un modem GSM expose une interface qui permet à des applications telles que NowSMS d'envoyer et de recevoir des messages par l'interface modem. Les frais des opérateurs mobiles pour l'envoi et la réception sont les mêmes que l'envoie et la réception directement sur un téléphone mobile. Pour effectuer ces tâches, un modem GSM doit soutenir un "jeu de commandes AT étendu" pour envoyer / recevoir des messages SMS, tel que défini dans la norme ETSI GSM 07.05 et 3GPP TS 27.005 et spécifications.

Les modems GSM peuvent être un moyen rapide et efficace de commencer avec SMS, car un abonnement spécial à un fournisseur de service SMS n'est pas nécessaire. Dans la plupart des régions du monde, les modems GSM sont une solution rentable pour la réception de SMS, parce que l'expéditeur paie pour la livraison de message.

Un modem GSM peut être un dispositif de modem dédié avec une série, USB ou une connexion Bluetooth, comme le Falcom Samba 75. (D'autres fabricants de dispositifs de modem GSM dédié comprennent Wavecom, Multitech et iTegno. Nous avons également examiné un certain nombre de modems sur notre blog de soutien technique.) Pour commencer, insérez une carte SIM GSM dans le modem et le brancher à un port USB disponible sur votre ordinateur.

Un modem GSM peut aussi être un téléphone mobile GSM standard avec le pilote du câble et des logiciels appropriés pour se connecter à un port série ou un port USB sur votre ordinateur. N'importe quel téléphone qui prend en charge le «jeu de commandes AT étendu" pour envoyer / recevoir des messages SMS, tel que défini dans ETSI GSM 07.05 et / ou 3GPP TS 27.005, peut être soutenu par les SMS et MMS Gateways. Notez que tous les téléphones mobiles prennent en charge cette interface modem.

En raison de certains problèmes de compatibilité qui peuvent exister avec les téléphones mobiles, il est préférable d'utiliser un modem GSM dédié à un téléphone mobile GSM. Il s'agit d'un problème avec la messagerie MMS, ou si vous voulez être en mesure de recevoir des messages MMS entrants avec la passerelle, l'interface du modem sur la plupart des téléphones GSM ne vous permet d'envoyer des messages MMS.

Il convient également de noter que tous les téléphones prennent en charge l'interface modem pour envoyer et recevoir des messages SMS.

En particulier, la plupart des téléphones intelligents, y compris Blackberry, iPhone et les appareils Windows Mobile, ne supportent pas cette interface modem GSM pour envoyer et recevoir des messages SMS. En outre, les téléphones Nokia qui utilisent l'interface S60 (série 60), qui est basé Symbian, ne prennent en charge l'envoi de messages SMS via l'interface modem, et ne prennent pas en charge la réception de SMS via l'interface.

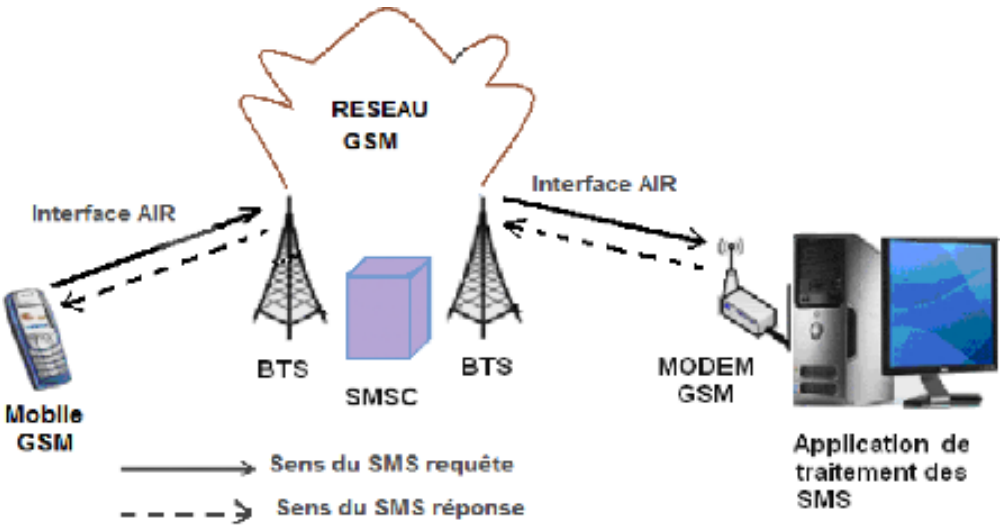

Figure 12 Schéma de fonctionnement des modems GSM dans le réseau

Les clés GSM sont montées dans le Raspbian comme étant des interfaces et qui se présentent sous forme d'un fichier binaire dans le dossier /dev/ car ce dossier permet aux utilisateurs de linux de lister toutes les interfaces liées au hardware.

Et comme nous voulons prendre le contrôle des modems GSM via la ligne de commande linux (raspian), on charge au début le driver embarqué de Windows et l'émule grâce à l'utilitaire usb_*modeswitch* pour le présenter dans le dossier /dev/ comme étant une des interfaces de ligne de commande USB .On les trouve sous le nom de ttyUSBx (x indique le numéro d'ordre de l'interface).

Dans notre cas l'USB monté dans le système donne accès à 3 interfaces USB la 1ère (ttyUSB1) est celle du modem GSM, la 2ème (ttyUSB2) est réservé au petit espace mémoire réservé aux drivers de la clé, quand à la 3ème interface, C'est pour le lecteur de la carte microSD.

Dans l'image suivante se présente une photo de la Gateway GSM.

Figure 13 Slot de la carte SD au dos de la Raspberry Pi

6) Conclusion

Dans ce chapitre on a vu en détail l'implémentation du hardware en utilisant une carte intelligente Raspberry pi ainsi que le hub USB qui nous permet d'utiliser plusieurs modems GSM et l'alimentation nécessaire pour faire fonctionner la Gateway GSM ainsi que les configurations nécessaires pour pouvoir démarrer la Raspberry Pi.

III) Chapitre 3 : L’environnement de travail & d’exécution

1) Introduction

Comme tout projet dans le domaine des systèmes embarqués, le travail se divise au minimum en 2 parties majeures, la 1ère est celle qui prend en compte l'ensemble d'outils, appareils, composants et montages ou réalisations, c'est la partieHardware. Cette dernière a été détaillée dans les chapitres précédents.

La 2ème partie c'est l'ensemble des outils logiciels, langages et environnement de travail ainsi que la logique et les algorithmes permettant au système de fonctionner correctement.

Dans ce chapitre on définira en détail les environnements de travail ainsi que les outils utilisés pour la réalisation de la Gateway GSM.

2) Vue d'ensemble du .NET Framework

Le .NET Framework est une technologie qui prend en charge la création et l'exécution de la nouvelle génération d'applications et de services Web XML. Le .NET Framework est conçu pour remplir les objectifs suivants :

Fournir un environnement cohérent de programmation orientée objet pour que le code objet soit stocké et exécuté localement, exécuté localement mais distribué sur Internet ou exécuté à distance.

Fournir un environnement d'exécution de code qui minimise le déploiement de logiciel et de conflits de version.

Fournir un environnement d'exécution de code qui promeut l'exécution sécurisée de code y compris le code créé par un tiers d'un niveau de confiance moyen ou un tiers inconnu.

Fournir un environnement d'exécution de code qui élimine les problèmes de performance des environnements interprétés ou écrits en scripts.

Fournir au développeur un environnement cohérent entre une grande variété de types d'applications comme les applications Windows et les applications Web.

Générer toutes les communications à partir des normes d'industries pour s'assurer que le code basé sur le .NET Framework puisse s'intégrer à n'importe quel autre code.

Le .NET Framework se compose du Common Language Runtime et de la bibliothèque de classes .NET Framework. Le Common Language Runtime est la base du .Net Framework. Le runtime peut être considéré comme un agent qui manage le code au moment de l'exécution, fournit des services essentiels comme la gestion de la mémoire, la gestion des threads et la communication à distance.

Il applique également une stricte sécurité des types et d'autres formes d'exactitude du code qui promeuvent un code sécurisé et robuste. En fait, le concept de gestion de code est un principe fondamental du runtime. Le code qui cible le runtime porte le nom de code managé, tandis que le code qui ne cible pas le runtime porte le nom de code non managé.

La bibliothèque de classes est une collection complète orientée objet de types réutilisables que vous pouvez utiliser pour développer des applications allant des traditionnelles applications en ligne de commande ou à interface utilisateur graphique jusqu'à des applications qui exploitent les dernières innovations fournies par ASP.NET, comme les services Web XML et Web Forms.

Le .NET Framework peut être intégré par des composants non managés qui chargent le Common Language Runtime dans leurs processus et initient l'exécution du code managé, créant ainsi un environnement logiciel qui peut exploiter à la fois les fonctionnalités managées et non managées. Le .NET Framework fournit non seulement plusieurs hôtes de runtime, mais il prend également en charge le développement d'hôtes de runtime tiers.

L'illustration suivante montre les relations du Common Language Runtime et de la bibliothèque de classes avec vos applications et avec l'ensemble du système. L'illustration montre également comment le code managé opère au sein d'une architecture plus grande.

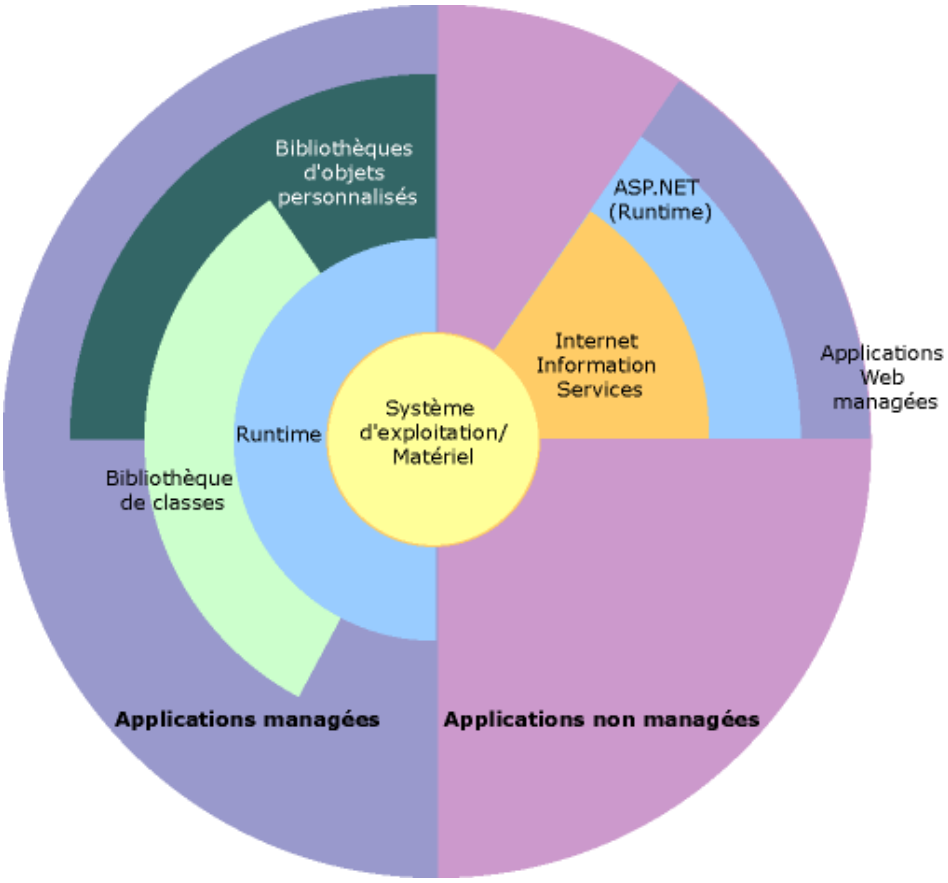

Figure 14 Schéma descriptif sur le .NET Framework

3) Fonctionnalités du Common Language Runtime (CLR)

Le Common Language Runtime gère la mémoire, l'exécution des threads, l'exécution du code, la vérification de la sécurité du code, la compilation et d'autres services du système. Ces fonctionnalités font partie intégrante du code managé qui s'exécute sous le Common Language Runtime.

En ce qui concerne la sécurité, les composants managés se voient attribués divers niveaux de confiance en fonction d'un nombre de facteurs qui comprennent leur origine (comme Internet, un réseau d'entreprise ou un ordinateur local). Cela signifie qu'un composant managé peut ou ne peut pas effectuer des opérations d'accès au fichier, des opérations d'accès au Registre ou d'autres fonctions délicates, même si ce composant est utilisé dans la même application active.

Le runtime fait appliquer la sécurité d'accès du code. Par exemple, les utilisateurs ont confiance dans le fait qu'un fichier exécutable incorporé dans une page Web peut afficher une animation sur l'écran ou chanter une chanson mais ne peut pas accéder à leurs données personnelles, leur système de fichiers ou leur réseau. Les fonctionnalités de sécurité du runtime permettent ainsi à des logiciels légitimes, déployés sur Internet de comporter un grand nombre de fonctionnalités.

Le runtime garantit également un code robuste en implémentant une infrastructure de vérification de code et de type stricte portant le nom de système de type commun (CTS, Common Type System). Le CTS garantit que le tout le code managé soit auto descriptif. Les différents compilateurs de langage Microsoft et tiers génèrent du code managé conforme au système de type commun (CTS, Common Type System). Cela signifie que le code managé peut consommer d'autres instances et types managés, tout en appliquant strictement le respect et la sécurité des types.

En outre, l'environnement managé du runtime élimine un grand nombre de problèmes logiciels courants. Par exemple, le runtime traite automatiquement la disposition des objets et gère les références aux objets, les libérant lorsqu'ils ne sont plus utilisés. Cette gestion automatique de la mémoire résout les deux erreurs d'application les plus courantes, le manque de mémoire et les références mémoires non valides.

Le runtime accélère également la productivité du développeur. Par exemple, les programmeurs peuvent écrire des applications dans le langage de développement qu'ils ont choisi, tout en tirant pleinement parti du runtime, de la bibliothèque de classes, et de composants écrits dans d'autres langages par d'autres développeurs. Tout fournisseur de compilateur qui choisit de cibler le runtime peut en faire de même. Les compilateurs de langage qui ciblent le .NET Framework rendent les fonctionnalités du .NET Framework disponibles au code existant écrit dans ce langage, ce qui simplifie considérablement le processus de migration pour les applications existantes.

Si le runtime est conçu pour les logiciels du futur, il prend également en charge les logiciels d'aujourd'hui et d'hier. L'interopérabilité entre les codes managés et non managés permet aux développeurs de continuer à utiliser des composants COM et des DLL nécessaires.

Le runtime est conçu pour améliorer les performances. Bien que le Common Language Runtime fournisse de nombreux services de runtime standard, le code managé n'est jamais interprété. Une fonctionnalité nommée la compilation juste-à-temps (JIT, Just-In-Time) permet à tout le code managé de s'exécuter dans le langage machine natif du système sur lequel il s'exécute.

De son côté, Le gestionnaire de mémoire élimine les risques de mémoire fragmentée et augmente la localité de référence mémoire afin de décupler les performances.

Enfin, le runtime peut être hébergé par des applications côté serveur hautement performantes, comme Microsoft SQL Server et les services IIS (Internet Information Services). Cette infrastructure vous permet d'utiliser du code managé pour écrire votre logique métier tout en profitant des performances supérieures du meilleur des serveurs d'entreprise prenant en charge l'hébergement runtime.

Donc le .NET reste une technologie de virtualisation qui nous donne la possibilité de créer un contenu facilement transportable. Dans la figure suivante se trouve un schéma synoptique du fonctionnement de .Net .

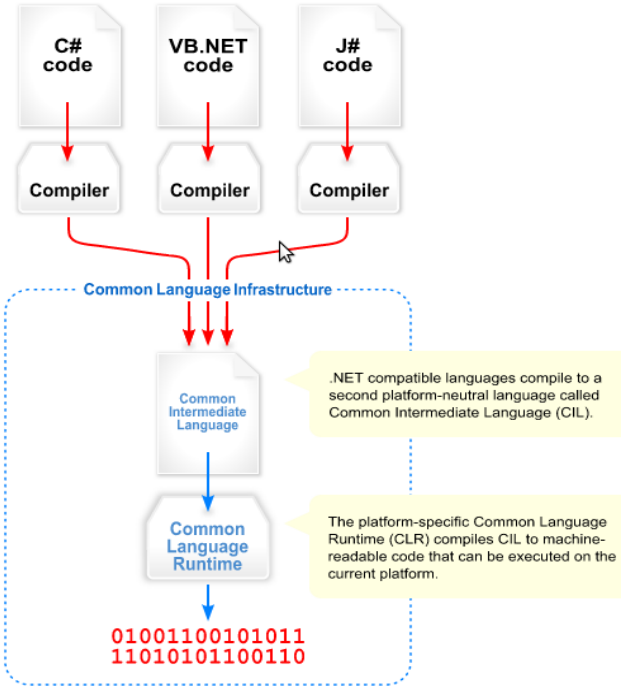

Figure 15 Schéma de fonctionnement d'un programme sous .Net

4) Le projet MONO Le .net et Raspbian

Le .net fonctionne dans les environnements Windows car il a été conçu par Microsoft, et à partir de Windows Vista, le .net est fourni par défaut avec le système d'exploitation. Mais dans ce projet on utilise le Raspbian comme OS qui est une distribution Linux comme défini au préalable et le .net n'est pas fourni par défaut avec le système d'exploitation.

Bien que Microsoft ait publier le mois dernier (Avril 2015) le .net Core (le noyau) en Open source ce qui évidemment le rend plus accessible par les autres systèmes d'exploitation mais cela n'est pas suffisant pour réaliser un travail tel que la Gateway GSM car cela touche tous les niveaux du niveau Bas (les Bytes) au niveau haut (La compilation et l'exécution).

Pour ces raisons on a préféré utiliser un projet plus stable permettant d'exécuter les 95% des avantages du .net sous linux. Ce projet s'appelle le Projet Mono.

Mono est un projet open source (sous licences GNU GPL, GNU LGPL ou X11 selon les éléments) de la plateforme de développement Microsoft .NET basée sur la CLI (Common Language Infrastructure).

Mono a été initié par Miguel de Icaza au sein de sa société Ximian qui a été rachetée par Novell en 2003. Suite au rachat de Novell par Attachmate en 2011, Mono a été repris par une société créée pour l'occasion : Xamarin.

Mono offre une plateforme de développement complète basée sur une mise en œuvre de l'environnement d'exécution de code .NET et des API de base définis à l'ECMA (également normes ISO). Mono supporte pour l'instant la version .NET 4.5.

Dans le schéma suivant se présente l'architecture du projet Mono et une brève comparaison avec l'architecture de base de Microsoft .NET.

5) Les langages de programmation utilisés

5.a - Le C# (C Sharp)

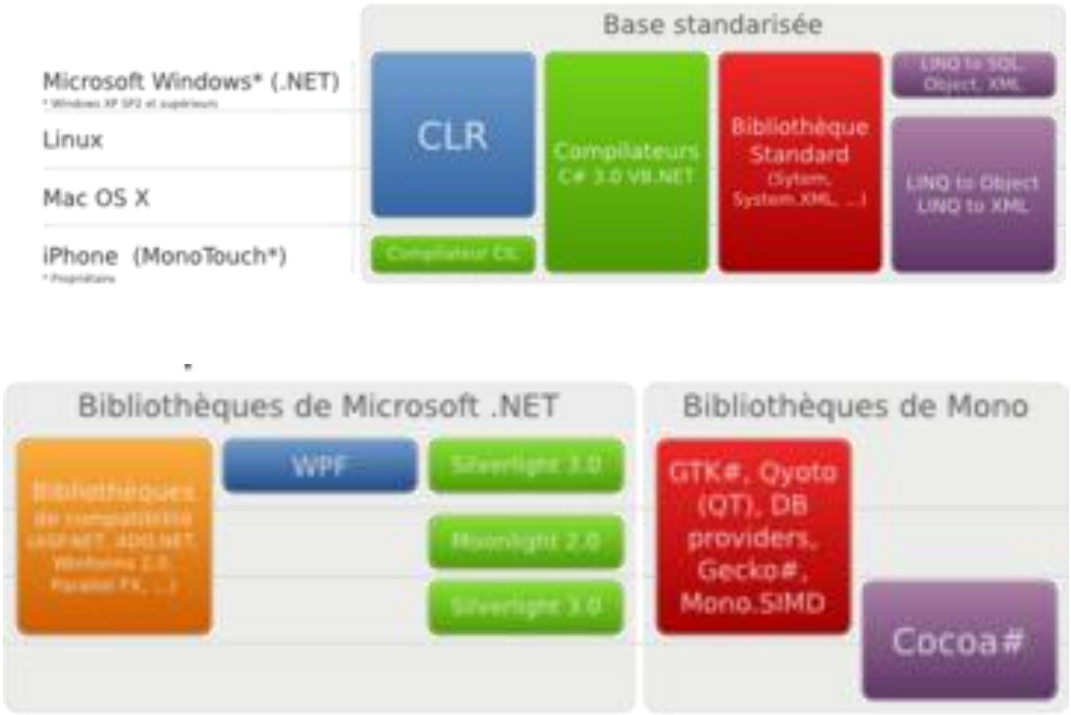

Figure 16 Structure du projet mono et comparaison avec la structure du .NET

Lors de la réalisation de la Gateway le langage majoritairement utilisé est le C#. Nous avons pris en considération plusieurs paramètres, principalement la flexibilité du langage et la capacité d'accès aux ressources du système ou dans l'application cliente sous Windows soit dans l'application qui gère la passerelle sous Raspbian. Pour cette raison la définition et la compréhension de ce langage est primordial pour pouvoir assimiler la logique suivie dans la réalisation de la Gateway.

Le C# est un langage orienté objet sécurisé qui permet aux développeurs de générer diverses applications qui s'exécutent sur l'environnement .NET comme expliqué au préalable. C# donne la possibilité de créer des applications Windows, des services Web, des composants distribués, des applications client-serveur comme dans notre cas et des applications ayant des interactions avec les différents types de bases de données.

La syntaxe C# est très expressive, et facile à reconnaître à ses accolades communes aux langages C, C++ ou Java. La syntaxe C# permet de répondre à de nombreuses complexités de C++ en fournissant des fonctionnalités puissantes telles que des types valeur Nullable, des énumérations, des Delegates, des expressions lambda et des accès directs à la mémoire qui n'existent pas dans d'autres langages utilisant des technologies de virtualisation tel que Java. C# prend en charge des méthodes et types génériques qui améliorent la cohérence et les performances des types, ainsi que des itérables (littérateurs), qui permettent aux implémenteurs de classes de collection de définir des comportements d'itération personnalisés simples à utiliser par le code client.

N'oublions pas aussi les expressions LINQ (Language Integrated Query) qui ont été utilisées à plusieurs reprises dans ce projet et transforment les requêtes fortement typées en construction de langage de premier ordre.

Outre ces principes orientés objet de base, C# permet de développer facilement des composants logiciel à travers plusieurs constructions de langage innovatrices, y compris les éléments suivants :

- Les signatures de méthodes encapsulées, appelées délégués, qui activent les notifications d'événement de type sécurisé.
- Les propriétés, utilisées comme accesseurs pour les variables membres privés.
- Les attributs, qui fournissent des métadonnées déclaratives à propos des types au moment de l'exécution.

Si on veut interagir avec d'autres ressources du système tels que des objets COM ou des DLL natives, on peut le faire en C# à l'aide d'un processus appelé « interopérabilité ». L'interopérabilité offre aux programmes C# autant de possibilités qu'une application C++ native. C# prend en charge les pointeurs et le concept de code « UNSAFE » lorsque l'accès direct à la mémoire est absolument essentiel. L'un des avantages offert par l'interopérabilité est le principe des noms d'espaces « namespaces » qui ont été utilisé dans les 2 applications de ce projet notamment pour appeler les Frameworks ou les bibliothèques du système ou même les bibliothèques créés dans la même solution comme les CL de modèle dans l'application cliente.

Dans le schéma suivant s'explique la relation entre la partie précédente et la partie C# :

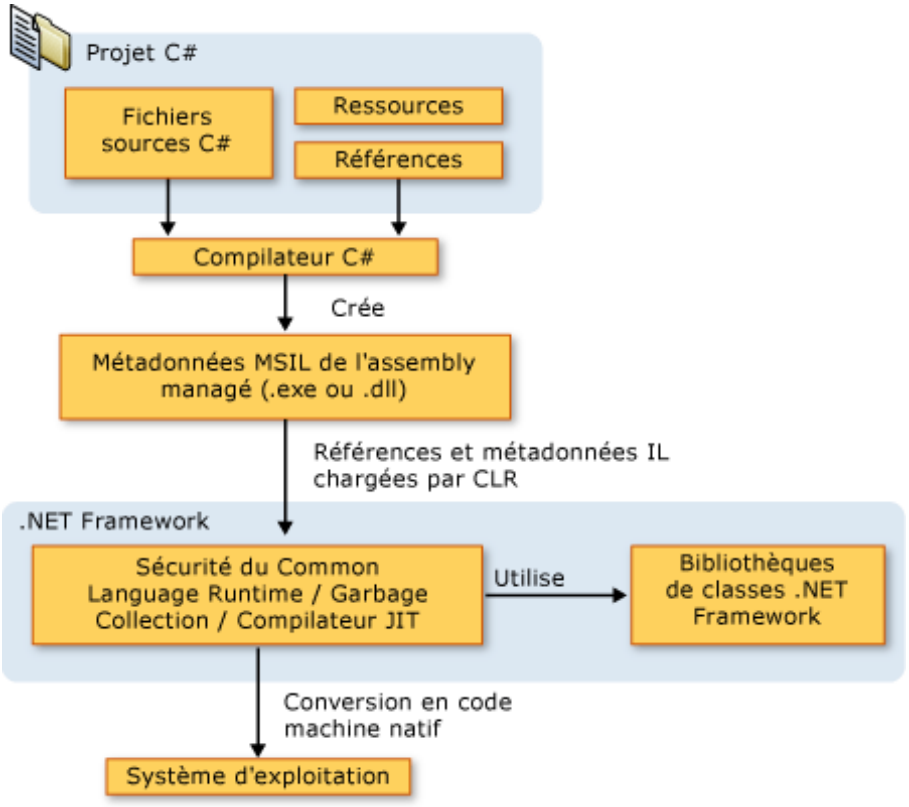

Figure 17 L'exécution d'un projet C# .NET

5.b - Le XAML

Pour la réalisation de l'application cliente on a besoin d'une technologie permettant de concevoir des interfaces graphique et non pas console, on a songé à utiliser la technologie « Windows Presentation Foundation » qui sera détaillée dans la partie suivante. Cette technologie repose sur le XAML pour concevoir les Users interfaces. XAML est l'acronyme de « eXtensible Application Markup Language », donc c'est un langage à balises pour applications extensibles. Il est appelé extensibles car les applications ont la possibilité d'ajouter des composants pendant l'exécution cette technique est appelée « RUNTIME ». Le XAML sert à donc décrire le contenu des fenêtres que nous allons voir à l'écran. En fait, un fichier XAML correspond à une page dans notre application cliente. Les éléments du langage XAML correspondent à des balises de positionnement, des composants d'interface utilisateur graphiques tel que les boutons, champ de texte, menu, liste, bouton radio, …etc.

Ces composants de même que le texte utilisent des graphismes vectoriels permettant d'ajuster leur taille sans perte de définition graphique.

6) WPF

Windows Presentation Foundation, plus communément connu sous le nom de WPF c'est un système de présentation nouvelle génération qui génère des applications clientes Windows avec des expériences utilisateur visuellement très évolué coté graphisme et consommation de ressources. Avec WPF, on peut créer une large gamme d'applications autonomes s'exécutant dans le système ou intégrées par un navigateur.

Le cœur de WPF est un moteur de rendu vectoriel indépendant de toute résolution, créé pour tirer parti du matériel graphique moderne. WPF étend le cœur avec un jeu complet de fonctionnalités de développement d'applications qui incluent XAML (eXtensible Application Markup Language), des contrôles, la liaison de données et la disposition ainsi que des graphiques 2D et 3D, l'animation, les styles, les modèles, les documents, les médias, le texte et la typographie. WPF est inclus dans le Microsoft .NET Framework. On peut donc générer des applications intégrant d'autres éléments de la bibliothèque de classes .NET Framework avec les différent langages de programmation. Dans notre cas c'est le C#.

WPF existe comme sous-ensemble du NameSpace de types .NET Framework qui pour la plupart sont situés dans l'espace de noms System.Windows.

On peut instancier des classes, définir des propriétés, appeler des méthodes et gérer des événements, entièrement à l'aide de C#, le code généré automatiquement derrière la page est connu sous le nom de code Behind.

Dans notre projet on a utilisé un design pattern permettant d'optimiser les ressources, concevoir l'application en utilisant le principe d'abstraction total ce qui en résulte le faible usage du code Behind. Dans notre cas 0 code Behind n'a été généré tout fonctionne en data binding avec les différents blocs.

7) Le patron de conception (Design pattern) MVVM

Dans toutes les conceptions logicielles professionnelles un design pattern est forcément suivi pour des raisons d'extensibilité, sécurité et optimisation de l'usage des ressources, ainsi que pour faciliter la maintenance et l'entretien. Dans ce projet on a suivi le DP Model-View-ViewModel pour la réalisation de l'application cliente sous WPF.

Le Modèle-Vue-VueModèle est une architecture et une méthode de conception utilisée dans le monde du génie logiciel. MVVM est un pattern intéressant à mettre en place dans les projets pour plusieurs raisons :

Le faible couplage entre la Vue et la VueModèle permet de modifier facilement la vue sans avoir d'impact sur la VueModèle (et vice versa) :

- Il permet de tester de manière séparée les différents éléments de la solution.
- Il permet une maintenance facilitée des projets
- Le même code dans les VueModèle et Modèle peut être facilement réutilisé dans d'autres projets tout en utilisant des vues différentes tel que les plateformes mobiles.

Pour se faire une meilleure idée de l'organisation du pattern MVVM, le schéma suivant explique de manière simple les échanges entre les différents éléments.

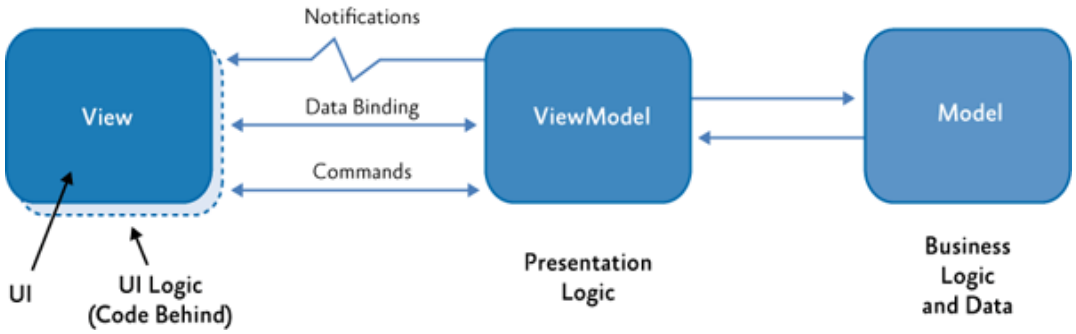

Figure 18 Schéma Descriptif du pattern MVVM

- Model : le modèle contient les données. Généralement, ces données proviennent d'une base de données ou d'un service externe comme une API.
- View : la vue correspond à ce qui est affiché (la page web dans notre cas). La vue contient les différents composants graphiques (boutons, liens, listes) ainsi que le texte.

- ViewModel : ce composant fait le lien entre le modèle et la vue. Il s'occupe de gérer les liaisons de données et les éventuelles conversions. C'est ici qu'intervient le binding.

8) Le Framework MVVM Light

Physiquement le Framework MVVM Light se présente actuellement sous la forme de deux assemblages,

- GalaSoft.MvvmLight.dll

- GalaSoft.MvvmLight.Extras.dll

Le second contient quelques ajouts récents d'une grande utilité, le premièr contient l'essentiel de la librairie. Ces assemblages sont disponibles en versions compilées pour les deux technologies dérivant de Xaml : WPF et Silverlight.

MVVM Light est complété d'une Template permettant de créer sous Visual Studio et Expression Blend de nouveaux projets intégrant de base l'ensemble des références nécessaires (ainsi que le squelette de quelques unités indispensables).

Tout repose donc sur la dll « GalaSoft.MvvmLight.dll », un exécutable pesant environ 25 Ko, ce n'est donc pas nuisible coté ressources.

Rappelons que MVVM Light s'architecture autour de quelques principes simples dont découlent des besoins spécifiques de mise en œuvre :

La séparation entre code fonctionnel et l'interface utilisateur ;

La gestion des commandes ;

La communication asynchrone entre les tiers ;

Le tiers « Modèle de Vue » ;

MVVM Light propose dans ce niveau d'ajouter une inversion de contrôle (Inversion Of Control, IoC) pour renforcer plus encore l'isolation entre Vue et Modèle de Vue. Parmi les deux solutions envisageables, le localisateur de services (Service Locator) et l'injection de dépendance

(Dependency Injection), MVVM Light a opté pour la plus simple : comprendre et mettre en œuvre, le localisateur de services. Le but est en fait de construire une application qui, plutôt qu'un monstre monolithique, se présente sous la forme de blocs de plus petite taille n'ayant aucune dépendance directe entre eux.

On comprend bien que l'esprit de l'IoC est avant tout de donner un moyen de maîtriser l'extravagante complexité galopante des logiciels modernes. Ce principe a été utilisé dans notre projet avec le Locator qui est implémenté par défaut dans l'application et qui gère les instances des ViewModel.

9) Visual Studio

Pour une utilisation correcte du C# un bon EDI est requis lors du développement et pour cela il n'y a pas mieux que l'EDI officiel qui est Visual Studio. C'est un ensemble complet d'outils de développement permettant de générer des applications Web ASP.NET, des Services Web, des applications bureautiques (Windows Forms ou WPF) et des applications mobiles (Windows Phone, Android, …etc).

Plusieurs langages tel que Visual Basic, le C# et le C++ utilisent le même environnement de développement intégré (IDE), qui permet le partage d'outils et facilite la création de solutions à plusieurs langages.

Par ailleurs, ces langages utilisent les fonctionnalités du .NET Framework, qui fournit un accès à des technologies clés simplifiant le développement d'applications.

Visual studio existe en plusieurs versions, on a utilisé la version Ultimate 2013 avec l'update 4. Ci-dessous vous se trouve une capture d'écran de Visual Studio.

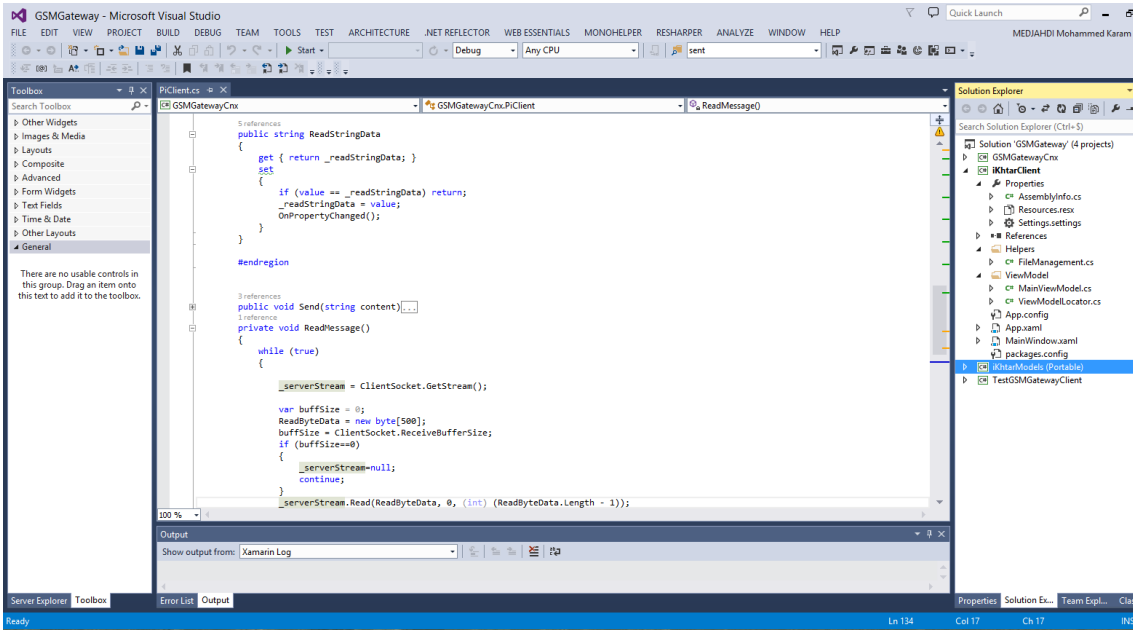

Figure 19 Imprimé d'écran sur Visual Studio

10) Conclusion

On peut donc conclure que l'ensemble des outils .NET est adéquat pour réaliser ce travail et les applications Serveur et Client seront codées avec le langage C# pour les avantage qu'il présente et qui ont été détaillé dans ce chapitre.

Quant à l'interface graphique (UI) on utilise XAML car ce denier est implémenté par la technologie WPF qui nous facilite la réalisation de l'application cliente en donnant une interface de qualité et un code très facile à comprendre et à maintenir.

La méthode suivie dans la réalisation de l'application est en utilisant le pattern MVVM et pour faciliter la tâche on utilise le Framework MVVM Light qui dispose d'un ensemble d'outils et bibliothèques prête à être utiliser.

Ce jeu d'outils et de méthodes sera productif et significatif que si on utilise de bons outils lors de développement et c'est pour cette raison que notre choix a été porté sur Visual Studio et ses extensions comme environnement de développement.

IV) Chapitre 4 : Les étapes de conception

1) Introduction

Après la compréhension de l'ensemble des outils Software et hardware faisant fonctionner la Gateway GSM il est temps de comprendre en détaille les étapes de conception ainsi que le fonctionnement logique gérant la Gateway et les clients exploitant cette dernière. Ce chapitre détaillera la logique et les algorithmes ainsi que la modélisation du projet.

L'organisation de ce projet se divise en 2 partie, les organigrammes et figures détaillant les différentes fonctionnalités et la 2ème partie explique en détaille ce que ces organigramme essayent de représenter.

2) Principe de fonctionnement

La Gateway se divise en sur deux parties, client et serveur.

Le serveur est un programme qui s'exécute en boucle infinie associé avec une Raspberry pi et le client est une application qui envoie au serveur des commandes. C'est au serveur de les interpréter et exécuter si elles sont valables.

Tout au début le serveur lance plusieurs opérations simultanément pour pouvoir gérer les tâches qui lui sont attribuées. Et pour cela on peut diviser la logique du serveur en 3 parties. La première est la partie qui s'occupe de la gestion et de l'interfaçage hardware. Cela concerne la détection des modems connectés grâce aux fichiers d'interfaces ttyUSBx du système Raspbian. Cette partie est orienté système d'exploitation et elle a été détaillée dans le chapitre 2.

Quant au programme il charge l'ensemble des fichiers d'interfaces et les sauvegardes dans des variables pour les tester si elles disposent des consoles permettant d'exécuter des commandes AT puis il les comparer aux fichiers de configuration mémorisées au préalable. L'ordre d'exécution et très important car si on inverse deux interfaces la Gateway acheminera mal les trames envoyés par le client.

La 2ème partie prend en charge les fichiers. Cela concerne des fichiers de configuration qui contiennent les informations nécessaire pour faire fonctionner chaque modem ainsi que les informations permettant aux algorithmes d'acheminement de reconnaitre le bon modem qui enverra les commandes, ces fonctions d'acheminement seront détaillé dans des parties antérieurs.

Les numéros favoris sont stocké dans des fichiers qui sont répertoriés dans la racine où se trouve l'application par défaut mais on peut changer de chemin et le modifier dans les fichiers de configuration.

Les fichiers de configuration sont des fichiers XML qui sont à l'origine persistée avec des collections d'objet de type Config et cette persistance est gérée par des classes de dépôts de données.

Pour la gestion de fichier les fichiers de configurations contiennent aussi le chemin des fichiers qui stockent les numéros favoris attribués à chaque modem. Ces fichiers permettent aux fonctions d'acheminement de donner la priorité à des numéros favoris pour chaque modem

Le 3ème partie du programme est orientée réseau elle prend en charge toutes les fonctionnalités qui gèrent la communication dans le réseau. Cela concerne principalement les sockets, le cryptage et la gestion des clients connectés. Et en gros le programme assure la coordination entre ces 3 parties parce qu'elles sont interconnectées l'une à l'autre et chacune dépend de l'autre.

Pour ce qui concerne le client il a été programmé pour faire passer les messages selon le format de trame déterminé. Il demande la connexion via l'adresse IP du serveur et le port de connexion acquis depuis l'interface graphique, puis il donne la main pour ses fonctionnalités.

Si l'utilisateur choisi d'envoyer des SMS l'application cliente charge des données dans des propriétés dans l'instance du ViewModel et fait appel aux fonctions qui formatent les trames avant de les crypter et de les envoyer à l'instance gérant la communication avec le serveur.

Il faut noter que chaque client dispose de son propre système gérant les numéros favoris en les enregistrant dans des fichiers et en les envoyant dans une trame spéciale qui sera interprété par le serveur qui lui à son tour l'enregistre les numéros favoris définis dans les fichiers de configurations comme cité dans la partie précédente.

3) Fonctionnement du serveur

Comme cité dans la première partie, le serveur commence à chaque démarrage par la lecture des fichiers de configuration qui contient principalement l'interface système du modem, le nom de l'opérateur, les paramètres concernant les indicatifs téléphoniques locaux et internationaux, le chemin des numéros favoris.

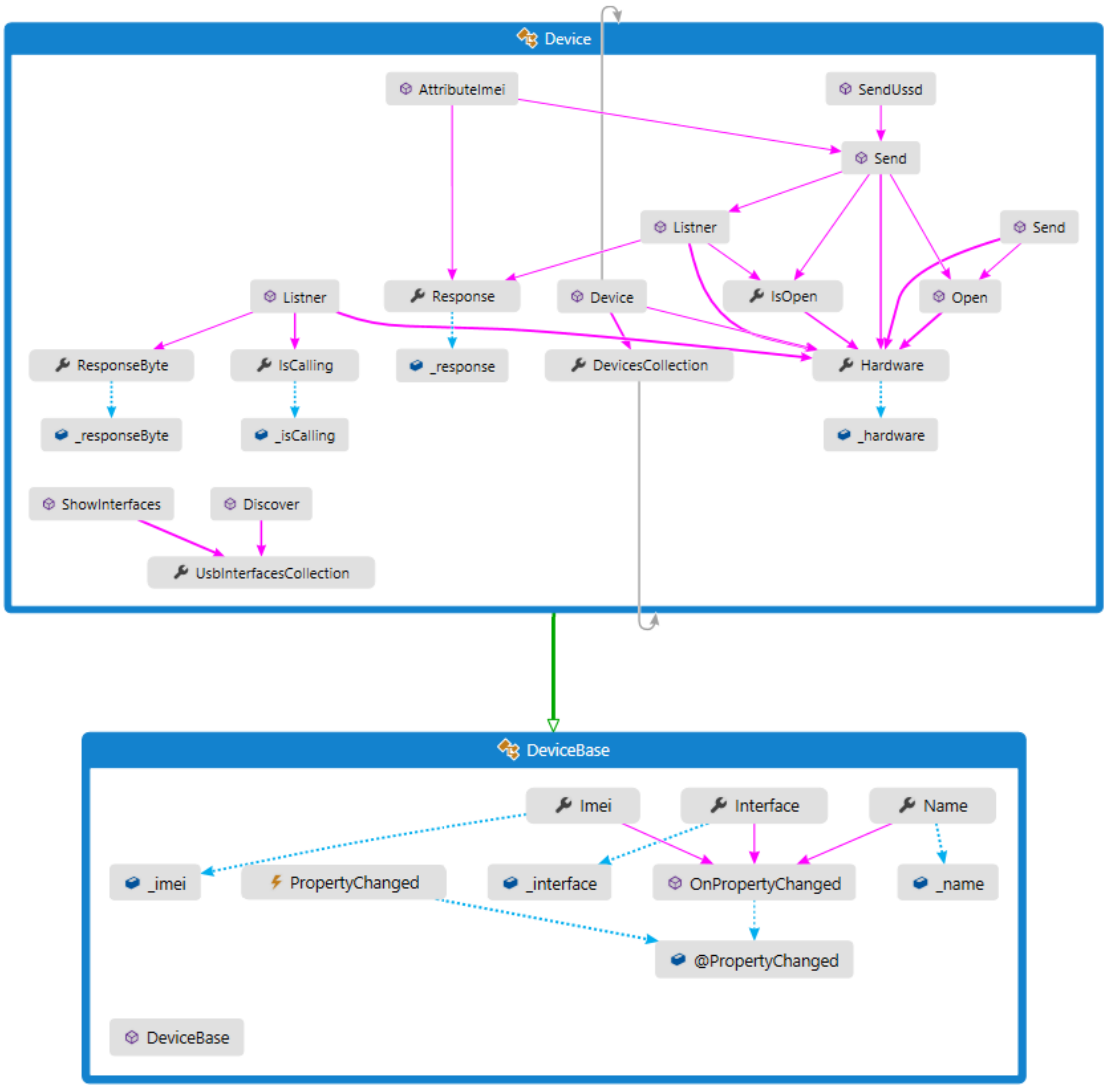

Figure 20 Code map de Device et DeviceBase

Ce code map représente « Device » et « DeviceBase » qui sont utilisées dans le fichier de configuration (DeviceBase). Les instances sont de type Device, elles sont créées après la lecture du fichier.

Ce fichier est réellement une persistance d'une collection itérable des instances de type config « Config ». Lors du démarrage l'application les charges dans la mémoire pour procéder à un algorithme de vérification de disponibilité pour chaque modem pour pouvoir les mettre en service.

Une autre tâche s'exécute aussi en même temps c'est le chargement des favoris. Une balise spéciale qui correspond à la persistance de la propriété ContactsPath de la classe Config, cette dernière indique le chemin du fichier des numéros favoris.

Il faut noter que la classe config dispose d'une propriété de type DeviceBase, les propriétés de ce type n'ont pas été implémenté directement dans la classe config pour pouvoir être réutilisée en héritant de leurs classe dans d'autres partie du projet gérant l'application server. On évite ainsi la redondance du code.

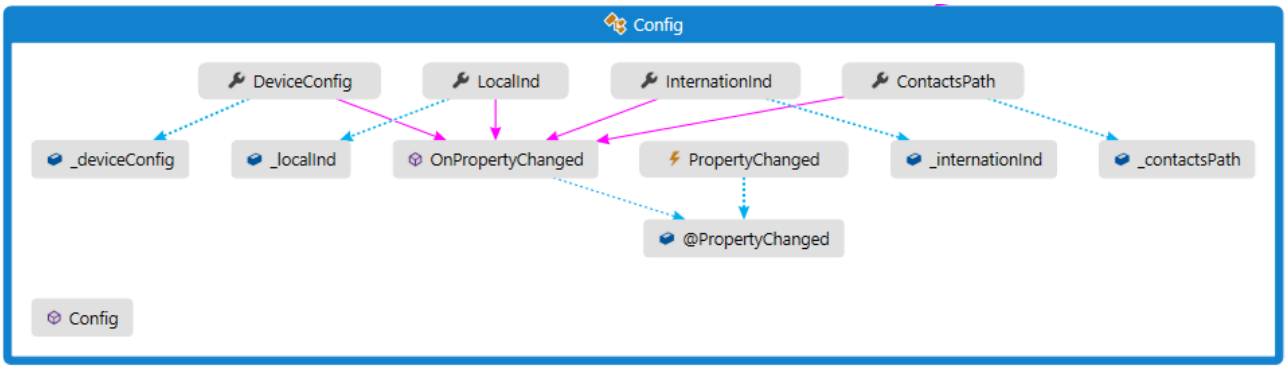

Figure 21 Code map de Config

Le code map est un DGML et les représentations contenues suivent le format conventionnel des DGML. En haut se présentent les propriétés encapsulées et les différentes fonctions. Quant à PropertyChanged elle représente un évènement de notification.

Les instances de cette classe ont la possibilité de notifier après avoir été instanciés cela sera utilisé lors de l'usage par d'autre classes ; ça permet de s'abonner au changement des propriétés qui ont la capacité de notifier. Donc cette partie est primordiale pour le démarrage du serveur.

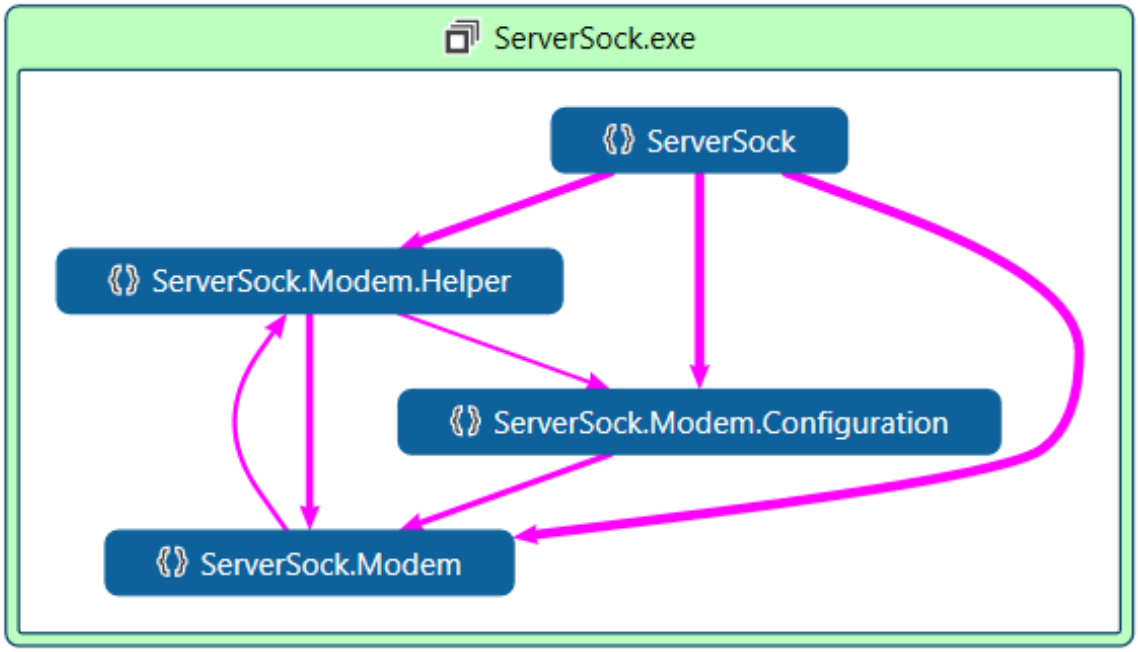

Figure 22 Code map de la solution

Dans ce code map on peut visualiser les différents NameSpace construisant la solution. ServerSock et le nom du NameSpace principale. Quant au Modem.Helper c'est un espace de nom contenant des classes qui contiennent des méthodes permettant de générer des commandes AT, parser et parcourir les trames,…etc.

On les a regroupé dans un même NameSpace parce qu'elle sont abstraite au sens fonctionnel et non pas programmatique, ce qui veut dire qu'on a la possibilité de les exécuter d'une façon séparée du projet et qu'elles ne dépendent pas des résultat des autres fonctions ou propriétés.

Les autres NameSpace sont détaillés par des organigrammes et dans les parties prochaines les diagrammes de classes présents dans l'UML de l'application vont détailler leurs contenue.

Voilà l'organigramme de l'algorithme de fonctionnement du serveur.

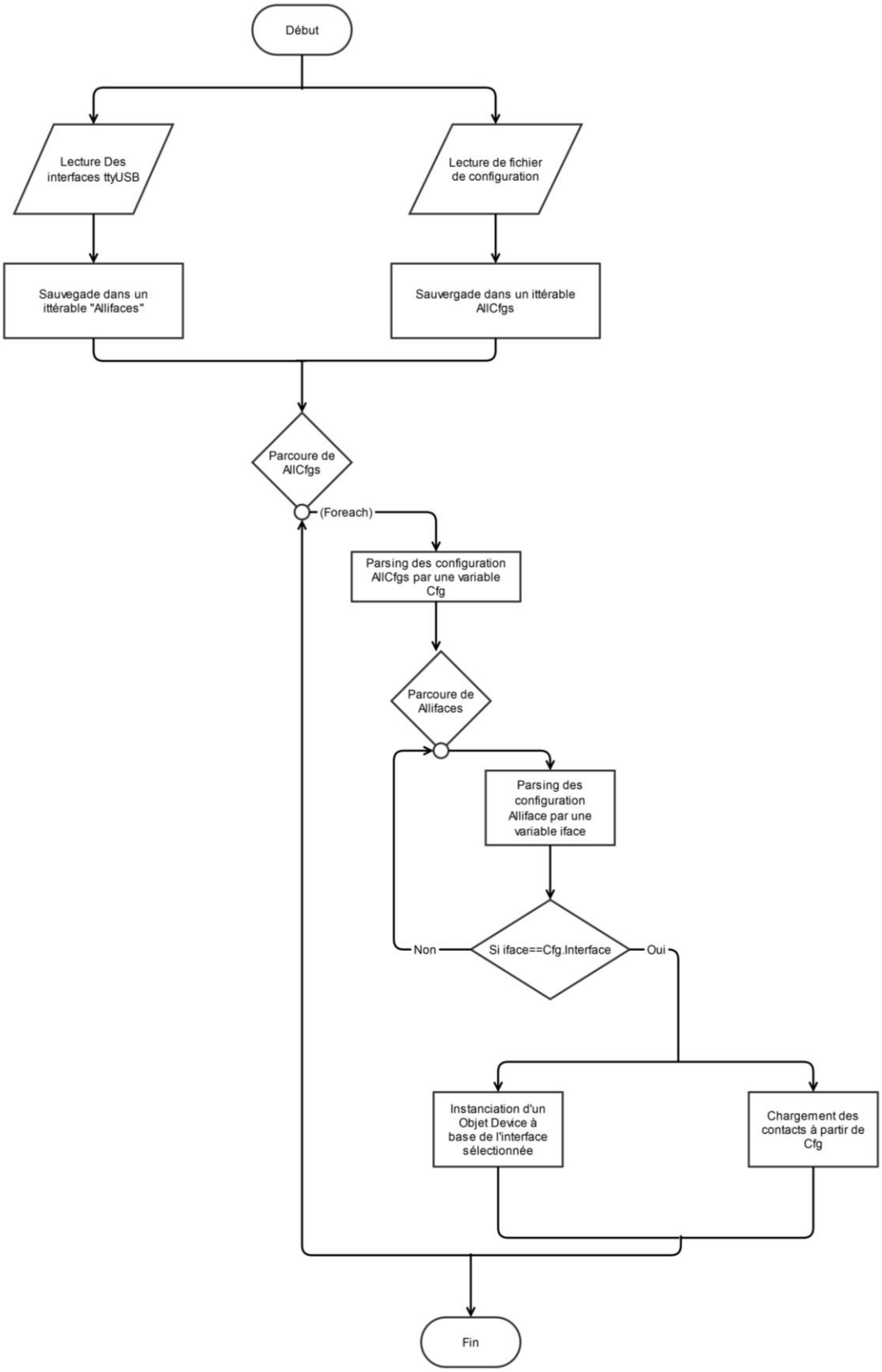

Figure 23 Algorithme de détection & création des instances

Après que l'application serveur démarre et charge les modems elle démarre d'autres processus d'écoute simultanément pour pouvoir exécuter ces commandes via le réseau et pour cette raison on a implémenté une socket serveur qui donnera la possibilité d'accéder au Raspberry pi dans le réseau via un moyen programmatique.

Mais pour rester dans le cadre de la Gateway GSM on a formalisé les commandes qui peuvent être acceptées et on les reformule en commandes AT pour les exécuter dans les modems chargées au préalable. Ces sockets ouvrent la communication dans le réseau et cela après avoir lu le port d'écoute depuis un fichier puis lancent l'écoute sur toutes les destinations. L'adresse réseau correspondant à toutes les destinations est 0.0.0.0 ce qui implique que peu importe le moyen d'accès au réseau (soit Ethernet Eth0, soit wifi Wlan0). Le port par défaut qu'on a pris est le 17000.

Pendant que le serveur démarre l'écoute et l'attente des clients les modems sont déjà chargés dans la mémoire et sont prêts pour exécuter des commandes AT. Après que le serveur démarre une attente de client est lancée. Si le serveur détecte qu'un client demande la connexion, il la valide et répond aléatoirement dans un port supérieur à 1024. La communication est lancée en Thread et le serveur libère le port pour que d'autres connexions puissent avoir lieu. Le serveur reste ainsi toujours en écoute.

Chaque client connecté peut exécuter les commandes dans un format bien spécifié, le format des messages sera détaillé dans la partie suivante. Il faut noter que le serveur doit conventionnellement être lancé dans une boucle infinie pour pouvoir maintenir la connexion et le lancement de plusieurs clients simultanément, mais il ne faut pas oublier que les boucles infinies engendre une perte de ressources malgré leurs faible consommation dans tel cas, et pour cela on a pensé à maintenir la connexion d'une façon différente : c'est la récursivité. Cette partie sera schématisée et expliquée avec le code map du serveur.

Et il n'y a pas que la partie réseau qui a la possibilité de lancer des commandes sur le serveur, on peut les exécuter localement ou par un accès SSH, et pour cela on a conçu une console qui peut exécuter des commandes via la CLI directement en lançant les commandes qui peuvent être listées en tapant la commande help.

Le diagramme suivant explique le fonctionnement :

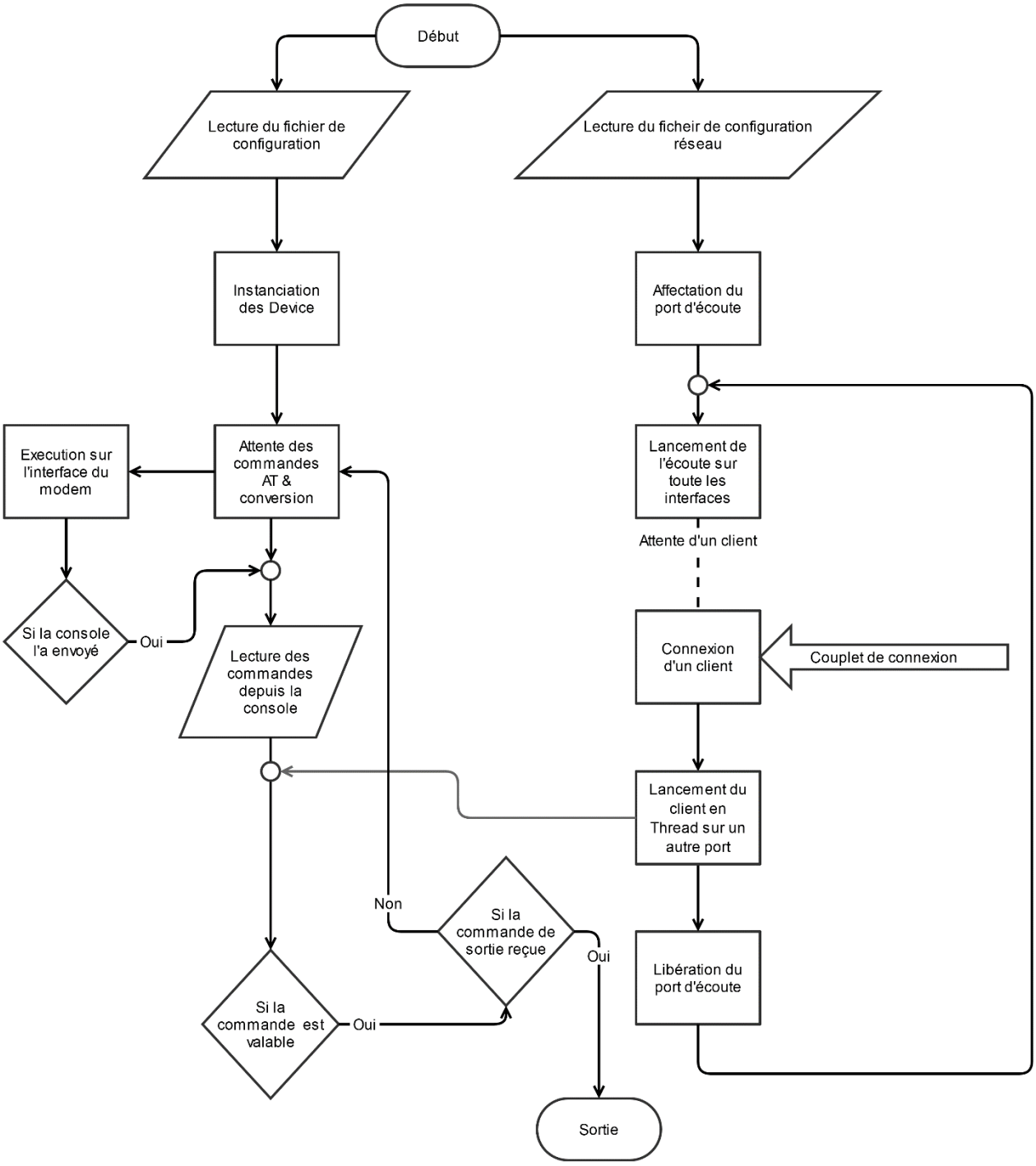

Figure 24 Organigramme de fonctionnement du serveur

Dans notre programme le serveur tourne à l'infinie non pas avec une boucle infinie mais une fonction récursive qui fait appel à elle-même à chaque fois que l'écoute est lancée elle permet donc d'une façon ou

d’une autre au serveur d’être en écoute d’une façon permanente et donne aux clients la possibilité d’être connecté à n’importe quel moment.

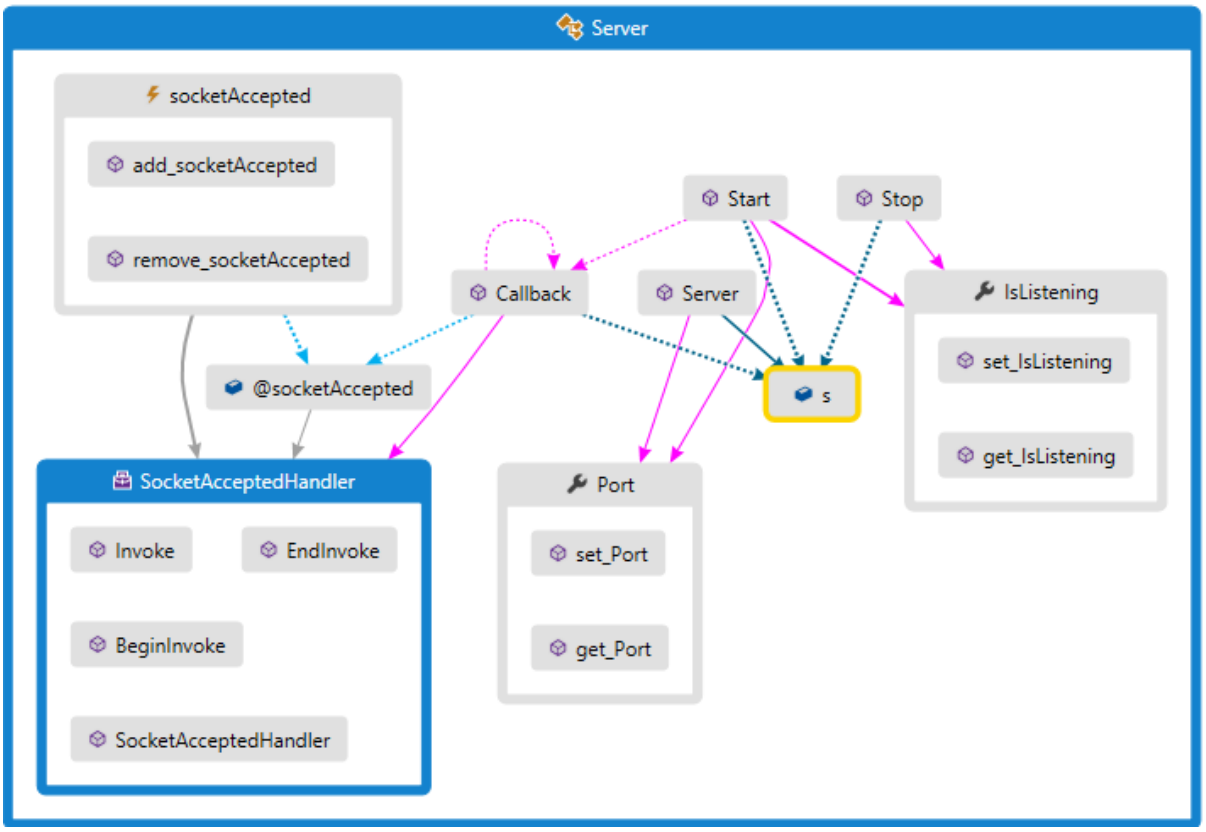

Figure 25 Code map de la classe Server

Il faut savoir que lors du lancement de server l’instance AcceptedClient issue d’une classe de ce même nom et qui a la capacité de gérer la communication lorsqu’un client est accepté est créé. Elle est dotée des Handler avec leurs évènements qui permettent aux autres instances d’envoyer des notifications et de s’abonner au comportement des clients surtout pour les 3 cas : Connexion, Déconnexion, envoie de messages. Ci-dessous le code map de cette classe.

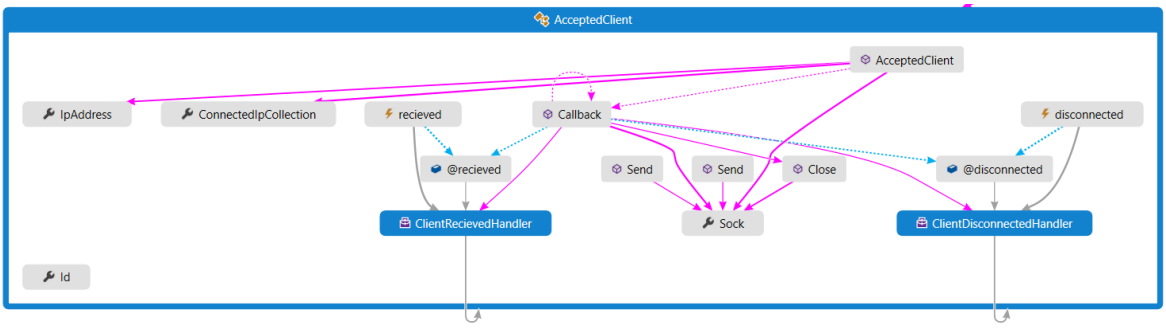

Figure 26 Code map de la classe AcceptedClient

4) Fonctions et algorithmes d'acheminement de services

Une fois une trame arrivée depuis le réseau ou la console son format doit être unique, une fois lue et vérifiée elle passe par un ensemble d'algorithmes permettant d'acheminer le service.

Rappelons-nous que ces trames sont dans la couche application sur le modèle OSI, car elle sont encapsulées et traitée après l'établissement d'une session client-serveur. Ci-dessous le modèle de trame.

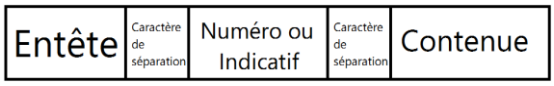

Figure 27 Modèle de trame acceptée par la Gateway

Un premier algorithme teste le nombre de caractères de séparation. Dans notre programme c'est le caractère pipeline « | » qui sépare entre les différentes parties. S'il y a deux caractères contenus dans une même trame c'est qu'elle peut être validée. Autrement elle sera automatiquement rejetée. Cette trame peut être soit construite par le client et envoyée soit construite en background de la console. Après le test des pipelines la trame est splittée en un tableau de chaîne de caractères de 3 parties entête, numéro ou indicatif, et le contenue utile. Ces 3 paramètres vont permettre à d'autres fonctions d'acheminer le message correctement.

Pour ce qui concerne l'entête c'est soit SMS soit USSD soit CTT. Au début le test de CTT est effectué si cet entête est détectée l'algorithme appel une fonction qui achemine le contenu (le 3ème élément du tableau) dans le favoris celons l'indicatif présent dans le 2ême élément du tableau.

Le 2ème cas d'entête et les USSD il fonctionne de la même façon que l'entête CTT sauf qu'au lieu d'enregistrer le contenu il exécute une commande USSD suivant l'indicatif envoyé.

Quant à l'entête SMS il exécute une suite de commandes AT correspondants aux commandes AT d'envoi de SMS. Deux algorithmes sont déclenchés le 1ère pour vérifie si le numéro n'existe pas dans l'un des favoris de modem, il est alors acheminé automatiquement selon l'indicatif présent dans le numéro (en Algérie c'est le 2ème numéro qui constitue l'indicatif local). Dans le cas contraire, il est acheminé vers le modem contenant ce numéro dans ses favoris. Les 2 organigrammes suivant l'expliquent clairement.

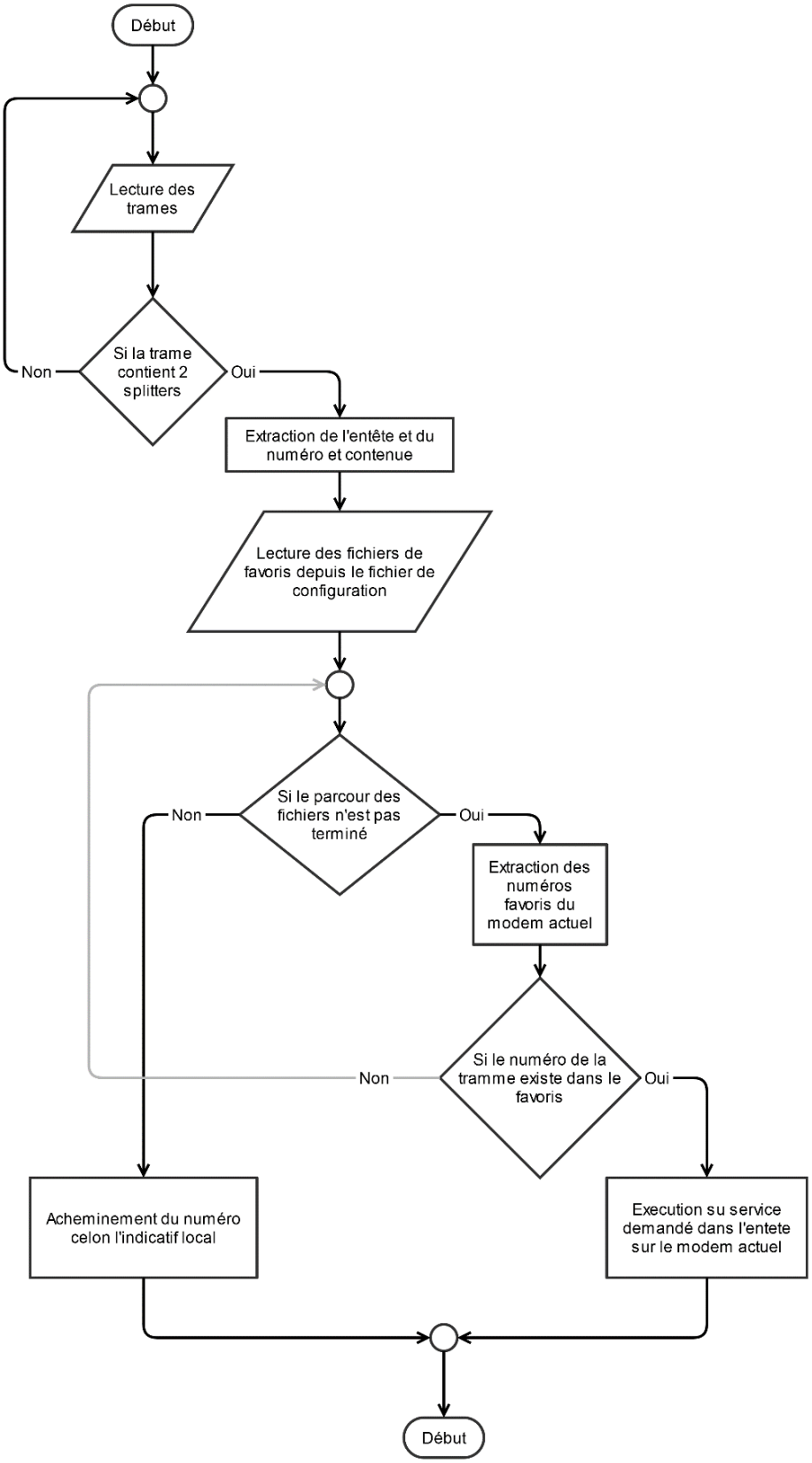

Figure 28 Algorithme d'acheminement global

L’organigramme précédent représente le comportement de la Gateway globalement lors de l’acheminement des services, et dans la partie « acheminement du numéro selon l’indicatif local » s’exécute un 2eme algorithme qui acheminera automatiquement le service et cela dans le cas où le numéro est introuvable dans l’un des favoris du modem.

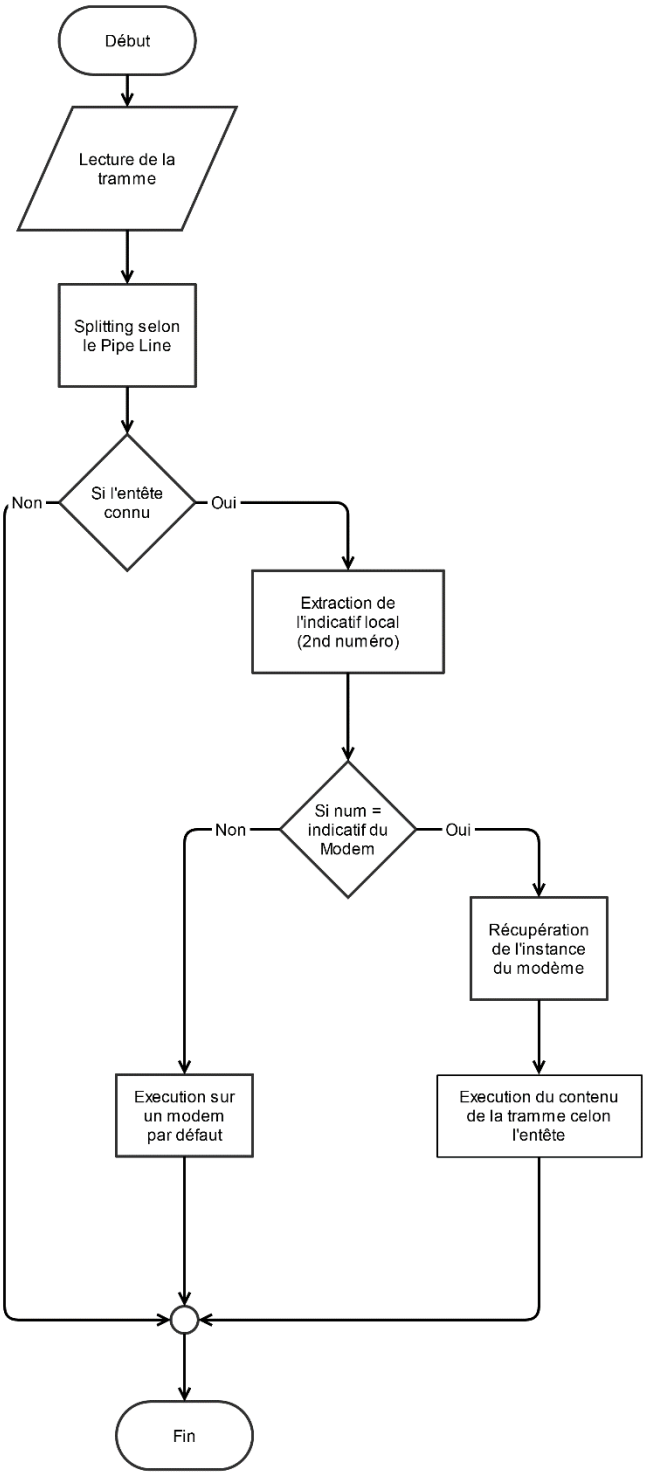

Figure 29 Algorithme d’acheminement automatique (par défaut)

5) Modélisation du serveur

Il ne faut pas oublier que cette conception implique la présence impérative des UML & diagrammes de classe qui représentent une norme de modélisation logicielle. Ci-dessous se trouve la modélisation sous forme d'un ensemble de diagrammes de NameSpace et de classes.

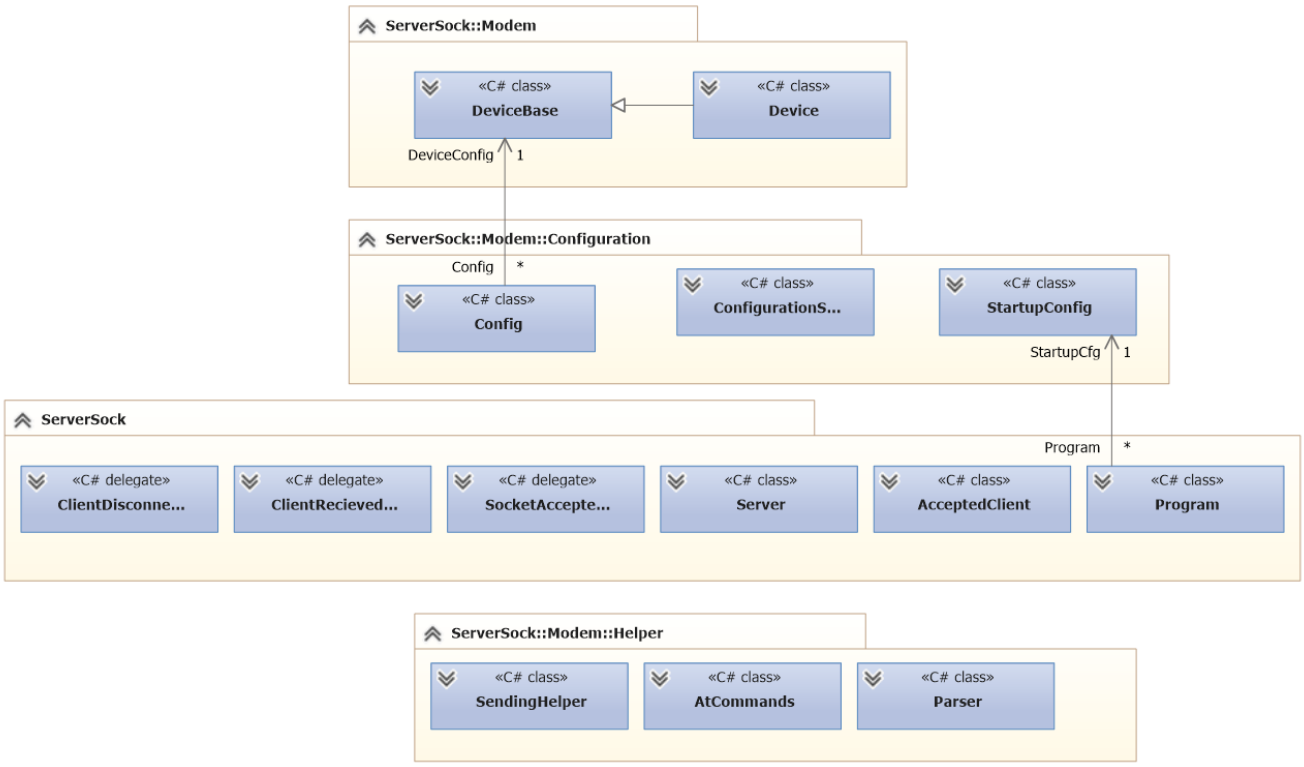

Figure 30 Diagramme de classes et NameSpaces global

Ce diagramme représente une vue global sur les classes présentes dans chaque NameSpace. Les diagrammes de classes de chaque NameSpace seront détaillés dans l'ensemble des diagrammes suivants.

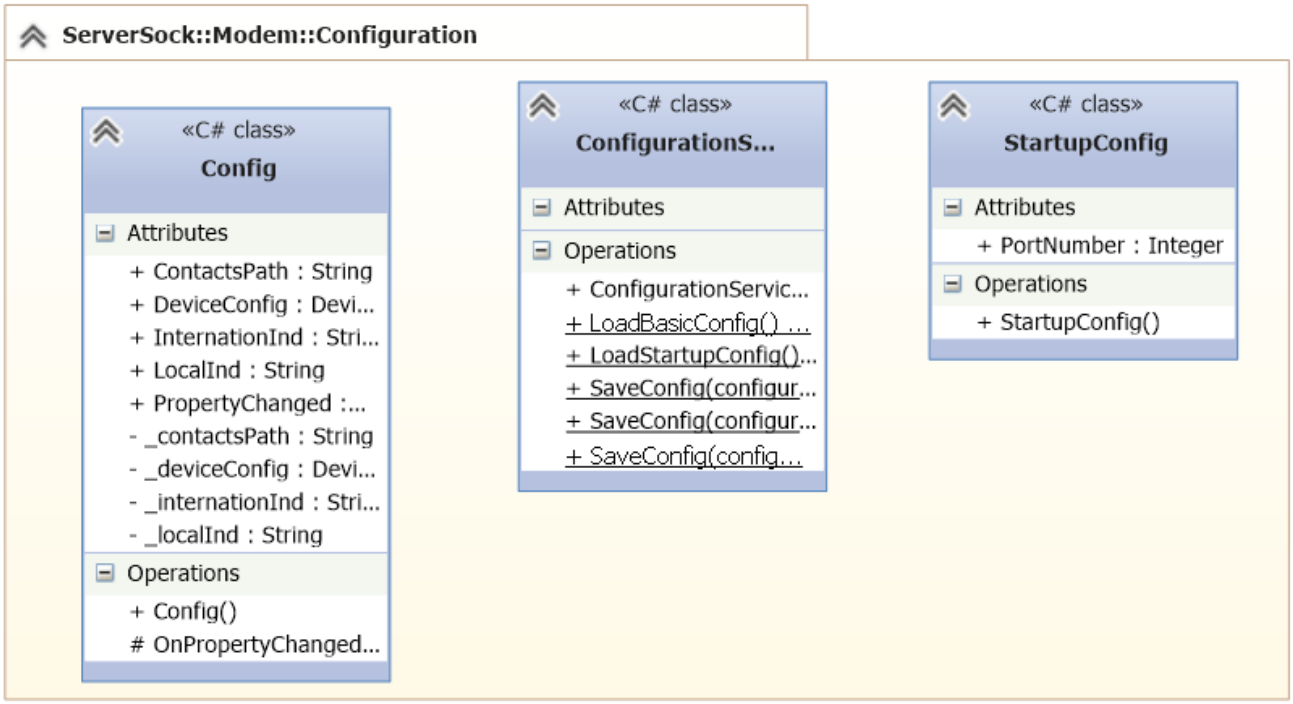

Figure 31 Diagramme de classes du NameSpace Modem.Configuration

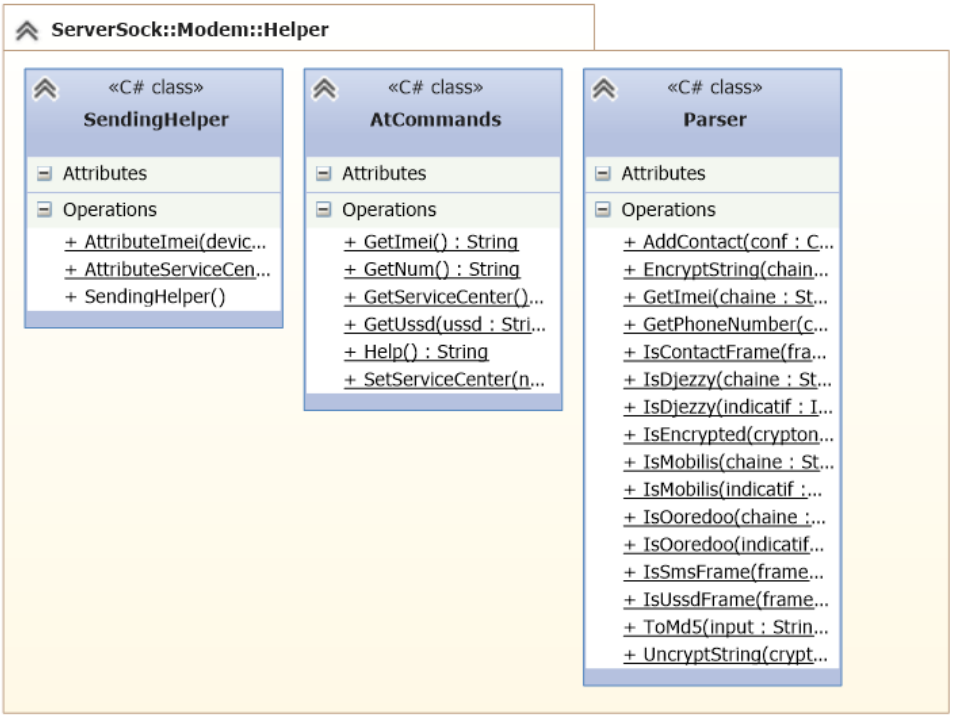

Figure 32 Diagramme de classes du NameSpace Modem.Helper

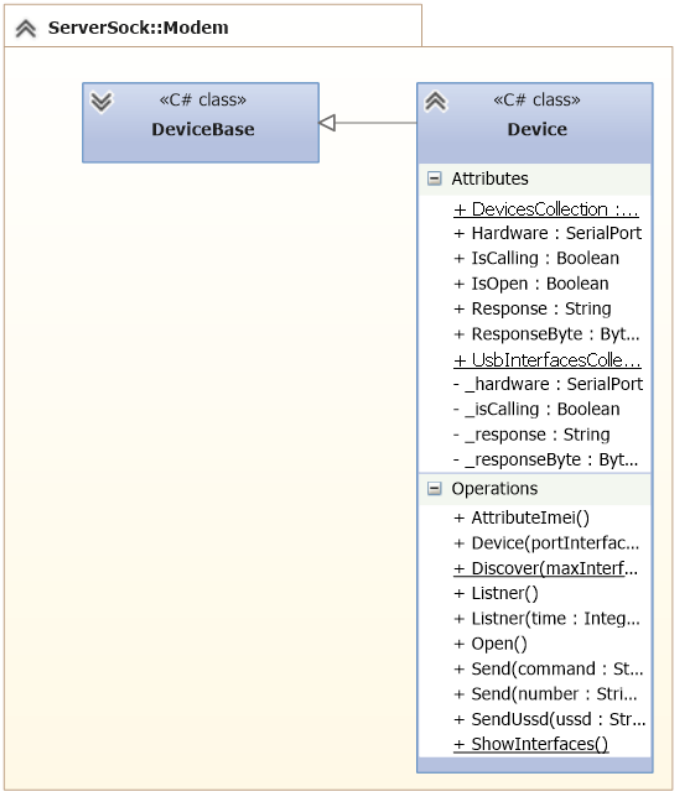

Figure 33 Diagramme de classes du NameSpace Modem

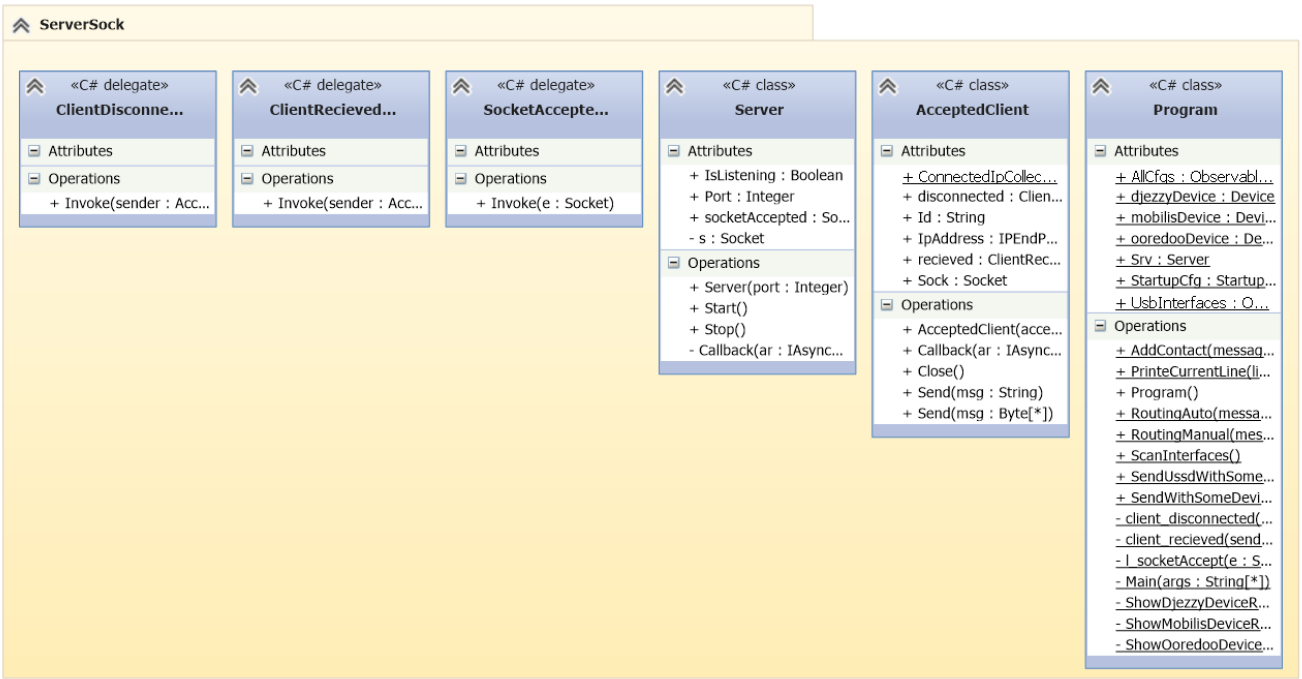

Figure 34 Diagramme de classes du NameSpace principal

Pour mieux expliquer le fonctionnement global le code qui contient les classes et les relations de chaque NameSpace peut améliorer la compréhension du fonctionnement de la solution.

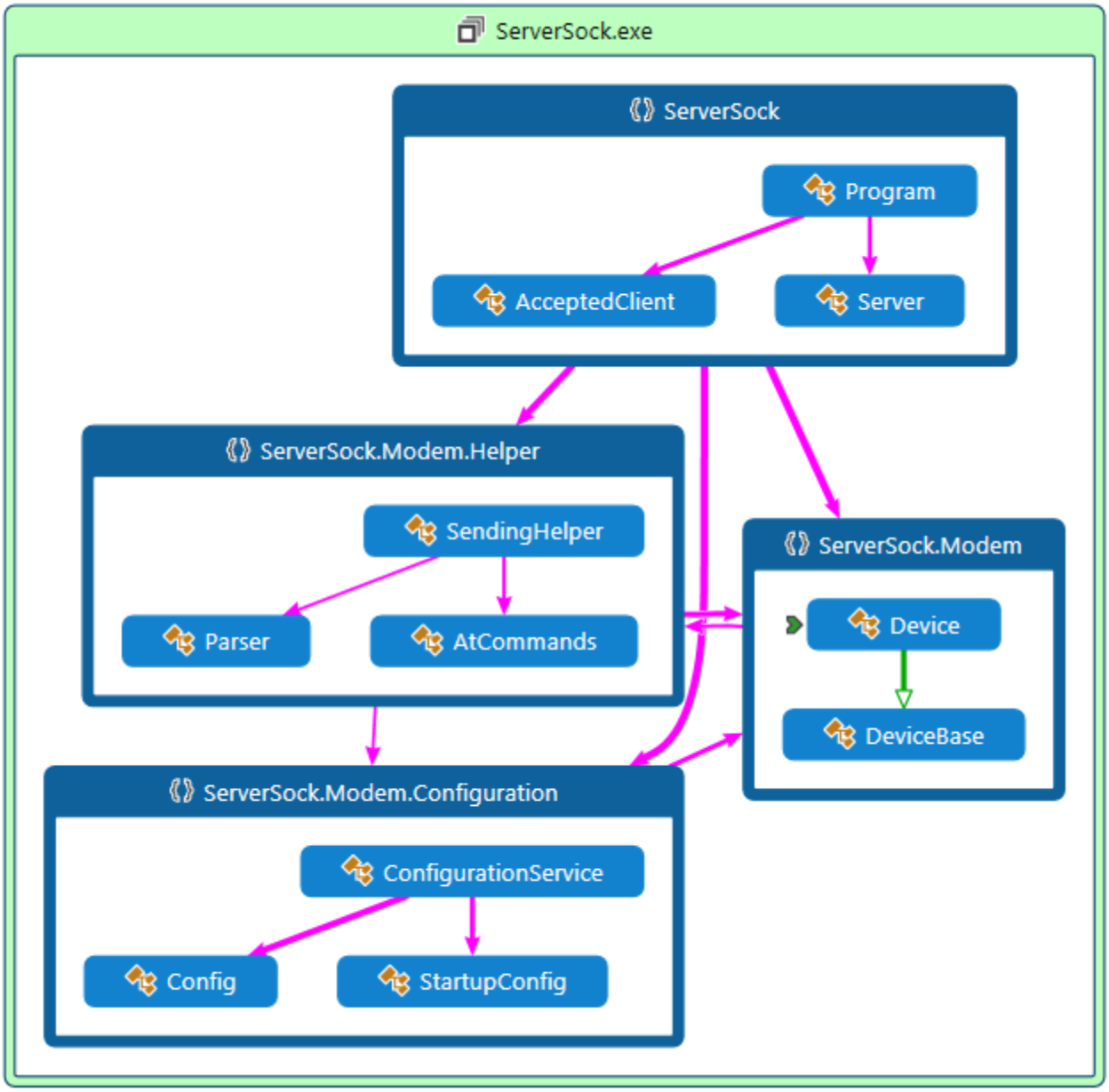

Figure 35 Diagramme de la solution avec les classes

6) Modélisation de et concepts de l'application cliente

Comme on l'a cité dans le chapitre précédent l'application cliente suit un design pattern appelée MVVM en appliquant le Framework MVVM light et cela pour plusieurs raison, on cite majoritairement l'abstraction des blocs. En résulte la facilité de maintenance et l'extensibilité du projet. Pour ce qui concerne l'extensibilité cliente on a pensé aux travaux qui peuvent être réalisés dans le futur en se basant sur ce travail, et on a conçu les model en PCL (Portable classe Library) et en une librairie gérant la communication. Ces 2 parties sont des projets à part dans la même solution que la solution cliente. Cela veut dire qu'un travail d'équipe peut être envisagé facilement car aucune dépendance directe n'est présente dans ces 3 projets.

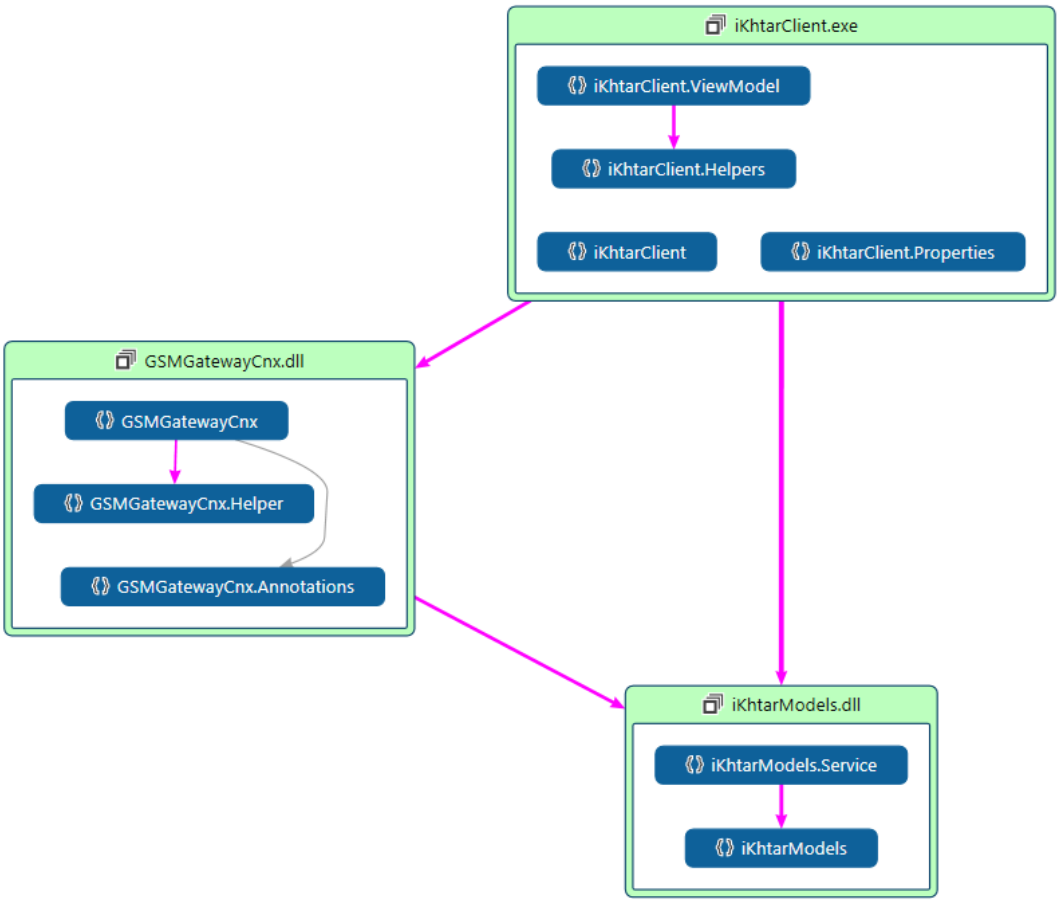

Figure 36 Code map de la solution

Il faut noter que la logique appliquée respecte strictement les normes de conception de MVVM et ce qui en résulte c'est l'absence totale du code Behind.

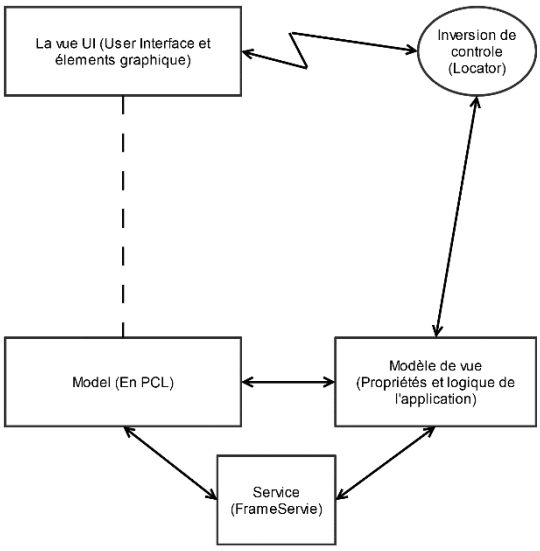

Figure 37 Schéma synoptique du concept de l'application cliente

Ce schéma se divise en plusieurs parties, et en concevant l'application cliente on s'est basé sur la méthode « Code First ».

Le modèle est un projet PCL, l'avantage de l'usage des PCL est la portabilité du projet. En exploitant ce type de projet on peut facilement transporter l'application dans d'autres plateformes comme Windows 8.1, 10, Windows phone, Android, iOS ou Silverlight. Ou même d'autres environnements qui intègrent le .NET tel que Delphi Prisme ce qui donne un très grand avantage.

Ce projet est constitué de deux parties :

1) La 1ère c'est les classes de base constituant les modèles :

   Le modèle comme l'indique son nom c'est des classes modèles contenant des propriétés encapsulées qui ne contiennent aucune logique ni fonctionnalités. Elles sont faites pour être utilisées par les autres classes. Les modèles présents dans notre cas sont Frame et SocketBase et c'est tout ce dont le client a besoin pour faire fonctionner le reste de l'application.

2) La 2ème c'est le service :

Elle est constituée d'une seule classe c'est la Frame Service. Elle gère la création et la lecture des trames envoyées dans le réseau il dépend des classes modèles.

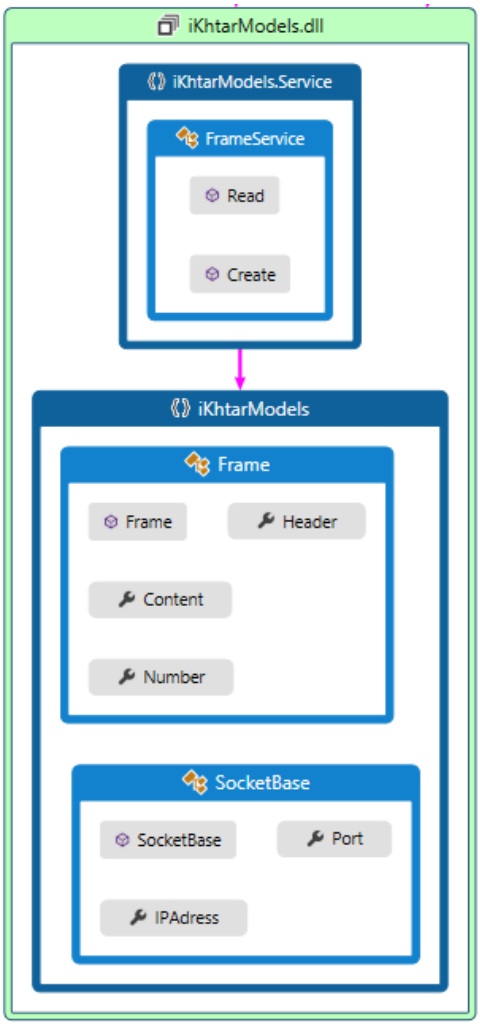

Figure 38 Code map du projet « model »

Dans ce code map on visualise les classes du modèle et l'absence des fonctions, les constructeurs par défaut, les propriétés et le service ne sont qu'une simple classe statique contenant 2 fonctions pour la création et la lecture de trames. Il se base sur le traitement de chaînes de caractères.

Le résultat du projet génère des DLL qui vont être importé dans les 2 autres projets.

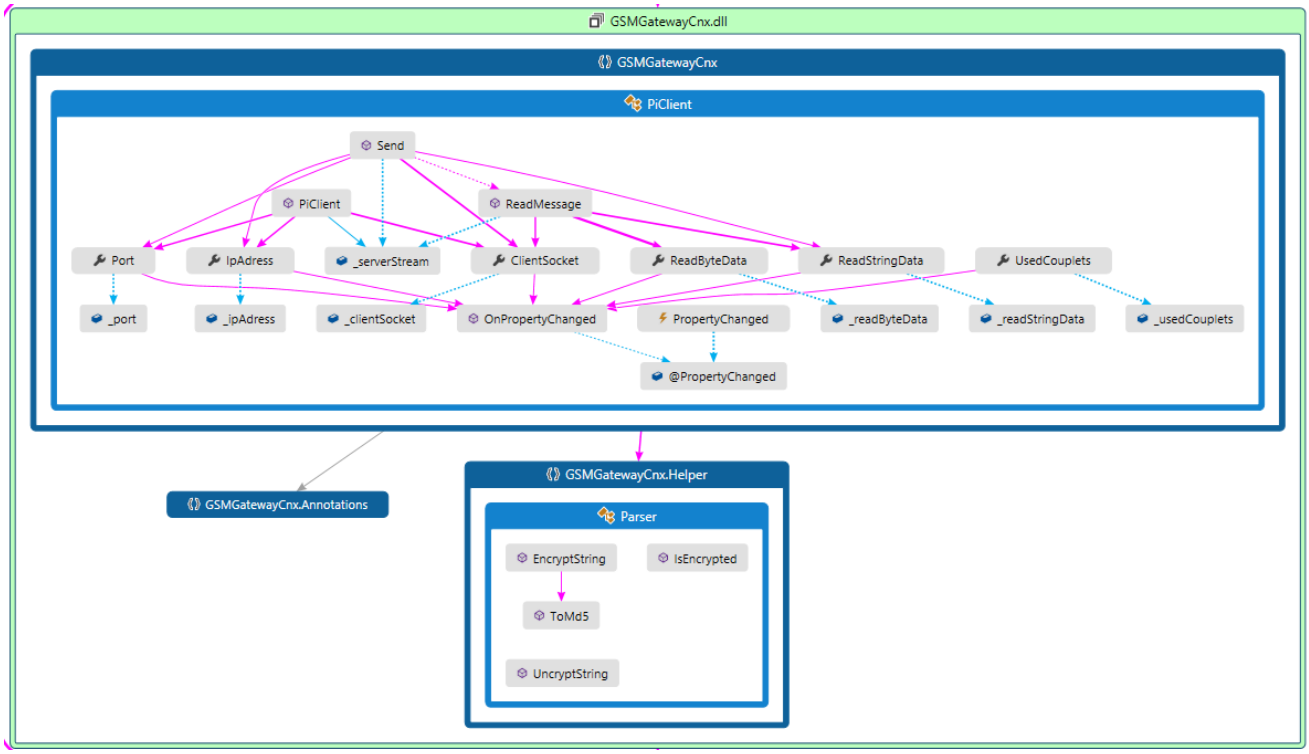

Figure 39 Code map du projet de gestion de connexion

Ce projet est aussi constitué de deux parties la 1ère qu'on a appelé PiClient et qui est responsable de la gestion de la communication en tant qu'instance cliente qui a la capacité de lire en envoyer des messages. Elle est constituée de plusieurs propriétés et fonctions qui interagissent principalement avec une socket. Cette dernière implémente l'interface INotifyPropertyChanged du NameSpace principale « system » et donc elle a aussi la capacité d'envoyer des notifications et cela est essentiel dans le pattern MVVM pour que l'UI puisse changer en temps réel et suivre l'état des variables et propriétés (grâce au DataBinding) sans devoir faire de boucles infinies ou des manipulations nuisibles pour les ressources du système. Cette entité à la possibilité d'envoyer et recevoir des données de tous type mais dans notre cas d'utilisation on n'utilise que les caractères en UTF8 pour les USSD et les SMS.

La 2ème partie est une classe Helper qui contient la classe Parser chargée de l'encryptions des données et qui sera détaillée dans le chapitre suivant.

Le 3ème Projet est un projet WPF qui fait appel par référence aux DLL des 2 projets précédents. Il est constitué principalement de la Vue et de Modèle de vue (View & ViewModel). Dans le code map suivant se trouve les principaux NameSpaces constituant ce projet. Cette logique est

l'incarnation du Framework MVVM Light le ViewModel n'a pas été ouvert comme aperçu détail à cause du nombre excessif des propriétés et méthodes présentes. L'espace réservé aux Properties a été affiché car la vue de notre projet n'utilise pas les chaînes de caractères en claire avec ses composants mais fait appel au fichier de ressources pour optimiser la visibilité dans le code XAML et donne la possibilité de produire des distributions dans des langues différentes.

La classe en bas est celle de gestion de fichier elle joue le rôle d'un repository contenant les fonctions de CRUD auxquelles l'instance du ViewModel peut faire appel d'un instant à un autre.

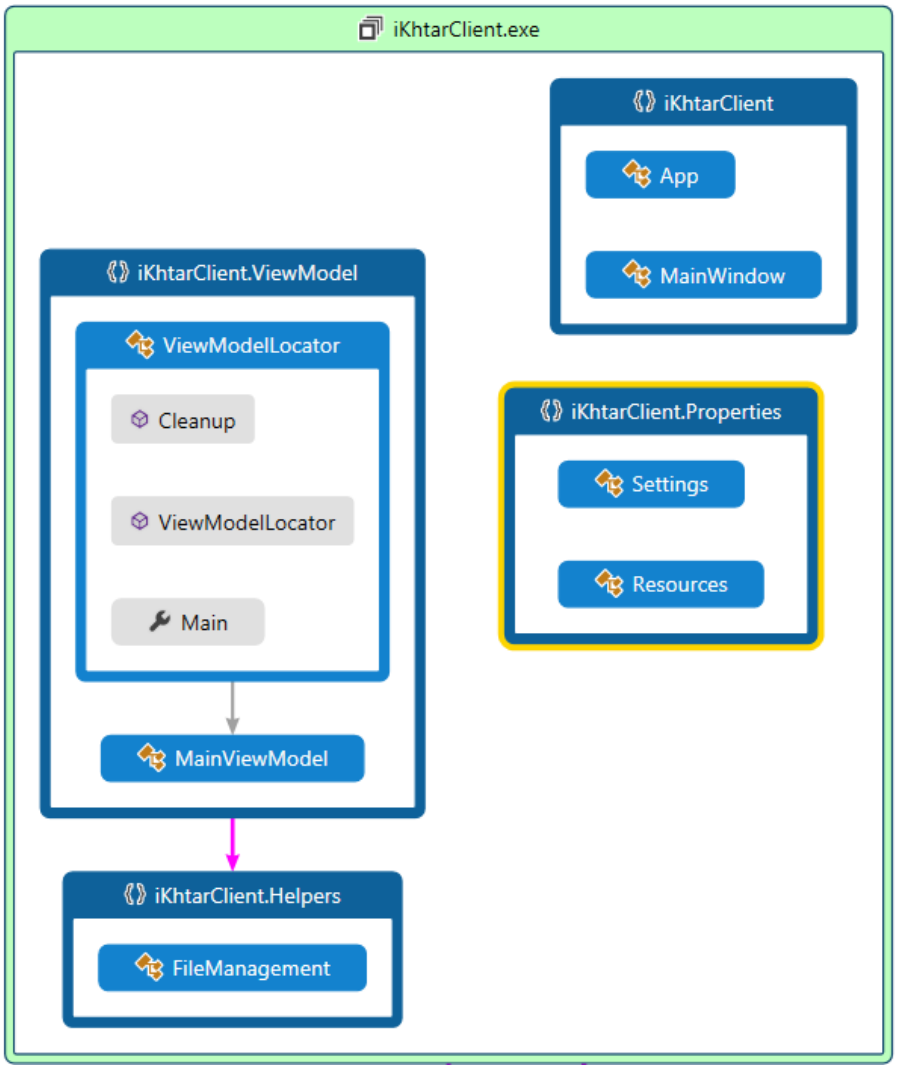

Figure 40 Code map du projet client en WPF

Les diagrammes de classes vont être représentés dans la partie suivante d'une façon abrégée car les NameSpaces et classes de MVVM Light sont connues et le diagramme sera encombré si on les intègre.

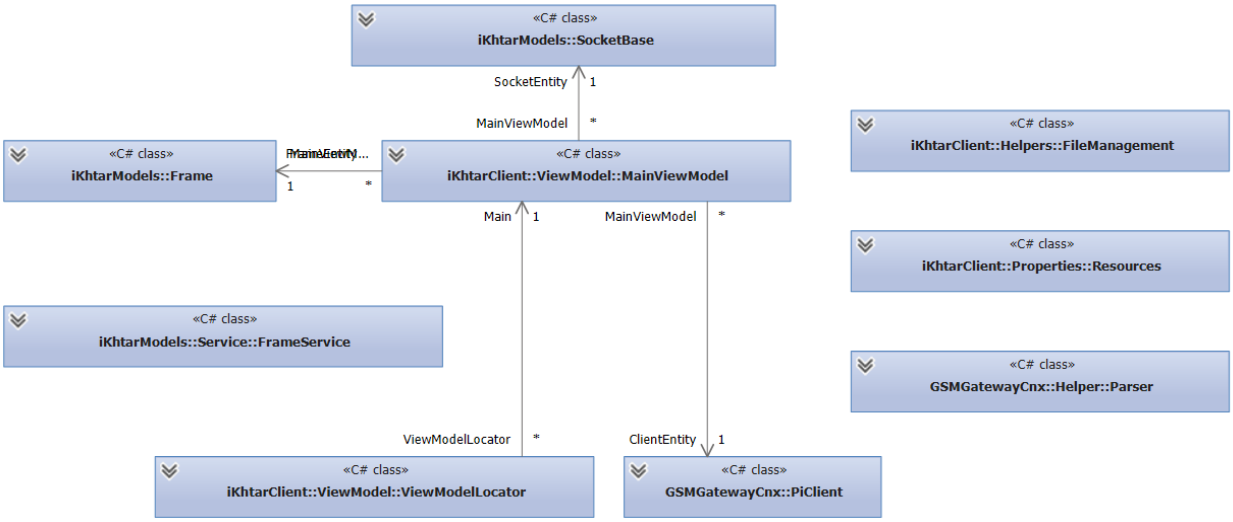

Figure 42 Diagramme de classe de la solution cliente

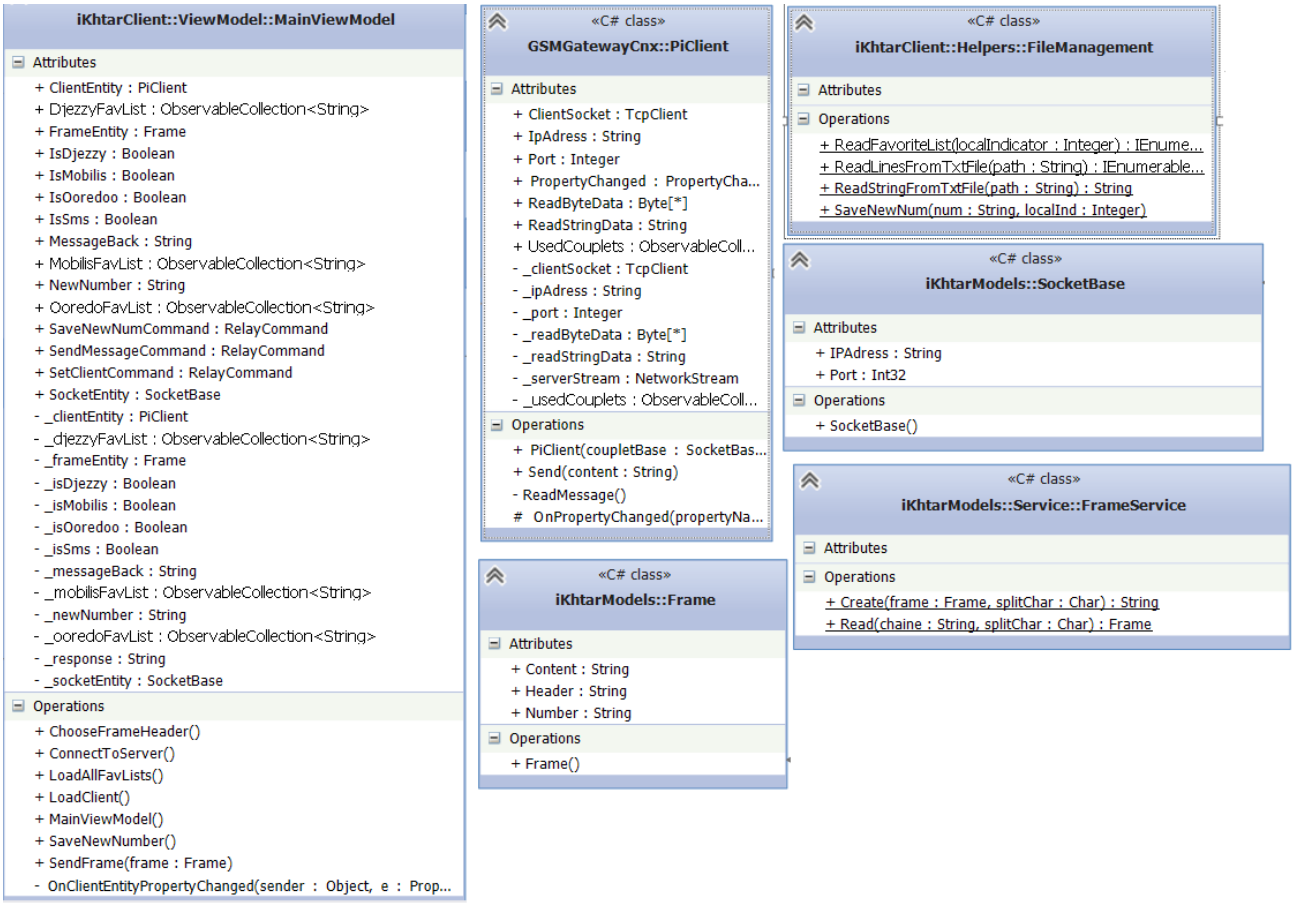

Figure 41 Diagramme des classes les plus importantes

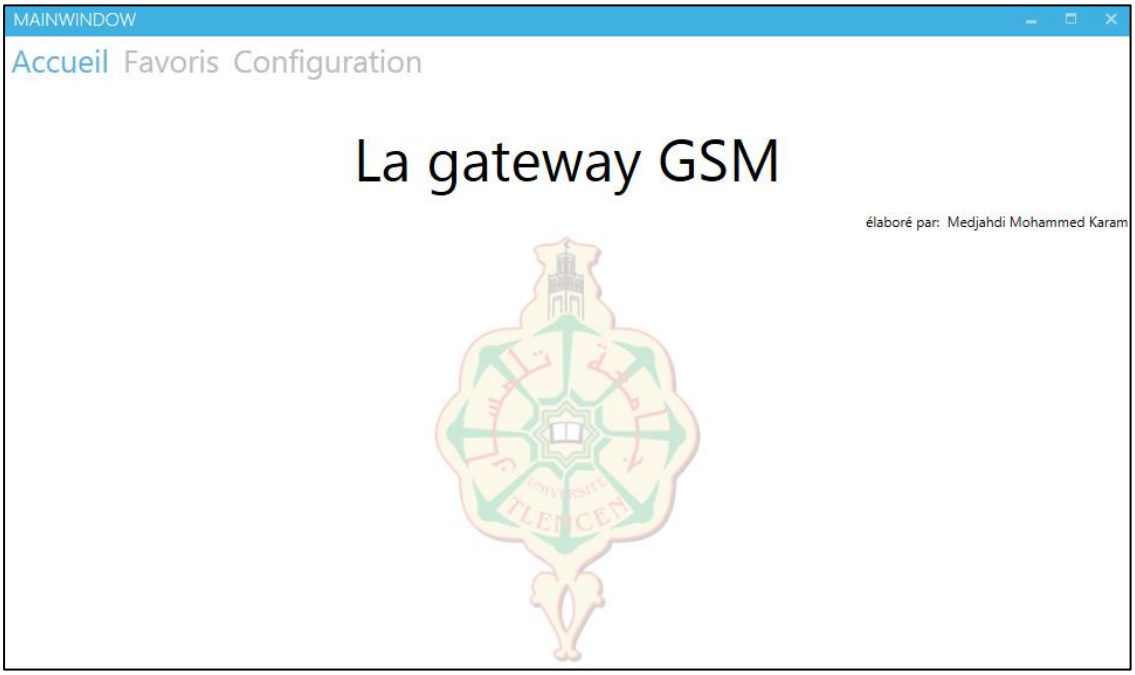

Figure 5. Ecran d'accueil de l'application cliente

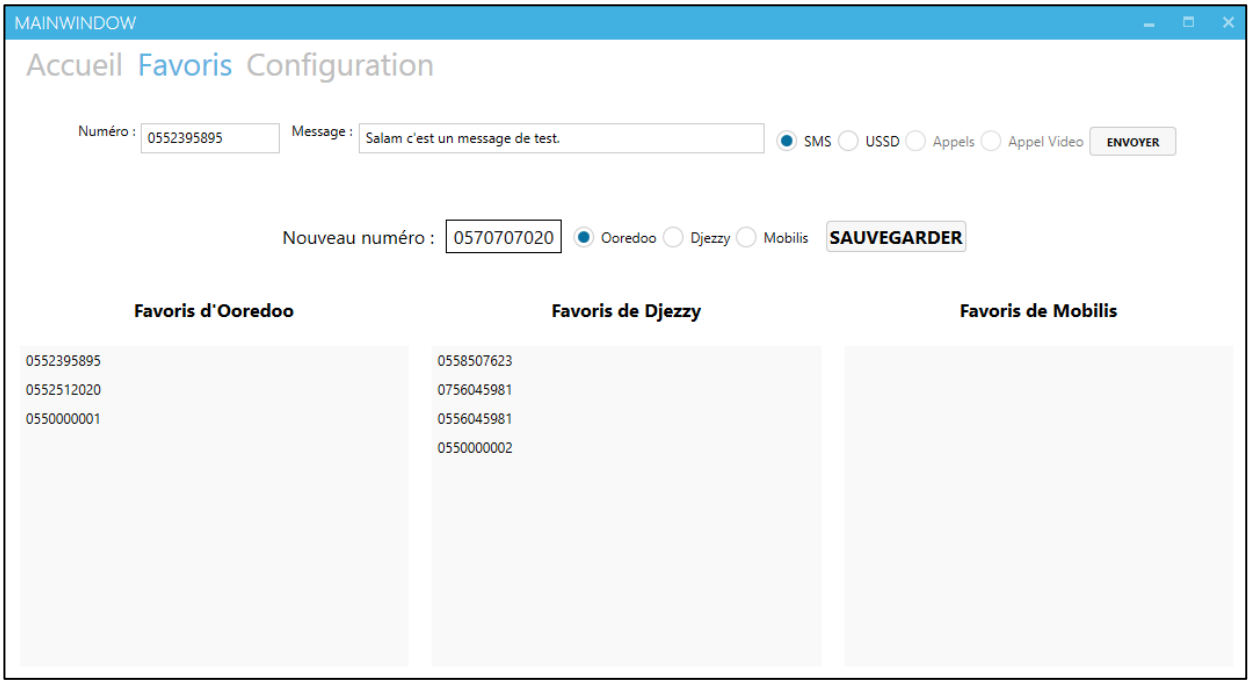

Figure 44 Ecran de services

7) Conclusion

On a vu dans ce chapitre l'ensemble de fonctionnalités gérant m'application serveur ainsi que les opportunités qu'elle offre pour exploiter la Gateway de façons différentes.

Cette Gateway qui est fabriquée à base des modules GSM 3G et exploité par un programme .NET donne la possibilité au client de l'exploiter à distance en ouvrant la communication réseau à de multiple clients. Ce type de communications donne la possibilité de développer des applications qui exploitent l'application cliente dans les différentes plateformes. Dans notre cas l'application WPF était le meilleur moyen pour démontrer la faisabilité des applications clientes qui exploitent la Gateway sur le réseau.

V) Chapitre 5 : Fiabilité et sécurité de la Gateway

1) Introduction

Dans les chapitres précédents tout ce qui concerne l'application, les protocoles de gestion, le hardware, la conception ont été détaillés mais une partie très importante dans ce genre d'applications n'a pas été prise en compte, il s'agit de la sécurité des données transmises. En effet, il est courant dans les réseaux de télécommunications d'assurer une intelligibilité des circuits en cours.

Dans ce chapitre on explique avec détail comment le système gérant la Gateway sécurise l'envoie des données et notre apport dans le concept de sécurité.

2) Problématique

De nos jours, les problèmes de sécurité sont multiples et varient d'un environnement à un autre et d'un concept à un autre. Mais dans notre cas les attaques pouvant être appliquées sur le système sont bien connus chez les hackers.

Puisqu'on est obligé de passer par un réseau on peut parler facilement du spoofing des données.

Si on laisse les messages passer en claire dans le réseau, les données peuvent être analysées et les hackers peuvent injecter des trames similaires à celle générées par l'application cliente et leurs donnent la possibilité d'envoyer des SMS à tort et à travers. Cela peut engendrer aussi le SMS Spamming.

Un autre type d'attaque peut être appliqué dans ce cas, c'est en changeant les SMS Header et imiter les providers de services téléphonique. Ils peuvent aussi passer des commandes AT pouvant compromettre ou bousier les modules 3G ou les cartes SIM installées.

Pour remédier à ces problèmes de sécurité on doit donc trouver un message pour empêcher les messages d'être envoyées en claire. La solution est donc de trouver un bon moyen de cryptage.

Ce chapitre détaille l'algorithme de cryptage et analyse les résultats après que le cryptage ait été appliqué.

3) L’algorithme de cryptage & décryptage

Tout au début on a pensé à des algorithmes de cryptages connus. On a besoin d’un algorithme fort et réversible à la fois, mais le problème de ces algorithmes c’est le faite qu’ils sont connus ce qui implique que les hackers peuvent faire des statistiques sur les messages envoyées dans le réseau et trouver le nom de l’algorithme appliqué. La solution donc est d’appliquer un algorithme personnalisé pour s’épargner d’éventuelles attaques.

Notre algorithme consiste à générer des clés qui permettent de changer le texte envoyé en un autre texte mais d’une façon dynamique. On s’inspirant du Protocol SSH qui envoie une clé à l’ouverture de la session et aux algorithmes de GSM (A3 et A5) qui sont présent dans la carte SIM et dans le réseau on a pu conclure une méthode de cryptage Hybride entre ces 2.

On introduit un algorithme de cryptage et de décryptage dans les applications client et serveurs et on génère dynamiquement des clés dans chaque message et non pas qu’au début de la session ce qui rend la reconnaissance des algorithmes difficilement reconnaissable.

Une vérification des chaines transmises est requise car des erreurs peuvent se produire au cours du cryptage ou de la transmission et c’est pour cela on a utilisé le MD5 Comme base de vérification et aussi comme base pour la génération de notre clé de cryptage.

L'algorithme MD5, pour Message Digest 5, est une fonction de hachage cryptographique qui permet d'obtenir l'empreinte numérique d'un fichier (on parle souvent de message).

Il génère des chaînes de caractères, sa taille est de 32 Octets quel que soit les paramètres d’entrée et sa particularité c’est qu’il est irréversible et donc solide comme algorithme pour générer d’autres clés qui seront réversible comme dans notre cas.

L’algorithme de cryptage sera détaillé grâce à l’organigramme suivant.

L’organigramme est présenté en 2 parties vu sa longueur.

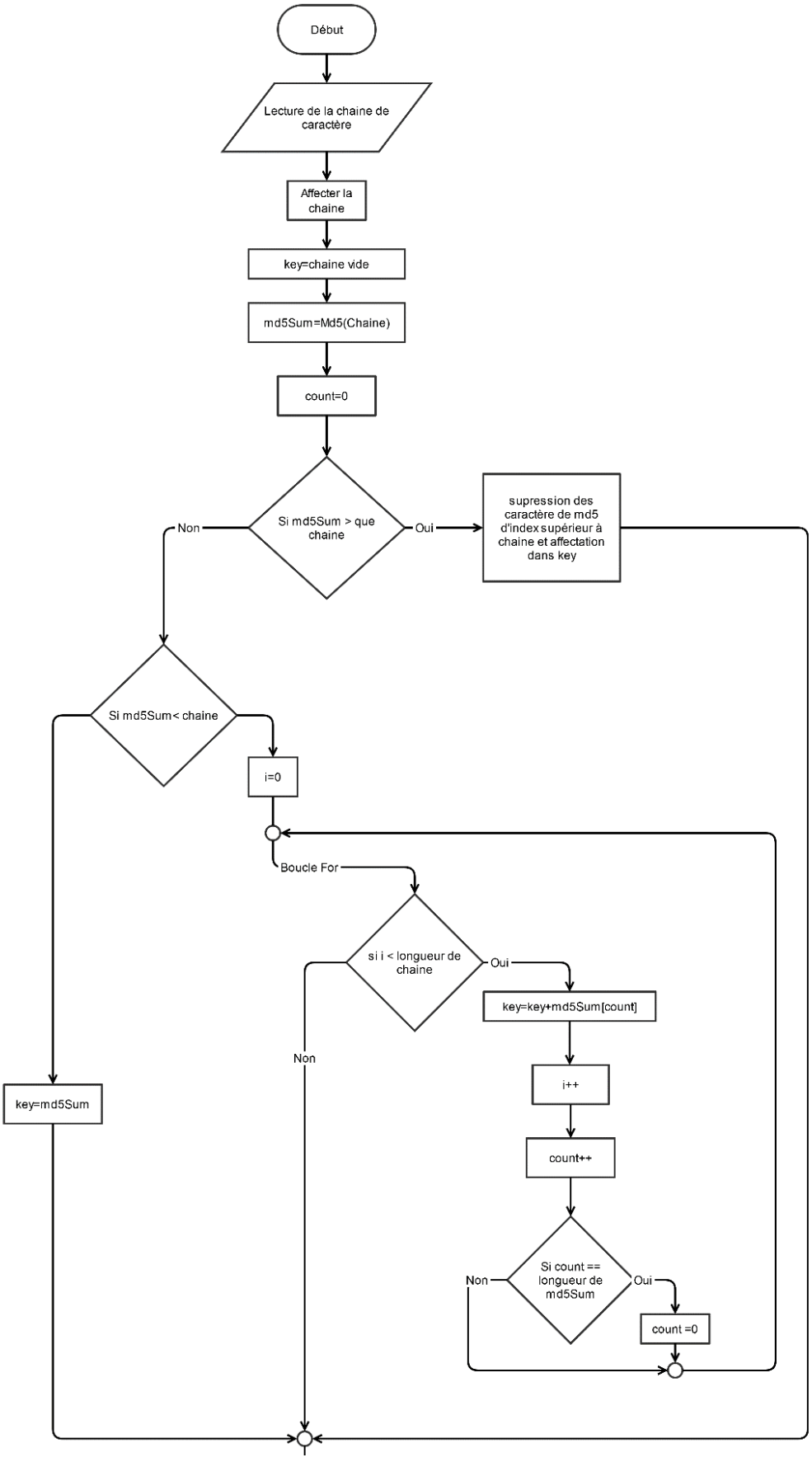

Figure 45 Algorithme de Cryptage (1ère partie)

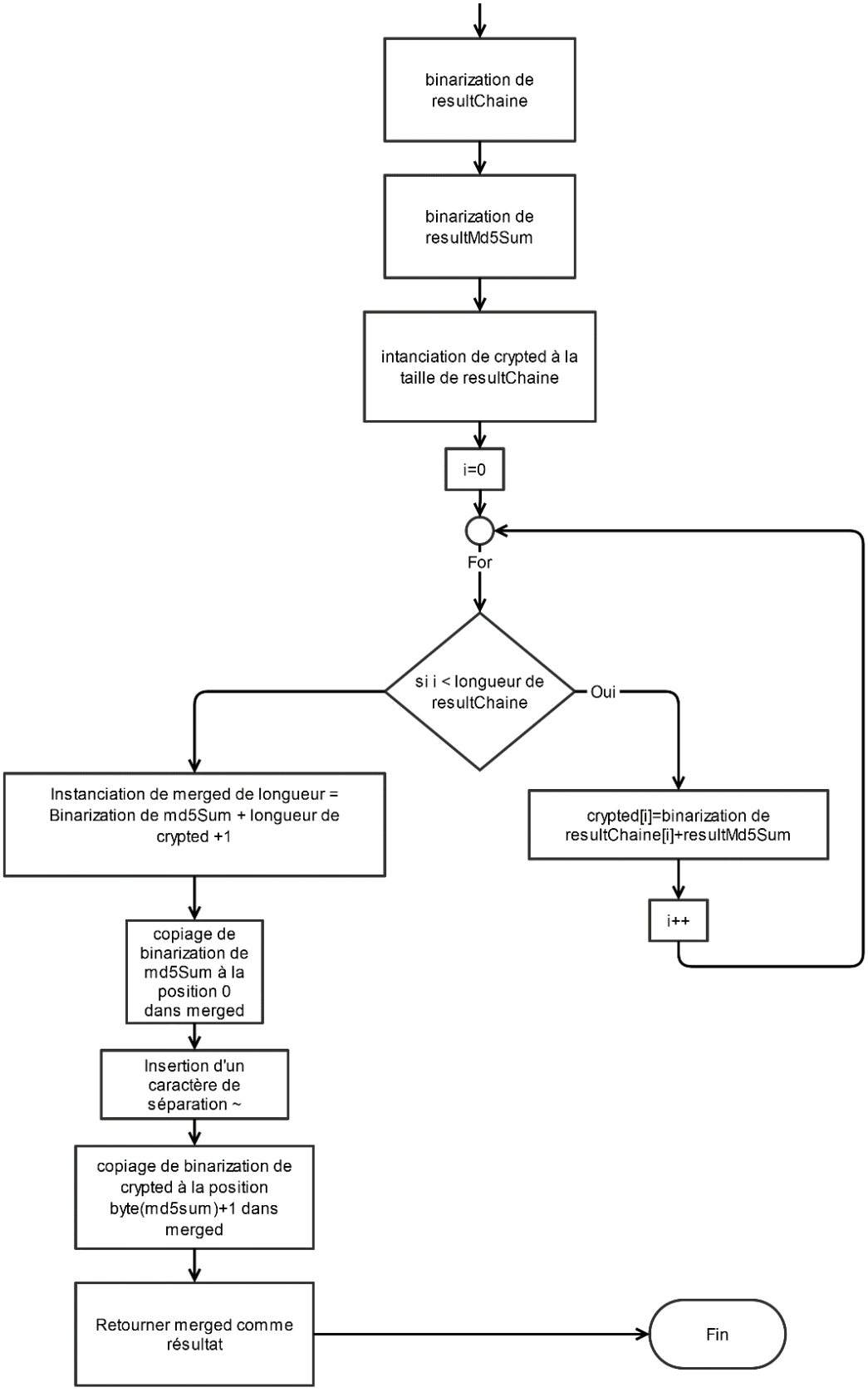

Figure 46 Algorithme de Cryptage (2ème partie)

On procède maintenant à une analyse pour vérifier l'effet du cryptage sur les messages.

On installe un logiciel de capture sur réseau appelé Wireshark. Et on lance le serveur puis le client en mode débogage. On tape un numéro donné et un message par exemple : « Salam. ».

Le résultat du break pointe affiche le contenu du message à envoyer avant le cryptage. Il est affiché dans la figure suivante.

Name	Value	Type
chaine	"SMS|0552395895|Salam."	string
md5Sum	null	string

Figure 48 Résultat du débogage avant le cryptage

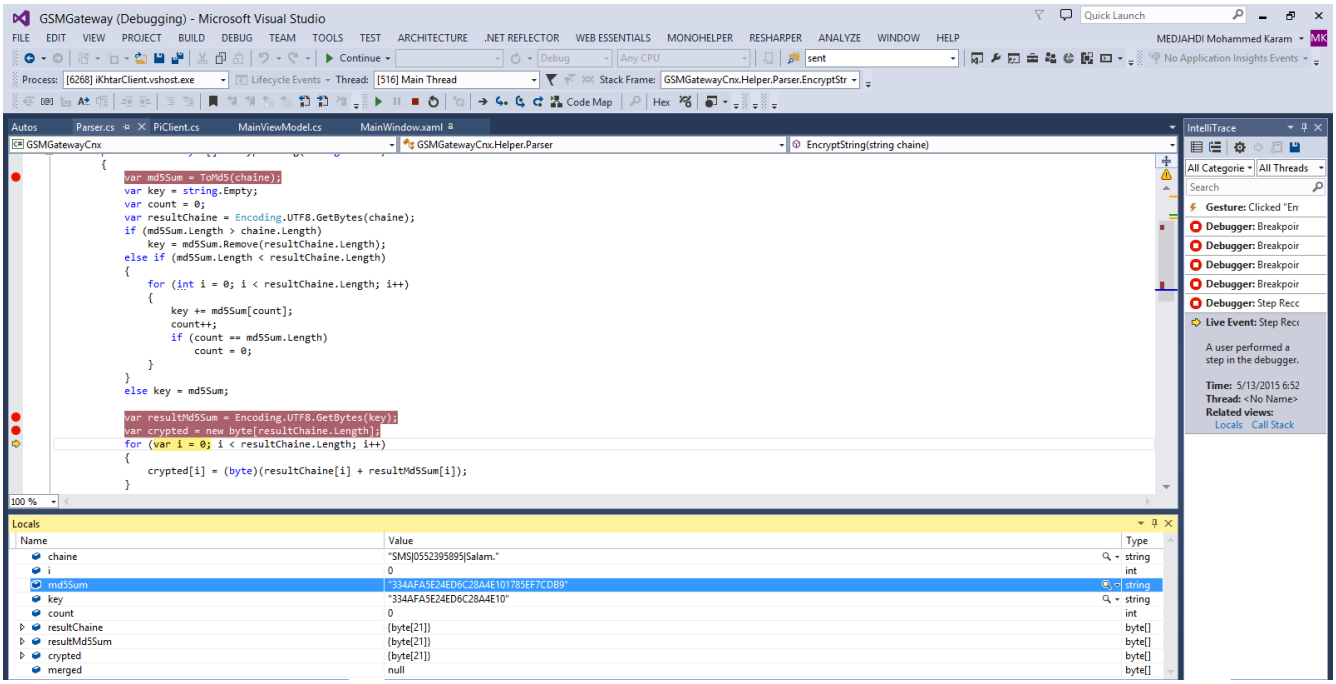

Figure 47 Débogage pendant le cryptage

La figure précédente illustre le résultat de la clé de cryptage appelée md5Sum elle est montrée en surbrillance dans le tableau du bas et la clé md5 dans la variable key.

Dans le tableau suivant vous trouvez le résultat du débogage du message globale après le cryptage (le contenue de la variable merged).

Index	Contenu de la case	Index	Contenu de la case
[0]	51	[27]	55
[1]	51	[28]	67
[2]	52	[29]	68
[3]	65	[30]	66
[4]	70	[31]	57
[5]	65	[32]	126
[6]	53	[33]	134
[7]	69	[34]	128
[8]	50	[35]	135
[9]	52	[36]	189
[10]	69	[37]	118
[11]	68	[38]	118
[12]	54	[39]	106
[13]	67	[40]	119
[14]	50	[41]	101
[15]	56	[42]	109
[16]	65	[43]	122
[17]	52	[44]	124
[18]	69	[45]	111
[19]	49	[46]	120
[20]	48	[47]	174
[21]	49	[48]	139
[22]	55	[49]	162
[23]	56	[50]	160
[24]	53	[51]	166
[25]	69	[52]	158
[26]	70	[53]	94

Figure 49 Contenu de « merged » contentant le message crypté

Après que le débogage de cette fonction se soit terminé l'algorithme continu son chemin d'exécution vers la fin qui se termine par l'envoi de ce tableau de Byte par réseau et c'est là qu'intervient le rôle de l'analyse réseau. On capture les trames sortantes depuis le pc qui possède l'application cliente qui envoi l'SMS vers la Raspberry et on filtre que les trames passantes vers l'adresse IP de ce dernier. Dans la figure suivante se trouve une capture d'écran de Wireshark : on peut voir clairement le filtrage de l'adresse en haut à gauche (l'IP 192.168.1.25).

Et les trames envoyées. Puis on visualise le contenue en bas. Elle sera détaillée dans la figure qui suit.

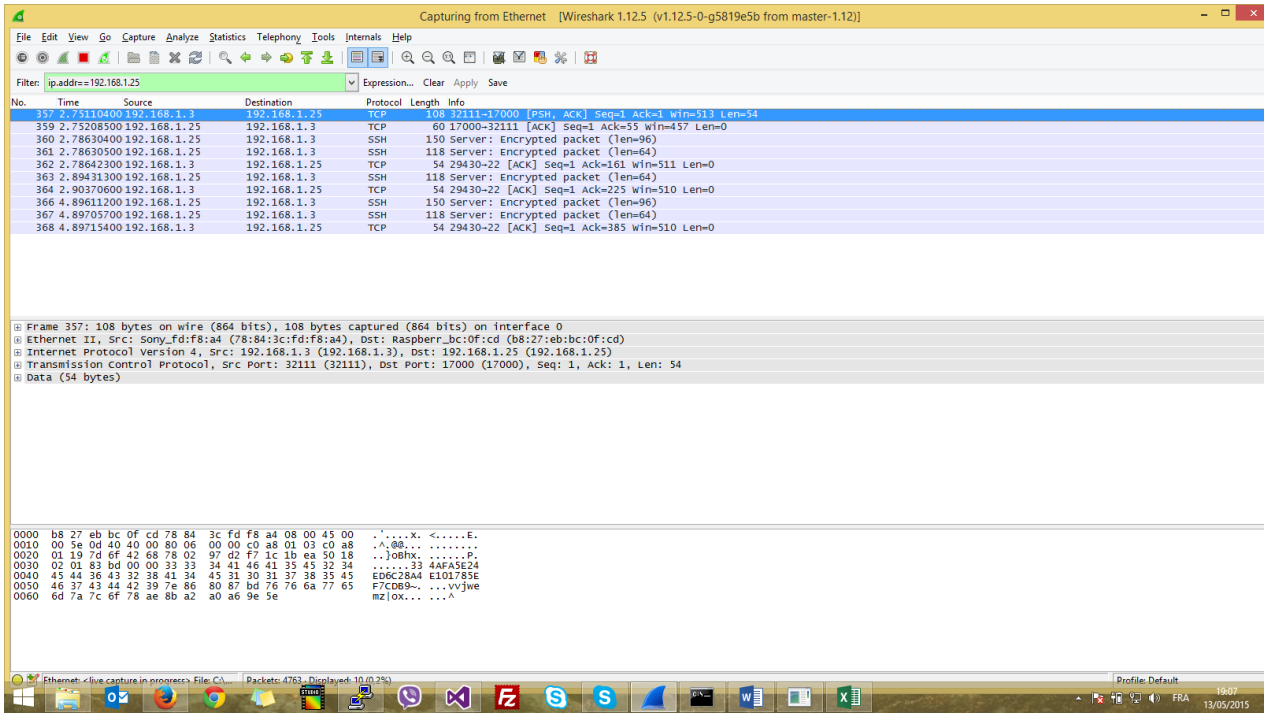

Figure 50 Capture d'écran de Wireshark lors l'envoi du message vers la Gateway

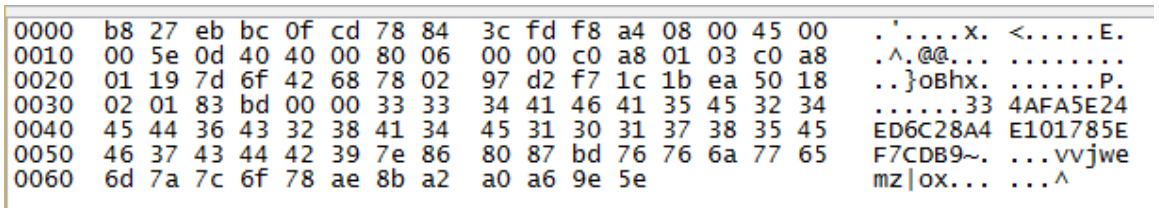

Figure 51 Capture d'une trame envoyée.

Dans la figure précédente on voit clairement le message envoyé, seul la clé MD5 qui est envoyée en claire avec le tilde de séparation (334AFA5E24ED6C28A4E101785ED7CDB9~) le reste du message n'est pas une chaine de caractère claire elle a été cryptée selon la clé de cryptage précédente et ne peut en en aucun cas être lu sans l'algorithme de décryptage.

Donc le cryptage a été effectué avec succès et le message a été acheminé par la Raspberry vers le numéro 0552395895 et lu correctement par le téléphone mobile.

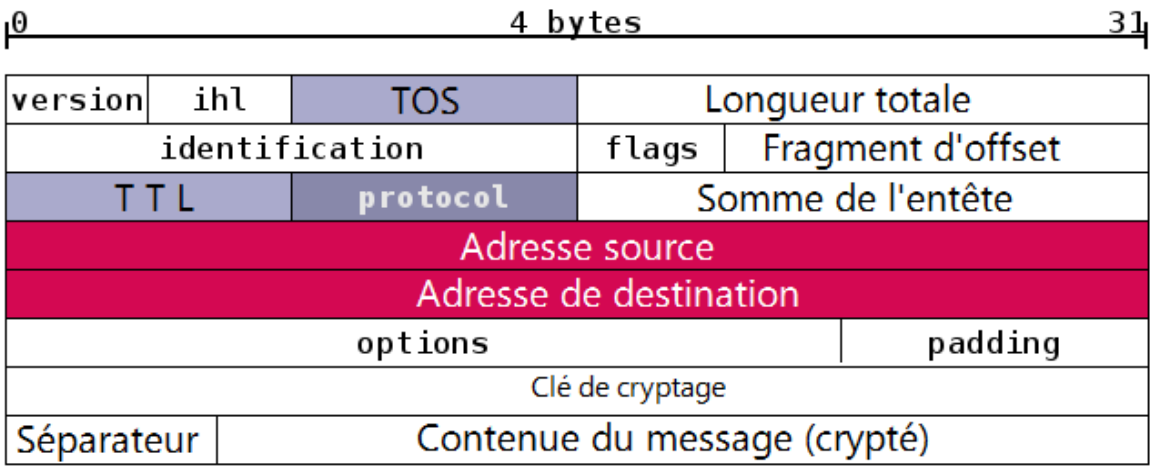

Figure 52 Modèle d'un paquet complet envoyé sur le réseau

La figure précédente représente un modèle d'un paquet envoyé dans le réseau, le message envoyé par le client se présente dans les deux dernières lignes. Sachant que le contenu du message est aussi encapsulé et cela a été détaillé dans le chapitre passé.

4) Conclusion

Ce qu'on peut conclure de ce chapitre c'est l'importance du cryptage lors d'envoie des messages et dans ce travail on a implémenté notre propre algorithmes de cryptage ce qui renforce la sécurité de la Gateway GSM et élimine une très grande possibilité d'attaques pouvant être appliquées dans le réseau, et cela a été vérifié en faisant l'analyse réseau en capturant les trames sortantes du client. Seule la clé de cryptage est envoyée en claire.

La force du cryptage utilisé c'est l'aspect dynamique car la clé change dans chaque message envoyé ce qui rend aussi les attaque de brute force très difficiles.

Conclusion Générale

Durant la réalisation de ce projet de fin d'étude on a rencontré plusieurs problèmes liés essentiellement aux ressources matériels et logiciels à la fois.

En segmentant les problèmes rencontrés on s'aperçoit qu'on distingue en premier plan les problèmes hardwares (Matériel), cela concerne principalement les modules GSM car la Gateway Gsm conventionnel doit être réalisée à base des modules GSM mais au cours de la réalisation du circuit on a trouvé du mal à trouver des modules GSM spécifique, d'où la solution des modems 3G qui a été détaillé dans ce mémoire. L'avantage de ces modules GSM, c'est qu'ils offrent plusieurs fonctionnalités qui restent indisponibles dans les clés 3G.

Le deuxième problème rencontré dans ce projet est l'indisponibilité de certains types de multiplexeurs. Sans ces derniers la solution électronique a été évitée, car le Protocol USB est un Protocol qui nécessite une connectivité quasi-permanente avec l'ordinateur ou la carte intelligente et le temps de coupure des données est critique ce qui nécessite des multiplexeurs à temps de transition très réduit de l'ordre de picoseconde en utilisant une cartes intelligente comme la Raspberry pi on passe directement vers le montage des clés 3G entant qu'interface de communication directement liée au système d'exploitation utilisé.

Après avoir choisi le développement sur une carte intelligente beaucoup de problèmes sont apparu et le problème majeur était le montage des clés, les clés disponible dans le marché Algérien ont été principalement importées par des opérateurs téléphoniques pour des buts commerciales, et pour ces mêmes buts ils ont été verrouillé de façon à ne pas pouvoir connecter plusieurs clés dans un même appareil que ce soit un ordinateur ou une carte intelligente.

Sachant que pour communiquer avec USB une carte intelligente telle que la Raspberry pi dotée d'une distribution linux émule le driver Windows de la clé pour pouvoir communiquer en mode console série. En ce qui concerne Windows le système d'exploitation copie le driver dans un répertoire dans le système puis détecte la clé à nouveau pour récupérer son VID et son PID puis associe dans le registre à ce couplet d'identifiants le driver qu'il faut et enregistre cette association dans la base des registres.

Linux émule ce fonctionnement grâce à de multiples outils, on cite principalement usbmodeswitch.

Et comme c'est noté au préalable les fabricants des clés 3G pour des buts commerciaux verrouillent les clés en laissant le même couplet d'identifiants pour toutes les clés ce qui bloque l'OS lors de l'association du driver.

Tous ces problèmes ont été rencontrés sans parler des problèmes liés à la qualité des clés trouvées et la compatibilité avec les différents réseaux téléphoniques.

La méthodologie suivie lors de la réalisation de ce projet a permis de décortiquer ces problèmes et procéder à des solutions orientées software.

Références et bibliographie

http://fr.wikipedia.org

http://en.wikipedia.org

http://www.statista.com

http://www.tutorialspoint.com/gsm

https://msdn.microsoft.com/

http://csharp-source.net/

http://www.paul-mcgee.me.uk

https://www.thefanclub.co.za

http://www.nowsms.com

http://www.voip4business.biz

https://github.com/giessweinapps/MonoDebugger

http://www.mono-project.com/docs/

http://csharp.2000things.com/

The Global System for Mobile Communication. J. Scourias . University of Waterloo

Le réseau GSM. Anne WEI. CNAM Paris

Cours Master 2 sur les réseaus GSM . Bahri Sidi Mohammed. Université de Tlemcen

Principes de base du fonctionnement du réseau GSM. Demoulin Cédric

Liste des figure

Résumé

Ce travail traite le fondement des Gateway GSM, et leurs applications dans les réseaux d'une entreprise. La conception d'une Gateway intelligente est proposée en exploitant une carte Raspberry PI connectée avec des modems GSM 3G. La solution proposée est particulièrement bien adaptée au contexte local. En effet, la Gateway conçue nécessite de très faibles investissements. Un puissant logiciel embarqué sur la Raspberry PI est proposé en exploitant la technologie mono et .Net. Un client qui peut être utilisé sur différente plateforme, développé en technologie .Net, a également été réalisé. L'aspect sécurité a été pris en compte en réalisant une solution de cryptage propre à la conception de la Gateway GSM.

Sumarry

This project is an attempt build a GSM Gateway while discussing its applications in the world of IT networks. This smart GSM Gateway was designed using the single board computer called the Raspberry Pi which have at least two 3G GSM modems connected to it via its USB interfaces. To help the local market and enrich the code repertoire compatible with local service providers, we adapted this solution to the needs of the local mobile network operators. One of the attractive aspects of our design is the low cost of this device; add to that the robust embedded software that we conceived on the Raspberry Pi using the C# Compiler and Mono Project, running on top of the latest .NET technology. To assure a maximum compatibility and the possibility of cross-platform porting, the client side application was also made around the .NET technology. In addition, for a secure data exchange, all outgoing or ingoing information is encrypted with a protocol that implements a new kind of encryption algorithms made especially for this project.

ملخص

هذا المشروع يهدف إلى صناعة جهاز يعمل كمعبر جي اس ام وتجسيد استعماله لدى مختلف الشركات والمؤسسات. تصميم المعبر المقترح في مشروعنا هذا قائم على البطاقة الذكية المسمات راسب بيري بي حيث تتصل هذه الأخيرة بأجهزة موديم من الجيل الثالث. العمل المقدم يتماشى ومتطلبات السوق المحلية، ولهذا السبب أنشأ الجهاز المقترح بأثمان ليست بالباهظة البتة ولا تتطلب استثمار مادي كبير. والبرمجيات الملحقة بالجهاز جد فعالة وسلسة من ناحية الإستخدام أو إعادة الإستعمال وقائمة على العديد من التقنيات أهمها الدوت نت ومشروع المونو ويستغل الملقم أو السيرفر عن طريق برنامج يلعب دور الزبون. هذا البرنامج المنشأ بنفس التقنيات يدعم معظم المنصات بما فيها المحمولة. أما من ناحية الحماية فالبيانات التي يتم معالجتها من طرف تطبيقات المشروع فهي مؤمّنة ومشفّرة عن طريق خوارزميات صممناها خصيصا لهذا المشروع.

Table des matières