Université Sidi Mohamed Ben Abdellah

Faculté des Sciences et Techniques de Fès

Département de Génie Industriel

Mémoire de Projet de fin d'études

Préparé par

KRIROU Jilali

Pour l'obtention du diplôme d'Ingénieur d'Etat

Spécialité : Ingénierie en Mécatronique

Intitulé

Etude et intégration de protocoles de communication entre les objets connectés et les terminaux mobiles au sein de plateforme

Lieu : Media Mobility

Réf : 21/IMT15

Soutenu le 02 Juillet 2015 devant le jury :

- Pr Nabih ElOUAZZANI (Encadrant FST)
- Mr Mehdi ALAOUI (Encadrant Media Mobility)
- Pr Belmajdoub (Examinateur)
- Pr Tahri (Examinateur)

وَقُلِ ٱعۡمَلُواْ فَسَيَرَى ٱللَّهُ عَمَلَكُمۡ وَرَسُولُهُۥ وَٱلۡمُؤۡمِنُونَۖ وَسَتُرَدُّونَ إِلَىٰ
عَٰلِمِ ٱلۡغَيۡبِ وَٱلشَّهَٰدَةِ فَيُنَبِّئُكُم بِمَا كُنتُمۡ تَعۡمَلُونَ ١٠٥

صَدَقَ اللهُ العَظِيمُ

Dédicace

A mes très chers parents

Votre confiance et vos encouragements ont été pour moi la première source de persévérance. Que vous trouvez dans ces modestes mots le témoignage de ma gratitude et ma sincère appréciation.

A mes très chers frères et sœurs

Vous tenez une place immense dans mon cœur.

A toute ma famille

Recevez ici le témoignage de ma gratitude et de mon profond attachement.

A mes chers amis

Merci pour l'amitié, pour votre écoute, gentillesse et altruisme. Les moments que nous avons

passé ensemble resteront un agréable souvenir.

jilali

Remerciements

Au terme de ce travail, j'ai le plaisir d'exprimer mes sincères remerciements à tout le personnel de **Media Mobility** qui m'ont permis de passer mon stage de fin d'études dans de bonnes conditions.

Mes remerciements vont particulièrement à **Mr. ALAOUI Mehdi** mon encadrent au sein de Media Mobility ainsi que **Mr. Nabih Elouazzani** mon encadrant pédagogique pour leurs conseils, leur suivi et leur grande contribution durant les différentes phases de la réalisation du projet.

Je tiens à présenter mes remerciements aux professeurs et responsables de la FST Fès pour l'effort continu qu'ils fournissent afin d'assurer une bonne formation aux élèves ingénieurs.

Je remercie aussi toute personne ayant contribué de près ou de loin à la réalisation de ce projet, en espérant que mon travail sera à la hauteur de vos attentes.

Liste des abréviations

Abréviation	Désignation
APK	Android application package
CRM	Customer Relationship Management.
CE	Embedded compact
CRC	Cyclic redundancy check
IOT	Internet of things
IOS	IPhone operating system
ISM	Industriel, Scientific and medical
OS	Operating system
PIR	Capteur passif d'infrarouge
PSK	Phase shift keying
RIM	Research in motion
RF	Radio frequency
SCCB	Serial camera control bus
USART	Le récepteur et transmetteur universel synchrone et asynchrone
WYSIWYG	What You See Is What You Get
1-Wire	Le unique simple fil interface mode

Table des figures

Liste des tableaux

Table des matières

Introduction générale

Actuellement, les systèmes d'exploitation (OS) sur lesquels reposent les applications mobiles sont nombreux, on en compte plus d'une centaine. Ces systèmes nécessitent de multiplier le développement sur chacun d'entre eux, car chaque système d'exploitation dispose de ses propres spécificités et comporte déjà de nombreuses versions.

Avec l'évolution des technologies, les objets connectés ont vu le jour et ne cessent d'évoluer jusqu'à présent. Ce marché, qui comptait 4 milliards de "objets" en 2010, devrait atteindre 80 milliards en 2020 dont 85 % seront des objets connectés.
Cette croissance exponentielle s'explique par l'avènement des terminaux mobiles, utilisables comme écrans de contrôle déportés pour les objets connectés, par la démocratisation de l'accès à Internet en haut débit et, enfin, par le développement des technologies « Big Data » pour tirer pleinement partie des données collectées.
ScreenDy demeure comme une solution innovante qui permet de répondre au problème de la multiplicité des systèmes d'exploitation ainsi, d'une part, elle augmente la productivité en simplifiant la création des applications mobiles, d'autre part, elle facilite le déploiement de ces applications sur divers technologies en compatibilité avec téléphone et objet connecté tout en apportant une garantie sur la pérennité des développements.

Dans cette vision s'inscrit le projet de fin d'études, qui consiste à mettre en place un noyau de la solution ScreenDy destiné aux objets connectés pour faciliter la communication entre les terminaux mobiles et les objets communicants via plusieurs protocoles.

Dans ce rapport, nous avons développé les différentes étapes du travail. Ainsi, le premier chapitre sera consacré au contexte général du projet regroupant une présentation de l'organisme d'accueil ainsi qu'une présentation du projet. Ensuite, le deuxième chapitre sera consacré à la technologie des objets connectés. Quant au troisième chapitre une étude technique pour la réalisation des prototypes de communication entre les Smartphones et les objets connectés. Le chapitre quatre représente une conception détaillée des objets de communication pour intégrer au studio ScreenDy. Avant de conclure, un cinquième chapitre sera dédié à la présentation d'une application type pour tester le bon fonctionnement.

Chapitre 1

Contexte général du projet

Dans la première section de ce chapitre, nous donnons un bref aperçu sur l'organisme d'accueil, son organisation et son domaine d'activité. La deuxième section sera consacrée à la définition du projet, ses objectifs ainsi que la démarche adoptée pour le mener à bien.

1. Contexte général du projet

1.1. Présentation de l'organisme d'accueil

Media-Mobility est un éditeur d'applications et de solutions innovantes sur le mobile. Depuis sa création en 2007, la société se positionne sur l'ensemble de la chaîne de création de valeur du multimédia mobile et le déploiement multiscreen : téléphone, tablette, smart car, TV connectée et objet connecté. Le siège social de l'entreprise se situe à Paris compte 50 personnes avec des bureaux à Dubaï, Casablanca et à San Francisco. [1]

1.1.1. Fiche signalétique

La fiche signalétique de Media Mobility est présentée dans le tableau suivant :

Raison sociale	**Media Mobility**
Forme juridique	S.A.R.L
Année de création	2007
Adresse	6, rue Caïd El Achtar - ex Girondins, Maarif - 20370 Casablanca– Maroc
Téléphone /Fax	Tel : +212 (0)5 22 252 449
Directeur	Mehdi Alaoui Hassani
Effectif	Entre 10 et 20
Chiffre d'affaires	500 000 euros
Activités	Solutions & Services mobiles

Tableau 1: fiche signalétique de Media Mobility

1.1.2. Organigramme de l'organisme

Media Mobility s'appuie sur son équipe interne de consultants qualifiés ainsi qu'un réseau de partenaires-expert nationaux et internationaux. L'organigramme ci-dessous présente les différentes directions de Media Mobility. [1]

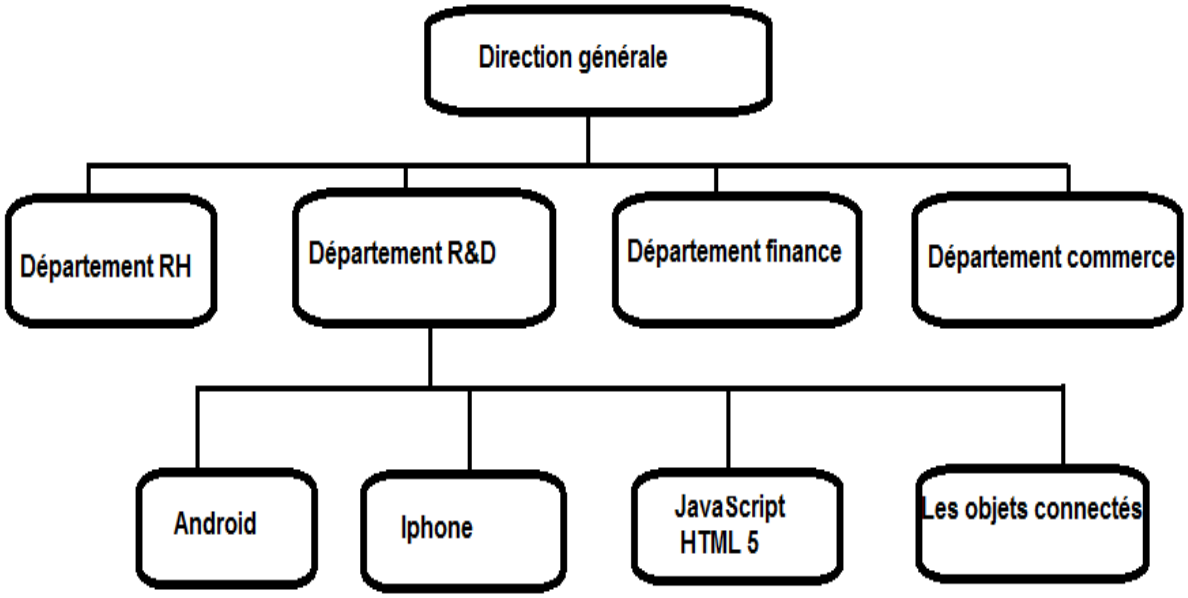

Figure 1: Organigramme de Media Mobility

Le projet a été développé au sein du service recherche et développement, qui a pour mission de suivre l'évolution du marché et de trouver des solutions innovantes.

.

1.2. Projet ScreenDy

1.2.1. Présentation de projet

Chaque année de nouveaux objets apparaissent avec différents OS, la fragmentation du marché continue à augmenter, cela complique la tâche aussi bien pour les développeurs que pour les donneurs d'ordres. C'est dans cette optique que la solution informatique ScreenDy a été proposée par la société Media Mobility. ScreenDy dispose d'un noyau qui implémente les fonctionnalités natives pour chaque OS majeur (figure 2), et met à la disposition des développeurs et des chercheurs un Kit de développements pour étendre le nombre d'objets et de systèmes d'exploitation compatibles.

ScreenDy permet de raccourcir les délais de création d'applications mobiles et d'assurer un déploiement sur tous les écrans : téléphone, tablettes, Tv Connectée, Desktops et les objets connectés. Cette solution dispose de son propre atelier de génie logiciel permettant de créer rapidement tout type d'applications mobiles de dernière génération. Les applications peuvent être entièrement natives, hybrides, mixtes ou web mobiles.

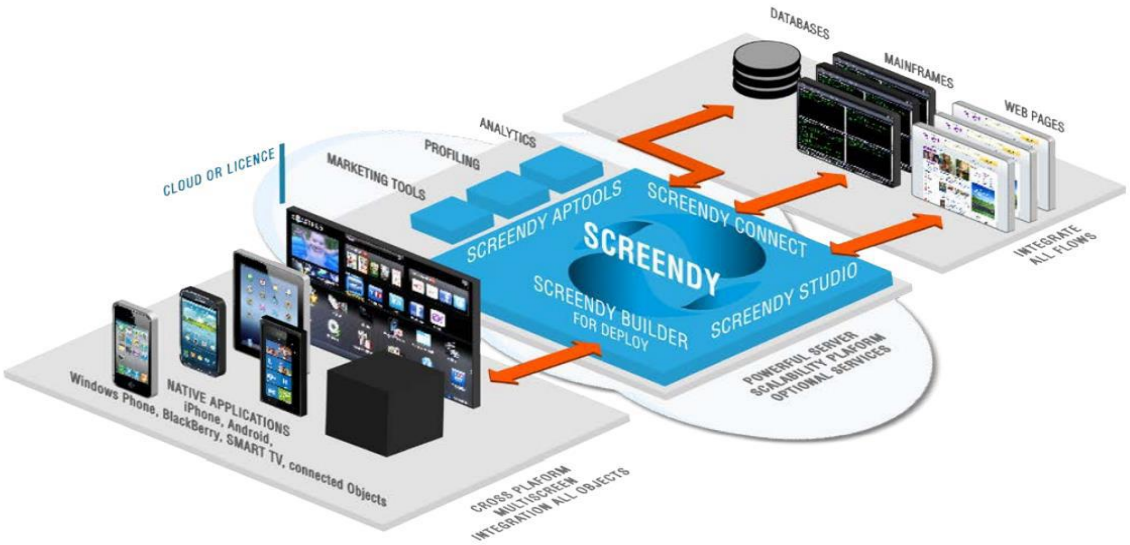

Figure 2: Solution Screendy

Cette technologie innovante, assure la portabilité des applications sur des écrans en fonction de la plateforme déployée, minimise les coûts et le temps de développement multiples.

L'application une fois développé, elle permet d'obtenir un investissement plus important.
Avec la solution ScreenDy, Media Mobility accompagne le débutant et le développeur dans leurs stratégies mobiles en prenant en charge la totalité du processus de conception, de création et de déploiement de leur application. Le CRM (Customer Relationship Management) directement intégré sur la plateforme permet de gérer en temps réel les utilisateurs et de déployer tous les outils du marketing mobile pour monétiser efficacement les applications. Avec des solutions clés en mains ou sur mesure, nous pouvons ajouter de nouvelles extensions et de nouveaux services aux applications à tout moment. [2]

1.2.2. Multiplicité des systèmes d'exploitation mobiles

Le marché du mobile a énormément évolué depuis le début de l'année 2012. Les entreprises Apple, Google, Windows et Nokia sont en compétition constante pour récupérer des parts de marché : Android rattrape et dépasse dans certains pays IOS, et Windows Phone commence à se développer. Le marché des OS sur mobile est donc particulièrement dynamique.

Ainsi, les OS se multiplient, en conséquence, les développeurs doivent faire face à un nombre croissant d'environnements de développement à maîtriser. Afin de réduire les délais et donc les coûts de développement, des plateformes de développement multi-OS (appelées cross-Platform) ont commencé à voir le jour. Néanmoins, ce n'est pas pour autant qu'elles ne nécessitent aucune compétence de développement : elles requièrent aussi un temps d'adaptation pour les développeurs.

Cette explosion des plateformes de développement sur mobile est donc un problème compliqué pour les entreprises, qui doivent faire preuve d'adaptation.

Ci-dessous se trouve une rapide description des systèmes d'exploitation mobiles :

1.2.2.1. Android

Android est un système d'exploitation Open Source pour terminaux mobiles conçu par Android, une startup rachetée par Google en juillet 2005. Cet OS se différencie principalement de ses concurrents par le fait qu'il est libre. Le modèle économique de Google semble très pertinent, l'adoption d'Android par les fabricants sera probablement rapide du fait de la gratuité d'utilisation pour le constructeur.

1.2.2.2. Symbian OS

Le Symbian OS est développé par la société éponyme qui est une propriété exclusive de Nokia. Bien que cette plateforme soit crée par la participation de plusieurs fabricants tels que Samsung ou Sony Ericsson, ce système est fortement connoté Nokia, ce qui est un frein à son adoption par d'autres constructeurs. Il est récemment passé en open source. C'est un système libre, open source se base sur un noyau Symbian.

1.2.2.3. Ios

IOS (*Internetwork Operating System*), qui était nommé iPhone OS, se trouve non seulement sur les différents générations de iPhone mais également sur d'autres produits de Apple iPad et iPod touch. Il est dérivé de Mac OS X dont il partage les fondations : kernel, les services Unix et Cocoa. Pour Apple, le succès est considérable : début 2009, il n'y avait pas moins de 5 millions de téléchargements par jour. Donc, il s'agit du concurrent numéro un pour Android.

1.2.2.4. BlackBerry OS

Le système d'exploitation BlackBerry est la plate-forme exclusive mobile développé par RIM (*Research In Motion*) exclusivement pour ses Smartphones BlackBerry et les appareils mobiles. RIM utilise ce système d'exploitation pour soutenir des fonctions spécialisées, notamment l'écran tactile.

1.2.2.5. Windows Phone

Windows Mobile est l'OS (système d'exploitation) mobile de Microsoft. C'est une évolution de Windows Pocket PC, ancêtre de Windows CE (embedded compact). Cet OS a réussi au fil des années à s'octroyer une part de marché honorable. Son succès est dû à son

affiliation à la famille d'OS Windows, ultra-dominante sur le bureau. Un autre avantage souvent cité est la facilité de développement apportée grâce à l'environnement cliquodrome de Visual Studio qui a su faire venir au développement mobile les développeurs VB (*Visual Basic*).

1.2.2.6. Palm webOS

Il y a quelques années, Palm a même cédé à Windows Mobile en proposant certains de ses appareils sous l'OS de Microsoft.

1.2.3. Objectif du projet

Notre contribution dans ce projet consiste à concevoir et à réaliser une solution permettant de développer des applications pour les objets connectés grâce à la solution ScreenDy.

En effet cette solution permet de créer des applications mobiles déployées sur toutes les plateformes d'une manière facile et simple. D'où la nécessité d'implémenter un noyau de la solution ScreenDy destiné aux applications des objets connectés.

On pourrait résumer la finalité de notre travail en termes d'objectifs suivants :

- ✓ Étudier et se familiariser avec le noyau destiné pour le téléphone de la solution ScreenDy.
- ✓ Développer des prototypes qui permettent de relier un Smartphone et un objet connecté.
- ✓ Concevoir une solution qui répond le mieux à la génération des applications pour les objets connectés.
- ✓ Implémenter le noyau réservé pour les objets connectés.

1.3. Conduite de projet

1.3.1. Ingénierie de système

L'ingénierie de système est une approche scientifique interdisciplinaire de formation récente, dont le but est de formaliser et d'appréhender la conception de systèmes complexes avec succès.

Les efforts en ingénierie des systèmes embrassent l'ensemble du cycle de vie du système et leur mise en cohérence et mobilisent l'ensemble des corpus théoriques (sciences de l'ingénieur, sciences humaines, sciences cognitives, génie logiciel, etc.)

Elle se focalise sur la définition des besoins du client et des exigences fonctionnelles, détectés tôt dans le cycle de vie, en documentant les exigences, puis en poursuivant avec la synthèse de la conception et la validation du système.

1.3.2. La mise en œuvre de l'ingénierie de système

L'approche d'ingénierie de systèmes fournit un langage commun aux diverses disciplines utilisées parallèlement les unes aux autres. Une vue système assure, dès le début du processus, que les comportements physiques et logiques du cahier des charges seront respectés

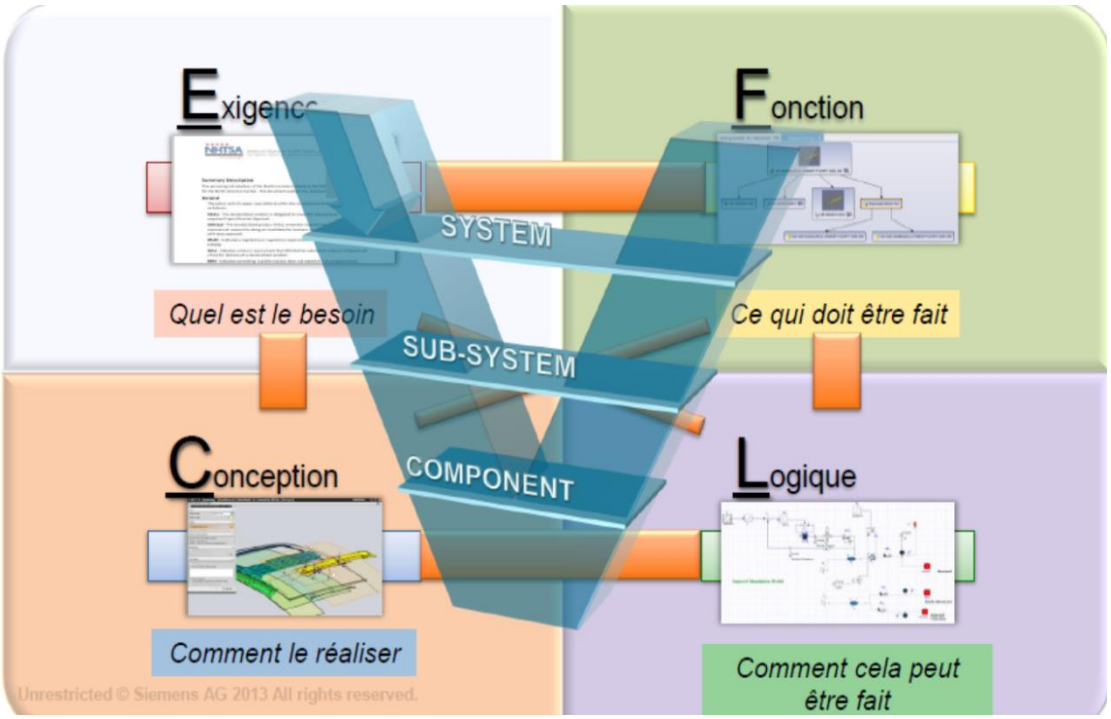

Figure 3: Processus de développement en V

Il faut s'assurer que le besoin est bien rempli et conforme, en suivant généralement le cycle en V (figure 3) de développement multi physique d'un produit. Au départ, les exigences permettent de préciser le besoin, l'aspect fonctionnel précise ce qui doit être fait, la partie de la conception définie comment le réaliser, entre ces deux derniers éléments, la partie logique précise comment cela peut être fait.

1.4. Planification du projet

La planification est parmi les phases d'avant-projet. Elle consiste à prévoir le déroulement du projet tout au long des phases constituant le cycle de développement.

Vu la taille du projet, et après plusieurs réunions avec les encadrants à Media Mobility, le planning suivant a été adopté. Ce planning respecte les spécificités du projet, ainsi, une grande importance a été donnée aux phases « choix des prototypes », « Intégration des codes» et « test de bon fonctionnement ».

Pour cela on a utilisé le diagramme GANTT, c'est un outil de gestion de projet qui permet de visualiser dans le temps les différentes tâches du projet. Cela permet de planifier le projet et par conséquent, de donner une dynamique au projet. Pour représenter ce GANTT, nous avons utilisé le logiciel GANTT Project (figure 4).

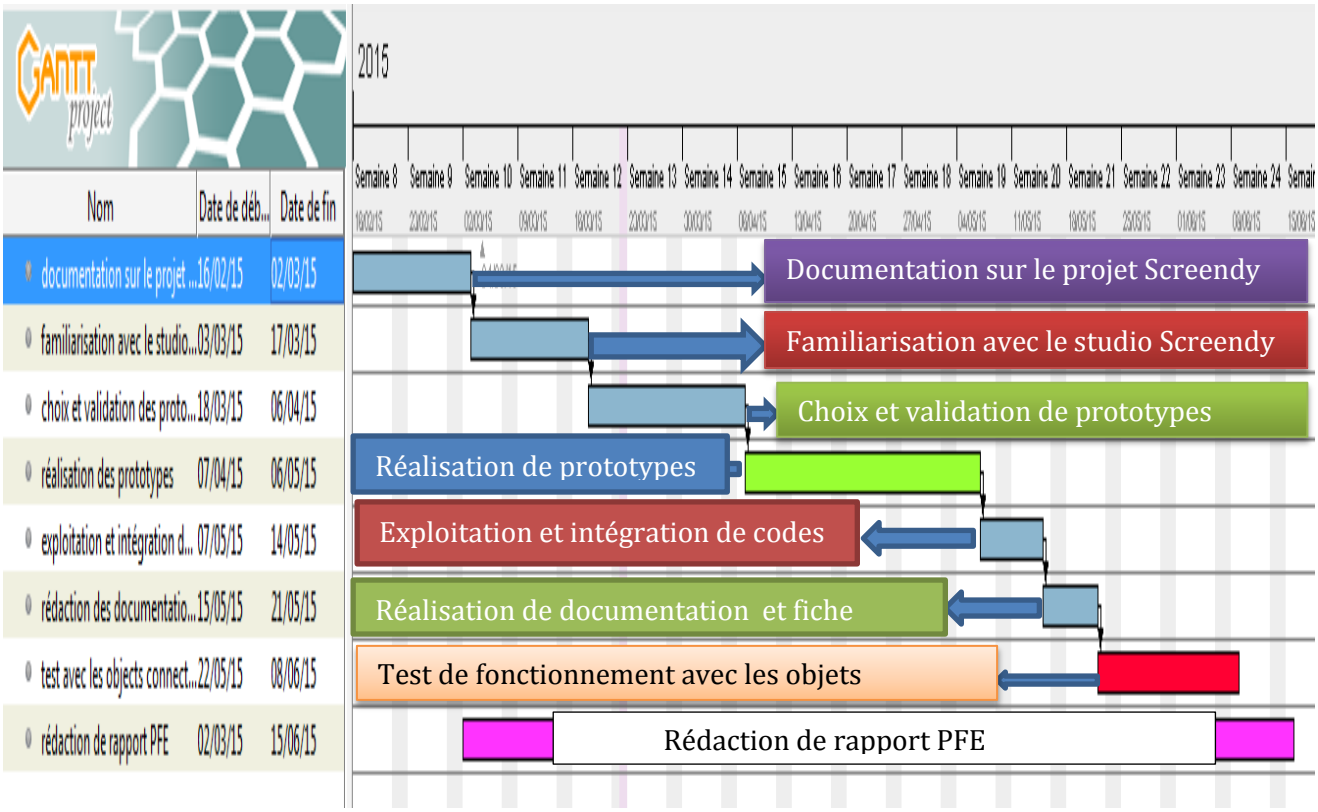

Figure 4: Fiche de planification de projet

Conclusion

Ce chapitre a été le point de départ pour l'élaboration du projet, dans la mesure où il décrivait son contexte général, en présentant successivement l'organisme d'accueil Media Mobility, les objectifs généraux à atteindre, ainsi que la démarche et les étapes de sa mise en œuvre.

Chapitre 2

Etude de la technologie des IOT

Ce chapitre comporte une description de marché des objets connectés et les composants systèmes qui interviennent pour la réalisation de la communication entre les objets.

2. les objets connectés

un objets connecté est un objet qui permet, via des systèmes d'identification électronique normalisés et unifiés, et des dispositifs mobiles sans fil, d'identifier directement et sans ambiguïté des entités numériques et des objets physiques et ainsi de pouvoir récupérer, stocker, transférer et traiter, sans discontinuité entre les mondes physiques et virtuels, les données s'y rattachant. [3]

2.1. Marché des objets connectés

2.1.1. Statistique sur les objets connectés

Les niveaux d'intérêt pour les objets connectés inégaux selon les cibles et les univers concernés sont présentés dans la figure suivante : [4]

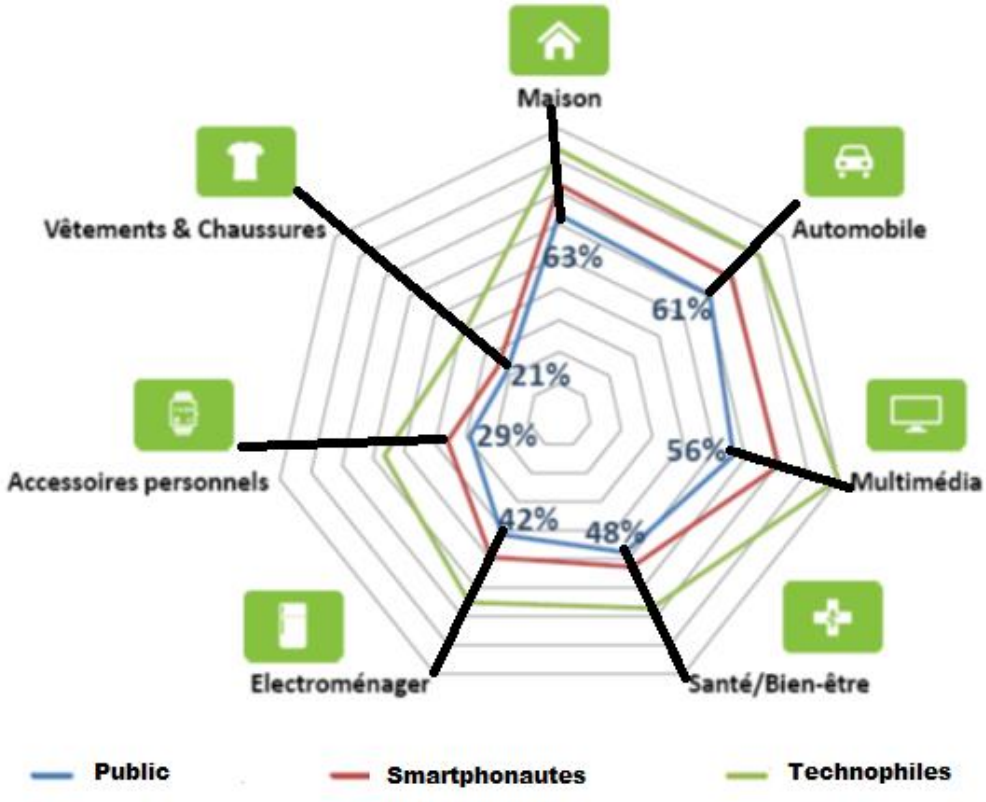

Figure 5: Les niveaux d'intérêt pour les objets connectés selon les cibles

Selon tous les experts et indicateurs (IDC, Cisco, Gartner, Idate, Xerfi, Gfk), le marché des objets connectés devrait connaitre une grande expansion dans 5 ans, l'estimation du marché est présentée dans la figure suivante :

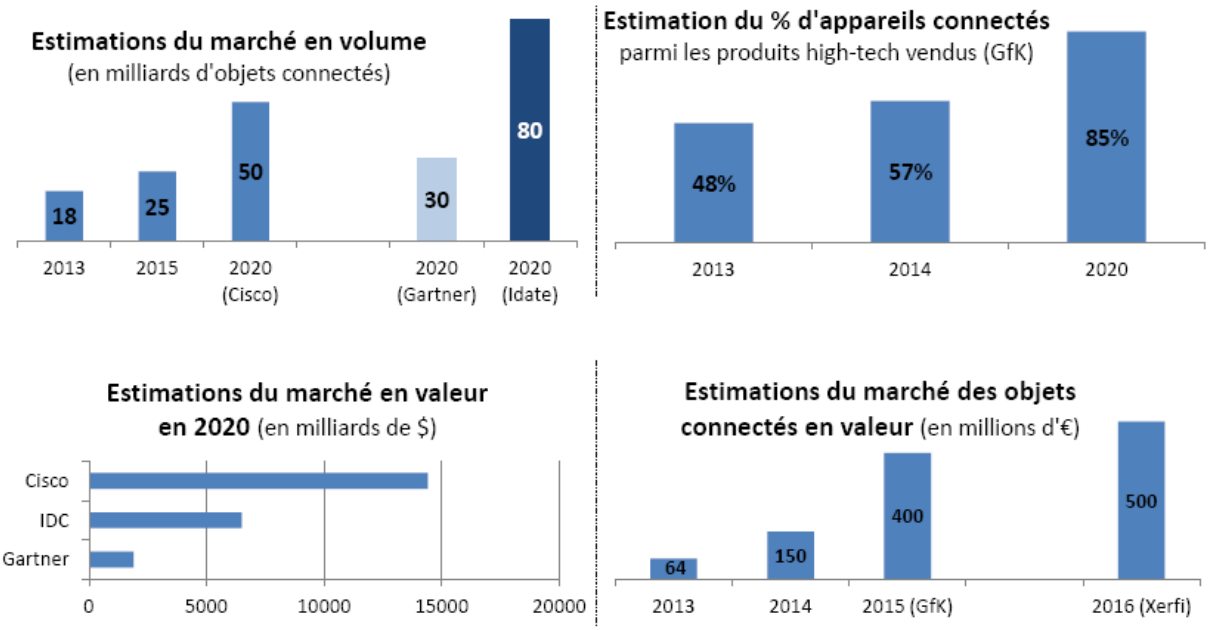

Figure 6: Marché des objets connectés

Grâce à un contexte très favorable tout se connectera et communiquera. Le concept autour d'un objet connecté est présenté dans la figure suivante :

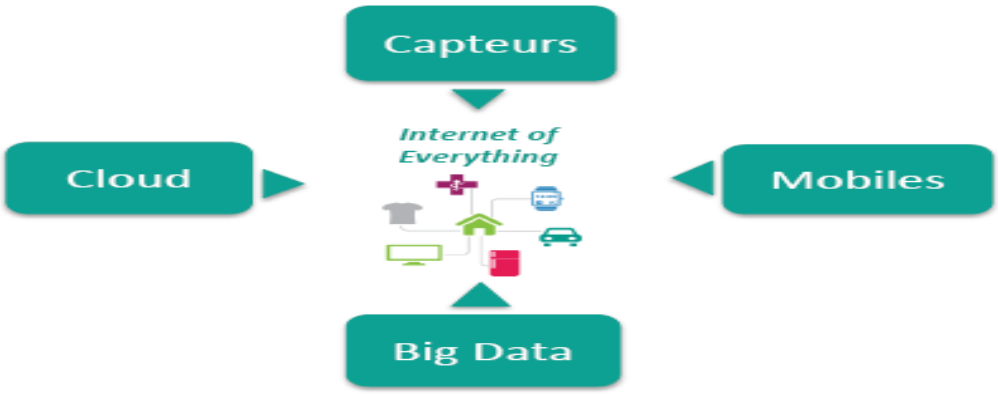

Figure 7: Le concept autour d'un objet connecté

Cloud : serveur informatique distant par l'intermédiaire d'un réseau internet.
Big Data : ensemble des données collectées, stockées et analysées afin de prendre des décisions utiles pour piloter un système.

2.2. Solution ScreenDy

Connaissant l'utilisation des téléphones portables, ainsi que leur commodité, ils représentent un bon moyen pour envoyer des commandes et recevoir des informations en temps réel, la figure suivante présente le noyau Screendy pour les objets connectés.

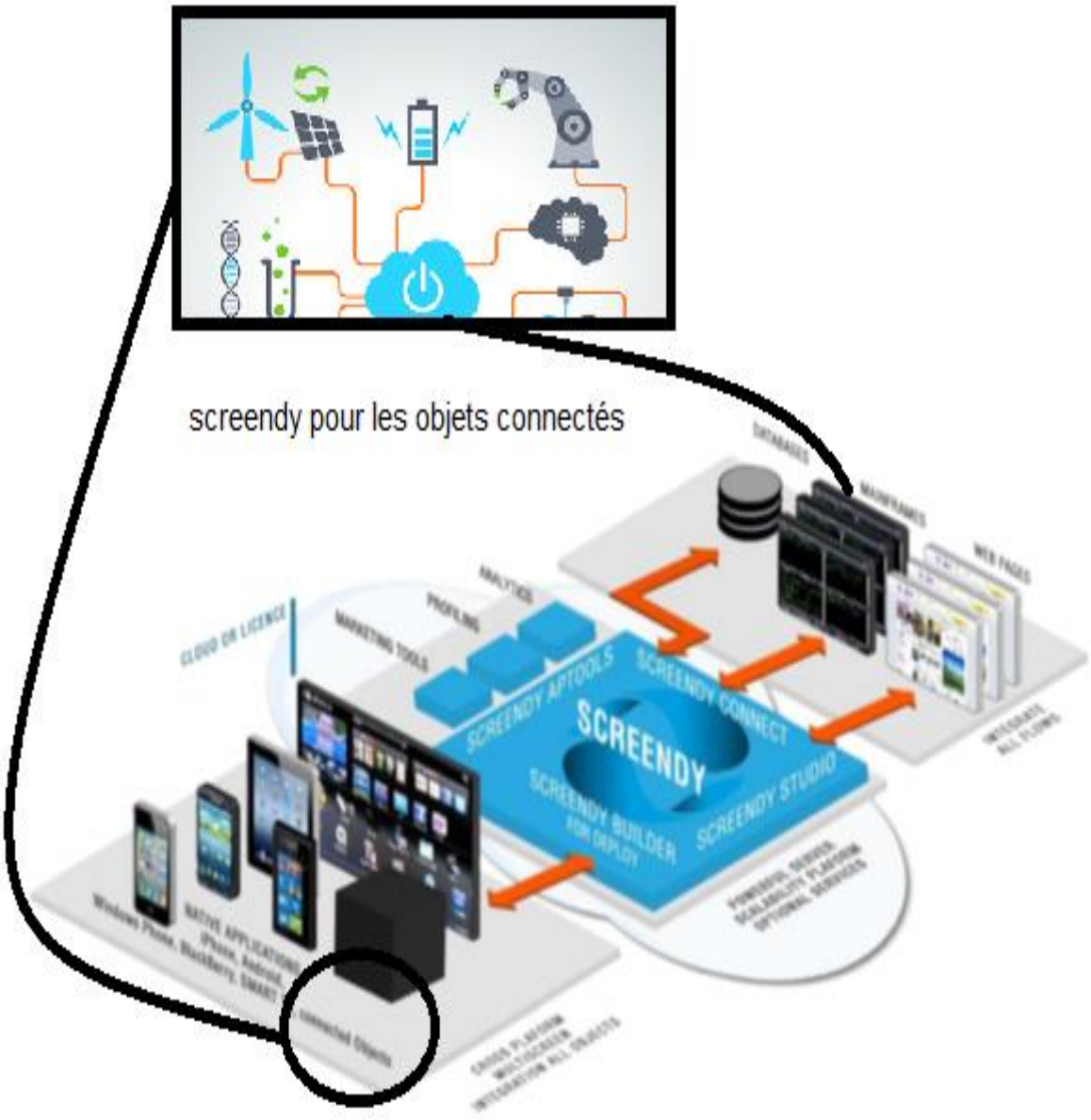

Figure 8: Solution Screendy pour les objets connectés

2.3. Composants système

IOT (internet of things) n'est pas une technologie mais un système de systèmes. L'interopérabilité entre ces systèmes et l'intégration de tous les composants induisent une complexité forte. La capacité à gérer les interfaces est donc déterminante. Voici les principaux systèmes technologiques nécessaires au fonctionnement de l'internet des objets connectés. [3]

Type de systèmes	Identification	Capteurs	Connexion	Intégration	Traitement de données	Réseaux
Technologies anciennes	Codes barre, solutions RFID simples, URI, Coordonnées GPS	Luxmètre, Capteur de Proximité Thermomètre, hydromètre…	Câbles, radio,	Middlewares	Base de données, tableur, Progiciel de gestion intégré,Gestion de la relation client…	Internet, Ethernet…
Technologies récentes	Solutions RFID complexes, puces optiques, ADN	Accéléromètre, Gyroscope, Capteurs miniaturisés nanotechnologies	Bluetooth, Communication en champ proche, WiFi, Zigbee	Middlewares évolués, Analyse décisionnelle des systèmes complexes	Entrepôt de données 3D (compatible avec les puces RFID), Web sémantique…	Réseau EPCglobal

Tableau 2: : principaux systèmes technologiques nécessaires au fonctionnement de l'IOT

2.3.1. Les types de communication

Les télécommunications sont définies comme la transmission à distance d'informations avec des moyens à base d'électronique et d'informatique. Ce terme a un sens plus large que son acception équivalente officielle « communication électronique ». Elles se distinguent ainsi du poste qui transmet des informations ou des objets sous forme physique. Parmi les communications on trouve :

Communication	Définition
Bluetooth	• c'est un procédé de communication basé sur un système d'onde radio à courte distance, permettant le partage ou l'échange d'information entre deux appareils.
ZigBee	• C'est un réseau sans fil à bas débit et à courte portée qui utilise les ondes hertziennes pour transporter des messages entre deux ou plusieurs entités réseaux.
Infrarouge	• c'est un faisceau de lumière commode, rapide mais sensible aux interférences lumineuses. Le faisceau ne doit jamais être coupé sinon la transmission est interrompue.
Wifi	• le réseau Wi-Fi permet de relier par ondes radio plusieurs appareils informatiques (ordinateur, routeur , Smartphone, décodeur Internet, etc.) au sein d'un réseau informatique afin de permettre la transmission de données entre eux.
GSM	• : (Global System for Mobile Communication) La norme GSM est utilisée pour les réseaux de communication sans fil à travers le monde.
NFC	• Near Field Communication, (ou communication en champ proche) est une technologie permettant d'échanger des données à moins de 10cm, entre deux appareils équipés de ce dispositif. Le mobile équipé du NFC est capable de lire des « tags » (étiquettes électroniques), pour récolter des informations pratiques, ou pour lancer une action de manière automatique sur un Smartphone.

Tableau 3: Les types de communication

2.3.2. Les types de cartes

Toutes les cartes électroniques équipées d'un composant intelligent comme le microcontrôleur sont capables de communiquer avec des modules de communication (Bluetooth, Wifi, GSM…) dont le but est de contrôler à distance des objets connectés.

Parmi les familles de microcontrôleurs nous avons : les AVR, ARM, PIC, INTEL 8051, STM32, freescale …

Parmi les cartes électroniques il y a : arduino, brique NXT, PICkit, Raspberry Pi, IOIO Android, arduino ADK, freescale FRDM-KL25Z...

2.3.3. Les types de capteurs

Un capteur est un transducteur capable de transformer une grandeur physique en une autre grandeur physique généralement électrique utilisable par l'homme ou par le biais d'un instrument approprié. Le capteur est le 1er élément d'une chaîne de mesure ou d'instrumentation.

Un capteur n'est jamais parfait, il convient de connaître avec la plus grande précision possible son état d'imperfection. De plus, il faut prendre en compte la perturbation apportée au système par la mesure. La chaine d'acquisition est présentée dans la figure suivante :

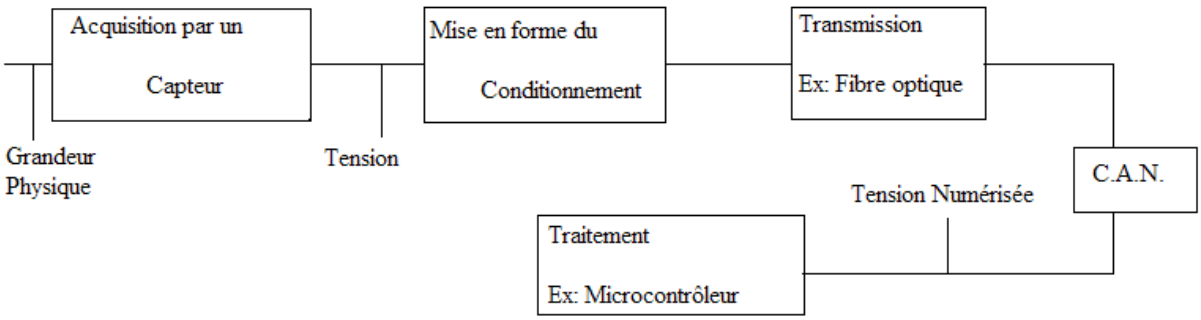

Figure 9: Chaine d'acquisition d'un capteur

Conclusion

Le but de ce chapitre était la présentation de la technologie des objets connectés et l'analyse du composant système. Les objets connectés ont été recensés dans un premier temps avant l'identification des acteurs interagissant avec le système et leurs rôles vis-à-vis de celui-ci.

Chapitre 3

Conception de prototypes IOT

Ce chapitre comporte l'ensemble des prototypes qui relie un Smartphone et un objet connecté. Le premier prototype sera une maison intelligente connectée et le second sera un véhicule connecté.

3. Prototype

L'objectif est de réaliser des prototypes pour mettre en réalité la notion d'un objet connecté et pour faire les tests de bon fonctionnement, la sécurité et les démarches à suivre pour généraliser ces applications sur tous les Smartphones et les adapter à tous les protocoles de communication disponibles.

3.1.1. Brainstorming sur les prototypes

Au cours d'une séance de brainstorming, l'équipe ScreenDy propose un ensemble d'idées sur les prototypes à réaliser; on cite les prototypes suivants (tableau 5) :

Numéro de prototype	prototype
1	ascenseur connecté
2	maison intelligente
3	voiture connectée
4	drone connecté

Tableau 4: Liste de prototypes

3.1.2. Choix des prototypes à réaliser

Après une phase de recherche dans le domaine des objets connectés, on a trouvé parmi les prototypes proposés deux qui présentent parfaitement la notion d'un objet connecté. Il s'agit de :

- ✓ Prototype1 : contrôle par Smartphone d'une maison intelligente
- ✓ Prototype2 : contrôle par Smartphone d'une voiture connectée

3.1.3. Matériel nécessaire pour la réalisation des prototypes

- Carte électronique et système d'exploitation mobile

Alors pour mettre en œuvre des prototypes fonctionnels des objets connectés, on va utiliser les types les plus populaires c'est pour cela on a décidé de travailler avec les types suivants (tableau 6) :

Système d'exploitation	Android
Communication	Bluetooth
Carte électronique	Arduino Uno

Tableau 5: La technologie choisi pour réaliser les prototypes

- Les composants

Pour éviter le retard de livraison des produits nécessaires pour la réalisation de ce projet, on a utilisé notre expérience pour estimer et élaborer la liste des articles afin de les recevoir avant la fin de la période de stage. La liste des articles est présentée dans le tableau suivant :

Nom article	**Reference**	**quantité**
Câble de prototypage F/F		**14**
Câble de prototypage M/F		**14**
Carte arduino Uno		**2**
Bluetooth arduino Uno	JY-MCU HC-06	**2**
Chargeur carte arduino Uno		**1**
LED Vert		**6**
LED rouge		**6**
Batterie pour arduino 9V		**1**
Interface L293D		**1**
Servomoteur Arduino	24 x 13 x 27 mm	**1**
Capteur de mouvement (PIR) pour arduino Uno		**1**
Capteur de température pour arduino Uno	DS18B20	**1**
Capteur camera pour arduino Uno		**1**
Résistance 4k7		**2**
Résistance 220		**4**
Câble usb arduino Uno		**1**

Tableau 6: liste des articles pour la réalisation de projet

3.2. Prototype 1 : La maison connectée

À l'avenir, le chauffage, la climatisation, l'éclairage, la gestion des flux (eau, énergie, aliments, déchets, information...) et la sécurité pourraient être pour tout ou partie gérées par un système informatique, auto-apprenant dédié (centralisé ou non), en interaction avec les besoins des occupants, éventuellement en utilisant des énergies et des ressources moins nuisantes pour l'environnement, avec ou sans câblage. La maison s'adapterait aux habitudes et aux goûts de ses habitants et invités (éventuellement malvoyants, handicapés, âgés, malades, etc.), grâce à un profilage de ces derniers, communiqué au système gérant la maison. Certains imaginent aussi une maison intelligente et autonome pour ses besoins en eau, thermies, frigories ou électricité, capable de détecter d'elle-même, des dysfonctionnements ou des changements de paramètres susceptibles de présenter un danger.[5]

C'est pour cela on a développé un prototype basé sur des capteurs et actionneurs les plus utilisés dans la domotique afin d'approcher le principe de la maison intelligente (figure10).

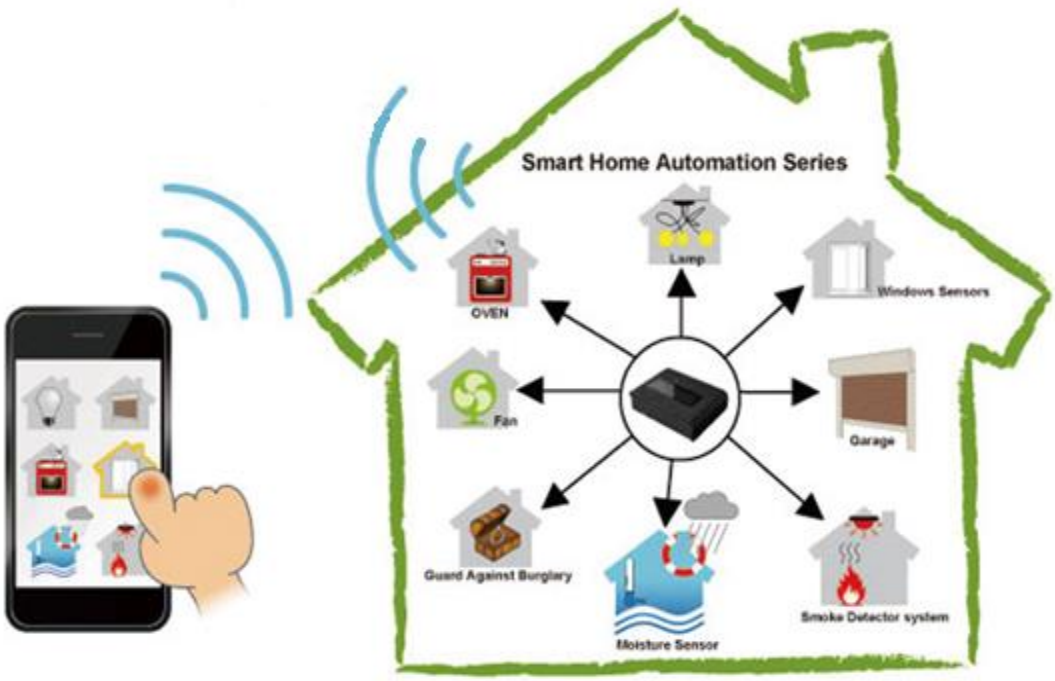

Figure 10: La maison connectée

3.2.1. Les équipements de la domotique

Le système domotique se divise, au sein d'une installation domotique, en six familles d'équipements à la fois complémentaires et interdépendantes :

- La série des automatismes : lesquels incluent divers cerveaux-moteurs comme ceux de la motorisation du portail, la motorisation des volets, ainsi que la domotique de la piscine ;
- Les types de programmation domotique : nécessaires entre autres pour piloter l'éclairage, le chauffage, etc. ;
- La famille des appareils domotiques proprement dits, tels que l'aspirateur, le home cinéma, etc. ;
- Les indispensables capteurs domotiques, aux formes et fonctionnalités multiples, ainsi que les alarmes qu'ils permettent de déclencher ;
- L'articulation incontournable du réseau qui garantit la communication mono ou bidirectionnelle de l'ensemble du système domotique. Boulevard de la maison communicante, le réseau inclut le câblage domotique, ou mieux la domotique sans fil.
- Le groupe des moyens de commande à distance : diverses télécommandes et écrans domotiques, le téléphone mobile et la domotique par internet.

3.2.2. Description de fonctionnement des capteurs et actionneurs

3.2.2.1. Capteur de mouvement

Un capteur PIR (capteur passif d'infrarouge) permet de détecter le mouvement d'un corps humain (en effet, la chaleur du corps produit suffisamment de lumière infrarouge pour être mesurée). Le capteur peut détecter un mouvement jusqu'à une distance maximale de 6m.

Ce capteur (figure 11) nécessite au moins 20 secondes de calibration automatique au démarrage. Durant ces 20 secondes, il est complètement inactif et ne doit pas être obstrué par des sources de chaleurs (main, corps, chat, etc).

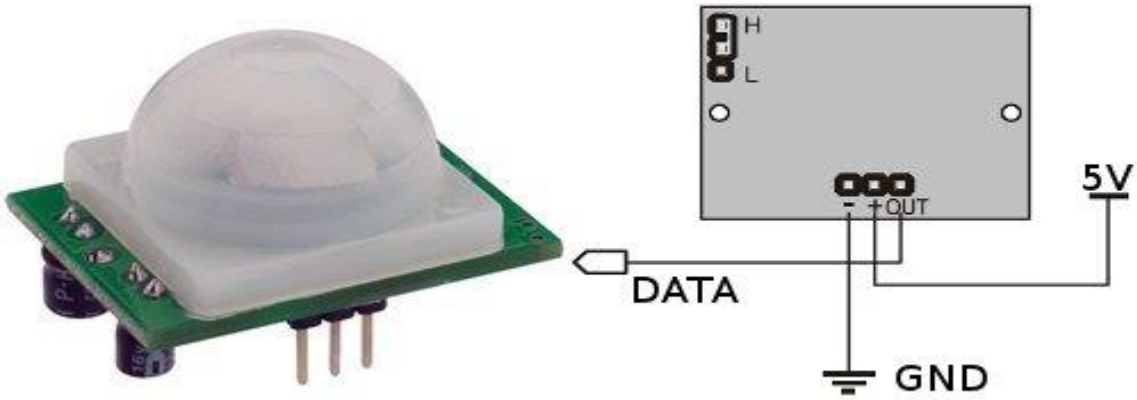

Figure 11: Capteur de mouvement

3.2.2.2. Capteur de température DS18B20

Le DS18B20 (figure 12) est un capteur de température du fabricant Dallas, il communique via un bus 1-Wire (la unique simple-fil interface mode). Ce capteur nécessite seulement un port lignes peut être réalisé avec le microprocesseur. Parmi ces caractéristiques nous avons :

- Communication bidirectionnelle avec le microprocesseur ;
- Plage de température: -55 ~ + 125 ;
- La température inhérente résolution 0.5 ;
- Puissance de travail 3 ~ 5 V/DC ;

.

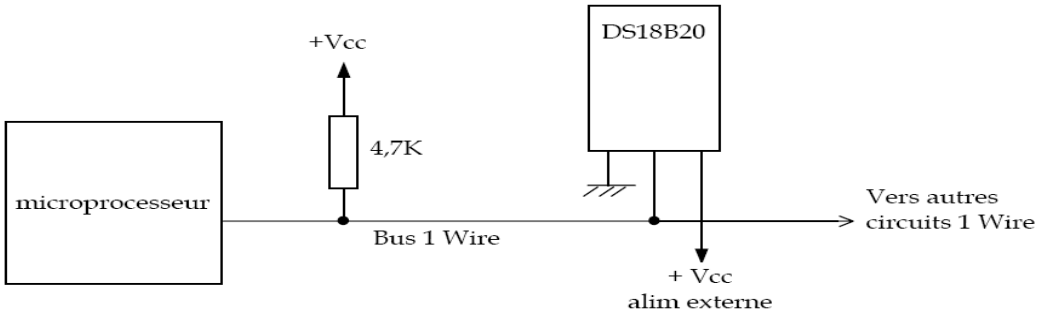

Figure 12: Capteur de température

3.2.2.3. Capteur d’image OV7670

L'OV7670 (figure 13) est un capteur d'image, de petite taille, avec une faible tension de fonctionnement qui offre toutefois toutes les fonctions de la caméra mono-puce VGA (Video Graphics Array) et du processeur d'images.

Grâce au contrôle du bus SCCB (serial camera control bus), il permet d’afficher l'image entière dans une fenêtre avec une résolution des données de 8 bits. La vitesse d'échantillonnage VGA est de 30 images par seconde. Il est possible de paramétrer la qualité des images, le format des données et le mode de transmission. [6]

Le système intégré Omnivision à capteur d'image permet d'améliorer la qualité de l'image en réduisant ou en éliminant les défauts optiques ou électroniques, tels que le bruit de motif fixe, la couleur ainsi que la clarté et la stabilité de l'image.

Figure 13: Module camera OV7670

3.2.3. Schéma général du montage

Le schéma général du montage de notre système est illustré par la figure suivante.

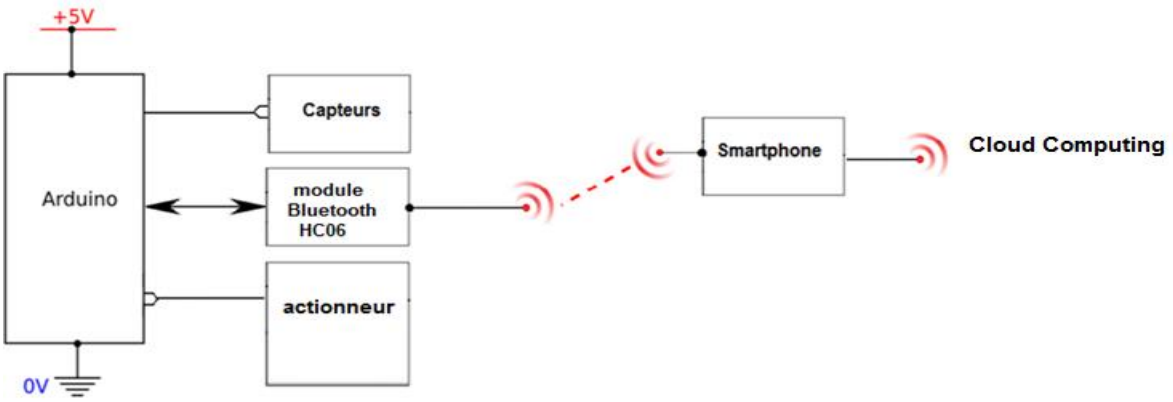

Figure 14: Schéma général du montage

La carte « Arduino » reçoit les commandes venues du « Smartphone » par l'intermédiaire « Module Bluetooth » qui est le responsable de la communication bidirectionnelle. En se basant sur les informations reçus des « capteurs », la carte « Arduino » envoie ses commandes aux « actionneurs ».

3.2.4. Le circuit du montage

La figure suivante présente le circuit du montage de notre système.

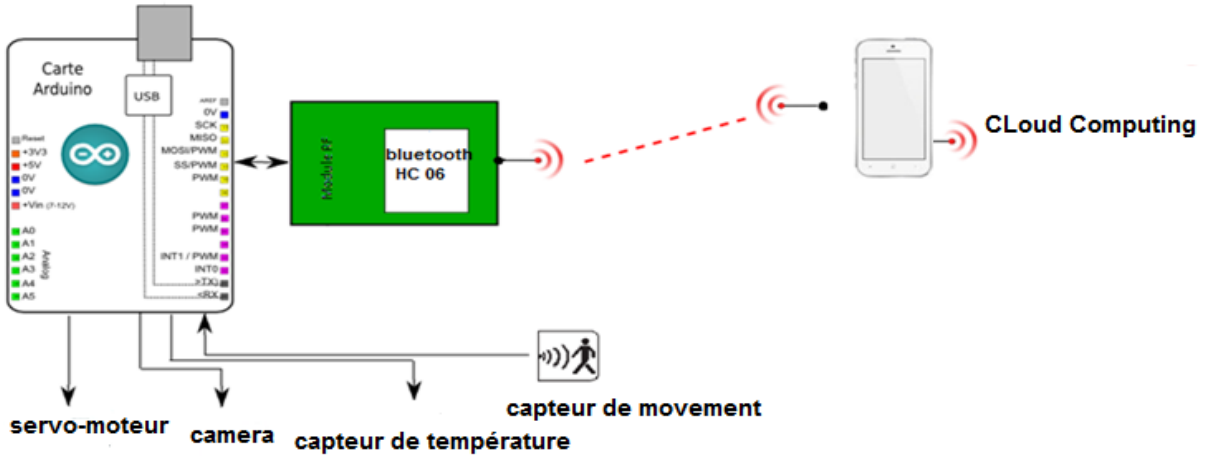

Figure 15: Circuit du montage capteurs et actionneurs

3.2.5. Schéma descriptif

Dans le schéma suivant (figure 16) on trouve deux types de communication intégré dans le projet :

- Communication entre le microcontrôleur et le module Bluetooth: communication usart

  (Voir annexe E.1)

- Communication entre le module Bluetooth et Smartphone : communication radio.

  (Voir annexe E.1)

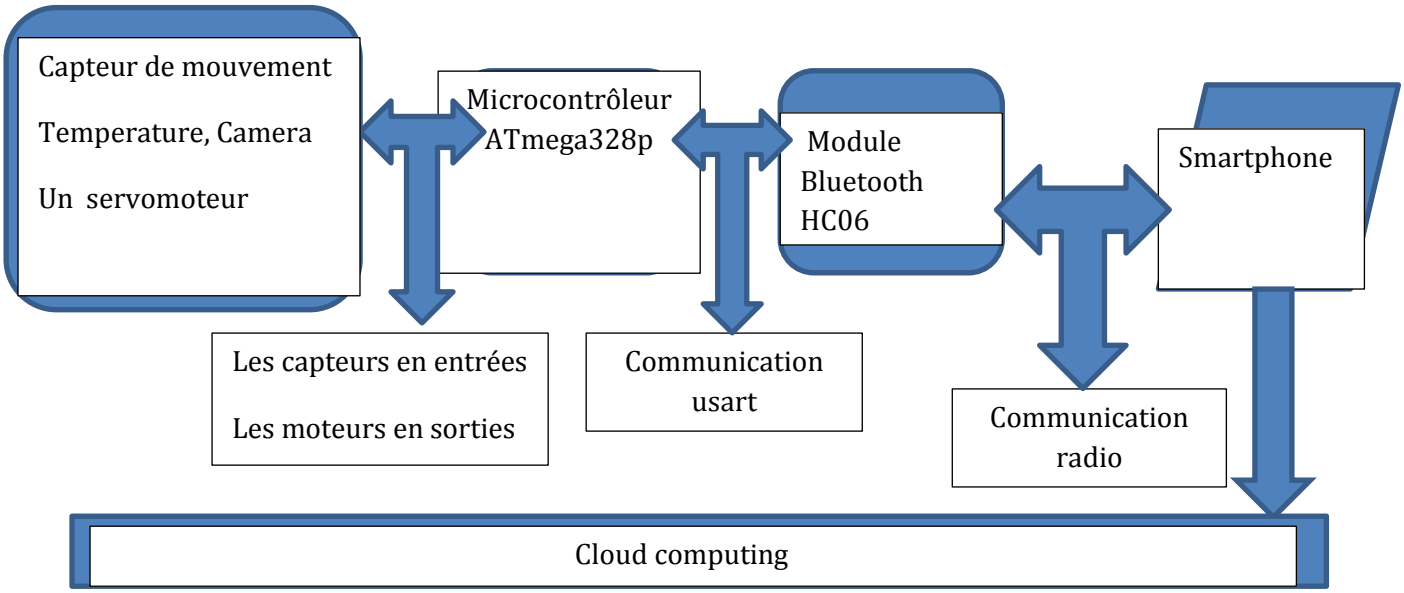

Figure 16: concept de communication dans le système

3.2.6. Schéma électrique de la maison connectée

Le schéma electrique illustré dans la figure suivante (figure 17) présente le premier prototype des objets connectés.

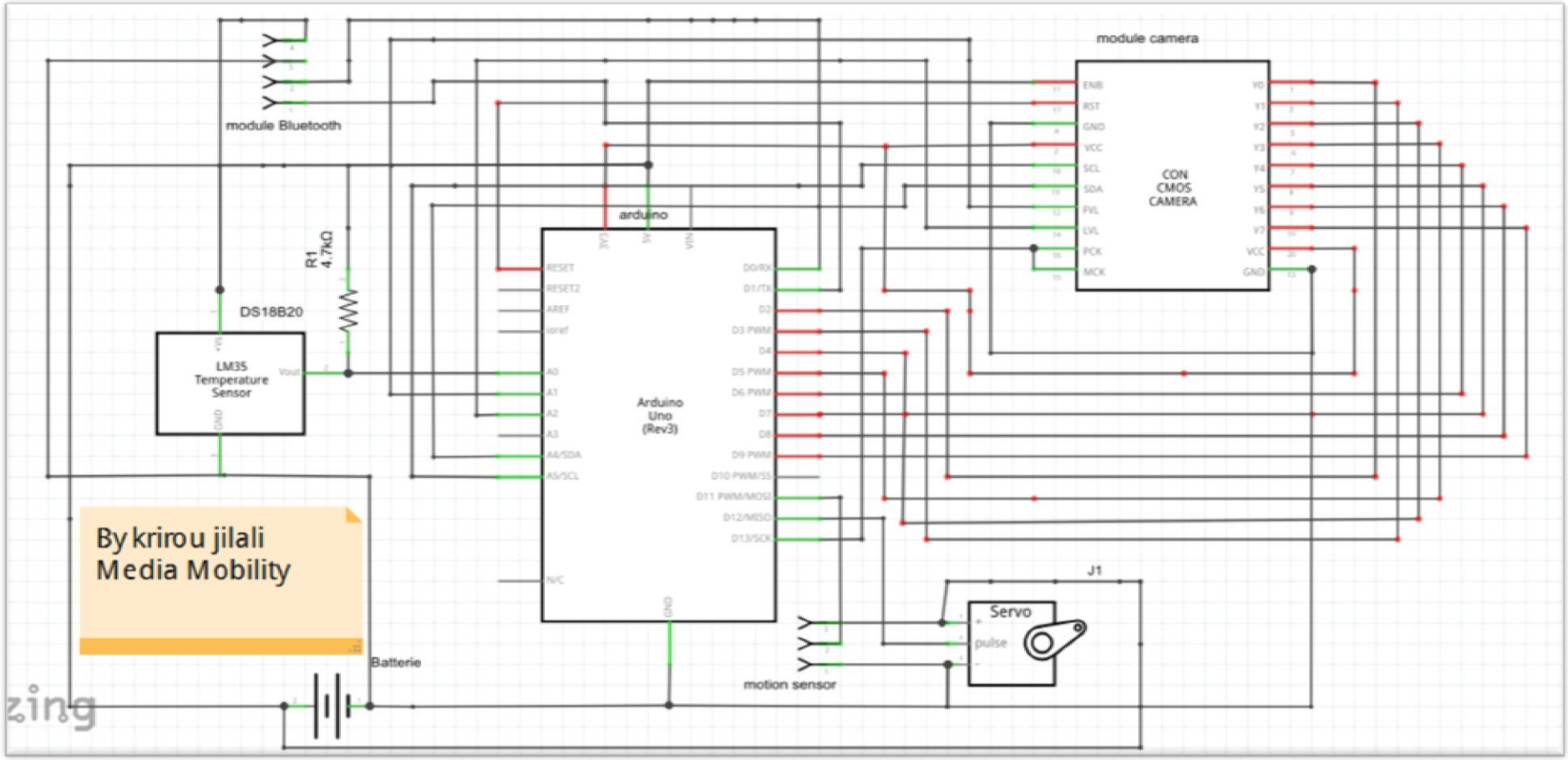

Figure 17: Schéma électrique pour les équipements de la domotique

3.2.7. Organigramme de commande

L'organigramme illustré dans la figure suivante présente l'architecture de code qui sera implémenté dans la carte arduino afin de commander le système.

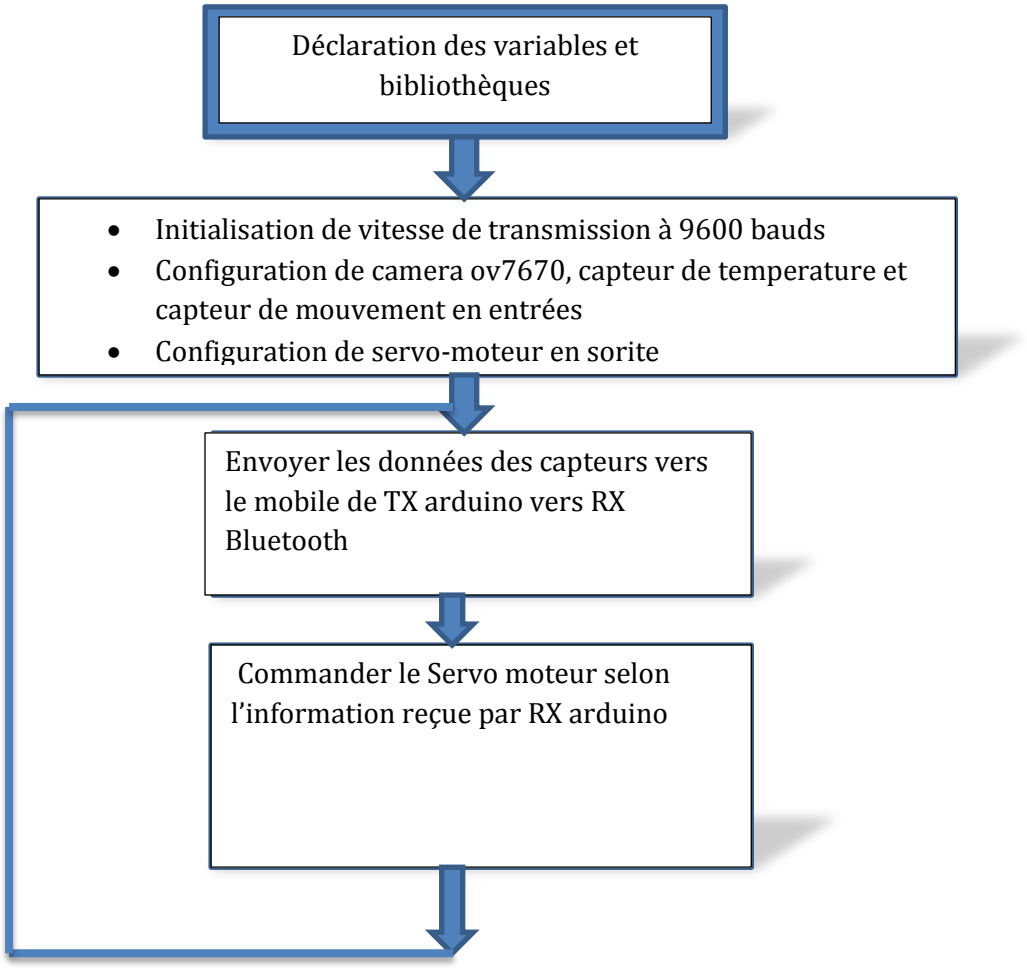

Figure 18: Organigramme de commande

3.2.8. Test de fonctionnement d'émetteur et récepteur

Studio Avrbot (figure 19) est une plateforme de développement intégré permet la programmation des microcontrôleurs en langage C/C++. Aussi, cette plateforme est utile pour établir plusieurs types de communication avec des cartes à microcontrôleur afin de contrôler un système.

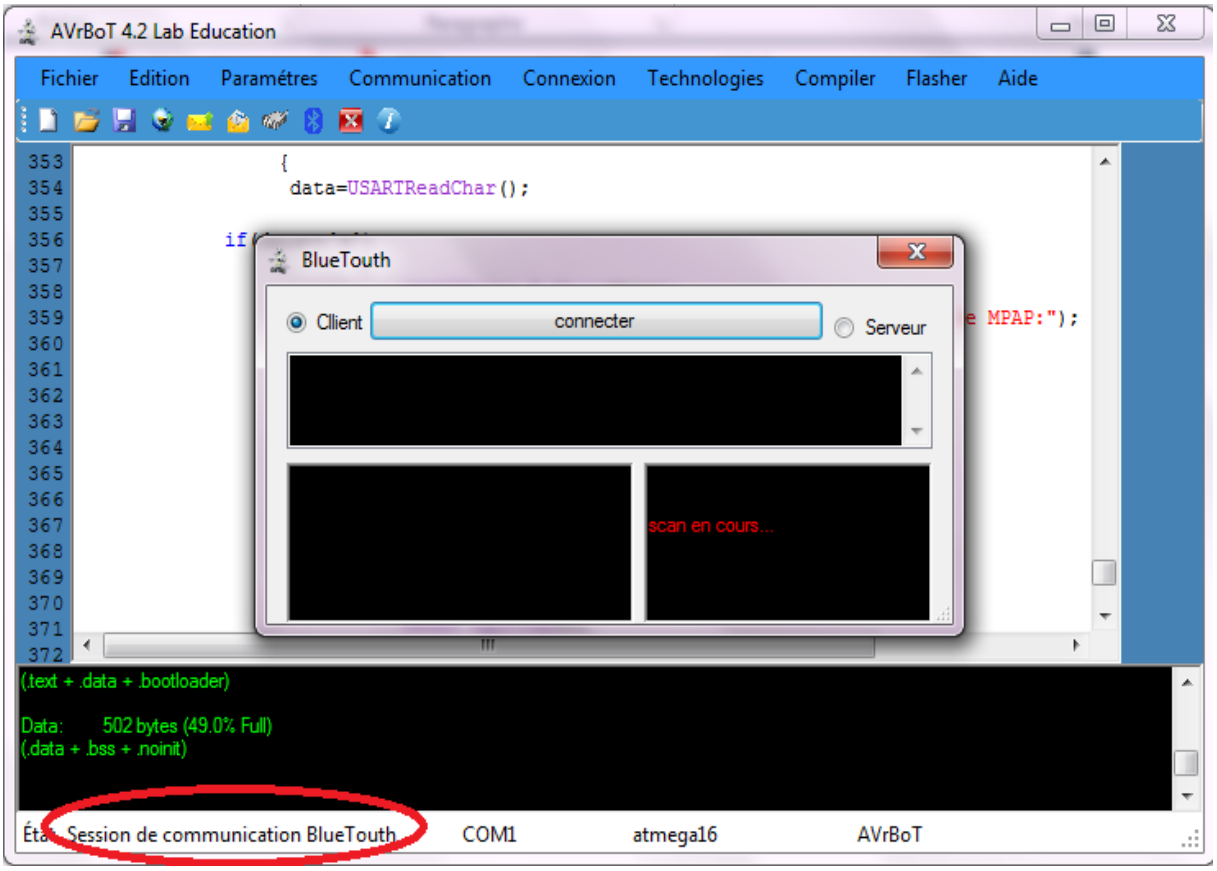

Figure 19: Studio AVRBOT

Dans ce projet on a utilisé le studio avrbot comme interface de test de communication Bluetooth afin de valider le bon fonctionnement de la partie électronique/système embarqué.

La figure(19) montre l'interface Bluetooth qui permet de réaliser la recherche du système embarqué afin d'établir la connexion. L'utilisateur peut envoyer alors des commandes pour tester la réponse de système.

3.3. Prototype 2 : voiture connectée

Dans les prochaines années, la voiture deviendra progressivement un véritable objet communicant. L'enjeu est crucial pour les constructeurs automobiles. Cette connectivité permettra d'offrir de nouvelles prestations et les automobilistes sont déjà en attente de services connectés, dans la continuité de ceux dont ils disposent déjà sur leurs smarts devices.

3.3.1. Les actionneurs et les capteurs utilisés

En plus des capteurs utilisés dans la maison connectée, pour ce prototype on ajoute des moteurs pour jouer le rôle des roues motrices de la voiture.

- 2 moteurs à coutant continu pour les roues motrices ;
- un servomoteur pour changer la direction de la voiture.

3.3.2. Schéma électrique

Le module suivant (figure 20) montre un exemple de schéma de commande d'une voiture connectée, la nouvelle architecture d'automobile impose d'utiliser des caméras embarquées qui remplacent les capteurs traditionnels. Le moteur de proportion sera remplacé par des moteurs qui sont relié à chaque roue motrice.

La camera permet de capturer des images instantanées et de les envoyer pour être traiter, des applications informatique sont chargés de traiter l'image pour donner des propositions et alertes pour le conducteur.

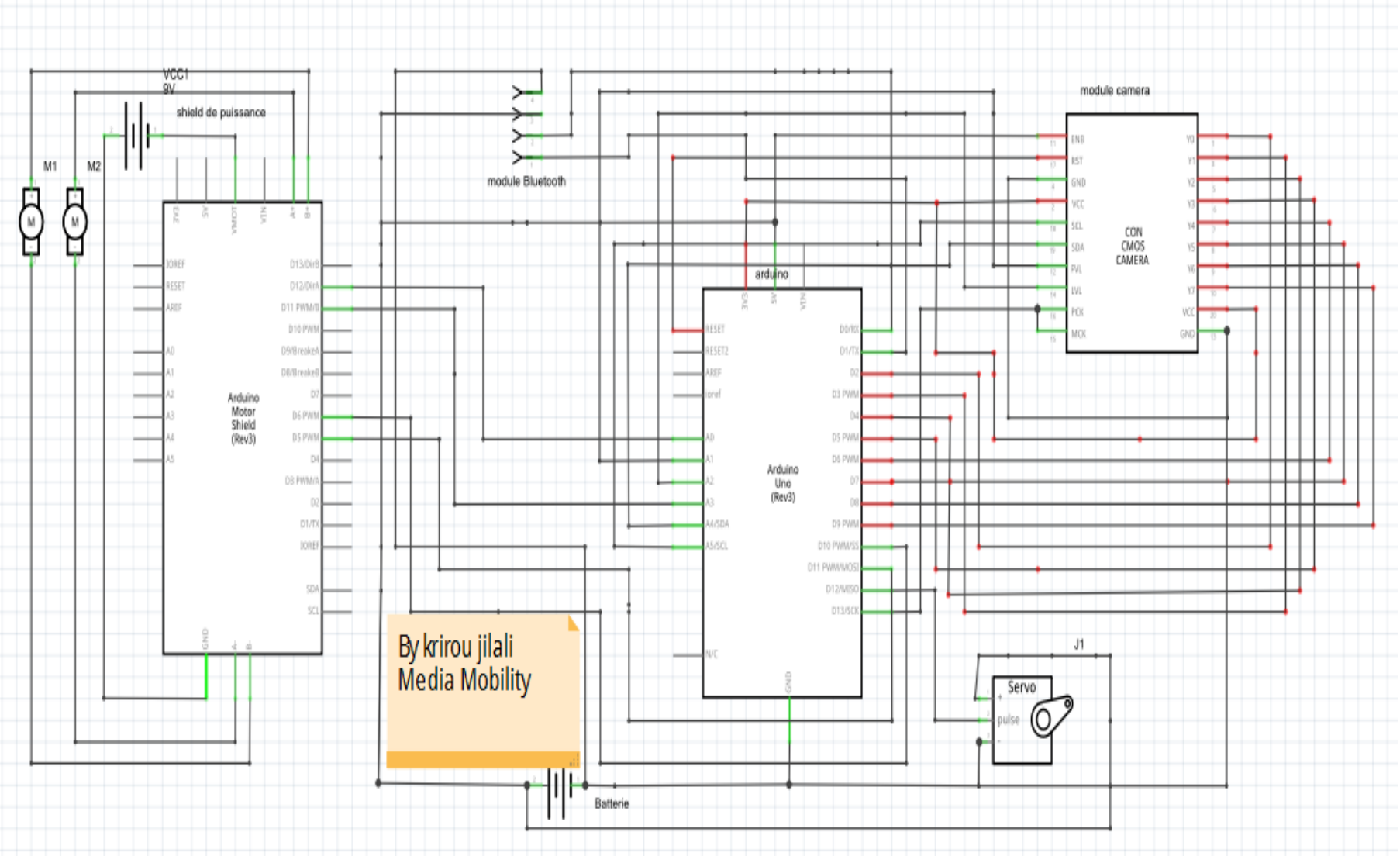

Figure 20: Schéma électrique de la voiture connectée

3.3.3. Développement de programme Android coté Smartphone

Cette architecture explique les étapes pour établir la connexion avec les périphériques

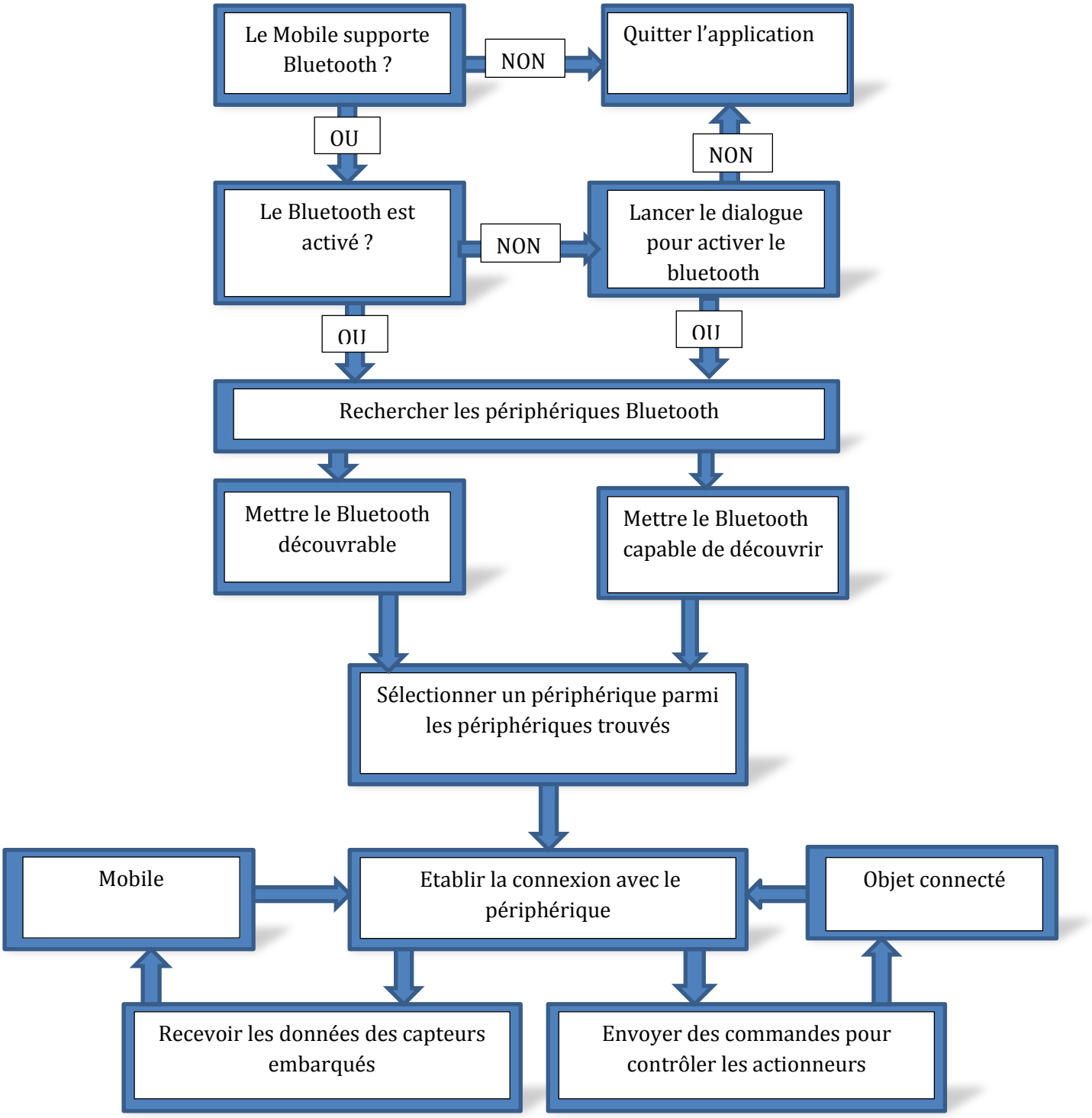

Figure 21: Architecture de la communication Bluetooth

L'objet Bluetooth teste d'abord si le mobile supporte un protocole Bluetooth, si oui il demande d'activer le Bluetooth (s'il est désactivé). Une fois le Bluetooth activé l'objet lance une Windows (popup) et commence à chercher les périphériques dans son entourage 10 m à 15 m, une liste des appareils trouvés s'affiche dans le Windows. L'utilisateur doit sélectionner un appareil dans la liste pour se connecter, par la suite on envoie la commande, les données reçues s'affichent dans une Windows.

Conclusion

Le but de ce chapitre était la présentation des prototypes pour montrer l'importance de la communication Bluetooth dans le domaine des objets connectés et la nécessité de développer des blocs de communication afin de simplifier le développement des applications.

Chapitre 4

Intégration de Bluetooth dans le studio ScreenDy

Ce chapitre comporte une description de la phase d'intégration de l'objet Bluetooth. Dans un premier temps, le fonctionnement de studio sera présenté, puis le code Bluetooth sera exploité afin de le généraliser et de l'intégrer.

4. Intégration du code dans le studio ScreenDy

4.1. Fonctionnement de studio ScreenDy

Le projet ScreenDy constitue une solution efficace pour développer plus aisément et plus rapidement des applications personnalisées avec des fonctionnalités d'une précision égale voire meilleure à celles développées à l'aide d'un code natif (figure 22).

Figure 22: Studio de développement Screendy

Le principe du fonctionnement du système (Noyau + Plugin) suit les étapes suivantes (Figure 23) :

Figure 23: Architecture simplifié du système

1. Le fonctionnement typique du système commence d'abord par la manipulation de la charte graphique de l'application mobile sur le «Studio» par l'utilisateur; il ajoute les différents composants de l'application: pages, styles, boutons, images, formulaires, …etc.

2. La partie Backend du studio génère par la suite un métalangage qui décrit l'ensemble de l'application avec ses différents composants ; Ce métalangage n'est autre qu'un simple fichier JSON bien structuré qui décrit chaque composant avec les différents paramètres associés à ce dernier.

3. Le fichier JSON généré est envoyé par la suite au noyau de chaque OS mobile pour le traiter (étape 4)

4. Le Core (noyau) manipule le fichier JSON (métalangage) en bouclant sur les différents composants décrits par ce fichier et fais la correspondance (via le nom) entre ses composants et les différents plugins supportés. Ainsi, le noyau instancie l'objet équivalent en lui attribuant les différents paramètres et options.

Ce projet englobe plusieurs phases qui s'achèvent par la création d'un exécutable «.APK » prêt à l'utilisation. Tout d'abord nous préparons les différentes pages qui vont alimenter le squelette de notre application et à l'aide de l'environnement de développement mobile « **STUDIO ».** Ensuite nous lançons une demande de la génération de notre application conçue dans la phase précédente à l'aide du bouton « **DOWNLOAD »** de l'environnement « **STUDIO ».** Une fois l'opération du « **DOWNLOAD »** est achevée, le deployeur s'occupe de la configuration du noyau à travers l'alimentation des différentes informations relatives à l'application conçue telles que le **nom d'application**, **l'identifiant de l'application**, **mode d'application** et le **type de lancement** et il exécute par la suite le noyau afin d'obtenir une application exécutable prête à employer.

La figure schématise les différents composants de notre solution **SCREENDY.**

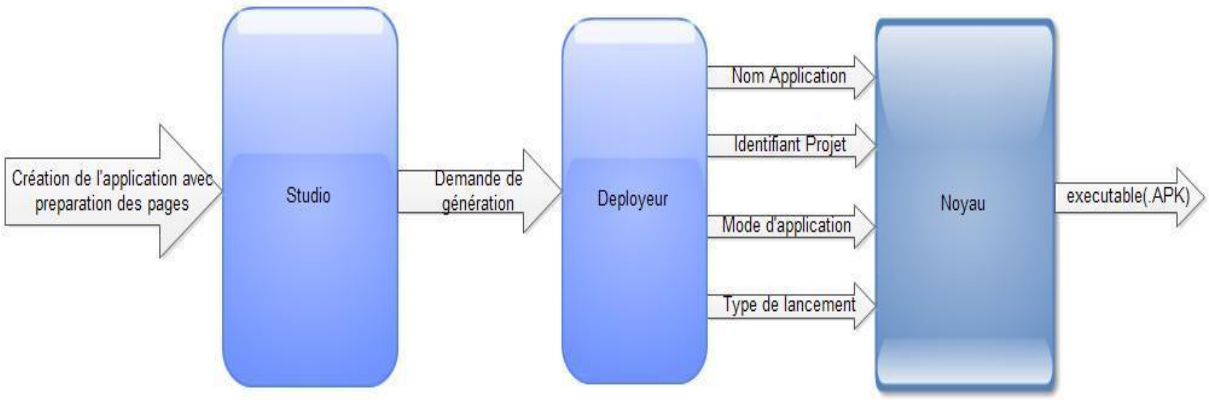

Figure 24: Les composants de solution ScreenDy

Studio : un éditeur wysiwyg «What you see is what you get», qui permet de Définir l'interface graphique de l'application en simple glisser déposer.

Deployeur : est un script s'occupe de la configuration et l'exécution du noyau.

Noyau : permet d'identifier chaque objet avec son propre code pour la création de l'application (fichier exécutable .APK)

Pour mieux comprendre le concept du noyau, nous devons présenter la figure suivante :

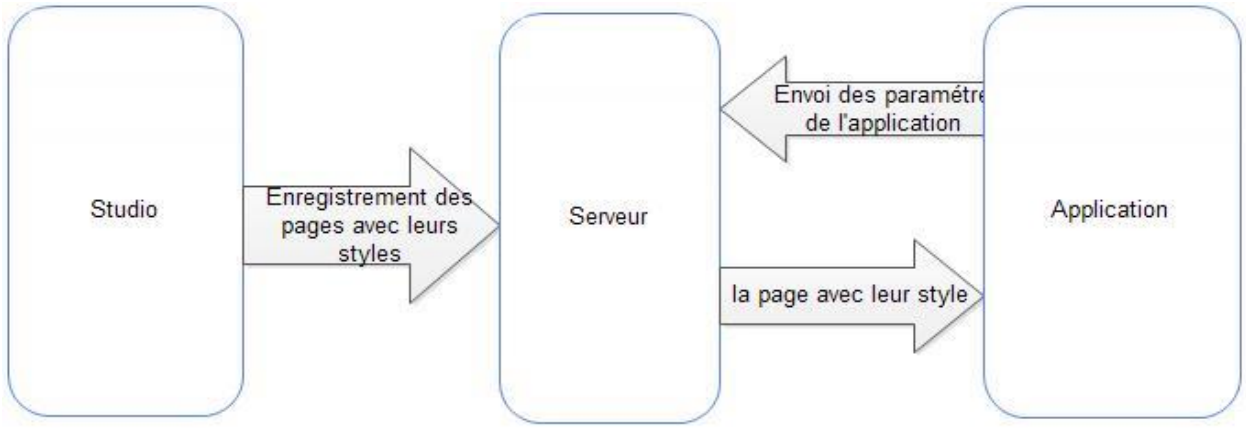

Figure 25:communication entre serveur, noyau et studio

L'application est un noyau configuré qui envoi les paramètres au serveur pour ramener la seule et la première page avec le style de l'application de serveur.

4.2. Intégration de code au noyau Android

Pour la mise en place d'un plugin (objet), il est essentiel de comprendre l'interaction qu'il va y avoir entre la classe de notre composant/plugin et ScreenDy, comme expliqué dans le schéma suivant :

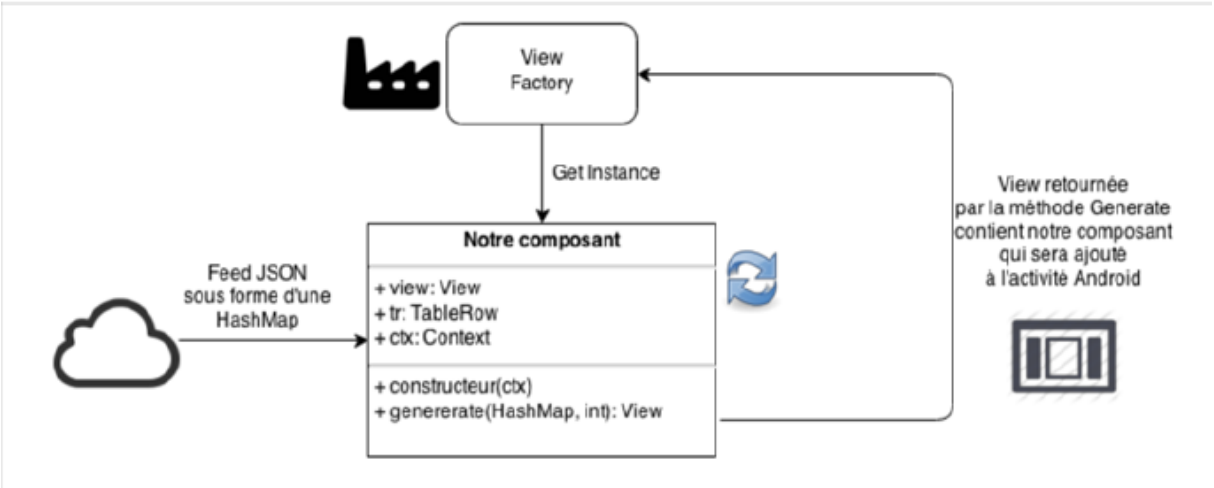

Figure 26: Interaction entre la classe du plugin et ScreenDy

On a divisé le code en trois grandes parties (figure 27) :

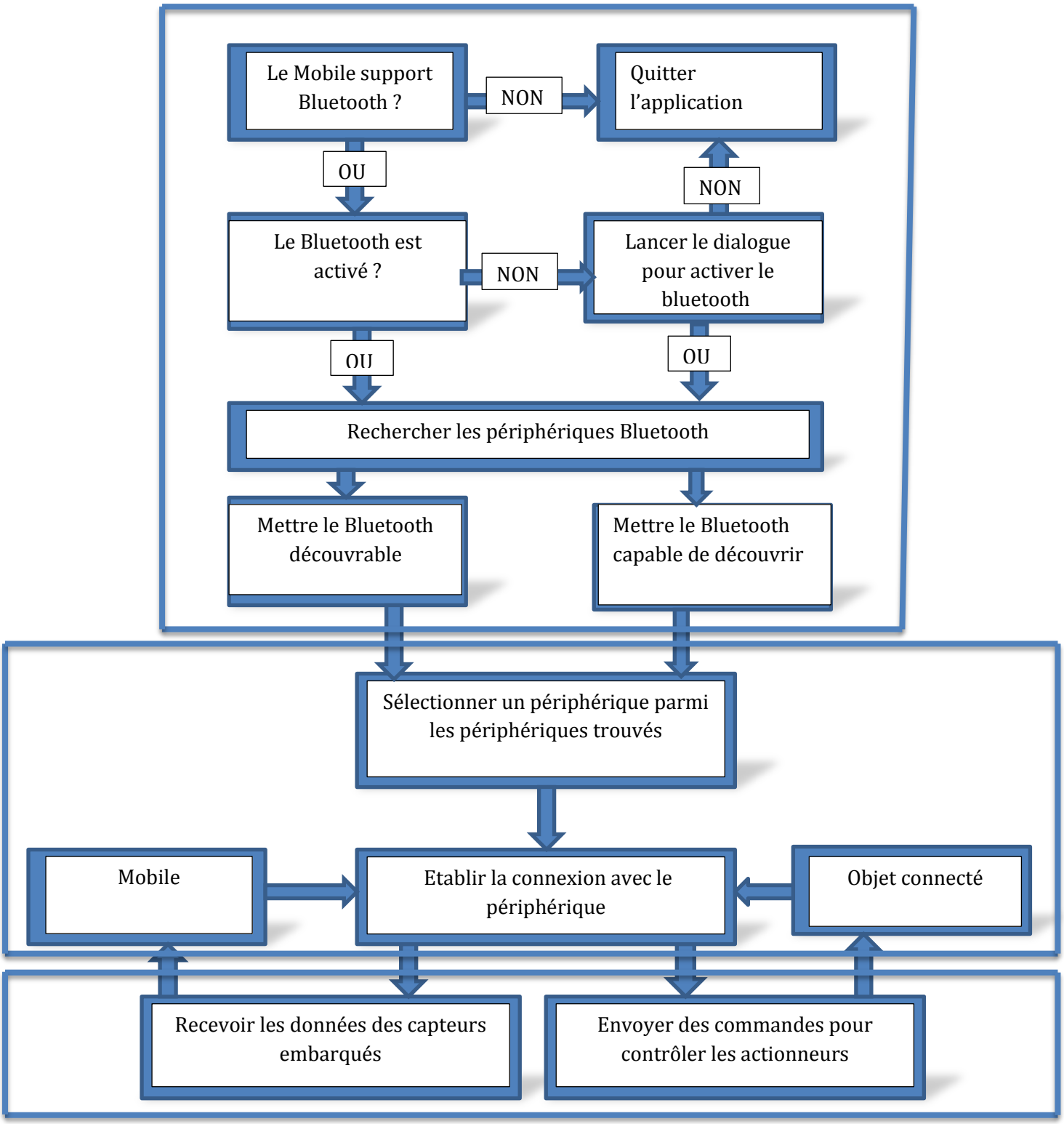

Figure 27:division de code Bluetooth

- Partie vérification et configuration de protocole Bluetooth
- Partie de connexion avec le périphérique
- Partie d'échange d'information

4.3. Création et utilisation de l'objet

A partir du code et son architecture Bluetooth, on a tiré toutes les informations de type variable lors d'une communication afin de développer un objet valable pour toutes les applications.

Les variables dans une communication Bluetooth sont le nom de pair périphérique Bluetooth et les informations lors d'échange.

La partie échange des informations doit configurer par l'utilisateur afin d'envoyer l'information exacte pour contrôler le système à base de communication Bluetooth.

Ce tableau résume les paramètres en entrées afin de créer et configurer au mieux l'objet.

Nom du paramètre	type	remarque	Exemple
messageToSend	String	La chaine de caractère à envoyer pour contrôler un objet connecté	HIGH LOW
buttonText	String	Le texte de Button	STOP GO
imageButton	String	L'image de Button	Image ()

Tableau 7: Les paramètres de configuration de l'objet Bluetooth

4.4. Mise en place de l'objet « BluetoothObject » dans le noyau android

La déclaration du composant sur ScreenDy se fait avant d'entamer le développement. Le nom de la classe, qui contiendra le code source du plugin par la suite, doit être le même que celui du composant dans ScreenDy. Dans le cas échéant, la « View Factory » ne sera pas capable d'instancier le composant et renvoyer une erreur (figure 28).

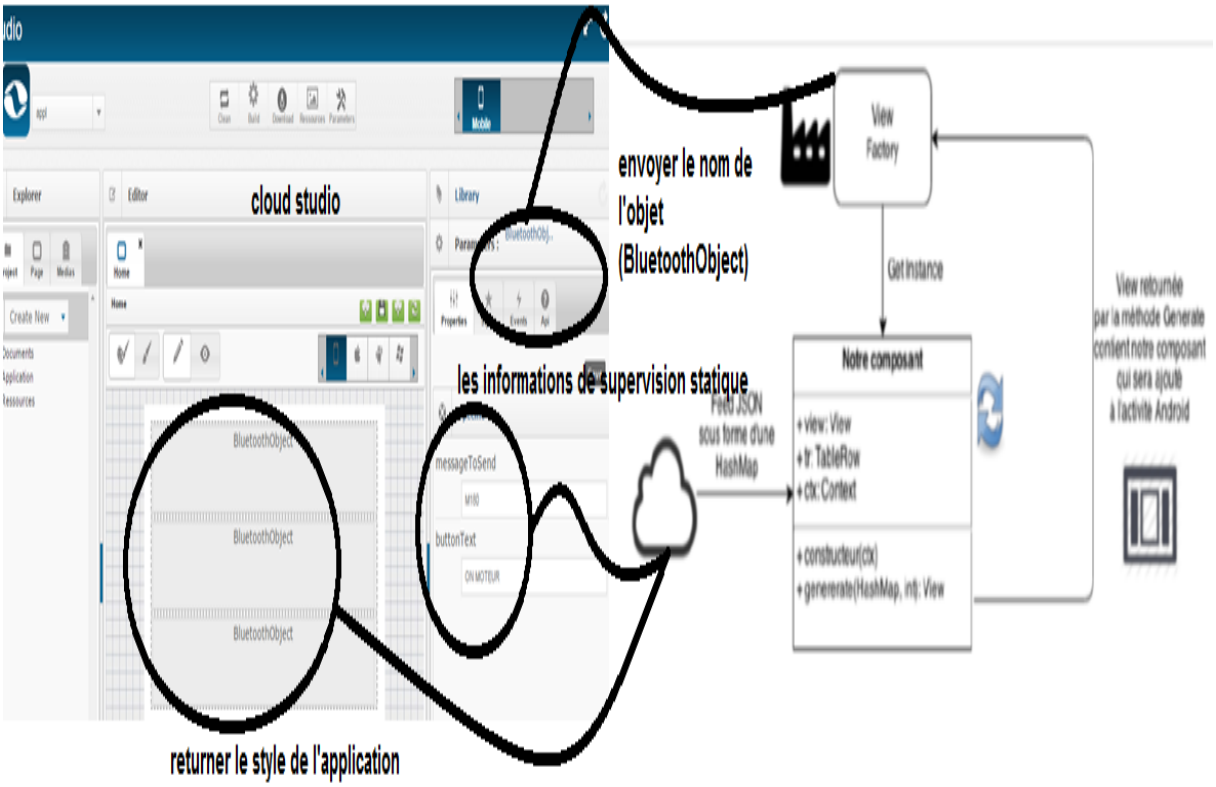

Figure 28: Composant Bluetooth et son noyau Android

4.5. Importance et utilité

L'utilisation de cet objet va permettre aux développeurs d'ajouter facilement la communication Bluetooth à des applications, et gagner de l'argent rapidement. Il a été conçu d'une manière configurable pour le mieux exploité.
A l'aide de cet objet, l'utilisateur a besoin seulement de 5min pour développer une application pour connecter et contrôler un objet connecté.

Conclusion

Le but de ce chapitre était la présentation de différentes étapes pour la généralisation de code Bluetooth afin de l'intégrer au studio pour être valable pour toutes les applications mobiles.

Chapitre 5

Test de bon fonctionnement

Ce chapitre comporte une description de la phase de test de bon fonctionnement de l'objet Bluetooth à partir de studio Screendy avec les prototypes réalisés.

5. Test de bon fonctionnement

5.1. Création d'une application type

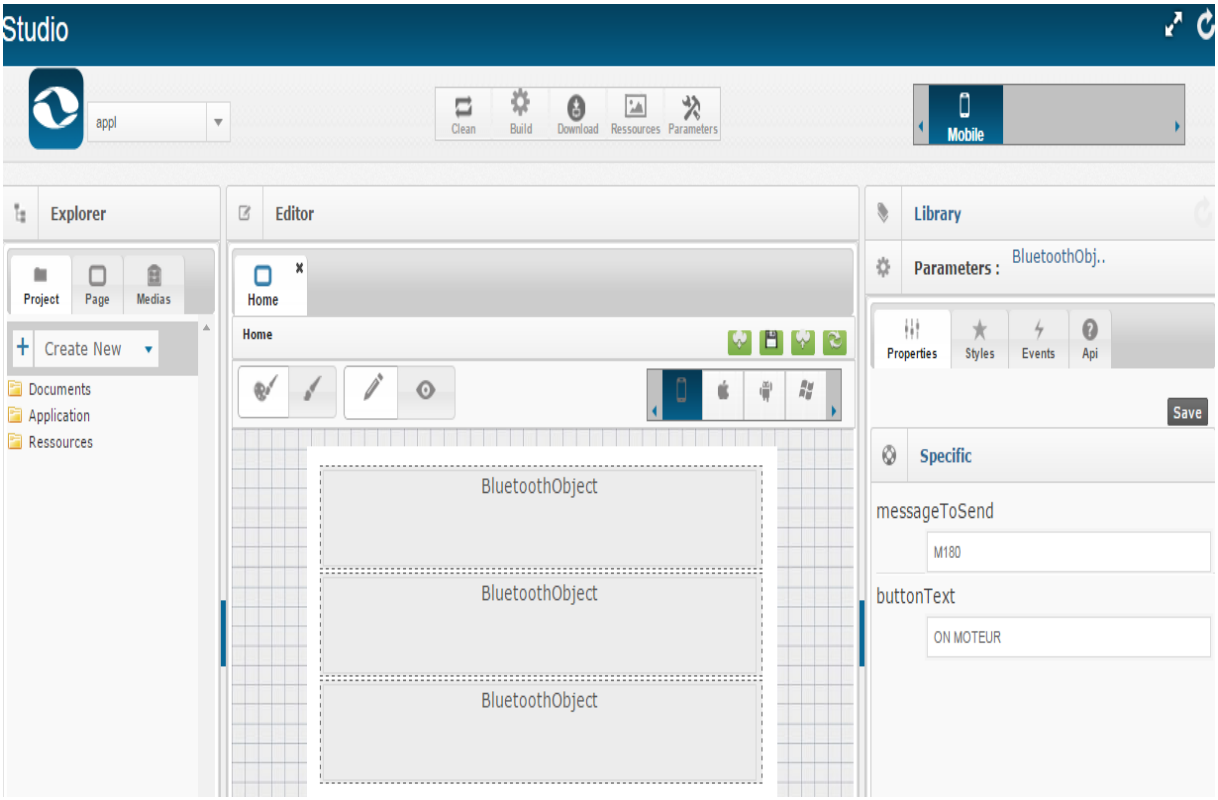

Figure 29:Création de l'application à base de l'objet Bluetooth

Ouvrir le studio Screendy (figure 29).

Dans « library » chercher l'objet « BluetoothObject ».

Glisser l'objet.

Aller dans « specific » ➔ « messageToSend » et entrer la commande à envoyer.

Aller dans « specific » ➔ « buttonText » et entrer le nom de button.

Enregistrer et charger l'application dans le mobile.

5.2. Conseil de l'utilisation de l'application Screendy

✓ Au cas où la liste des périphériques est vide ou aucun périphérique trouvé :

- Vérifier le schéma du montage ;
- Cliquer une autre fois sur le Button pour relancer la recherche ;
- Sinon relancer l'application.

✓ Il faut attendre jusqu'à l'arrêt de recherche des périphériques.
✓ Cliquer sur le périphérique souhaité pour faire la connexion.

✓ Vérifier que la connexion est établie.

- Une fois la connexion établis entre le mobile et le module Bluetooth ;
- Cliquer sur le Button pour envoyer les commandes et recevoir les données.

Conclusion

Le but de ce chapitre était la réalisation d'une application à partir de studio afin de montrer l'utilité de l'objet développé et pour la mise en place d'un guide utilisateur qui aide à comprendre le fonctionnement de l'objet communicant dans le studio ScreenDy.

Conclusion générale

A l'issus de ce travail nous avons mis en place un noyau pour ScreenDy dédié aux objets connectés. Cette solution vise à générer des applications en se basant sur la configuration du noyau pour établir la communication via plusieurs protocoles.

Pour concrétiser ce projet, nous avons développé un composant qui est capable de chercher le périphérique communicant et de se connecter pour échanger les informations entre le système embarqué et les terminaux mobiles.
Pour réaliser ce projet, nous avons commencé par une étude détaillée des besoins requis. Ensuite, nous avons réalisé des prototypes fonctionnels des objets connectés. Par la suite une conception des objets s'est avérée nécessaire pour l'intégration au studio ScreenDy. Enfin nous avons abordé la phase de la réalisation d'une application à partir de studio pour tester le bon fonctionnement.

Nous avons réussi à terminer le projet représentant la solution ScreenDy pour résoudre le problème de multiplicité du système d'exploitation ainsi que la complexité de la communication avec les objets connectés.

En perspective du travail réalisé, nous envisageons d'ajouter des composants plus complexes et très avancés sur le développement des applications Android comme la communication inter-objets sans intervention humaine, et les composants de type intelligence artificiel.

Bibliographie

[1]Omar Alaoui, Media Mobility, 2010, media-mobility.com.

[2] Mehdi Alaoui screendy.com

[3] Internet des objets

http://fr.wikipedia.org/wiki/Internet_des_objets

[4]Objets connectés : ***Harris Interactive*** 3 juin 2014

[5] **Domotics**

http://www.touteladomotique.com/index.php?option=com_content&view=article&id=1030:par-ou-commencer-nd1-quel-protocole-choisir&catid=5:domotique&Itemid=89#.U6F75JYcyZ

[6] OV7670 Datasheet

Annexes

Annexe A: Studio

Annexe B: les codes sources

Annexe C: Camera

Annexe D: Capteur de température

Annexe E: Communication

A.1: Presentation

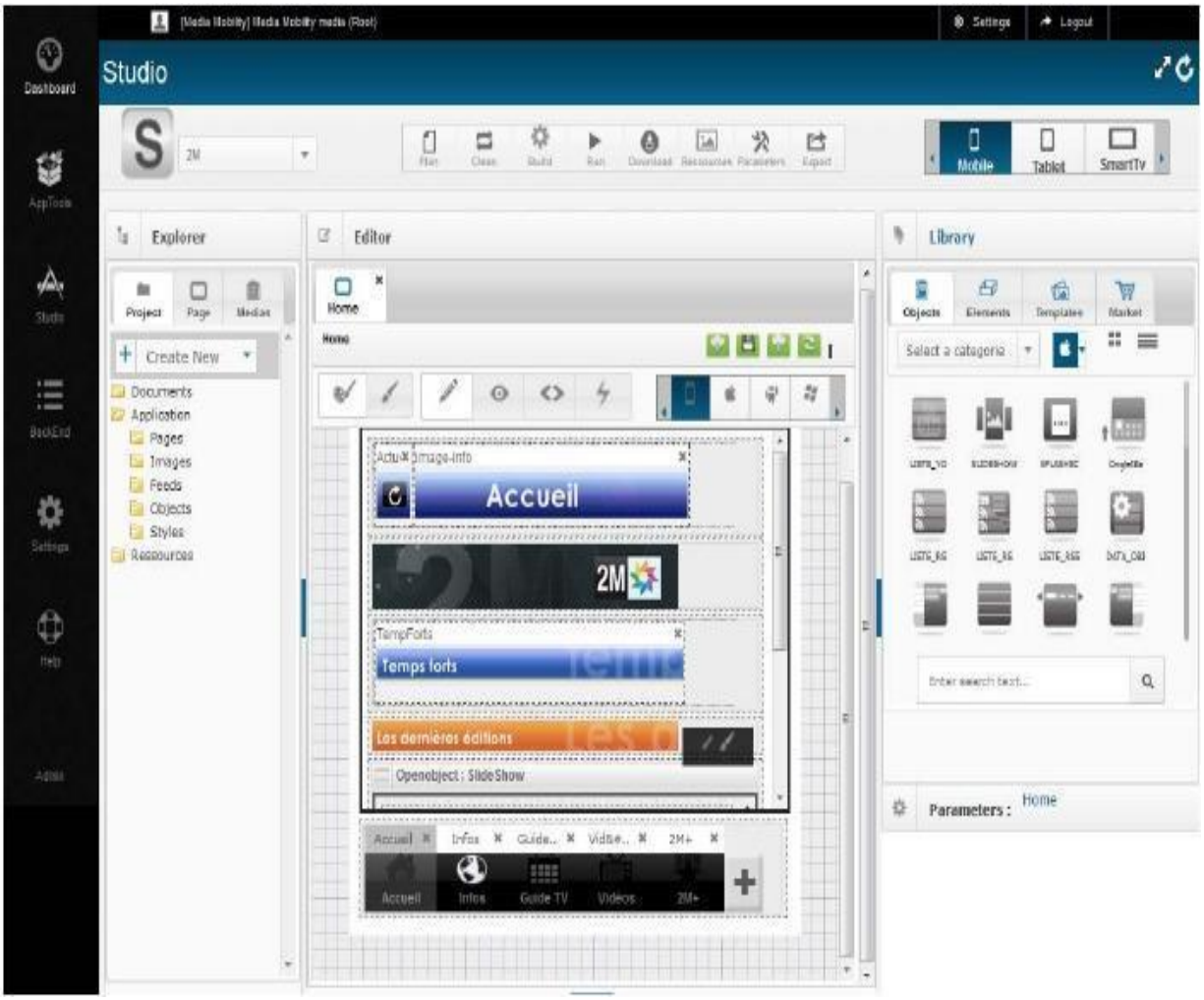

Figure 30: Présentation de Studio

A.2: Overview

-

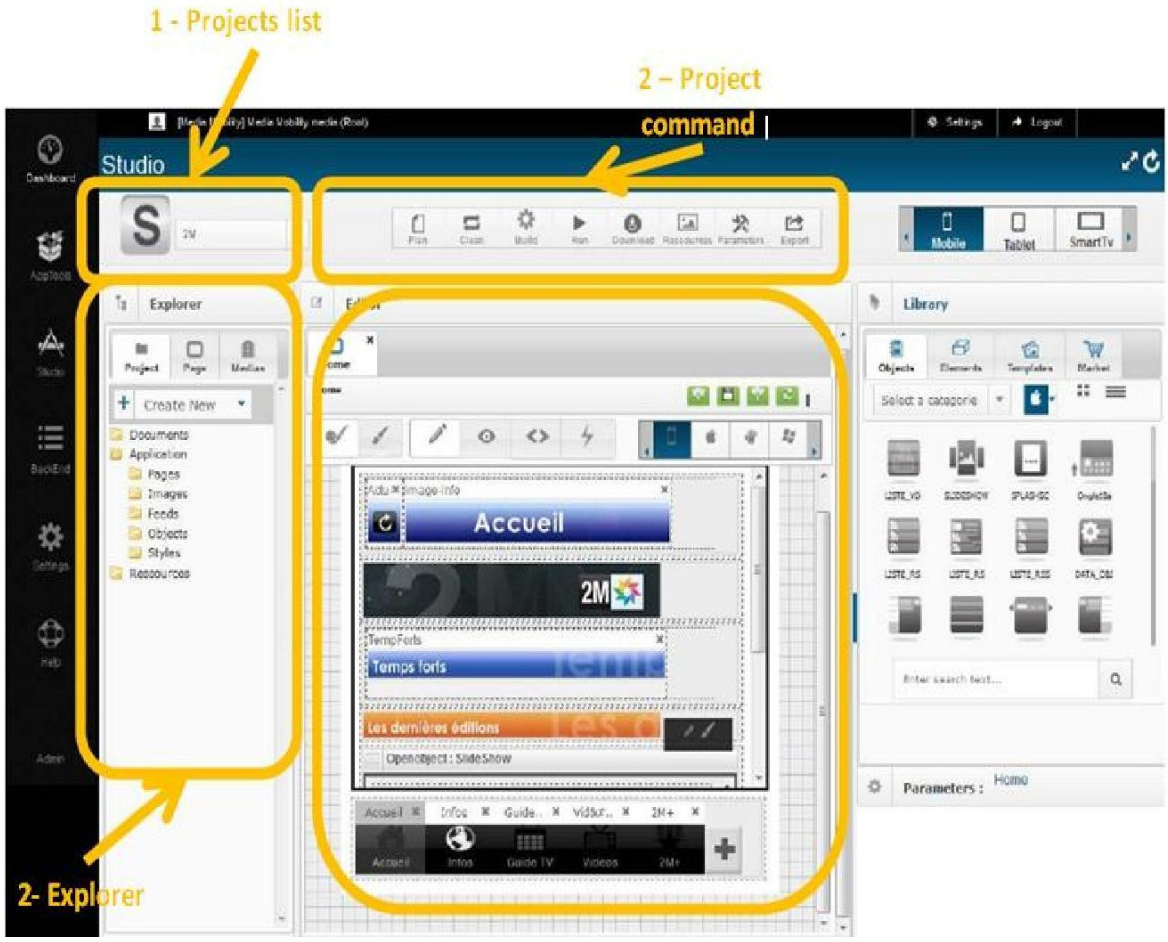

Figure 31: Overview

A.3:Project List

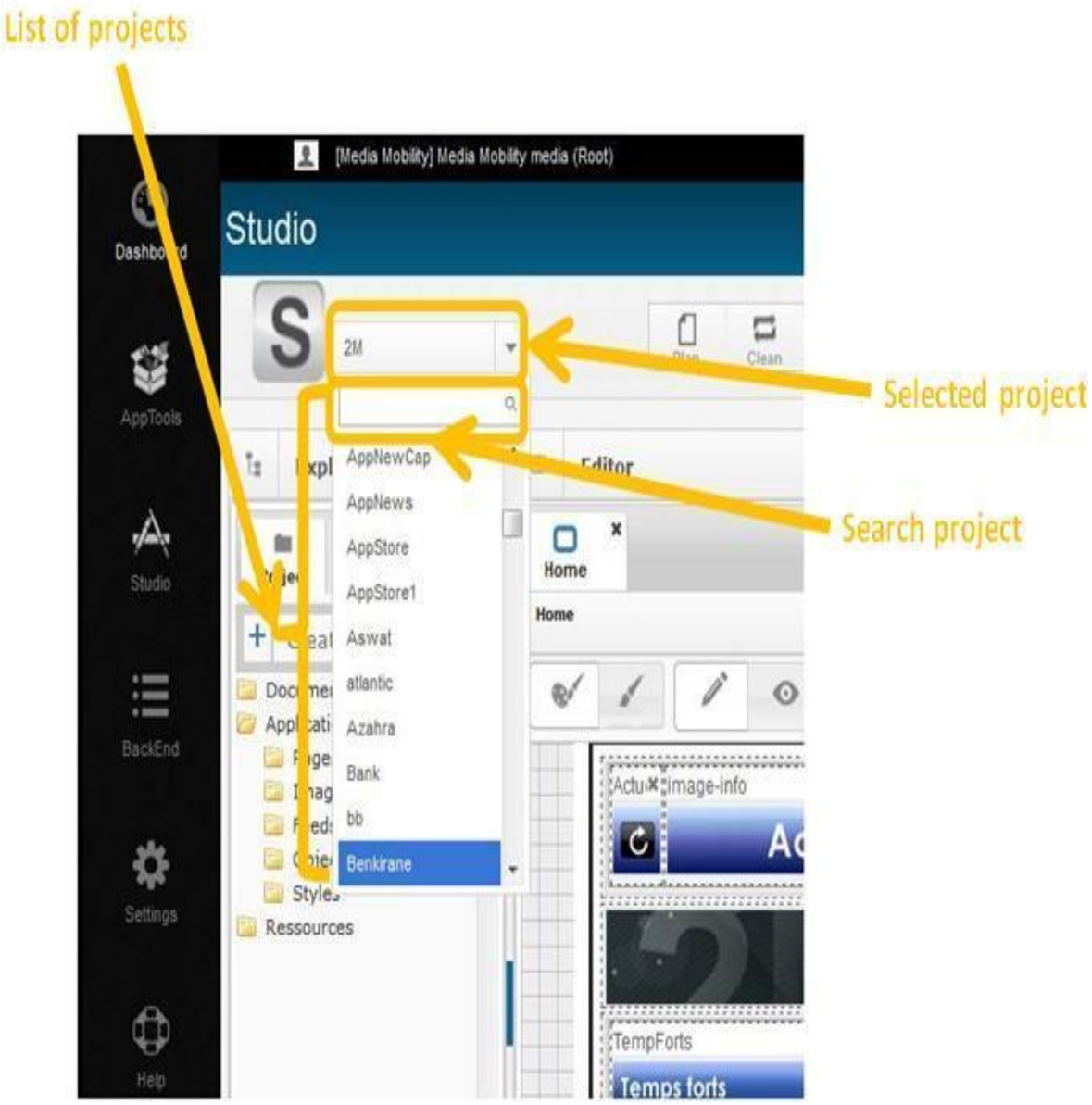

Figure 32: Project List

B.1 code Bluetooth Android

```
/**
 * l'objet bluetooth permet la communication avec les objets connectées on
 * utilisant la communication bluetooth developpé par krirou jilali 2015
 * jkrirou@hotmail.com Media Mobility
 */

@SuppressLint("NewApi")
public class BluetoothObject extends Widget {
    /*****************************************************************************************/
    private String TAG = "Bluetooth";

    private String messageToSend; // static configuration
    private String buttonText;
    // private String oneDevice;
    // private String imageButton;

    /*****************************************************************************************/
    String webServiceUrl;
    String Key;                          // data object webservice
    String methode = "GET";
    /*****************************************************************************************/
    Context context; // user application

    private static boolean starscan = true; // control bluetooth scan device
    private static boolean disconnect = false; // control bluetooth disconnexion
    /*****************************************************************************************/
    BluetoothDevice mydevice;
    BluetoothAdapter mBluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
    Set<BluetoothDevice> pairedDevices = mBluetoothAdapter.getBondedDevices(); // bluetooth
    private InputStream receiveStream = null;
    private OutputStream sendStream = null;
    private static BluetoothSocket mysocket = null;
    UUID MY_UUID = UUID.fromString("00001101-0000-1000-8000-00805f9b34fb");
    /*****************************************************************************************/
```

Figure 33: code Bluetooth Android

B.2 code capteur de temperature arduino

prog_ard

```
DeviceAddress insideThermometer = { 0x28, 0x94, 0xE2, 0xDF, 0x02, 0x00, 0x00, 0xFE };
DeviceAddress outsideThermometer = { 0x28, 0x6B, 0xDF, 0xDF, 0x02, 0x00, 0x00, 0xC0 };
DeviceAddress dogHouseThermometer = { 0x28, 0x59, 0xBE, 0xDF, 0x02, 0x00, 0x00, 0x9F };

void setup(void)
{
  // start serial port
  Serial.begin(9600);
  // Start up the library
  sensors.begin();
  // set the resolution to 10 bit (good enough?)
  sensors.setResolution(insideThermometer, 10);
  sensors.setResolution(outsideThermometer, 10);
  sensors.setResolution(dogHouseThermometer, 10);
}

void printTemperature(DeviceAddress deviceAddress)
{
  float tempC = sensors.getTempC(deviceAddress);
  if (tempC == -127.00) {
    Serial.print("Error getting temperature");
  } else {
    Serial.print("C: ");
    Serial.print(tempC);
    Serial.print(" F: ");
    Serial.print(DallasTemperature::toFahrenheit(tempC));
  }
```

Figure 34: code capteur de température

B.3 code camera arduino

Camera_OV0706_TEST §

```
#define chipSelect 10
#if ARDUINO >= 100
SoftwareSerial cameraconnection = SoftwareSerial(2, 3);
#else
NewSoftSerial cameraconnection = NewSoftSerial(2, 3);
#endif
camera_VC0706 cam = camera_VC0706(&cameraconnection);
void setup() {


#if !defined(SOFTWARE_SPI)
#if defined(__AVR_ATmega1280__) || defined(__AVR_ATmega2560__)
        if(chipSelect != 53) pinMode(53, OUTPUT); // SS on Mega
#else
        if(chipSelect != 10) pinMode(10, OUTPUT); // SS on Uno, etc.
#endif
#endif

        pinMode(7,INPUT_PULLUP);
        Serial.begin(9600);
        Serial.println("VC0706 Camera test");


        if (!SD.begin(chipSelect)) {
                Serial.println("Card failed, or not present");
                return;
        }
```

Figure 35: code caméra

C.1 Figure functional block diagram

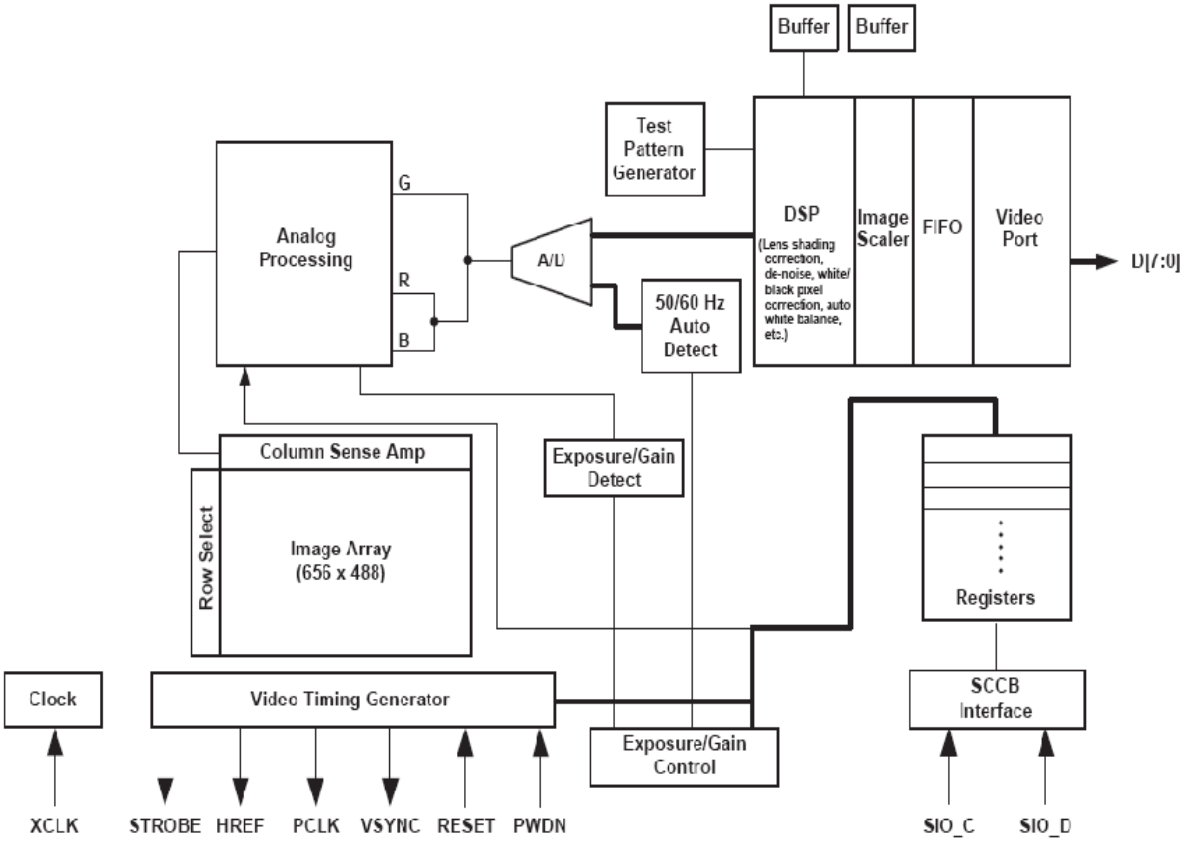

Figure 36: block diagram camera

C.2 Figure Pin description

Pin	Type	Description
VDD**	Supply	Power supply
GND	Supply	Ground level
SDIOC	Input	SCCB clock
SDIOD	Input/Output	SCCB data
VSYNC	Output	Vertical synchronization
HREF	Output	Horizontal synchronization
PCLK	Output	Pixel clock
XCLK	Input	System clock
D0-D7	Output	Video parallel output
RESET	Input	Reset (Active low)
PWDN	Input	Power down (Active high)

Figure 37: Pin description

D.1 Fonctionnement de DS 18B20:

Ces circuits possèdent un code unique sur 64 bits comme tous les circuits 1 Wire. Le code famille est h"28", suivi de 6 octets propre au circuit et d'un octet de CRC. La détection de présence de ce circuit se fait en envoyant l'impulsion de Reset, qui est un état bas pendant au moins 480 µs. Quand un circuit DS 18B20 est présent sur le bus il le signale en maintenant le bus à l'état bas pendant 60 à 240 µs. Toute transaction avec un tel circuit doit démarrer par un pulse de Reset suivi de l'envoi d'un commande ROM. On pourra après envoyer une commande de fonction propre à ce type de circuit. Si le circuit est seul sur le bus 1 Wire, la commande ROM peut être l'appel général SKIP ROM = h"CC". Si ce n'est pas le cas, il faudra connaître les 64 bits propre du circuit que l'on veut atteindre et utiliser la commande MATCH ROM = h'55" suivi des 8 octets du code. Une recherche préalable des 8 octets de code sera faite par la commande READ ROM = h'33' si le circuit est seul ou bien par SEARCH ROM = h"F0" s'il y a plusieurs circuits sur le bus.

- **Mémoire interne :**

Elle est constituée d'une zone RAM de 9 octets et d'une zone EEPROM non volatile de 3 octets.

- **Le bus 1 Wire :**

Le bus 1 WIRE de DALLAS, permet de connecter et de faire dialoguer entre eux des circuits sur un seul fil. Ce système de bus utilise un seul maître, qui pourra dialoguer avec un ou plusieurs esclaves. Toutes les commandes et données sont envoyées avec le bit LSB en tête.
Le fil unique du bus doit être tiré au +Vcc par une résistance de 4,7KΩ. L'état repos du bus est donc un état haut.

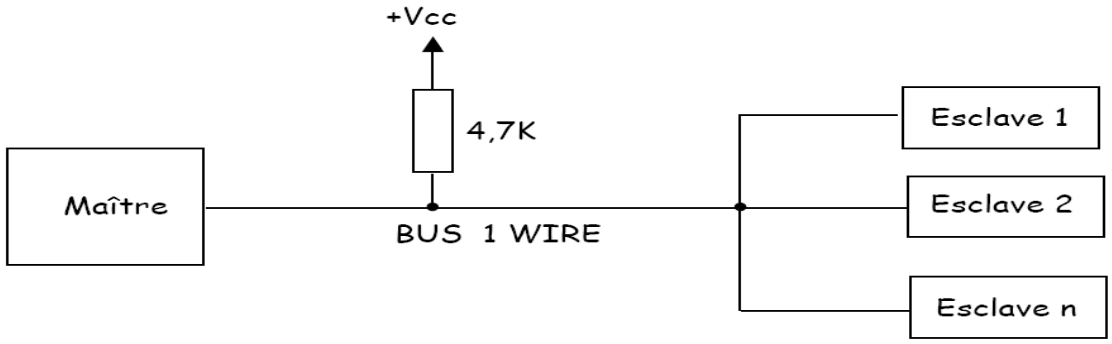

Figure 38: Le bus 1-wire

Si le bus est maintenu à l'état bas plus de 480 µs par le maître, tous les composants sur le bus sont remis à zéro. C'est le pulse d'initialisation ou de Reset. Après un délai de 15 à 60 µs, le ou les esclaves raccordés, forcent le bus à l'état bas pendant 60 à 240 µs pour signaler leur présence.

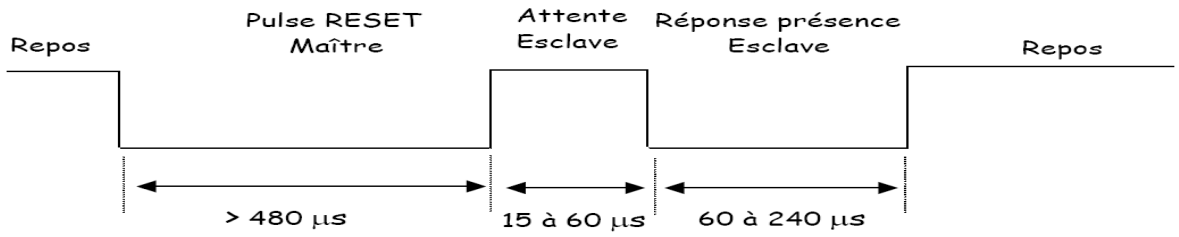

Figure 39: Communication entre maitre et esclave

Chaque circuit possède une adresse physique unique, gravée dans la puce à la fabrication. Cette adresse est constituée de 64 bits soit 8 octets. Le premier octet détermine le type de famille auquel appartient le circuit. Les 6 octets suivants, constituent le code propre du circuit. Le dernier octet est le CRC. C'est un octet de contrôle calculé à partir des 56 bits précédents.

E.1 Communication

Communication usart

L'USART de l'AVR permet d'utiliser l'ensemble de la communication en duplex, des mots de 5- à 9-bits (un mot de 8-bits = byte), 1 ou 2 bits de stop, trois modes paritaires et une grande variété de *baud rates*. Les micro-contrôleurs AVR typiques ont 2 interfaces USART, bien que certain n'ait pas d'USART. La transmission de données est réalisée un mot par cycle - le AVR convertit le mot qu'il reçoit de l'utilisateur en bits et le transmet indépendamment et vice et versa. L'utilisateur contrôle l'USART en lisant et en écrivant une configuration, le statut et les données registres.

Chaque option de configuration a des registres correspondants, qui sont relativement faciles à configurer en utilisant la fiche technique. La vitesse de transmission, cependant est un peu plus difficile à configurer. Le signal d'horloge pour la transmission de données est généré à partir de la propre horloge du contrôleur et l'utilisateur peut spécifier un nombre de 1 à 4096, par lequel le cycle d'horloge du contrôleur est divisé. Le résultat est de plus divisé par 2, 8 ou 16, selon le mode. Le problème est que les fréquences d'horloge ne peuvent pas toutes être divisées pour que le résultat soit une vitesse de transmission standard. Avec quelques fréquences, la vitesse de transmission peut différer de du standard avec une hausse de 10 %. Les fiches techniques de l'AVR contiennent des tables avec des fréquences d'horloge typiques, des vitesses de transmission, le multiplicateur nécessaire pour atteindre cette vitesse de transmission et l'erreur de calcul possible.

Puisque la transmission de données a lieu indépendamment du processeur et beaucoup plus lentement, il est nécessaire de confirmer que l'interface est prête pour le mot suivant avant la transmission. Cela peut être fait en surveillant le bit prêt transmit par le buffer, qui a de l'importance si le buffer est prêt à accepter un nouveau mot ou non. Le contrôleur commence par le bit prêt valable. Aussitôt qu'un mot est transmis et que le buffer est vide, le bit prêt est modifié.

L'arrivée d'un mot est signifiée aussi par un bit de statut spécial. En plus de cela, il y a des bits de statut pour signifier des erreurs de cadrage, des erreurs paritaires et des dépassements de capacité du buffer. Il arrive que le buffer soit en dépassement de capacité, quand le dernier

mot doit encore être lu à partir du buffer alors qu'un nouveau arrive - c'est pourquoi il est toujours important de lire les mots entrants dans le programme dès que possible, par exemple en utilisant un interruption. Il y a trois raisons d'interruptions possibles: transmettre que le buffer est prêt, transmission réussie et réception réussie.
Les buffers de transmission et de réception sont des registres séparés, mais ils transmettent les mêmes adresses de mémoires et les mêmes noms. En écrivant dans un registre de partage, la donnée est stockée dans un buffer de transmission pour la lire ensuite à partir du buffer de réception. En utilisant un mot de 9-bits, le neuvième bit est transmit et lu en utilisant l'un des registres de configuration.

Communication radio

C'est elle qui s'occupe de l'émission et de la réception des ondes radio. Elle définit les caractéristiques telles que la bande de fréquence et l'arrangement des canaux, les caractéristiques du transmetteur, de la modulation, du récepteur, etc.
Le système Bluetooth opère dans les bandes de fréquences ISM* (*Industrial, Scientific and Medical*) 2,4 GHz dont l'exploitation ne nécessite pas de licence vu la faible puissance d'émission et le risque faible d'interférences. Cette bande de fréquences est comprise entre 2 400 et 2 483,5 MHz. Un *transceiver* à sauts de fréquences est utilisé pour limiter les interférences et l'atténuation.
Deux modulations sont définies : une modulation obligatoire utilise une modulation de fréquence binaire pour minimiser la complexité de l'émetteur ; une modulation optionnelle utilise une modulation de phase (PSK à 4 et 8 symboles). La rapidité de modulation est de 1 Mbaud pour toutes les modulations. La transmission duplex utilise une division temporelle.
Les 79 canaux RF sont numérotés de 0 à 78 et séparés par 1 MHz en commençant par 2 402 MHz. Le codage de l'information se fait par sauts de fréquences et la période est de 625 µs, ce qui permet 1 600 sauts par seconde.