Projet Sous-Bock Connecté

SBOCO

PROJET EI4 AGI - 2015/2016

Alan Corduan, Julien Regnier, Adrien Boigné, Scot Scriven

EI4 - 2015/2016

Sommaire

INTRODUCTION

L'éclosion exponentielle de nouvelles technologies métamorphose les habitudes de vie à travers le monde. Certains outils qui n'étaient que superflus hier deviennent de réelles nécessités aujourd'hui et la manière d'appréhender un domaine d'activité en est fortement impactée. Au-delà des relations entretenues par les Hommes depuis des millénaires, la communication s'élargit dorénavant au monde des objets. Tous les secteurs se retrouvent dopés par l'aide apportée par cette recrudescence d'adresses MAC et le monde des objets connectés constitue véritablement une source intarissable pour l'avenir.

Loin de toutes ces considérations technologiques, l'univers des cocktails semble surtout chercher l'innovation à l'intérieur du verre au lieu de repenser la manière même de les créer. L'idée du sous-bock connecté s'inscrit donc dans l'idée d'apporter une véritable plus-value pour la prise de décision dans la réalisation de multiples recettes.

Amorcer un tel projet a nécessité une certaine réflexion concernant notre propre organisation afin de privilégier 2 paramètres qui nous semblaient essentiels : l'efficacité et l'apprentissage. Dans cette optique et devant l'ampleur des tâches à effectuer, nous avons rapidement opté pour une approche sans rôle prédéfini. Qu'importe notre expérience et nos compétences, nous nous sommes personnellement incités à manipuler et comprendre les différents axes de développement de notre projet. De l'électronique à la programmation mobile, chaque membre du groupe a disposé de micro-tâches à accomplir.

A travers ce compte rendu, nous traiterons les démarches qui ont jalonnées la réalisation du sous-bock à travers 4 grands paragraphes. Après avoir expliqué les spécifications et la répartition des tâches, nous expliquerons les étapes de programmation et la manière dont nous avons abordé la partie software de notre projet. Nous décrirons ensuite les diverses phases qui se sont inscrites dans la fabrication du prototype électronique en éclaircissant le rôle de chaque composant. Bien que nous présenterons ce rapport selon cet ordre donné, les 2 tâches (programmation et conception sous-bock) étant bien distinctes, nous avons - en réalité - travaillé parallèlement sur ces dernières. La liaison entre les 2 entités qui constituent notre projet (application et sous-bock) sera le fruit de notre quatrième chapitre qui permettra de commenter le langage commun que nous avons pu mettre en place.

ORGANISATION DU PROJET

Spécifications

L'application mobile "Sous-bock App" sera en interaction avec le sous-bock. Il aura pour but de fournir et récupérer les informations de l'objet. Il permettra à l'utilisateur de choisir un cocktail puis de gérer le dosage de celui-ci.

Contraintes techniques:

- l'application sera hybride, c'est à dire qu'elle sera multiplateforme (Android, IOS, Windows Phone), utilisation de cordova/ionic framework.
- utilisation du Bluetooth pour l'échange de données
- utilisation d'internet pour la récupération des cocktails ⇒site web

Fonctionnalités attendues :

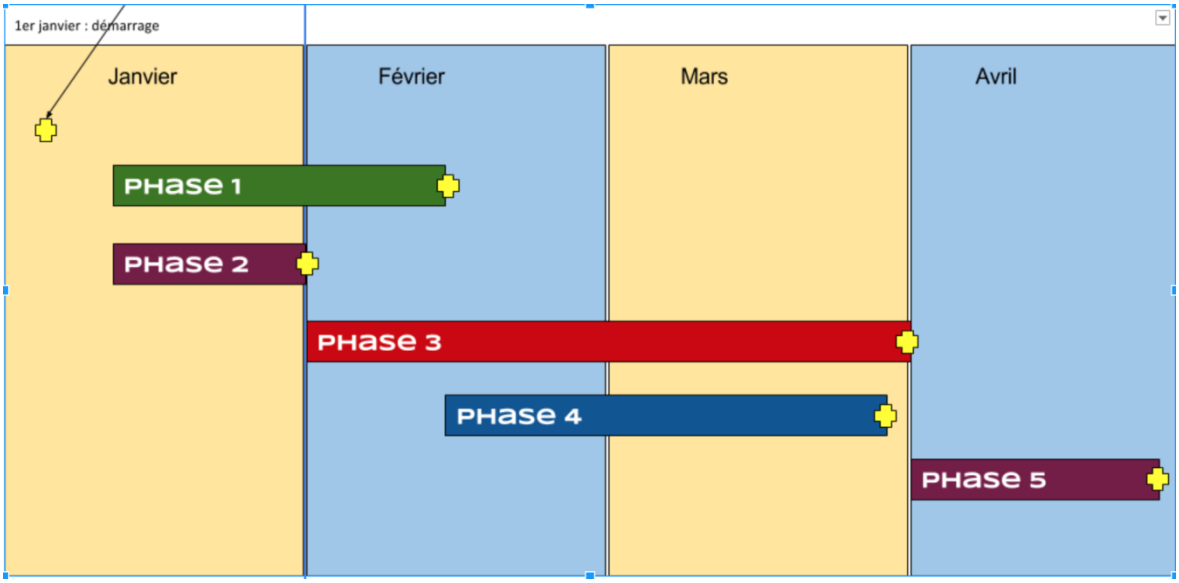
- être capable de communiquer avec le sous-bock :
    - Le sous bock nous envoie les données de manière régulière lorsque l'on donne son choix de cocktail.
    - Gérer la création des cocktails depuis le téléphone : utilisation de la tare, étapes, fin
- l'utilisateur doit pouvoir choisir un cocktail pour le sous-bock
- sauvegarder 1 ou plusieurs sous bock à l'application
- l'utilisateur doit être capable d'ajouter/enregistrer les ingrédients d'un cocktail (sans sous-bock)
- l'utilisateur doit être capable d'ajouter un cocktail en mode "freestyle" (avec sous-bock)
- mettre en favoris les cocktails
- avoir son historique
- prendre en photo ou récupérer une photo pour l'ajout un cocktail
- être capable de rechercher les cocktails (par nom, date et type)
- système de suivis du cocktail en temps réel (interface visuel, ex : verre qui se remplit au fur et à mesure)
- conversion ml/l en g

Fonctionnalités futures :

- être capable de partager avec des personnes utilisant l'application (partage de cocktails, de jeux à boire, ou même de rendez-vous de soirées...)
- générer des cocktails aléatoires (en fonction des goûts/préférences et avec des conditions pour éviter les cocktails imbuvables)
- voir les consommations et cocktails de ses amis
- système d'amis
- affichage de l'alcoolémie du verre

Gestion de projet

Voici le planning prévisionnel écrit au début du projet. Nous pouvons voir les jalons et les phases que nous avons pensé être importants. Les détails des jalons et phases sont listés ci-dessous.



GESTION DES PHASES		
Intitulé	*Descriptions des phases (macro tâches)*	*Livrables*
PHASE 1 :	Conception de l'objet sous-bock	Objet sous-bock réalisé. (Système de balance au point, carte Arduino + module Bluetooth programmés)
PHASE 2:	Développement de l'IHM (front-end) sans les algorithmes de gestions	Un 1er jet graphique de l'appli : couleurs, design, interaction, effets possibles.
PHASE 3:	Développement du back-end (algorithmes, gestions des BDD, etc...)	Application opérationnel sur toutes les tâches internes (ex : ajout de cocktails, récupération de nouveau cocktails, etc...)

PHASE 4:	Développement de la communication sous-bock/appli	Documentation sur le système d'échange des données. Et Application mobile arrivant à communiquer avec le sous-bock
PHASE 5:	Test unitaire / Débogage / Puis amélioration de l'UX/UI si possible	Sous bock et application mobile fonctionnelle (dans sa version finale)

Livrables	*Responsable*	*Etat*	*Progrès.*	*Date*	
				prévue	*réalisée*
Jalon 1					
Objet sous-bock réalisé. (Système de balance au point, carte Arduino + module Bluetooth programmés)	Adrien, Julien, Scot	terminé	100%	15/02/2016	22/03/2016
Jalon 2					
Un 1er jet graphique de l'appli : couleurs, design, interaction, effets possibles.	Adrien, Julien, Scot, Alan	terminé	100%	31/01/2016	03/02/2016
Jalon 3					
Application opérationnel sur toutes les tâches internes (ex : ajout de cocktails, récupération de nouveau cocktails, etc...)	Adrien, Julien, Scot, Alan	terminé	100%	31/03/2016	16/03/2016
Jalon 4					
Documentation sur le système d'échange des données. Et Application mobile arrivant à communiquer avec le sous-bock	Adrien, Julien, Scot, Alan	terminé	100%	21/03/2016	22/03/2016

Jalon 5					
Sous bock et application mobile fonctionnelle (dans sa version finale)	Adrien, Julien, Scot, Alan	terminé	100%	18/04/2016	06/04/2016

Nous pouvons remarquer ci-dessus que certaines phases ont été retardées et d'autres avancées. Ainsi l'écart le plus important fût la conception du sous-bock, en effet nous avons dû faire face à un problème électronique au niveau de l'amplificateur d'instrumentation (voir la partie "Module HX711"). Hormis cela, le planning fût assez bien respecté à 2-3 jours près.

PROGRAMMATION DE L'APPLICATION

Les différents types d'applications

Application native

Une application native est une application mobile développée pour un des systèmes d'exploitation utilisés par les smartphones et tablettes (iOS, Android, Windows Phone). Elle est développée avec un langage spécifique à son système d'exploitation et l'utilisation de l'application est limitée à ce même système d'exploitation.

Développer une application native permet d'utiliser la mémoire du smartphone ainsi que toutes les fonctionnalités liées au système d'exploitation visé (GPS, accéléromètre...). Cela permet également de proposer des applications plus riches en termes de fonctionnalité, de qualité, de performance et de résolution.

Application web

Une application web est une application mobile développée en HTML accessible par le biais d'un navigateur Internet pour téléphone mobile.

Elle utilise le navigateur du smartphone et ne nécessite pas forcément de télécharger l'application. Elle est accessible par tous les smartphones quelques soient leur marque et leur système d'exploitation.

Application hybride

Une application hybride combine des éléments HTML5 sous forme d'application web et des éléments d'une application native. Ceux-ci permettent d'utiliser les fonctionnalités natives des smartphones.

Une application hybride permet de réduire les coûts et délais de développement, en effet il suffit de développer qu'une seule fois et ensuite de générer l'application pour toutes les plateformes. Mais la qualité, la performance et la résolution de ces applications sont nettement inférieures à celles des applications natives.

Dans notre cas nous allons créer une application hybride, tout d'abord car c'est une exigence de notre de professeur référent mais aussi car cela permet de générer une application sur différentes plateformes telles qu'IOS, Android, Windows Phone en ayant programmé l'application avec un seul langage de programmation.

Les outils et langages de développement

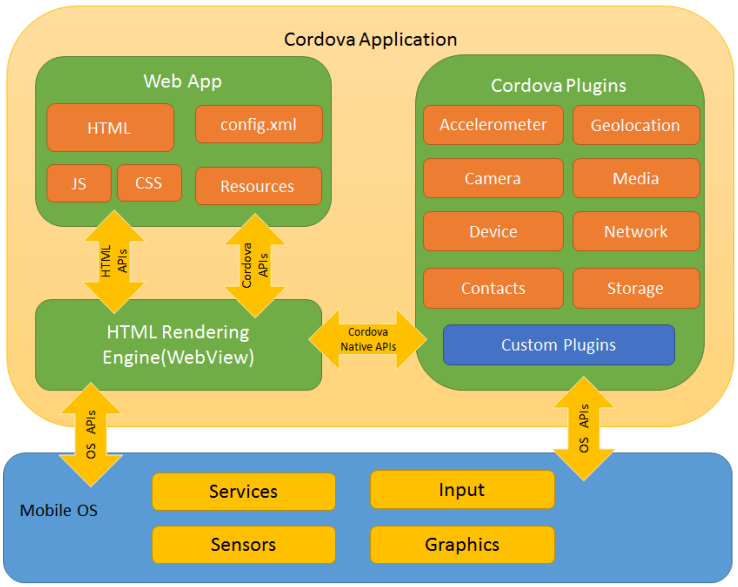
Cordova

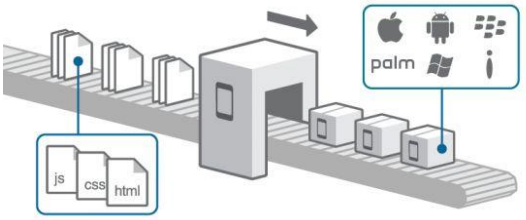
Présentation

Apache Cordova est un framework open-source qui est développé par la Fondation Apache. Il nous permet de créer des applications pour différentes plateformes, telles qu'Android, iOS, Ubuntu OS, Windows, avec HTML, CSS et le JavaScript. Ce framework permet d'accéder à un certain nombre de fonctionnalités natives du smartphone/tablette comme par exemple l'appareil photo, notifications et bien d'autres.

Le fonctionnement

Cordova fonctionne comme un conteneur web permettant au développeur d'écrire son code en HTML, CSS et Javascript. Il utilise simplement une "WebView" disponible nativement sur toutes les plateformes et va interpréter et afficher notre code.





Architecture d'une application Cordova

Ionic Framework

Comme son nom l'indique Ionic est un framework qui est open source et gratuit. Il s'appuie sur AngularJS pour le "front-end" (partie utilisateur et visuelle de l'application) et sur Cordova pour le "back-end".

Pour pallier à certains problèmes au niveau performance de Cordova, Ionic a été développé avec peu de manipulation de l'arbre DOM, aucun jQuery et des transitions/animations utilisant l'accélération matérielle.

Le développement grâce à ce Framework devient plus facile et plus structuré avec AngularJS (voir paragraphe "AngularJS"). Le SDK d'AngularJS permet de développer une application riche et robuste.

En plus de toutes ces fonctionnalités, Ionic nous offre un ensemble de "composants" de design permettant non seulement de rendre l'application "magnifiquement beau" mais aussi de faciliter les développeurs ayant peu de compétences en design.

De plus une Interface de commande(CLI) est fournie nous permettant de créer, générer, tester et déployer une application sur n'importe quelle plateforme.

Python & Scrapy

Python est un langage de programmation orienté objet open source et multiplateformes. Scrapy lui, est un framework open source écrit en python permettant l'extraction de données contenu dans les sites web.

AngularJS

AngularJS est un framework Javascript gratuit et open-source fondé sur l'extension du langage HTML par de nouvelles balises et attributs.

Voici certains concepts d'AngularJS :

- Une architecture MVC (Modèle-Vue-Contrôleur) qui consiste à avoir une stricte séparation entre les données (M), la présentation des données (V), et le actions que l'on peut effectuer sur ces données (C).
- Le Data Binding qui consiste à lier des variables du Javascript au HTML.
- L'Injection de dépendances permet de ne plus se soucier d'instancier les dépendances (modules)
- La manipulation du DOM grâce à des directives.

Git et Github

Git est un logiciel de gestion de versions décentralisé. Ce logiciel permet de gérer des codes sources en permettant de suivre l'évolution d'un fichier ligne par ligne.

Voici les utilités principales de ce logiciel :

- Il permet de retenir chaque modification effectuée et connaître les commentaires de ces modifications.
- Facilite le travail à plusieurs sur un même projet, en effet il est possible de fusionner les modifications avec celles des autres personnes sans risque de supprimer leur travail. Il n'y a donc pas perte d'information.

La gestion de versions décentralisée consiste à voir l'outil de gestion de versions comme un outil permettant à chacun de travailler à son rythme, de façon désynchronisée des autres, puis d'offrir un moyen à ces développeurs de s'échanger leurs travaux respectifs.

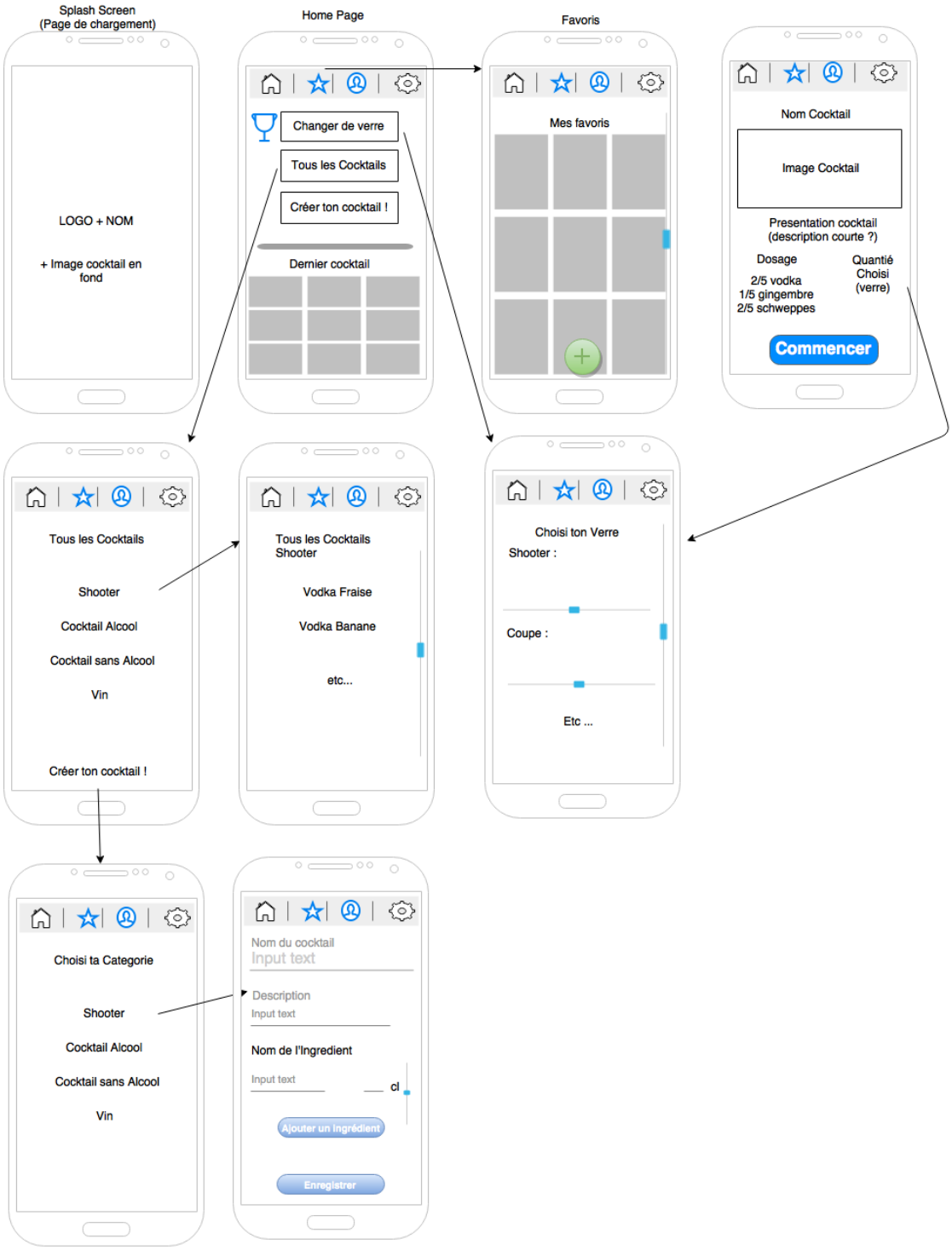
Github est un site permettant à n'importe qui de déposer un projet informatique (code source) grâce au logiciel de gestion de versions Git. La plupart des projets qui sont sur Github sont en libre accès, tout le monde peut donc récupérer les sources d'un projet pour les inclure dans leur propre projet ou bien pour les améliorer d'une manière ou d'une autre.

Le site offre de nombreuses fonctionnalités habituellement retrouvées sur les réseaux sociaux comme les flux, la possibilité de suivre des personnes ou des projets ainsi que des graphes de réseaux pour les dépôts. Github offre aussi la possibilité de créer un wiki et une page web pour chaque dépôt. Le site offre aussi un logiciel de suivi de problèmes.

Démarche et fonctionnement

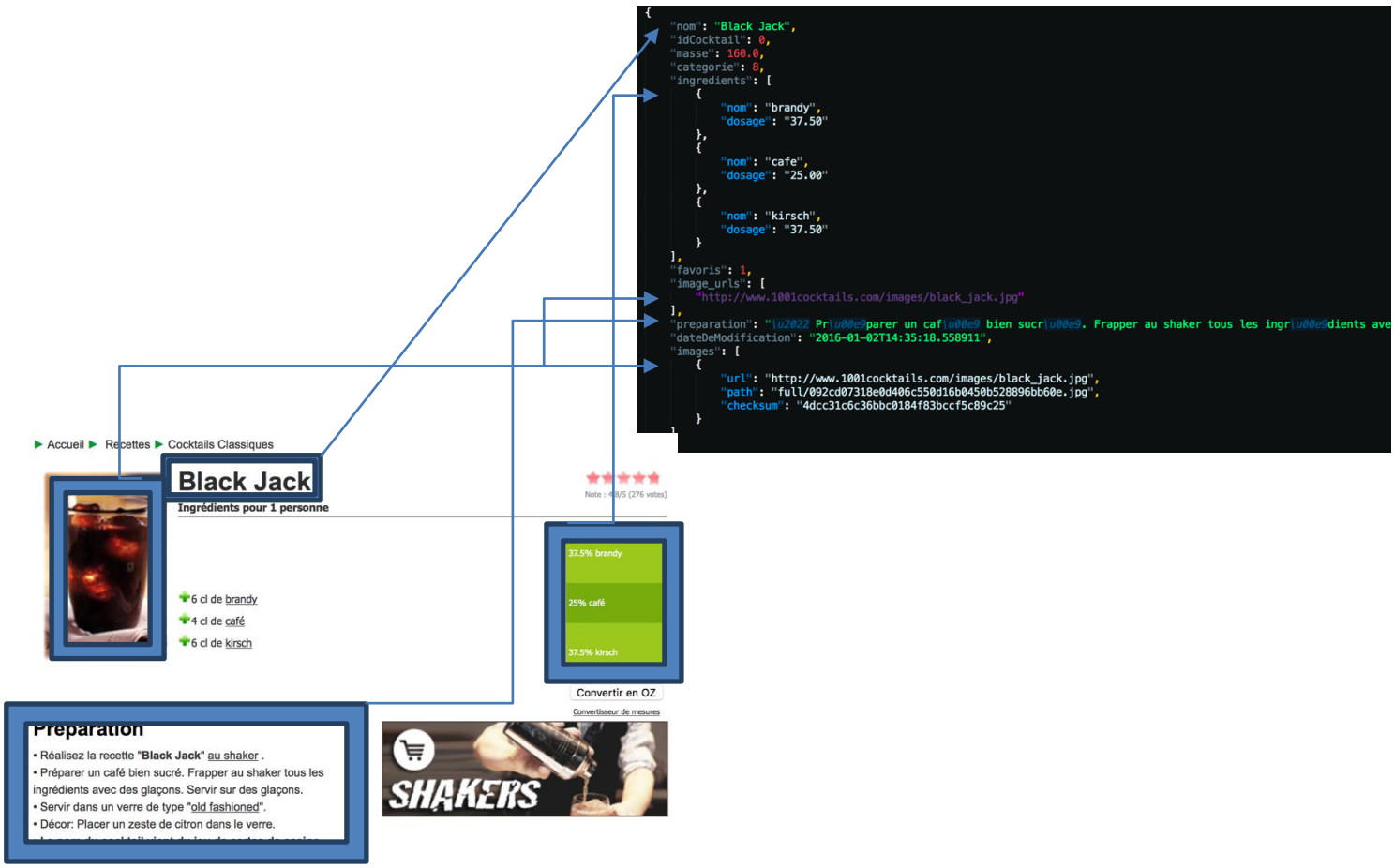
Avant de commencer à décrire le fonctionnement de notre application, il nous a fallu tout d'abord définir son maquettage. Puis apprendre le fonctionnement des langages, outils et frameworks cités plus haut.

Maquettage

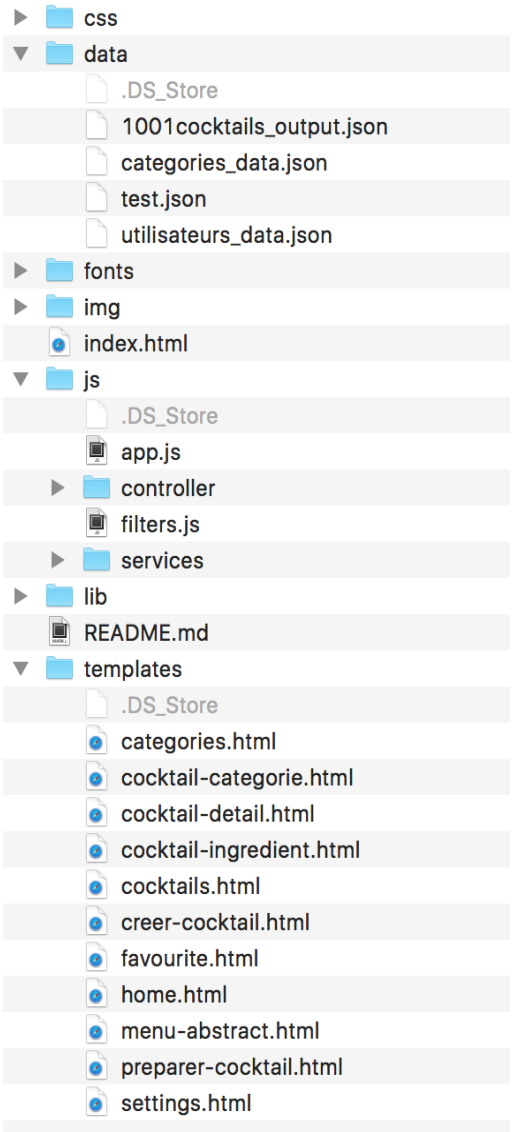


Structure du code

Pour pouvoir fournir des cocktails de base dans notre application il nous a fallu faire du « web scraping ». Pour cela nous avons pris le site : http://www.1001cocktails.com et ainsi nous avons pu récupérer les données de chaque cocktail en format json. L'utilisation de python et du framework « Scrapy » nous a permis d'automatiser et de rendre plus efficace le processus.



L'application est écrite en javascript, html, css avec Cordova et le framework Ionic. Ce framework nous permet de structurer notre code de manière Modèle Vue Contrôleur (Voir la partie sur AngularJS).



- **Css** : dossier contenant les fichiers de styles de l'application.
- **Data** : dossier contenant les fichiers de données en format json
- **Fonts** : dossier contenant les fichiers de style des polices
- **Img** : dossier contenant les images
- **Index.html** : fichier d'amorçage de l'application
- **Js** : dossier contenant le app.js (fichier de configuration), le dossier des contrôleurs, le dossier des services (modèles)
- **Lib** : dossier contenant les librairies (camera…)
- **Templates** : dossier contenant les fichier html, c'est à dire la structure d'une page (les vues)

Pour récupérer le projet contenant l'application, le python (web scraping) et le programme arduino, il faut aller sur github et le récupérer sur ce lien : https://github.com/aboigne/sousbockistia

REALISATION PROTOTYPE ELECTRONIQUE

Système balance

Sur le prototype que nous avons réalisé, seul le plateau est en interaction avec l'objet que nous souhaitons peser. Ce plateau est issu d'une balance et comprend tout le système électronique nécessaire qui nous permet d'acquérir une masse. La description des composants qui interviennent dans ce processus va donc nous permettre de comprendre le fonctionnement global du système afin de pouvoir traiter par la suite les données issues de la balance.

Jauges de déformation

La complexité d'une balance réside dans la nécessité de transformer une donnée physique en information électrique, traitable numériquement. Cette première étape est assurée par des jauges de déformation (ou jauges de contrainte). Une telle jauge agit comme une résistance dont la valeur varie en fonction de sa déformation. En effet, un fil enroulé opère comme une résistance grâce à 2 paramètres : sa longueur et sa résistivité. Ainsi, lorsque l'on modifie sa longueur en y exerçant un effort, la valeur de la résistance totale s'en trouve impactée.

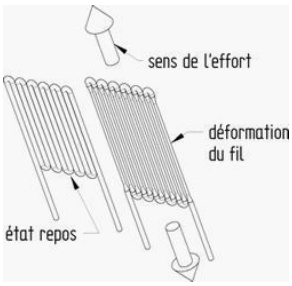
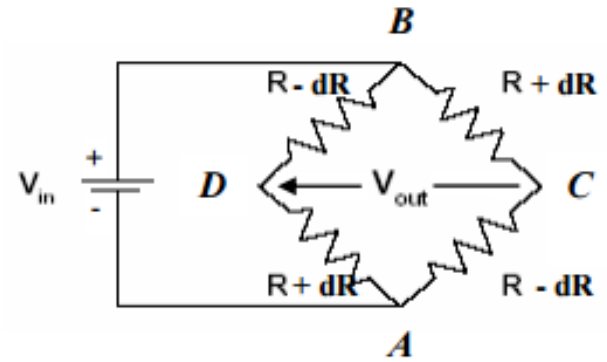


Illustration du fonctionnement d'une jauge de déformation

Cependant, nous pouvons aisément remarquer que la déformation exercée sur le fil n'entrainera jamais d'immense variation en termes de résistance : les informations que nous pourrons extraire de ces capteurs seront vraiment minimes. Il est donc nécessaire de redonder ces données en plaçant plusieurs jauges de déformation à des endroits stratégiques sur la plateforme en contact avec l'objet à peser. Ces données pourront ensuite être amplifiées pour pouvoir être traitées efficacement. Le branchement des capteurs (résistances) entre eux intervient dans cette optique d'intensification de l'information : c'est le pont de Wheatstone.

Pont de Wheatstone

L'objectif du pont de Wheatstone est de pouvoir extraire une tension de manière précise malgré de faibles variations des résistances. Le schéma de branchement des jauges (résistances) est donc le suivant :



Quelques calculs s'appuyant sur les lois élémentaires d'électricité devraient nous permettre de mieux comprendre le rôle de ce pont :

- On souhaite en premier lieu connaitre la valeur du courant i qui circule dans les branches BDA et BCA :

  V_{in} - (R-dR)i - (R+dR)i = 0 (car la chute de tension entre B et A est de V_{in})

  $\Rightarrow i = \frac{Vin}{2R}$

- On cherche maintenant à connaitre la chute de tension entre B et D :

  $V_{BD} = (R-dR)\frac{Vin}{2R}$ (loi d'Ohm : U = Ri avec i trouvé précédemment)

- On suit le même raisonnement pour la chute de tension entre B et C :

  $V_{BC} = (R+dR)\frac{Vin}{2R}$

- On calcule la valeur de V_{out} :

  $V_{out} = V_C - V_D$

  Avec $V_C = V_{in} - V_{BC}$ et $V_D = V_{in} - V_{BD} \Rightarrow V_{out} = V_{in} - V_{BC} - (V_{in} - V_{BD})$

  $\Rightarrow V_{out} = V_{BD} - V_{BC}$

  $\Rightarrow V_{out} = (R-dR)\frac{Vin}{2R} - (R+dR)\frac{Vin}{2R} \Rightarrow \underline{V_{out} = -dR * V_{in}}$

Cette dernière équation met en exergue l'importance du pont de Wheatstone pour notre cas. En effet, les variations de résistances étant très faibles pour les jauges de contraintes, un tel système nous permet de les amplifier en agissant sur la tension d'entrée V_{in}. Ainsi, plus V_{in} sera important, plus les petites variations seront visibles à travers la sortie V_{out}.

Module HX711

La sortie de la balance étant une différence entre 2 tensions très similaires, il nous est maintenant nécessaire de l'amplifier et de la convertir en une sortie digitale pour qu'elle puisse ensuite être traitée numériquement.

De nombreuses recherches ont alors été effectuées dans l'optique d'inclure un amplificateur d'instrumentation (couplé à une carte Arduino) à notre système.

Le composant INA125 a – en premier lieu – constitué une solution à notre problème. Ainsi, nous pouvons voir ci-dessous que cet amplificateur prend en entrée les valeurs de tensions issues du pont de Wheatstone, ainsi qu'une résistance R_G qui permet de fixer le gain que l'on souhaite appliquer. La sortie analogique est visible sur la broche 10.

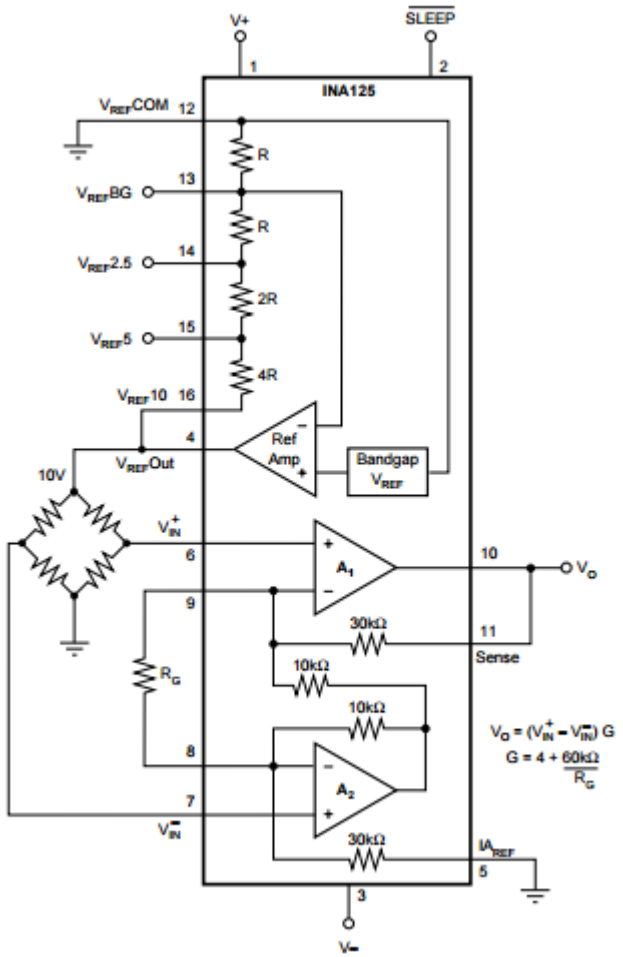


Schéma descriptif INA 125

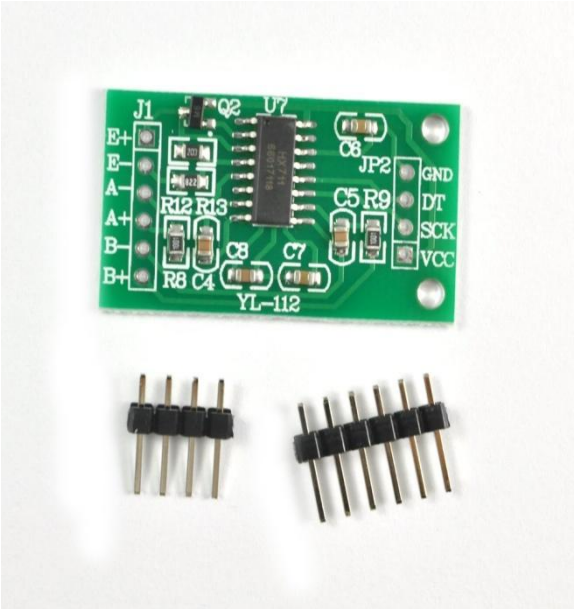
DESIRED GAIN (V/V)	R_G (Ω)	NEAREST 1% R_G VALUE (Ω)
4	NC	NC
5	60k	60.4k
10	10k	10k
20	3750	3740
50	1304	1300
100	625	619
200	306	309
500	121	121
1000	60	60.4
2000	30	30.1
10000	6	6.04

NC: No Connection.

Tableau relation valeur R_G / Gain

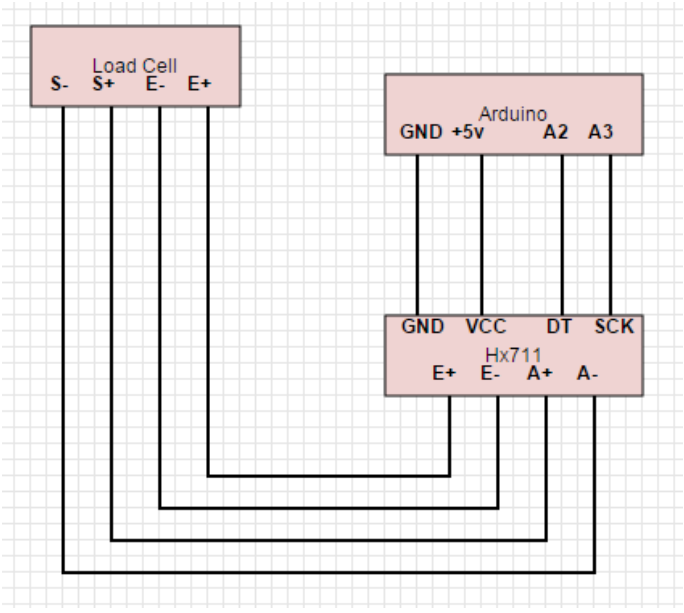
Cependant, après une grande quantité de mesures, de modifications des branchements et de changements du gain, la sortie analogue semblait toujours renvoyer des données inexactes : la différence entre les broches 6 et 7 (sorties du pont de Wheatstone) étant tellement petite, certaines perturbations minimes avaient probablement un impact irréversible sur la sortie V_O.

Nous avons donc finalement opté pour le module HX711 qui se présente sous forme d'une carte (à la différence du composant INA 125, nous n'avons pas à effectuer de soudure, donc moins de risque de perturbation et de mauvaise manipulation).



Comment marche cette carte ? Tout d'abord il nous faut la relier à notre jauge de contrainte ainsi qu'à notre carte Arduino. Les broches E+ et E- ont été reliées aux fils de tension de la jauge (E- pour la masse et E+ pour la tension). A+ et A- aux fils de sortie du pont de Wheatstone. B+ et B- servent uniquement si l'on veut rajouter un 2ème pont de Wheatstone, dans notre prototype nous ne les utiliserons pas. A droite, GND est à relié à la masse et VCC à la sortie 5V de la carte Arduino. DT et SCK à 2pins de lecture analogique.

Voici le schéma démonstratif :



Une fois les branchements effectués, il faut régler la valeur du gain du module HX711. Pour cela, nous devons importer la librairie de la carte, car celle-ci utilise ses propres fonctions (tare, lecture du poids, unité de lecture, etc). Il nous suffit donc de lancer un programme spécial fourni par le fabricant de la carte qui sert uniquement à régler le gain, pour cela nous mettons un poids d'une valeur connue et fixe (pour nous 100g), puis nous réglons sur le programme la valeur du gain jusqu'à ce que la carte nous renvois la bonne valeur, nous la notons, puis sur notre algorithme de communication Bluetooth entre l'application et le sous-bock, nous rentrons la bonne valeur.

Au final, ce module est très facile d'utilisation avec peu de branchement, réglable à souhait et fourni avec une librairie contenant de nombreuses fonctions très pratiquent à l'utilisation.

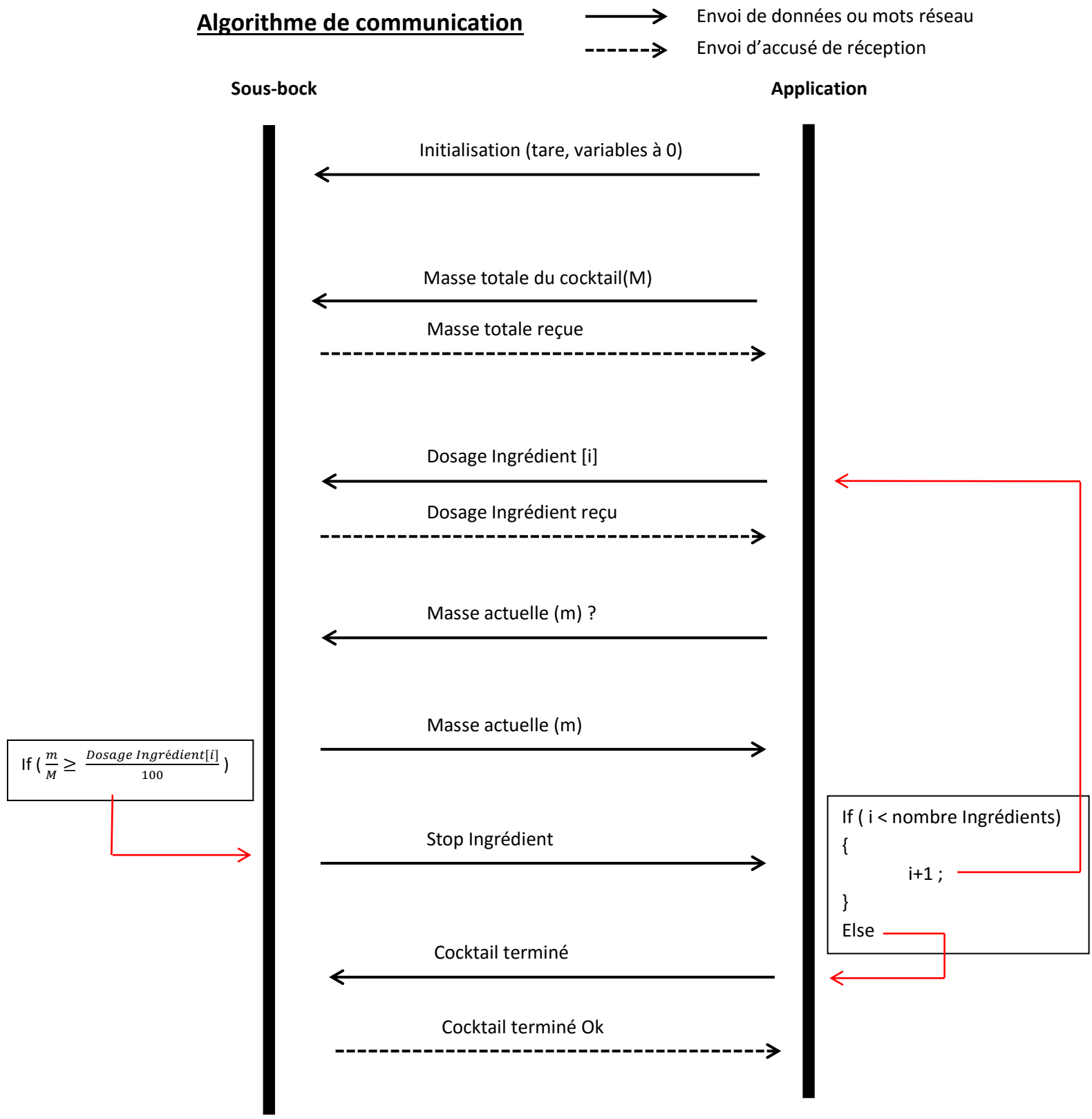
COMMUNICATION BLUETOOTH

Module Bluetooth HC06

Nous avons choisi d'assurer la communication Bluetooth à l'aide du module HC06. Le faible coût et la vitesse de ce dernier sont deux paramètres qui ont motivé notre sélection. Cependant, sa facilité d'implémentation est le critère qui a réellement permis d'opter pour ce module. En effet, étant compatible avec Arduino, une librairie interne nous permet d'utiliser des fonctions préexistantes.

Il reste pourtant un point noir au tableau : ce module n'inclut pas la technologie « Bluetooth Low Energy » qui s'impose maintenant comme la référence chez les nouveaux Smartphones. Afin d'être compatibles – aujourd'hui et demain - avec un maximum d'utilisateurs, il nous faudra donc reconsidérer ce choix.

Algorithme de communication



CONCLUSION

C'est l'idée même de "l'objet connecté" qui nous a réuni tous les quatre autour de ce projet. En effet, un tel projet est très riche en domaine de compétences, il va passer par des langages de programmations variés, de l'électronique et même de l'innovation. C'est donc avec une envie d'apprendre et de découvrir que nous avons attaqué ce projet très ambitieux.

Partant de rien, et n'ayant jamais programmé de telles applications, nous avons pu énormément développer nos connaissances informatiques ainsi que nos compétences organisationnelles (qui sont plus que vitales pour avancer rigoureusement). C'est donc dans cette optique que nous avons découvert et utilisé des outils comme Github (qui est le roi incontesté dans la multi programmations) et Cordova (facilite la programmation multiplateforme).

Etat du projet

Notre but était clair pour tous : réussir et terminer ce sous-bock connecté. Nous pouvons dire aujourd'hui que ce but est atteint, nous avons une ébauche de sous-bock qui communique avec notre application mobile. Nous pouvons y créer des cocktails, en inventer, en découvrir, et bien sûr les rendre réel pour le plus grand plaisir du consommateur. Tout cela sans la moindre formation de barman

Objectifs et idées futures

Nous avons pleins d'idées que nous n'avons pas pu mettre en œuvre et qui seront probablement l'objet de futures améliorations. Nous souhaitons créer un réseau social autour de ce produit, ainsi les utilisateurs pourraient partager, noter et commenter les différents cocktails. Nous pouvons aussi imaginer toutes sortes de jeux, ou calculer le taux d'alcool de l'utilisateur, ou même diffuser des informations (happy-hour de notre bar favori). Les idées ne manquent pas et représentent autant d'améliorations possibles.

Dans un premier temps nous voudrions rendre l'application native. Cette dernière est considérablement lente au niveau de la communication Bluetooth, dû au framework Cordova, et il est essentiel aujourd'hui d'avoir un rendu fluide pour les consommateurs qui sont de plus en plus exigeant sur la rapidité des outils informatiques.

Feed-back sur la réalisation globale du projet

Ce projet nous a rendu plus rigoureux, et nous avons compris qu'il était essentiel d'être multitâche et force de propositions, car les idées fusent, il faut donc savoir les analyser et en déterminer leurs faisabilités. Un esprit d'équipe et une cohésion de groupe sont nécessaires pour l'aboutissement de nos objectifs. Cette expérience nous a donc permis à tous, non seulement d'apprécier ce secteur informatique (objet connecté) mais aussi de nous parfaire et de nous améliorer dans ce même domaine.

BIBLIOGRAPHIE

Application native, hybride, web :

- http://www.mobizel.com/2015/02/webapp-application-hybride-native-quelle-est-la-difference/
- http://www.supinfo.com/articles/single/145-application-mobile-native-web-hybride

Les outils et langages de développement :

- Ionic Framework : http://ionicframework.com
- AngularJS : https://angular.io et https://angularjs.org
- Git et Github : https://git-scm.com et https://github.com
- Cordova : https://cordova.apache.org
- Scrapy : http://scrapy.org

Module HX711 :

- http://arduinotronics.blogspot.fr/2015/06/arduino-hx711-digital-scale.html

Module Bluetooth HC06 :

- http://tiptopboards.com/205-module-bluetooth-hc-06-compatible-avec-arduino.html