Refonte des systèmes internes du InMoov

10/07/2018

Cahier des charges réalisé par :

Brice PARILUSYAN

Hugo POUSSEUR

Dernière modification : 5 septembre 2018

Rapport réalisé sous LaTeX

Table des matières

1 Mise en contexte :

1.1 Le robot InMoov

InMoov est un robot humanoïde à taille humaine créé par Gaël Langevin, sculpteur et designer français. Le robot est en open source sur internet et a la particularité d'être entièrement imprimable en 3D sur la majeure partie des imprimantes 3Ds existantes. Le robot est composé de près de 300 pièces différentes et est continuellement amélioré par la communauté, il ne possède actuellement pas de jambe motorisées.

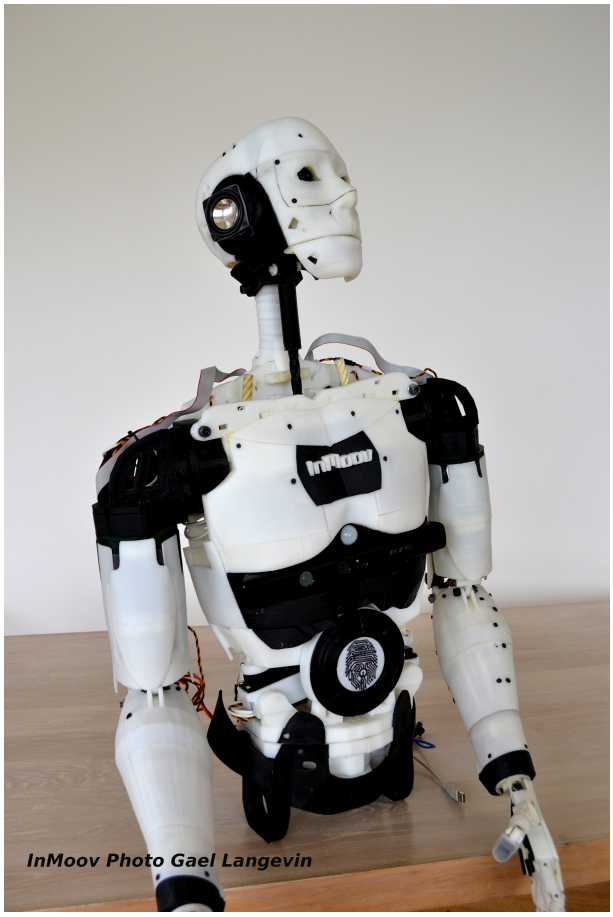

FIGURE 1 – Robot InMoov

L'association DaVinciBot a construit son propre clone lors de l'année 2017 - 2018 dans le but d'avoir à la fois une vitrine pertinente et une plateforme d'innovation pour les générations futures. Le robot étant imprimé en 3D, il est facilement démontable et modifiable, cela fait de lui le parfait candidat pour accueillir des projets d'innovation étudiant.

1.2 Le système électronique actuel

le robot InMoov étant Motorisé, il possède un système de distribution d'information et énergétique. Celui-ci fonctionne avec pour composant principal deux shields (des cartes électroniques conçues pour s'ajouter directement sur des microcontrôleurs, ici des arduino mega) ainsi qu'un réseau de relai.

Les shields ont pour but de faire la distribution d'informations entre les cartes arduino et les différents servo-moteurs/capteurs du robot. De plus, elles assurent aussi la distribution énergétique aux différentes parties. Elles ont été crée par la communauté InMoov et sont accessibles par l'achat en pièces détaché.

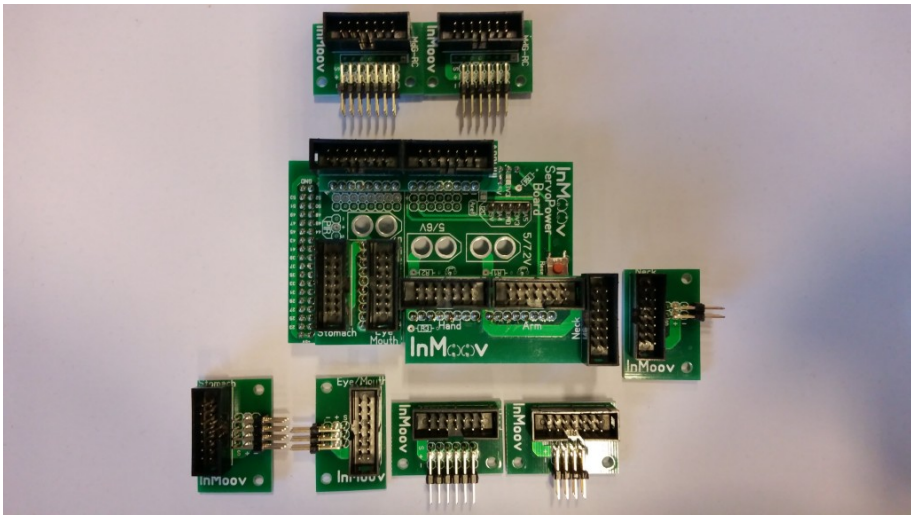

FIGURE 2 – Shield Inmoov avec relais

1.3 Le système informatique actuel

Le robot inMoov étant un projet Open-source, son système informatique a aussi été développé par la communauté. Il se compose d'un module de langage (que nous avons remplacé) et d'un API moteur appelé MyRobotLab [1].

1.4 L'objectif du projet

Le système de shields actuel a trois défauts majeurs :

1. Les plans et schéma électrique des shields et relais ne sont pas en libre accès. Par conséquent, si nous souhaitons ajouter des éléments au robot nous ne pouvons pas modifier les shields sans avoir à tous recréer depuis le début. Cela constitue aussi une difficulté car nous travaillons avec un élément que nous ne comprenons pas entièrement, ce qui conduit irrémédiablement à des problèmes.
2. Les shields constituent à la fois le réseau d'information, mais aussi le réseau d'alimentation du robot. en plus de ne pas être propre, ça a pour conséquence de complexifier le système et d'augmenter la difficulté pour repérer la source de certains problèmes, voire d'en créer.
3. Le robot est voué à évoluer au cours des années, hors un circuit imprimé n'est pas modifiable une fois produit. En conséquence, il faudrait recréer et réimprimer une shield à chaque fois qu'un ajout est fait au robot, ce qui n'est pas envisageable. Ce problème est dû au fait que les shields n'ont pas été désigné pour être modulaire mais spécifique au robot actuel.

Ces raisons nous ont poussé à choisir de refondre ce système de distribution en un autre plus performant, adaptable et modulaire. Il est à préciser que le système informatique du robot est lui aussi refondu. La partie électronique et informatique étant lié, vous aurez à travailler conjointement avec le groupe qui s'en occupe et vice-versa.

1. Page de MyrobotLab

Parmi les différentes caractéristiques recherchées pour le nouveau système électrique (fiabilité, performance, simplicité etc...), la modularité est la plus importante. Ce système doit être capable de s'étendre à l'infini sans avoir à refaire le moindre élément. Le robot est voué à évoluer en continu, mais de manière inconnue, c'est pour cette raison qu'il faut un système utilisant peu de composants différents, simple à utiliser et capable de s'agrandir au besoin. Ce système devra être encore viable d'ici 5 ans, et ce malgré les améliorations que le robot aura reçues.

Pour ce qui est de la partie informatique, le problème est relativement similaire, MyRobotLab est un système peu efficient, difficilement modifiable et qui ne rentre pas dans notre objectif d'évolution future. La solution consiste donc à remplacer entièrement le système informatique du robot par une architecture MQTT et de refaire une API moteur qui soit simple d'utilisation (pour humain comme programme informatique).
La nouvelle architecture aura pour but de favoriser l'expansion future du robot et de faciliter l'adhésion de nouveau programmes.

Première partie

Energy network

2 Le réseau d'information :

Le réseau d'information que nous avons imaginé est composé de deux éléments distincts :

- Des interfaces plugable directement sur des cartes arduino ou Raspberry afin de traduire ses signaux de la manière voulu.
- Des Gateway faisait office de relai entre les microcontrôleurs et un ou plusieurs éléments actifs du robot.

Le périphérique de communication sera le RJ45. Les communications devront être bidirectionnelles.

2.1 Interface de communication pour microcontrôleurs

Cette interface est une petite shield qui a pour but de traduire les communications émises par la carte dans plusieurs sorties/entrées RJ45. Cette shield se doit d'être le plus adaptable possible afin de se connecter à d'autres shields montés sur d'autres carte et ainsi assurer la communication entre elles.

Nous sommes amenés à retrouver plusieurs types de microcontrôleurs sur le robot, il est donc possible d'avoir plusieurs versions différentes de la shield afin d'avoir une version adaptée pour chaque carte. Majoritairement, nous avons utilisé des Arduinos mega et des Raspberry PI3. Il est vrai aussi que suivant le programme exécuté, la shield sollicitera peut-être plus l'entrée que la sortie et inversement. Ce sera à vous de prendre en compte ces données et créer des shields capables de pallier le plus de situation possible. Pour des raisons de production, nous ne souhaitons pas un trop grand nombre de versions différentes.

2.2 Les Gateway

Les Gateway font office de relais afin de rediriger une communication spécifique à un élément spécifique du robot. C'est un composant qui sera présent en grande quantité dans le robot et donc qui devra être de petite taille et adapter à la communication avec tous les éléments actifs possibles (capteur, moteur, servomoteur etc..).

Par souci de simplicité, nous avons imaginé les Gateway avec le rôle de multiplexeur. Ils sont similaires à des collecteurs en plomberie, leur rôle est de recevoir l'information d'un côté et le renvoyé de l'autre tout en sélectionnant l'information qui les intéresse. Leur but est d'être spécifique à un élément actif et de sélectionner l'information qui le concerne parmi le flux d'informations qu'il reçoit.

FIGURE 3 – Collecteur de plomberie

Toujours dans la même idée, les Gateway supportant un flux, elles devront avoir un point d'entrée et un point de sortie. De plus, elles devront accueillir un microcontrôleur afin de sélectionner l'information qui les concerne. Cela implique qu'elles soient paramétrables et alimentables en énergie.

3 Le réseau d'alimentation :

Le réseau d'alimentation électrique va permettre de distribuer l'énergie des batteries là ou elle est nécessaire. Le système est composé de deux parties, la carte d'alimentation qui récupère le courant directement depuis la batterie et les Gateway d'alimentations.

3.1 Carte d'alimentation générale

La carte d'alimentation a pour but de servir d'interface entre le réseau d'alimentation du robot et une batterie, il assurera aussi la distribution de l'énergie. La carte devra être la plus compact possible et facile à mettre en place. Chacun devra avoir deux emplacements pour y brancher des batteries.

Comme nous parlons d'alimentation électrique, des mesures de sécurité devront être nécessaires :

- Des diodes de protections doivent être mises là où il y a un quelconque risque de court-circuit, d'intégrité pour une batterie ou un composant électronique et toute autre situation à risques.
- Sur chaque sortie où entrée doit figurer la valeur en volts qui lui correspond, de manière indélébile. De plus, les câbles doivent avoir des couleurs explicites (conventions de couleurs à respecter) et être relié ensemble s'ils vont par groupes (aucun câble volant risquant de s'emmêler).
- Les gaines et connectiques doivent être conformes aux normes, ne pas présenter de risques et facilement manipulables.

3.2 Gateway & alimentation

Les Gateway d'alimentation sont basée sur le même concept que celle d'information, celui du collecteur. Chacune doit pouvoir alimenter un composant actif et une gateway d'information. Ce sont des composants passifs et doivent donc être le plus simple, petit et pratique possible. On n'oublie pas qu'il s'agit encore de courant électrique, les règles ci-dessus sont donc encore de mise.

pour faciliter les montages, il serait pratique d'avoir un système permettant d'accrocher une gateway d'alimentation à une d'information. attention, les deux doivent rester distinctes.

Deuxième partie

Neural backbone

4 Le cerveau d'InMoov :

4.1 Initiative de cette partie

Comme expliqué au-dessus, le but de ce projet est de recréer une base propice à nos attentes. Cette reconstruction comprend aussi une partie software. Ce projet consiste à créer une structure très épurée, facilement utilisable, rendant le robot modulable.

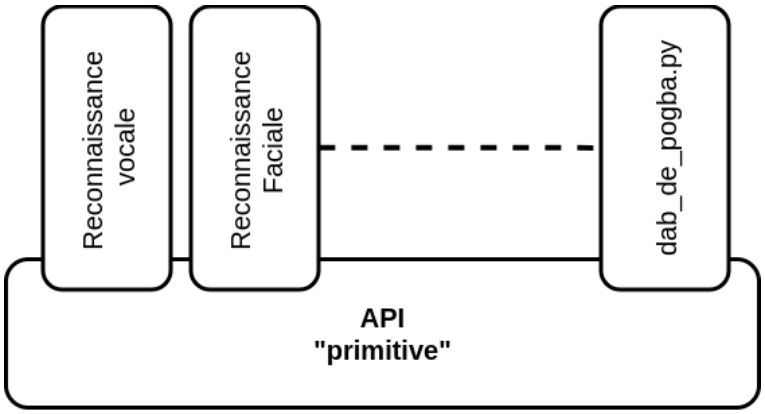

FIGURE 4 – Structure modulaire

Nous (le bureau) avons beaucoup discuté entre nous pour choisir la solution qui nous semble la plus optimale. Nous avons longuement hésité entre l'implémentation d'une structure ROS [2] (Robot Operating System) ou d'une structure basée sur le protocole MQTT [3](Message Queuing Telemetry Transport). Souhaitant un système facilement exploitable avec peu de bagage informatique, c'est sur cette deuxième option que nous nous sommes orientés. De plus il s'agit d'un protocole très utilisé dans le monde de l'IOT.

2. Wiki officiel de ROS
3. Site officiel de MQTT

Le but est de créer une API [4] (Application Programming Interface) permettant facilement de contrôler l'humanoïde.

4.2 Mise en place de l'API

La réalisation de cette API comprend :

La réalisation d'un Broker [5] MQTT : en faisant peu de recherches nous avons trouvé un projet open source qui le permet : Mosquitto [6], ce n'est qu'une proposition, il est possible d'utiliser un autre projet open source.

Les étudiants en charge de ce projet devront réfléchir au support utilisé pour héberger le Broker, en respectant 3 contraintes :

- la taille, le Broker doit être incorporé au robot
- la capacité à gérer la connexion sans perturbation, en fonction de la structure choisie il est possible que plusieurs dizaines de gateways communiquent avec le Brokers en même temps
- la communication avec le monde extérieur

En effet nous souhaitons que la communication avec le Broker soit possible en filaire (par Ethernet) et sans fil (par Wi-Fi). Nous ne souhaitons pas que seul l'accès en SSH soit possible en Wi-Fi, nous devons être capable de faire des requêtes directement en Wi-Fi.

MQTT est un protocole de communication, vous devez alors choisir un langage de programmation. Le langage doit être choisi en vue de sa librairie permettant l'utilisation du protocole MQTT, de la vitesse d'exécution et la gestion de multi-connexions (multi-threads ? ...).

La mise en forme des requêtes : il est important d'aborder cette partie sans se précipiter, il faudra réfléchir à une convention de nommage et d'ordre que respecteront les trames, applicable

4. API (lien Wikipédia)
5. Explication d'un Broker
6. Site officiel Mosquitto

à toutes les situations. Par exemple si chaque requête comprend un nom de composant et une valeur, il ne sera pas acceptable que pour un type de moteur le nom apparaisse avant la valeur pour un type de moteur et dans un l'autre sens pour un autre moteur. Il est vivement conseillé d'étudier les composants utilisés actuellement et ceux qui pourront être implémentés ultérieurement. Certains éléments (très peu) du robot ne passeront peut-être pas par une gateway, par exemple la caméra qui pourrait se brancher en USB sur le Broker.

Vous devez prendre compte que la communication entre le Broker et les différents composants peut être bidirectionnelle. Du capteur au client, dans le cas où le capteur retourne une valeur (par exemple capteur de proximité) et du client au capteur, quand l'utilisateur souhaite ordonner un mouvement.

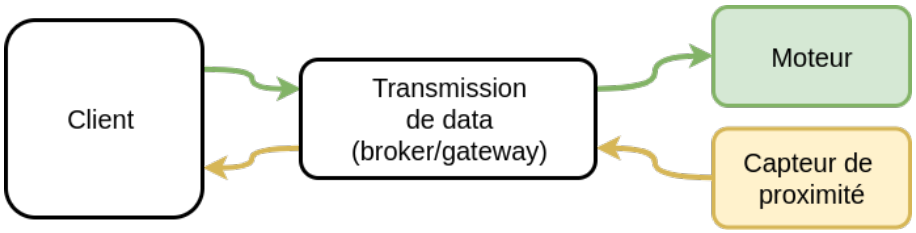

FIGURE 5 – Communication Bidirectionnelle

L'une des raisons qui nous pousse à faire ce projet est l'aspect modulable. Mais l'ajout de divers modules peut comprendre des risques, il est demandé de réfléchir à une procédure (par exemple un système de token [7]) évitant que certaines (ou toutes) parties du robot soit contrôlable en même temps par 2 ou plus clients.

7. Explication d'utilisation de tokens avec MQTT

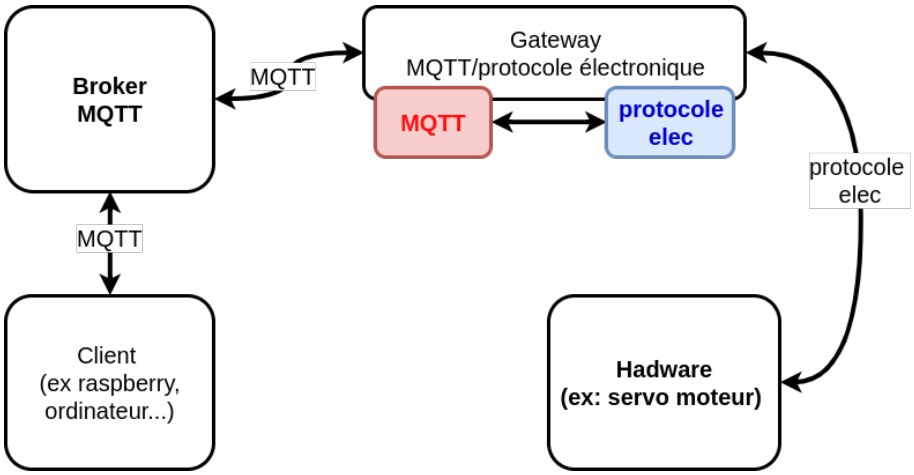

FIGURE 6 – Illustration du but

L'architecture de l'API : L'API doit être exploitable par un grand nombre de personnes, d'une personne ayant quelques notions en programmation mais aussi d'une personne ayant des connaissances en robotique. Pour toucher un large public, nous sommes obligé de mettre en place une API à 2 niveaux.

- Premier niveau : aussi appelé niveau bas, permet de contrôler précisément chaque moteur/capteur.
- Deuxième niveau : aussi appelé niveau haut, permet de réaliser des mouvements prédéfinis. Cette partie d'API devra exploiter des fonctions de la première API.

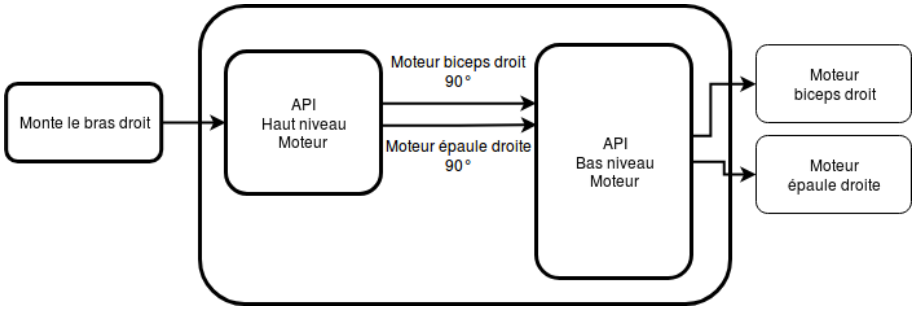

FIGURE 7 – Illustration du but

4.3 Ne jamais oublier

Nous tenons à souligner 3 points à ne pas oublier lors de la réalisation de ce projet, l'API doit être :

1. facile d'utilisation, une personne n'ayant aucune notion en robotique doit être capable de l'utiliser
2. sécuritaire, mettre des contraintes d'angles... Pour éviter toute catastrophe. Il nous a fallu un an de travail pour réaliser ce robot, ne l'oubliez pas !
3. documenté, ce travail sera le support d'autres projets, d'années futures. Pour exploiter à 100% votre travail, il sera demandé de réaliser une doc d'utilisation et une doc du code.

4.4 Exemple d'utilisation

Afin de montrer la puissance de l'API, il est demandé aux étudiants en charge du projet de réaliser une interface. Cette interface permettra de commander le robot à l'aide d'un ordinateur (ou d'un téléphone) à distance.

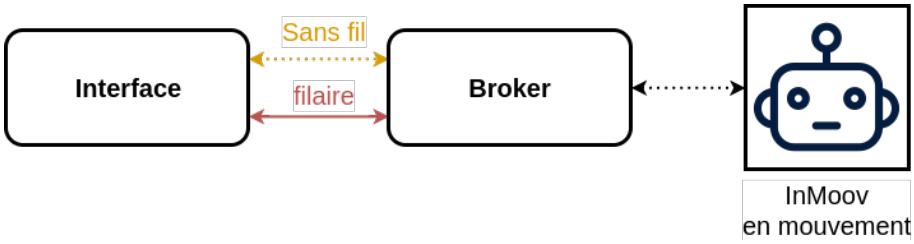

FIGURE 8 – Interface

La technologie utilisée pour cette interface est à vous de choisir, le rendu final doit cependant être jolie, rapide et intuitive. Il est fort probable que cette interface soit utilisée lors de nos promotions.

Troisième partie

Règles communes

5 L'organisation du projet :

Comme tous projets, vous aurez certaines contraintes. Étant un projet PI², vous aurez un client et un mentor. Le mentor sera un professeur et le client sera l'association DaVinciBot représenté par un membre. Vous aurez une réunion *obligatoire* avec le client une semaine sur 2, lors de cette réunion vous pourrez vérifier si vos travaux sont en adéquation avec ce qui est attendu et présenter votre avancement. Il est fortement conseillé à ce que vous vous réunissiez de manière hebdomadaire.

5.1 les contraintes de temps et logistiques

Le showroom de fin de projet aura lieu le 28 mars 2019, votre projet devra être terminé *impérativement* avant le 4 mars 2019. Le projet fini signifie que la documentation est complète, des plans sont accessibles et utilisables, les composants et programmes ont été crée, installé sur le robot, testé et validé comme fonctionnels par le client. Sans une seule de ces étapes, le projet ne sera pas considéré comme terminé et le client insatisfait.

Il est aussi à prendre en compte que le robot InMoov ne sera pas disponible à usage exclusif d'un groupe. En plus des autres groupes qui travailleront dessus, le robot sert de vitrine pour l'association et le DVIC, il devra donc constamment être fonctionnel (avec l'ancien ou le nouveau système) et mis en exposition. Vous devrez donc demander l'autorisation au préalable avant d'y avoir accès et devrez le remettre en état avant la fin de la journée. Une demande d'accès sera limité dans le temps. Il vous est donc conseillé de ne pas vous y prendre au dernier moment.

5.2 Coopération

Lors de ce projet, l'équipe pour l'électronique et celle pour l'informatique seront amener à co-opérer. Les deux projets sont liés et vous aurez à travailler en coordination. Il vous est fortement conseillé de ne pas négliger de vous mettre au courant des avances de l'autre équipe, vous aurez certaines décisions communes à prendre. Les réunions avec le client seront en commun avec les deux équipes.

5.3 Outils coopératifs

Communication : Pour faciliter le suivi le long du projet, il est demandé aux groupes de communiquer avec Teams

Développement informatique : L'utilisation de GIT [8] est primordiale pour développer en équipe. Il est demander aux étudiants de se former par eux-même sur l'utilisation de GIT. Internet est gorgé de tuto [9]

8. GIT (lien wikipedia)
9. Vidéos de tutorials GIT