Université de Claude Bernard Lyon 1
Département informatique

Master 2 en Technologie de l'Information et Web
Spécialité Technologie de l'Information
Projet TI5

Couche d'Interopérabilité Matériel Application (CIMA)

Membres du groupe D:
FARAH CHETOUANE
HAROLD NGALEU TOUMENI
MAMADOU HADY DIALLO
MAXIME BAUDIN
REMI DESMARGEZ

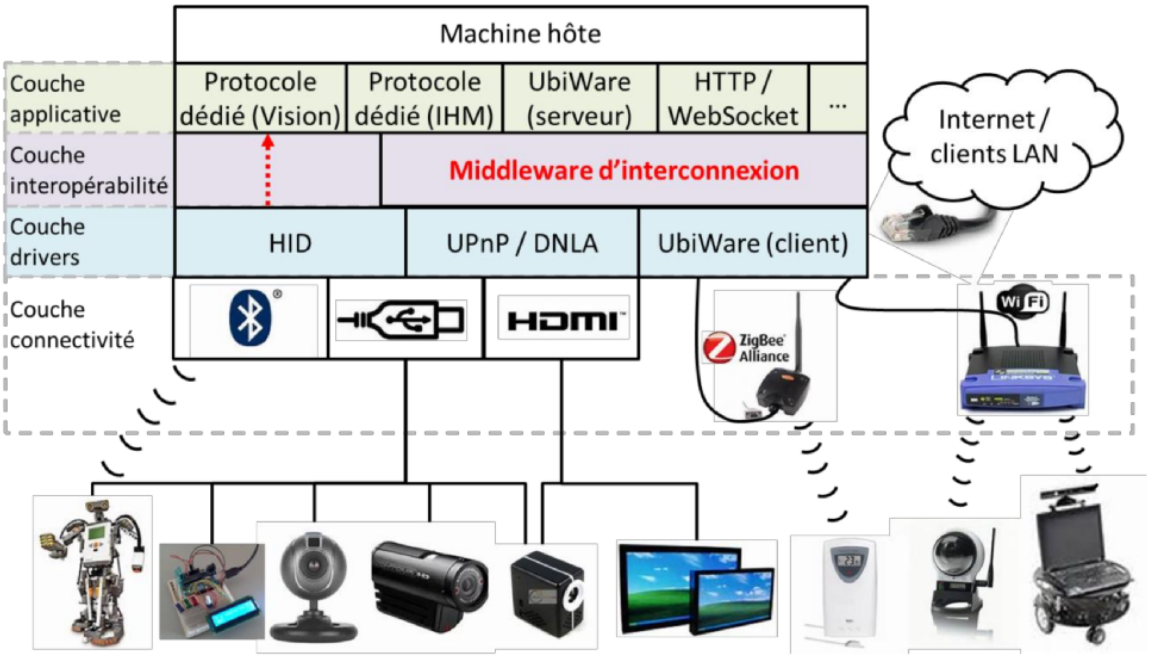
Encadrants:
LIONEL MEDINI
MICHAEL MRISSA

2014 - 2015

1 Objet et Contexte

1.1 Contexte

Le projet TI5 s'inscrit dans le cadre de l'extension des différents modules de la plateforme CIMA. Cette plateforme CIMA vise à connecter différents matériels (caméra, périphérique d'IHM, réseaux de capteurs, robots, etc...) à différentes applications développées ou soutenues par le LIRIS (Laboratoire d'InfoRmatique en Image et Systèmes d'information) comme par exemple "LIRIS-Vision", "Voir", "OASIS" ou bien "UbiWare". L'objectif de cette plateforme est de répondre à de nouveaux besoins non encore satisfaits par les plateformes existantes et de doter le LIRIS d'une plateforme facilitant la réutilisation de matériels communs destinés à différentes applications dans le cadre de travaux de recherche ou de démonstrations.
CIMA sera installée sur une machine localisée dans une salle de démonstration.
Son architecture consistera en une machine dédiée à un framework structuré selon plusieurs couches.



1.2 Positionnement

Nous avons la position de prestataire dans le cadre de ce projet. CIMA est un projet du laboratoire LIRIS. Dans ce cadre, l'équipe doit s'approprier l'existant afin de pouvoir réaliser des modules complémentaires.

1.3 Objet

L'objectif de ce projet TI5 sera d'étendre la couche d'interopérabilité (la plateforme CIMA) c'est-à-dire :

- Permettre la découverte des appareils non connectés en IP (Bluetooth, USB, HDMI)
- Étendre la partie Gateway

- Étendre le module cour-circuitant le middleware d'interconnectivité.
- Étendre panel de matériels compatibles(robot MindStorms, caméra, écran, capteur)
- Créer une interface intuitive de configuration et d'appariement des appareils et des outils de démonstration.

2.1 Livrables de gestion de projet

Pour s'assurer de la réussite de ce projet, nous allons livrer cinq documents (celui-ci compris) tout au long de notre avancement :

- Le dossier d'initialisation (ce document)
- Trois dossiers post-sprints :

Ces dossiers sont à rédiger à la fin de chaque sprints. Ils vont nous permettre de comparer nos estimations pré-sprint avec nos résultats post-sprint. Ils décriront les tâches que nous allons traiter avec leurs durées d'estimations, leurs répartitions ainsi que les livrables techniques à remettre à la fin de ses sprints. Ils comprendront également leurs états d'avancements, les écarts par rapport à ces précisions et les éventuelles modifications mis en oeuvre.

- Le dossier de synthèse finale
- La présentation finale

2.2 Livrables techniques

Voici les réalisations que nous allons implémenter dans le cadre de notre projet :

- **Maquette de l'interface de configuration manuelle à livrer pour le 23 octobre 2014**

L'interface de configuration a pour but de connecter un objet qui ne pourrait pas être automatiquement découvert.
En effet, dans certains cas il sera nécessaire de développer une interface personnalisable afin de définir ce qu'est l'objet, ce qu'il est capable de faire pour en exploiter ses capacités.
Elle fonctionnera de la manière suivante :

1. L'utilisateur, sélectionnera l'objet qu'il souhaite configurer.
2. Un champ permettra de spécifier les capacités propres à l'objet dans une liste déroulante.
    2.1. Dans le cas ou les capacités de l'objet ne seraient pas disponibles, l'utilisateur pourra en ajouter. Il pourra accéder à une seconde interface où il définira la capacité (Nom de la capacité, fonctionnalité implémentée, protocole de communication, etc).
3. Enfin il sera possible de tester le fonctionnement de la capacité définie.

- **Infrastructure serveur pour répondre à l'interface à livrer pour le 23 octobre 2014**

L'interface, afin de récupérer certaines données ou d'en ajouter, communiquera avec le serveur par l'intermediaire de requêtes REST. Le serveur se chargera alors d'interréagir avec la plateforme CIMA pour effectuer les comportements nécessaires à la configuration d'un objet.

- **Création d'un serveur sur un objet à livrer pour le 27 novembre 2014**

La norme M2M prévue par ETSI prévoit plusieurs cas d'utilisation de la plateforme. Il est nécessaire de démontrer avec des exemples que la plateforme CIMA sera capable de fonctionner dans ces cas d'utilisations. Le cas d'utilisation normal serait d'avoir un objet qui sait

communiquer avec la plateforme CIMA. Un autre cas prévoit d'avoir une partie de la plateforme qui est déportée sur l'objet.
C'est pourquoi le second sprint sera consacré dans le développement d'un serveur sur un objet. Le serveur pourra par exemple remplacer en partie la "Gateway". Il est aussi prévu de continuer le développement déjà existant d'un cas d'utilisation normal qui prévoit la communication entre la plateforme CIMA et un robot ev3.

- **Tests d'intégration et fonctionnels sur CIMA pour le 29 janvier 2014 :**

Les étudiants de l'IUT informatique travaillent en parallèle sur la découverte de capacités d'objets en USB ou en bluetooth à l'aide du protocole HID. Une fois le code développé il faudra l'intégrer à l'application CIMA. Ils devront développer un bundle OSGi, qu'il faudrat que nous utilisions depuis les autres modules de la plateforme. Nous devrons alors mettre en place plusieurs tests pour vérifier le bon fonctionnement de la plateforme.

2.3 Autres livrables

- **Effectuer une démonstration de la plateforme CIMA**

Afin de montrer le bon fonctionnement de la plateforme nous avons souhaité créer un test utilisant en partie les fonctionnalitées développées pendant le projet.

- **Présentation finale**

Une présentation aura lieu pour présenter le travail effectué par notre groupe au cours des trois sprints. Nous utiliserons un support de type diaporama.

3 Méthodes et outils

3.1 Contraintes

3.1.1 Langages

- Côté CIMA :
  - Gateway : cette partie sera developpée en Java, car la solution sur laquelle elle se repose a été developpé elle même en Java.
  - Module court-circuitant la "gateway" : Pour des raisons de performances et techniques cette partie sera développée en langage C.
- Côté cloud : pour les mêmes raisons que la partie gateway de la plateforme CIMA, elle sera également developpée en Java.
- Côté client : Le développement de cette partie se fera en utilisant le HTML et le Javascript.

3.1.2 Frameworks

Tout au long du projet nous allons utiliser deux frameworks, un pour la gestion côté serveur/plateforme, et l'autre pour la gestion du client graphique.

3.1.2.1 OSGI

OSGI est un framework java permettant (entre autres) le chargement à chaud de ses composants (bundles) et la gestion de leurs cycle de vie.

Dans le cadre du projet CIMA, OSGI s'avère efficace car il permet de développer plusieurs modules, pouvant se trouver à plusieurs endroits. En effet suivant les capacités techniques d'un objet on va déployer du code sur l'objet lui-même ou sur la plateforme CIMA. OSGI permet d'être assez indépendant de l'endroit où se trouve le code et de ne pas avoir à réécrire des bouts d'applications suivant qu'on soit sur l'objet ou sur la plateforme.

3.1.2.2 AngularJS

AngularJS est un framework JavaScript, libre, open-source développé par Google. Il permet de délocaliser une grande partie de sa logique côté client. Il est efficace lorsque l'application contient une multitude de saisies clients, des mises à jour de la vue dynamique.
Nous utilisons AngularJS pour l'interface web permettant de configurer de nouveaux objets.

3.1.3 Plateforme

3.1.3.1 OM2M

OM2M est une implémentation de la norme ETSI M2M. Le but d'OM2M est de faire communiquer plusieurs machines entre elles.
L'une des fortes contraintes de la plateforme CIMA est le fait de pouvoir ajouter des composants de façon dynamique et le framework OM2M offre cette possibilité, car il est basé sur OSGI, le fait qu'elle soit : OS agnostique, soutenu par la fondation Eclipse et possède donc une forte communauté, sous licence libre l'a fortement imposée comme solution de base pour le développement de CIMA.
La plateforme OM2M est divisée en 3 parties :

- La partie "Gateway" qui est la partie sur laquelle on branche les objets,
- la partie "Cloud" qui est la partie qui expose les objets aux applications,
- Et la partie "Device" qui est sur l'objet et qui permet la communication entre l'objet et la "Gateway"

3.1.4 Délais

Nous disposons de trois sprints de sept demi-journées. À chaque fin de sprint nous disposeront de quelques jours pour rendre le document de post-sprint ainsi que les différents livrables nécessaires.
De plus il serait apréciable d'avoir une démonstration fonctionnelle à la fin du mois de décembre.

3.2 Méthode

La méthode de gestion de projet choisi est la méthode agile et se déroulera en trois sprints de quatre jours. Avant chaque sprint on définira l'ensemble des tâches à réaliser pendant celui-ci. A la fin de chaque sprint on rendra compte de l'avancement des tâches.

3.3 Outils

3.3.1 La forge

Nous avons besoin d'une plateforme qui nous offrira des outils de gestion de projet, avec un outil de versioning, de gestion de demandes et de planning partagé.

Pour ce faire nous allons utiliser la forge qui de plus de ses outils comporte aussi un wiki (http://forge.univ-lyon1.fr/projects/cima-ti5-2014). De plus nous allons utiliser le wiki proposé par la forge pour noter les comptes rendus de réunion et toute documentation utile.

3.3.2 Communication

En plus de notre groupe TI5 composé de cinq étudiants, nous collaborons aussi avec quatre étudiants de l'IUT informatique de Lyon 1. Le principal moyen de communication que nous allons utiliser est l'échange par mail via la messagerie interne de l'université. Bien-sûr, nous aurons aussi des réunions avec nos encadrant une fois par semaine.

3.3.3 Serveur de test

Pour développer l'interface de configuration manuelle, nous avons besoin de déployer un serveur fonctionnel. Chaque membre aura donc son instance de CIMA sur sa propre machine. Cela permettra de paralléliser la modification du serveur et la mise en place de l'interface.

3.3.4 Intégration continue

Tout au long du projet nous utiliserons l'intégration continue des fonctionnalités développées. En particulier sur l'intégration de nos modules et celui des étudiants de l'IUT.

4 Macro-Planning

4.1 Lots de travail

Nous avons défini plusieurs lots de travail qui sont plus axé sur le premier sprint, car nous avons plus réfléchi à ce que nous allions faire dans ce dernier que dans les autres :

- Lot pour la configuration manuelle :
  - Conception de la partie configuration manuelle
  - Conception du modèle coté client (interface)
  - Mise en place des tests fonctionnels
  - Avoir une architecture AngularJS qui fonctionne
  - Production de la partie cliente
  - Etude et si nécéssaire développement de la partie persistance des objets
  - Développement des modules nécessaires à la configuration manuelle côté OM2M
- Lot pour le developpement d'un serveur sur un objet :
  - Conception et choix techniques
  - Mise en place des tests fonctionnels
  - Etude et intégration du code existant
  - Développement sur un objet test
- Lot pour la livraison d'une version fonctionnelle :
  - Exécution des tests fonctionnels
  - Vérification et finalisation de l'intégration du code de l'équipe des étudiants de DUT
- Lot Integration continu
  - Intégration des différents bundles développés par les étudiants de DUT

    - Tests d'intégration de ses bundles
- Lot conception
    - Expression des besoins utilisateurs
    - Analyse des besoins utilisateurs
    - Définition des tâches de travail

Il nous reste une tâche qui sera "en cours" tout au long du projet qui est l'intégration du code développé par les étudiants de DUT.

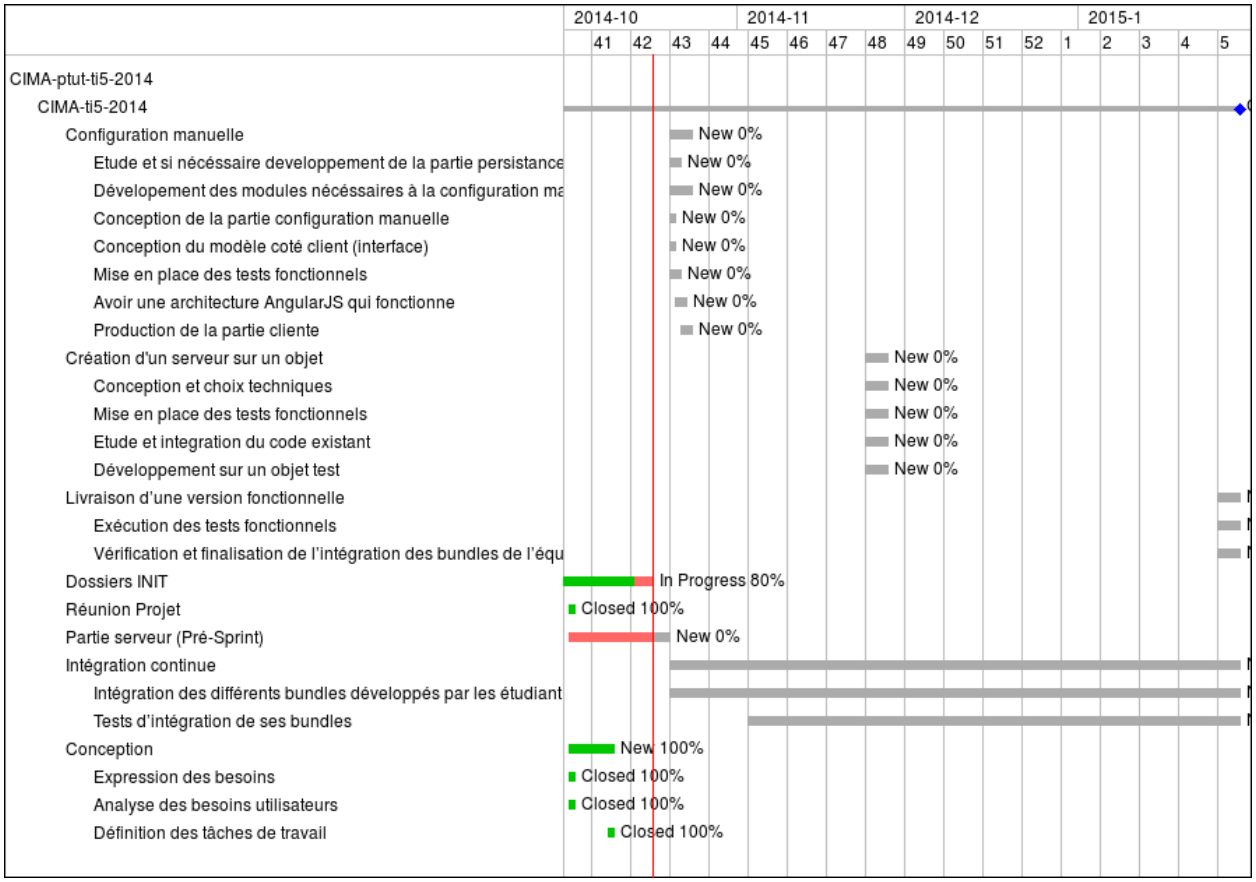
4.2 Phasage

Nous avons prévu de livrer un lot par sprint. Ci-dessous le tableau récapitulatif :

Lot	Date de remise
Conception	Pré-sprint 1 (19/10/2014)
Configuration manuelle	Sprint 1 (23/10/2014)
Serveur sur un objet	Sprint 2 (27/11/2014)
Livraison d'une version fonctionnelle	Sprint 3 (29/01/2014)
Integration continu	fin Sprint 3 (29/01/2014)

La tâche d'intégration continue du code développé par les étudiants de DUT sera réalisée en fonction de leurs disponibilités et de leurs avancés.

Annexes


2014-10
2014-11
2014-12
2015-1
41 42 43 44 45 46 47 48 49 50 51 52 1 2 3 4 5
CIMA-ptut-ti5-2014
CIMA-ti5-2014
Configuration manuelle
Etude et si nécéssaire developpement de la partie persistance
Dévelopement des modules nécéssaires à la configuration ma
Conception de la partie configuration manuelle
Conception du modèle coté client (interface)
Mise en place des tests fonctionnels
Avoir une architecture AngularJS qui fonctionne
Production de la partie cliente
Création d'un serveur sur un objet
Conception et choix techniques
Mise en place des tests fonctionnels
Etude et integration du code existant
Développement sur un objet test
Livraison d'une version fonctionnelle
Exécution des tests fonctionnels
Vérification et finalisation de l'intégration des bundles de l'équ
Dossiers INIT
Réunion Projet
Partie serveur (Pré-Sprint)
Intégration continue
Intégration des différents bundles développés par les étudiant
Tests d'intégration de ses bundles
Conception
Expression des besoins
Analyse des besoins utilisateurs
Définition des tâches de travail
New 0%
New 0%
New 0%
New 0%
New 0%
New 0%
New 0%
New 0%
New 0%
New 0%
New 0%
New 0%
In Progress 80%
Closed 100%
New 0%
New 100%
Closed 100%
Closed 100%
Closed 100%