Second Rapport INF6404A : Présentation du Middleware

Alexandre Mao
David Johannès
Philippe Troclet
Fabien Berquez
Département Génie Informatique et Génie Logiciel
École Polytechnique de Montréal, Québec, Canada
alexandre.mao@polymtl.ca
david.johannes@polymtl.ca
philippe.troclet@polymtl.ca
fabien.berquez@polymtl.ca

19 mai 2016

1 Introduction

Le monde d'IoT représente par définition un système "intégré", où les différents objets interagissent entre eux, sont inter-connectés, à travers l'échange d'informations et de commande (requête, demande). Ces objets ont une forte probabilité d'être hétérogènes en termes de niveau de sécurité, de privacité minimal garanti, de technologie, de protocole de communication, et de politique d'exécution. Le challenge est ainsi davantage lié au besoin d'avoir une structure horizontale capable de gérer les spécifications de sécurité et de privacité de manière unique et homogène. Ces spécifications auront besoin en effet d'être instanciées sur des "entités" et auront potentiellement des interfaces d'implémentation, de spécification et de communication différentes.

Les caractéristiques de IoT comprennent donc un réseau à très grande échelle des objets, une grande hétérogénéité au niveau des dispositifs et du réseau, et un grand nombre d'événements générés spontanément par ces objets. Malheureusement, toutes ces caractéristiques feront du développement des diverses applications et des services une tâche très difficile. En général, le middleware peut faciliter un processus de développement en intégrant des dispositifs informatiques et de communication hétérogènes, et en soutenant l'interopérabilité au sein des diverses applications et services.

En effet, un middleware fait abstraction de la complexité du système ou du matériel, permettant au développeur d'applications de concentrer tous ses efforts sur la tâche à résoudre, sans la distraction des préoccupations orthogonales au niveau du système ou du matériel. Ces complexités peuvent être liées à des préoccupations de communication ou au calcul plus généralement. Un middleware fournit une couche logicielle entre les applications, le système d'exploitation, les couches de communication réseau et les différents dispositifs du système, ce qui facilite et coordonne certains aspects du traitement coopératif. Du point de vue informatique, un middleware fournit une couche entre les logiciels d'application et des logiciels système. Dans l'IoT, l'hétérogénéité des dispositifs implique très souvent une hétérogénéité considérable dans les technologies de communication utilisées, ainsi que dans les technologies au niveau du système, c'est pourquoi un middleware devrait supporter ces deux types d'hétérogénéité. Nous avons donc besoin d'un middleware qui respecte des caractéristiques, que nous décrirons par des modules. Ces modules seront divisés en trois groupes : les modules fonctionnels, liés aux services et aux fonctions que notre middleware doit fournir ; les modules non-fonctionnels, liés à la Quality of Service (QoS), aux performances et aux différentes exigences que notre middleware devra prendre en compte ; et les modules architecturaux, liés à l'architecture de notre middleware.

Rappelons que notre sujet consiste à établir un système IoT dans le domaine Smart Health, et plus précisément dans les services de soins intensifs des hôpitaux, afin de résoudre le problèmes de congestion et de surveillance en continu dans ces services. La Figure 1 nous permet de visualiser les choix de dispositifs et de protocoles de communication et de réseau que nous avons établis dans le rapport précédent. Avant de commencer à décrire les différents modules dont nous auront besoin dans notre middleware, ce que nous feront dans les section 2, 3, et 4, il est important de préciser le choix que nous avons fait concernant l'architecture générale de notre middleware. En effet, nous avons décidé de diviser notre middleware en deux couches différentes, la première étant liée au moniteur qui centralise toutes les informations des divers dispositifs présents dans la chambre du patient, et qui donc va gérer l'hétérogénéité entre les dispositifs présents dans l'environnement du patient. La seconde couche middleware est liée au Gateway de notre système, qui s'occupe de centraliser les informations de tous les moniteurs (le problème d'hétérogénéité se pose moins ici), et qui va faire le lien entre la couche de stockage et celle de liaison à la couche physique. Cette seconde couche va être celle qui permettra d'identifier les différents groupements de dispositifs à travers la connaissance des différents moniteurs intelligents.

La Figure 2 nous permet de visualiser la structure de notre système en considérant seulement les couches dispositifs, réseaux et communication, middleware, et architecture. Ainsi, dans les trois prochaines sections, notre tâche ne sera pas seulement de décrire les différents modules dont nous avons besoin dans notre middleware, mais aussi de décrire dans quelle(s) couche(s) middleware nous en avons besoin (possiblement les deux).

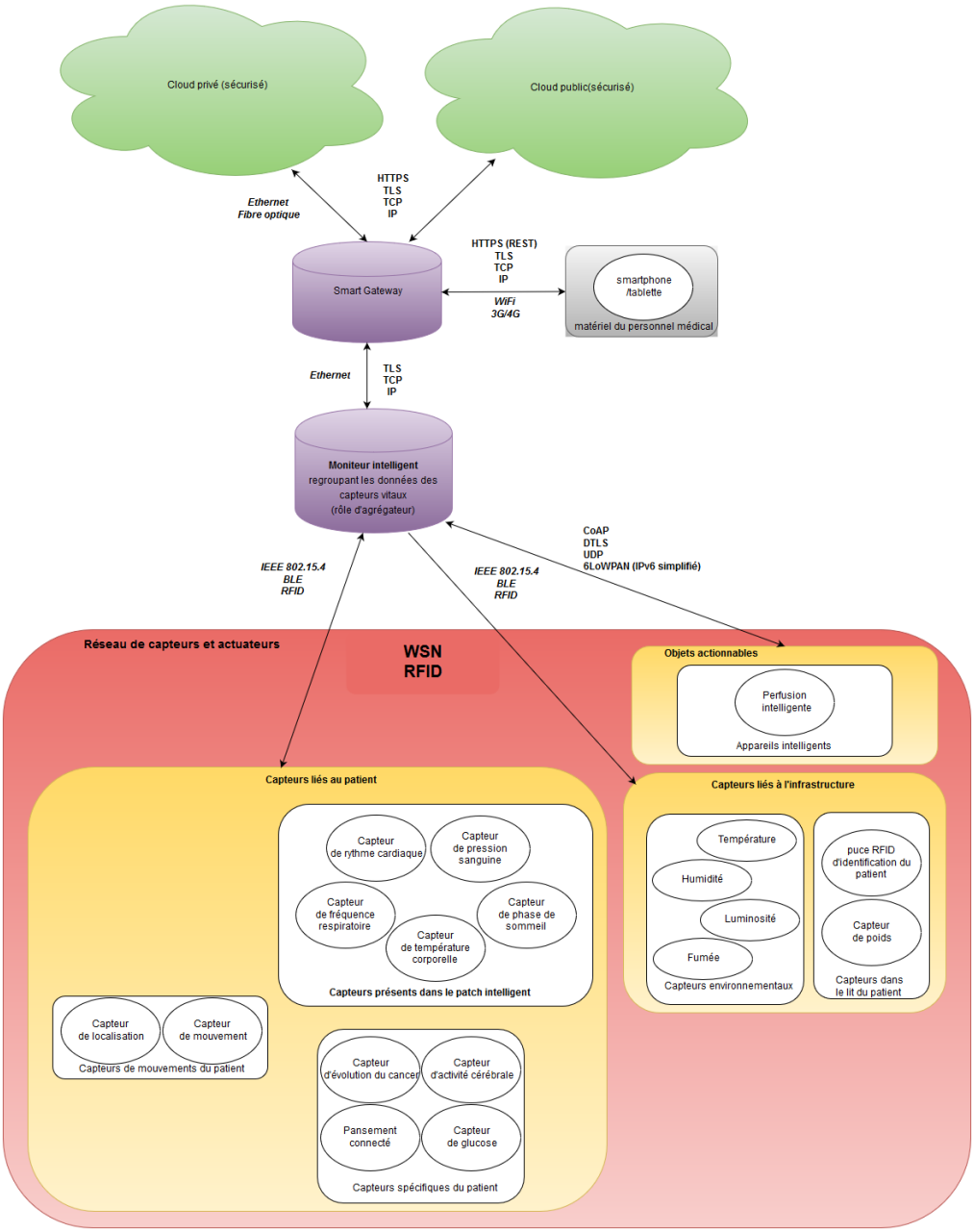

FIGURE 1 – Couches Dispositifs, et Protocoles de Réseau et de Communication de notre Système

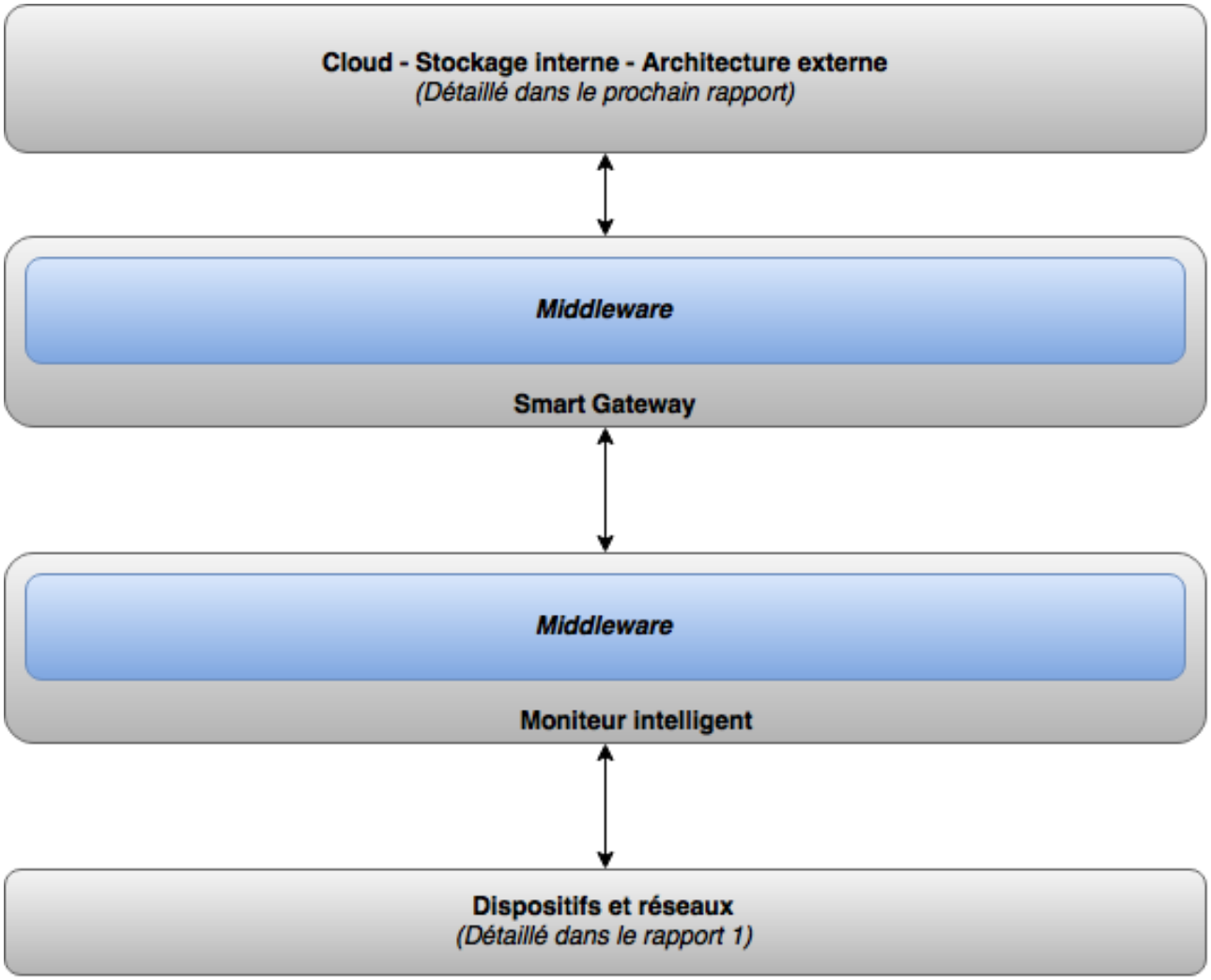

FIGURE 2 – Vue Globale de notre Middleware au sein de notre Système

2 Les Éxigences du Middleware

2.1 Les Éxigences Fonctionnelles

2.2 Les Éxigences Non-Fonctionnelles

2.3 Les Éxigences Architecturales

Le middleware requiert la prise en compte d'un certain nombre de problématiques architecturales qui ont influé sur nos choix. Nous en donnons la description ci-

dessous :

- ***Abstraction de programmation :*** Il est important d'offrir, dans n'importe quel middleware (pas uniquement ceux concernant IoT), des interfaces de programmation (API) afin de proposer aux développeurs de services ou d'applications des interfaces de programmation haut niveau qui permettent d'abstraire l'hétérogénéité des systèmes sous-jacents et ainsi d'isoler le développement des applications et des services de ces considérations bas-niveau.

- ***Interopérabilité :*** En raison de la grande diversité des protocoles, objets, applications qui sont très hétérogènes, particulièrement dans le domaine de l'IoT, un middleware se doit de permettre aux développeurs d'application ou de services de ne pas se préoccuper de ses problématiques et de se concentrer sur les données et services qu'il est possible d'obtenir et d'échanger avec les composantes du système. Cette interopérabilité doit être prise en compte au niveau des réseaux, de la sémantique et de la syntaxe des requêtes/ordres.

- ***Basé sur les services :*** Une architecture de middleware devrait être découpée en modules s'appuyant sur des services qui permettent une grande flexibilité et un ajout facilité de nouvelles fonctionnalités (soit plus avancées, soit récentes). Ce requis intervient également dans la prise en compte des deux requis précédents, car les services favorisent l'abstraction de programmation et l'interopérabilité en substituant le point d'accès qu'est le service aux composantes du système.

- ***Adaptabilité :*** Un middleware doit être évolutif et permettre de s'adapter à des nouvelles conditions d'environnement technique, technologique, ... Dans l'IoT spécifiquement, le réseau et l'environnement du middleware sont susceptibles de changer rapidement et fréquemment. Le contexte au niveau de l'application ou les requêtes possibles sont également susceptibles d'évoluer selon les changements apportés. Il est alors important que le middleware puisse s'adapter dynamiquement, ou au moins s'ajuster à ces variations.

- ***Connaissance du contexte (Context-awareness) :*** Le middleware IoT doit être conscient du contexte dans lequel les données, les événements, les ordres, ... ont été émis. Ceci comprend une conscience des dispositifs, utilisateurs et environnement du middleware ainsi que l'utilisation adéquate de ces informations afin d'offrir des services précis, efficaces et utiles pour l'utilisateur.

- ***Autonomie :*** Les appareils, technologies et applications intervenant dans l'IoT sont des agents actifs et intelligents, qui doivent pouvoir communiquer, échanger des informations et interagir sans nécessité d'inter-

vention humaine directe.

— ***Distribuée :*** Les applications, dispositifs et utilisateurs sont susceptibles d'être distribués géographiquement, et donc un middleware centralisé peut ne pas être suffisant pour supporter les services ou applications distribuées qui s'appuient sur lui. Une architecture de middleware pour IoT doit donc prendre en charge des fonctionnalités distribuées à travers l'infrastructure physique IoT.

3 Intergiciel - Middleware

3.1 Middleware niveau moniteur

3.2 Middleware niveau gateway

3.3 La sécurité dans le middleware

La vie privée et la sécurité sont des préoccupations majeures dans les systèmes IoT, et en particulier dans notre système centré autour de la santé. En effet, les données, en cas de fuite ou de corruption par un agent extérieur, sont susceptibles d'avoir des conséquences graves sur la vie du patient. Notre middleware doit donc prendre en compte cette problématique et répondre à ces défis. Comme la sécurité est un problème transverse, n'étant pas limité à l'un des maillons du système, nous avons choisi de l'évoquer dans cette partie séparée.
Après l'étude des risques potentiels, nous avons défini les modules suivants pour offrir des services et des garanties concernant la sécurité :

— ***Authentification :*** Il s'agit d'un module destiné à s'assurer et attester de l'identité d'un service, d'une application ou d'un utilisateur faisant une requête ou donnant un ordre à notre système, et particulièrement à notre middleware. L'objectif derrière ces considérations est de protéger le reste du système de requêtes provenant d'agents non-identifiés, tout en pouvant garantir l'origine des ordres, c'est à dire s'assurer qu'il n'est pas possible, pour une personne ou un service ayant fait une requête ou ayant transmis un ordre, de répudier l'avoir fait. Ceci est nécessaire à la fois pour la sécurité, mais aussi parfois pour des raisons de législation et de responsabilité devant la loi.
— ***Contrôle d'accès :*** Ce module, qui s'appuie tout d'abord sur le module d'*authentification* intervient à l'étape suivante. Une fois le requérant identifié, il convient de s'assurer qu'il a le droit d'accéder, de visionner, de modifier ou de supprimer les données visées, qu'elles soient dans le Cloud ou dans le stockage local mis en place dans notre architecture. Parallèlement, ce module s'assure également qu'un ordre passé ne soit transmis que si l'utilisateur ou l'application demandeuse est autorisée à effectuer cette action. Ce module répond aux considérations à la fois de

sécurité, mais aussi de privacité, en s'assurant que les données ne soit transmises qu'à des personnes ou des applications identifiées, autorisées et légitimes.

— ***Chiffrement des communications :*** Ce module a pour rôle de chiffrer les communications en utilisant les techniques de chiffrage appropriés afin de s'assurer qu'en cas de détournement des paquets, ou d'attaques de type Man In The Middle, les données sensibles comme le profil des patients, leur historique, leur dossier, les éléments venant de la base de données interne, ou du Cloud ou encore les informations de localisation des médecins ou des patients ne puissent être utilisés ou même consultés. Ce module agit pleinement pour favoriser la sécurité, mais plus encore la privacité. Il s'agit d'éviter ici les fuites de données personnelles, l'un des risques majeurs de l'IoT, renforcé par le caractère sensible des données transitant.

— ***Sécurité et intégrité des communications :*** Ce module est conçu pour s'assurer de la sécurité des communications, particulièrement celles orientées vers et depuis l'extérieur (par exemple le Cloud), mais aussi en interne entre les moniteurs et la Smart Gateway, ainsi que de l'intégrité des données reçues. En effet, une fois les données des capteurs agrégées (notamment avec le contexte) sur le moniteur, celles ci sont transmises à la Smart Gateway qui doit s'assurer de leur intégrité afin de ne pas enregistrer de fausses entrées. De même, certaines alertes étant générées par les moniteurs intelligents et transmises à la Smart Gateway pour leur transmission finale vers l'équipe médicale, il est important qu'aucun renseignement ne soit perdu au cours de ces transmissions. Ce module se concentre donc sur la sécurité, plus que la privacité des communications, car celle ci est renforcée par le module de chiffrement.

Ainsi, nous avons conçu cette partie transverse du middleware afin de s'assurer d'une sécurité et d'une protection de la vie privée maximale dans la limite des possibilités techniques. Les problématiques liées à la Qualité de Service (QoS) sont également pris en compte par la vérification de l'intégrité, donc de la fiabilité des données transmises, afin d'éviter des situations potentiellement risquées.

3.4 Récapitulatif

4 Conclusion

Nous avons présenté dans ce premier rapport notre solution IoT orientée Smart Health, afin de résoudre le problème de congestion des hôpitaux dans les services de soins intensifs. Plus précisément, nous avons proposé les dispositifs et les protocoles de communication et de réseau de notre solution. Nous avons aussi abordé les problématiques liées à notre solution, principalement celles liées au

QoS, à la privacité et à la sécurité. Ceci nous permettra ainsi pour nos prochains rapports de nous concentrer autour de ces problématiques afin de permettre à notre solution de respecter le plus possible ces paramètres.

Dans notre prochain rapport, nous aborderons ainsi les détails de conception concernant la couche située au dessus de celles présentées dans ce rapport, à savoir la couche Middleware. Nous verrons alors comment nous allons gérer le stockage des données et comment nous allons essayer de résoudre les problématiques liées à notre solution notamment avec la mise en place d'un environnement de stockage approprié. Nous détaillerons tout cela dans le deuxième rapport.