Intégration d'objets connectés hétérogènes et démonstrateurs

Étude des API et explication des regroupements

Adrien Casanova Victor Sallé
Marie-Catherine Turchini
Polytech Nice Sophia, SI5
{acasanov,salle,turchini}@polytech.unice.fr

Alexia Llorens
Université Nice Sophia Antipolis, M2
la002769@etu.unice.fr

Résumé

On assiste actuellement à une expansion et une démocratisation phénoménale des objets connectés qui envahissent notre quotidien. Beaucoup ont ainsi fait l'acquisition d'un bracelet sportif, d'une station météorologique connectée ou encore d'une montre intelligente. Au-delà de la multitude d'applications hétérogènes avec laquelle doit interagir l'utilisateur, se cache une autre problématique, plus technique : comment permettre aux développeurs d'homogénéiser l'accès aux objets et à leurs données ? La réponse à cette problématique se fera en fournissant un cadre de travail permettant de décloisonner leur usage afin de remonter l'ensemble des informations à un même niveau. Ce document présente ainsi la première étape nécessaire au développement de ce framework, à savoir la classification des objets connectés en catégories.

Mots-clés Internet of Things, Objets connectés, API, Framework

1. Introduction

Il existe de plus en plus d'objets connectés et la toile qu'ils constituent est très hétérogène. Il survient alors un réel besoin pour les développeurs d'avoir un cadre commun permettant de gérer l'hétérogénéité des dispositifs et de disposer d'un accès unifié à leur paramétrage et à leur utilisation. Ceci permettra alors de créer de nouveaux usages et de nouvelles applications mettant en œuvre non plus un seul dispositif mais une constellation de dispositifs. Pour un utilisateur lambda, notre projet représente la possibilité de pouvoir, sans connaissances particulières, interagir naturellement avec les objets, et surtout pouvoir accéder à toutes les informations qui l'intéressent de façon globale et non plus en consultant une multitude d'applications dédiées.

La première étape dans le développement de ce framework est une étude approfondie des objets connectés et des différentes contraintes qui leur sont liées. Le but principal de cette étude est d'établir une classification de ces objets en familles, ou catégories, qui constitueront les composants logiciels sur lesquels se basera notre framework. Cette classification sera ensuite appliquée aux objets mis à notre disposition afin de préparer leur utilisation dans le cadre d'une démonstration du framework.

2. Problématique

Afin donc d'homogénéiser l'utilisation des objets connectés, la première étape est de mettre en place des blocs correspondant aux différentes "familles", sur lesquels vont venir se connecter les nouveaux objets, et ce en fonction de ses caractéristiques. Cette connexion est à l'origine de la réunification des API en une unifiée.

Pour cela, il faut tout d'abord définir de façon précise ce qu'est une "famille" d'objets. Or, on se rend très rapidement compte du caractère flou, voire subjectif, de cette notion. En effet, il existe de nombreuses façons d'aborder les regroupements d'individus : faut-il le faire par moyens de communication ? Faut-il au contraire les rassembler par fonctionnalités similaires ? Répondre à ces questions n'est pas tâche aisée lorsque l'on constate le grand nombre de caractéristiques, diverses et variées : ce sont ces dernières qui vont nous permettre d'extraire plusieurs catégories. Il est donc primordial que l'étude se fasse sur l'ensemble des objets et leurs familles, et non pas de façon isolée.

Au-delà des premières questions qui viennent à l'esprit, d'autres apparaissent au cours de l'étude. Ayant en autre possession deux objets de la même marque, à savoir Withings, nous notons un point commun non négligeable : la présence d'une couche d'authentification (API *OAuth*) entre l'envoi des informations et la réception de celles-ci. Il serait donc également possible de rassembler les objets en fonction de leur constructeur, puisqu'au-delà de la technologie utilisée, nous soulignons des caractéristiques communes aux objets répondant à une même API provenant d'une unique source.

De plus, grâce à des recherches plus approfondies, nous avons retrouvé l'utilisation de cette API d'authentification pour d'autres objets connectés de constructeurs différents. Dans une optique d'optimisation et de factorisation, il faut donc envisager, au-delà des briques de connexion, de placer des blocs communs à plusieurs familles. Ainsi, une brique *OAuth* pourrait être mise à disposition de plusieurs regroupements d'objets et éviter par la même occasion une duplication de code inutile.

3. Familles

Différentes classifications émergent donc : il est possible de classifier les objets par protocole de communication, par fonctionnalités, par constructeur, etc. Le choix des familles permettant de regrouper nos objets s'est donc avéré important. Un choix porté sur une classification par domaine d'application aurait obligé à réécrire entièrement la partie communication, tandis qu'une classification par moyen de communication aurait impliqué une remontée à un niveau d'abstraction très haut pour que le framework puisse prendre en charge une majorité d'objets.

Le choix que nous faisons est donc d'attribuer non pas une mais plusieurs familles à chaque objet. De cette manière, un objet sera caractérisé par son protocole de communication, les fonctionnalités qu'il offre et ses aspects qui lui sont plus spécifiques comme les particularités offertes ou requises de la part de son constructeur. Ce choix devrait à terme nous permettre de prendre en compte une majorité des objets. Nous allons détailler les trois classifications fondamentales à mettre en place, à savoir une classification par domaines d'applications, par fonctionnalités, et par modes de com-

munication ; l'approfondissement des autres sera rendu disponible dans le document final de cette étude préliminaire[1].

3.1 Classification par domaines d'application

Objets dédiés Même si le nombre d'objets disponibles sur le marché augmente jour après jour, il est de manière générale possible de regrouper en familles de fonctionnalités les services qu'ils proposent. On pense évidemment à la famille des objets liés au sport, celle de ceux liés à santé, ou encore à la famille des objets d'intérêt domotique. Pour le moment, nous avons identifié les domaines suivants : sport - santé - domotique - loisirs - vêtements et accessoires - auto et moto - multimédia - animaux. Le fait de prendre en considération ce type de fonctionnalités nous permet de descendre en abstraction, en offrant de cette manière davantage de services, eux-mêmes plus proches de ceux réellement proposés par les objets. Aussi, il est tout à fait envisageable de descendre encore en abstraction en subdivisant ces grandes familles afin de faire naître des sous-familles comme pourrait l'être ≪ sécurité ≫ pour la famille des objets liés à la domotique.

Objets ouverts Cependant certains objets ne sont pas liés à un domaine d'application spécifique et peuvent être paramétrés afin de proposer différentes fonctions. On pense par exemple à l'objet Mother de Sen.se[2], qui utilise des capteurs pouvant servir de tracker d'activité, de thermomètre, détecteur de mouvement, etc en fonction de leur paramétrage. Le Nabaztag[3] quant à lui est très ouvert. De nombreux usages différents peuvent en être faits grâce à l'insertion de scripts.

3.2 Classification par modes de communication

Deux modes de communication sont à distinguer. Le premier cas concerne les objets qui communiquent directement avec l'application dédiée, dans une majorité des cas via Bluetooth mais parfois via NFC ou ZigBee par exemple. Cependant de plus en plus de constructeurs ne permettent plus une communication directe avec l'objet, et obligent à passer par leurs serveurs. Le raisonnement peut parfois être étonnant, puisque le bracelet Withings par exemple transfère d'abord ses données à l'application via Bluetooth, qui elle-même se charge de les transmettre aux serveurs de la marque, afin de permettre à d'autres dispositifs de récupérer ces données.

Il est donc primordial de gérer ces différents modes de communication d'une manière assez générale, pour que lors d'un échange via Bluetooth par exemple il ne soit pas obligatoire de réécrire l'intégralité du protocole de communication. De même, un appel à un service web semblant devenir la norme ces dernières années, il doit être simple à mettre en place dans le cadre de notre framework.

3.3 Classification par fonctionnalités

Authentification On observe une utilisation de protocoles d'authentification identiques chez différents constructeurs. Toujours dans l'exemple de Withings, l'accès aux services web se fait après une procédure d'authentification via le protocole OAuth. Ainsi les objets peuvent être regroupés selon le protocole d'authentification utilisé.

Communication événementielle Pour certains objets, il est possible de programmer des notifications ou actions qui seront déclenchées par des événements. Par exemple, le lapin Nabaztag en notre possession, peut être programmé pour réagir à la réception de données précisément choisies par son utilisateur, comme un nouveau Tweet ou email reçu, qui fera s'agiter l'oreille de l'animal.

Statut de l'objet Pour de nombreux objets il est possible d'obtenir des informations sur leur statut - allumé/éteint, actif/inactif - mais également de le modifier grâce à des fonctions d'allumage ou d'extinction par exemple. Ces fonctionnalités pourraient donc être exposées par une interface commune, qui permettrait ainsi de posséder l'ensemble des informations relatives par exemple à la présence ou l'absence d'un objet du réseau.

Récupération de données sur requête Certains objets utilisent des données qu'ils récupèrent de services distants, via le Wi-Fi par exemple, grâce à des requêtes automatiques. On pense par exemple au lapin qui pourrait, sur programmation de son utilisateur, énoncer à haute voix le temps du jour ; pour cela, il consommerait quotidiennement un service météorologique.

3.4 Les autres classifications

Certains constructeurs tel que Withings offrent une API commune[4] pour leurs objets. Celle-ci, souvent en REST, permet d'accéder aux données récoltées par l'ensemble des objets et transmises à un serveur. D'autres en revanche, comme le fabricant de stations météo Netatmo[5], mettent des SDK à disposition des développeurs. On pourrait donc envisager de classifier les produits selon les constructeurs.

4. Application/exemple objets

La matrice ci-dessous met en avant les caractéristiques des objets à notre disposition ainsi que les possibles regroupements qui peuvent en découler. On voit facilement, qu'avec à peine trois objets, s'offrent à nous plusieurs choix de familles.

	Pulse O2	Balance	Nabaztag
Communication			
WiFi		X	X
Bluetooth	X	X	
Constructeur			
Withings	X	X	
Karotz			X
Domaines			
Santé	X	X	
Sport	X		
Ouvert			X

5. Conclusion

Cette classification n'est qu'une première étape dans le développement de notre framework et est amenée à évoluer et à s'affiner au fur et à mesure de l'étude et de l'utilisation d'objets. Elle est présentée plus en détails dans un document annexe[1] que nous vous invitons à consulter afin d'en suivre l'évolution.

C'est donc sur les études faites jusqu'ici que s'appuieront la conception et l'implémentation, effectuées en parallèle, du framework et d'un démonstrateur lors de la phase suivante du projet. Plus précisément, une prise en main approfondie des objets à notre disposition et de leurs technologies permettront de guider la conception de notre solution logicielle. Cette dernière sera rapidement suivie par l'étude et la conception du démonstrateur dont le but est de réunir des données provenant des objets étudiés par le biais du framework. S'en suivront finalement les phases d'implémentation.

Références

[1] A. Casanova, A. Llorens, V. Sallé et M.-C. Turchini. *D2 - Étude des API et explication des regroupements*, 2014

[2] Mother de Sen.se. URL `www.sen.se`

[3] Karotz. URL `www.karotz.com`

[4] Withings. *API*, URL `http://oauth.withings.com/api/doc`

[5] Netatmo. *API & SDK*, URL `https://dev.netatmo.com/doc`