IUT des Pays de l'Adour - RT2
Module M4207C - Application informatique dédiée aux R&T

Projet UML / Java - 2021
Installation IoT

Version 6 octobre 2021 (12:24)

Expression des besoins

On désire gérer par des moyens informatiques (par exemple une application Java issue d'un projet de génie logiciel utilisant UML...) une installation IoT. Celle-ci sera composée de différents éléments :

- d'une centrale de commande assurant la supervision de l'installation : collecte des informations et éventuellement prise de décisions
- d'un ensemble de capteurs/actionneurs capables de remonter des informations à la centrale de commande (soit à sa demande en mode *pull*, soit de leur propre initiative en mode *push*) ou de déclencher une certaine action à la demande de la centrale
- des tableaux de bord destinés à afficher les informations de la centrale (soit en allant les récupérer sur la centrale, soit c'est la centrale qui "pousse" les infos vers la tableau de bord)

Le système que vous devez développer doit permettre d'enregistrer différents capteurs/actionneurs sur la centrale de commande. Celle-ci les interrogera alors régulièrement (si vous le souhaitez, la fréquence peut éventuellement être réglée différemment selon les capteurs). Pour simplifier, la valeur transmise par un capteur sera réduite dans un premier temps à un réel. Certains capteurs peuvent décider eux-mêmes de transmettre leurs données à la centrale. Soit pour envoyer une alerte, soit à intervalle régulier.

La centrale de commande devra stocker les valeurs reçues avec, pour chacune d'entre elles, au minimum la date et l'heure ainsi que la référence du capteur. Libre à vous de définir la structure de données pour stocker toutes ces informations.

Sur cette centrale de commande s'enregistreront également différents "tableaux de bord". Ce sont des composants destinés à diffuser de l'information :

- → valeur instantanée d'un capteur (au sens "temps réel mou", i.e. dernière valeur connue du capteur)
- → courbe/histogramme/... des valeurs d'un capteur sur une période donnée
- → exportation des valeurs d'un capteur dans un format quelconque pour ré-importation dans un autre logiciel (tout aussi quelconque... ☺)
- → etc...

Du point de vue de l'infrastructure à développer, le sujet est volontairement ouvert (certains diront peu précis à leur goût...) pour que vous puissiez laisser libre cours à votre imagination : structures de données utilisées (tableau(x), Vector, Hashtable,...), format des communications entre les différents composants (paramètres des appels de méthodes, résultats retournés), organisation de ces composants (tous les objets dans une même JVM, répartition sur le réseau via des appels RMI), etc...

Afin que votre projet puisse interopérer avec d'autres briques logicielles éventuellement écrites dans d'autres langages de programmation (ex : Python, NodeJS), il vous est demandé de mettre en place des communications selon le protocole MQTT.

Travail à réaliser

1. Vous développerez cette application dans les règles de l'art du génie logiciel.
   a) À partir des besoins exprimés ci-dessus, à vous d'analyser et concevoir une solution fonctionnelle. Vous vous aiderez pour cela du langage UML pour conceptualiser les différentes facettes de votre projet. NB : Bien entendu, cela ne vous dispense pas de la rédaction de documentations... Un cycle de vie en spirale (prototypes successifs) pourrait être un bon choix...
   b) Vous développerez ensuite la solution proposée en utilisant le langage Java (vu au cours du module M2207 intitulé "consolidation des bases de le programmation"). Les communications réseau entre les différents composants pourront se faire :
      - soit via le mécanisme des RMI [1]
      - soit via le protocole MQTT [2]
      - voire les 2 simultanément...
   c) Avant de livrer votre solution, vous prendrez bien évidemment soin de réaliser les tests nécessaires pour s'assurer de son bon fonctionnement.

Évaluation

Pour ce projet vous travaillerez en binôme afin de simuler une équipe de développement. Toute décision devra faire l'objet d'une discussion et d'un choix argumenté (consigné par écrit). Votre évaluation sur le module M4207C pour ce projet se fera en deux parties : une évaluation finale du projet terminé (lors de la dernière séance) et un suivi de l'avancement de votre projet au fur et à mesure des séances.

Pour ce projet vous disposez de 5 séances de TP (3h) et de 3 séances de TD (1h30). Ce qui fait globalement 13 créneaux d'1h30. Le projet se déroulant en binôme, cela vous fait donc **39 heures-hommes**. Voire 50% de plus si vous êtes en trinôme. Le planning (prévisionnel) de travail, et donc des évaluations, sera le suivant, sachant qu'il vous est tout à fait possible de prendre de l'avance si vous travaillez de manière assidue :

1. Diagramme des cas d'utilisation et quelques scénarii.
2. Petit à petit, détailler les scénarii en diagrammes de séquence et commencer à identifier les principaux composants de votre projet.
3. Diagramme de classes.
4. Commencer à (essayer de) développer les 1$^{\text{ères}}$ classes en Java.
5. Retour sur le diagramme de classes. Réfléchir aux diagrammes de collaboration, de déploiement ($\Rightarrow$ identifier les communications RMI). S'aider de diagrammes états-transitions et d'activités pour "fixer" les idées (ex : états d'une valeur d'un capteur).
6. Développement Java.
7. Développement Java.
8. 1$^{\text{ers}}$ tests du système. Réfléchir aux "statistiques" qu'il faudra calculer dans les tableaux de bord : A-t-on les informations nécessaire ? Quels algorithmes pour calculer les statistiques ? $\Rightarrow$ Formalisation de toutes ces "idées" sur les différents diagrammes UML.
9. Développement Java.
10. Séance d'évaluation du projet réalisé : produit livré (démo du projet) + réponse à des questions sur le développement et/ou le fonctionnement du projet.
11. ???

1. Java RMI : Remote Invocation Method
2. MQTT : Message Queuing Telemetry Transport

12. ???
13. ???

Liens utiles

— **UML**
  ▷ Support de cours M4207C (MM)
  http://munier.perso.univ-pau.fr/temp/M4207C/M4207C-MM-2021.pdf
  ▷ Unified Modeling Language à l'OMG
  http://uml.org/
  ▷ Logiciel UMLet
  http://www.umlet.com/
  http://munier.perso.univ-pau.fr/temp/M4207C/umlet_13.1.zip

— **Java**
  ▷ Support de cours M2207 (MM)
  http://munier.perso.univ-pau.fr/temp/M2207/M2207-MM-2021.pdf
  ▷ Java Platform, API Specification (version 7u45)
  http://munier.perso.univ-pau.fr/temp/jdk-7u45-apidocs/api/index.html
  ▷ Tutorial sur les RMI (MM)
  http://munier.perso.univ-pau.fr/temp/I6/rmi.pdf

— **MQTT**
  ▷ MQTT Essentials
  https://www.hivemq.com/mqtt-essentials/
  ▷ Eclipse Mosquitto™ - An open source MQTT broker
  http://mosquitto.org/