Mobile Embedded Group Project

Ilias Boulif, Jérémy Bricq, George Rusu

May 4, 2019

Contents

1 Introduction

The aim of this project is to propose an implementation of an IoT network using *Rime* where the sensor data is published through an MQTT-Rime gateway to normal MQTT subscribers. To achieve that we have implemented as requested by the project's statement, a tree-based routing protocol using Rime and an MQTT-Rime gateway.

Firstly, we are going to explain the general structure of our system. Secondly, we are going to present and explain the different messages exchanged in a normal run. Then, we will explain the organization and the rounting in our environement. And finaly, we will discuss about the optimizations we manage to implement.

The implementation source code of this project is available at [1].

2 General structure of our system

Our infrastructure is divised in 2 sides: over air side which cover the motes (root and sensor) and a wired network which covers the root, the gateway the broket and the subscribers.

Our system is a network composed of one root node and a number of sensor nodes. The network depends on the root which is the node responsible for transmitting the information to the subscribers. The sensor nodes for their part, collect the information for the root and forward them to him. using rime ! The nodes communicate by sending different kind of messages (see Section 3. Some of these messages are used to ensure a good working of the system. The network is built to keep sending information to the root even if there is a node is moved or deleted. The root is connected to the gateway in order to send the information to the subscribers. Some modes are built in the system. These modes describe the system's behaviour. The root receives the mode through the gateway, and forward it the nodes by the mean of the alive response packets.

3 Message format

In our implementation we have 3 packets structures. The node interprets the packet according to his type. We will explain each kind of packets and discuss the different types used by nodes to communicate between them.

Description of *packet*

1. Type : [DISCOVERY_REQUEST, DISCOVERY_RESPONSE, ALIVE_REQUEST, ALIVE_RESPONSE]
2. Rank : the rank of the emission node
3. Mode : the mode of sent [DATA_ON_CHANGE, DATA_PERIODICALLY]
4. HaveSubscriber : Boolean that indicates if someone is subscribed to some topics

Description of *data_packet*

1. Type : [SENSOR_DATA]
2. NodeSrc : The node where the information come from
3. NodeRank : The rank of the node source
4. DataTemp : The temperature sent
5. DataOthers : A place used for another kind of data

Description of *data_packet_aggregate*

1. Type : [SENSOR_DATA_AGGREGATE]
2. numberPacket : The number of packets
3. packet1 : The first packet
4. packet2 : The second packet

DISCOVERY_REQUEST When a node starts, he is alone and has a rank of 0. To connect to a network, Each sensor node need to send in broadcast a discovery request to find a parent connected to that network. Even when connected, the node will still keep sending discovery request in order to find a better parent, if exists. The lower is the rank (except a rank of 0) and the nearest is the node from the root node.

DISCOVERY_RESPONSE When a node receives a discovery request packet he replies with a discovery response with his rank. The node that has made the request receive now the response and assign the his rank the value of the rank of his parent + 1.

ALIVE_REQUEST Each node has to keep sending an alive request in order to know if his parent is still available in the network. After 4 alive requests without any response, the child considers that he has no parent. If he has no more parent, his rank return back to 0 and he has to research for a new one.

ALIVE_RESPONSE When a node receives an alive request, he has to replies with an alive response to his child to validate his presence. Each alive response is accompanied with the rank of the parent. In this way, if the parent is disconnected from the network (rank = 0), the child can also put his rank to 0 and can disconnect.

SENSOR_DATA A sensor data packet is a packet that contains some information collected by the sensor. This packet travels from his sensor to the subscribers by going forwarded by the nodes on the path to the root.

SENSOR_DATA_AGGREGATE A sensor data aggregate packet is a packet that contains many sensors data packets in one. This packet will be explained in details in the Section 5 : Optimizations.

4 Network organization and routing

5 Gateway

6 MQTT-Subscriber

7 Optimizations

8 Conclusion

References

[1] Github repository, *MobileEmbeddedComputing*, Available at : `https://github.com/georgesrusu/MobileEmbeddedComputing`.