IoT Projects - 2019/2020

Edoardo Longo
edoardo.longo@polimi.it

May 27, 2020

1 General Rules, Grading and Deadlines

Grade composition of the entire course:

- 25 out of 30 points are assigned based on the written exams
- up to 5 points are assigned based on the home projects done during the hand-on activities
- up to 3 points are assigned based on final projects

General rules:

- Projects are NOT mandatory. One student can decide not to take any project; in this case, that maximum grade he/she can get will be 25/30.
- Projects can be developed in groups of maximum 3 people.
- Project deadline is September 1st 2020

IMPORTANT 1: ONLY THE PROJECTS REGISTERED ON THE ONLINE FORM WILL BE CONSIDERED. LATE REGISTRATION WILL NOT BE ACCEPTED.

IMPORTANT 2: THE DELIVERY IS ONE AND ONLY ONE. WHEN YOU DELIVER THE PROJECT YOU CANNOT RE-DELIVER THE PROJECT FOR A BETTER GRADE. Send an email to `edoardo.longo@polimi.it` to deliver the project.

IMPORTANT 3: REGISTRATION IS NOT BINDING. IF YOU REGISTER FOR A PROJECT AND THEN DECIDE AFTERWARDS YOU DON'T WANT TO DELIVER IT, THAT'S OK.

Students willing to take the project assignment must choose among the project proposals listed in the following section or propose an original project, using the online form available at `https://forms.gle/jtFfD1peSa8bNMCu8` by **June 17th, 2020**.

For original projects, write to `edoardo.longo@polimi.it` describing the project you intend to implement.

2 Proposed Projects

The following projects proposal are thought for being implemented with the tools seen during the hands-on lectures.

You have to deliver the following items:

- Complete source code of the project
- Self-explanatory log file, showing that your project works. Try to be as detailed as possible when preparing the log file (i.e., use debug/print statements in all the crucial phases of your project)
- Project report (max. 3 pages), which summarizes your approach in solving the problem, including figures when needed. Don't include source code in the project report.

Projects will be evaluated based on the rationale and technical depth of the design choices, correctness of the source code and organization and clarity of the project report.

2.1 Project 1. Keep your distance

You are requested to design and implement a software prototype for a social distancing application using TinyOS and Node-Red and test it with Cooja. The application is meant to understand and to alert you when two people (motes) are close to each other. The operation of the software is as follow:

- Each mote broadcasts its presence every 500ms with a message containing the ID number.
- When a mote is in the proximity area of another mote and receive a message, it (1) stores the received mote ID and (2) triggers an alarm. Such alarm contains the ID number of the near mote. It is shown in Cooja and forwarded to Node-Red with a different socket for each mote.
- Upon the reception of the alert, Node-red sends a notification through IFTTT[1] to your mobile phone.

Use at least 5 motes. Start the simulation with all the mote far away from each other an move them with the mouse testing different configurations.

[1]IFTTT is a service to create conditional statements. It can be easily integrated with Node-Red and it's an easy way to send notification to your smartphone. This is a useful reference: `https://wiki.instar.com/Advanced_User/Node-RED_and_IFTTT/`, but don't be afraid to discover it by yourself. It's very easy and enjoyable.

2.2 Project 2. LoraWAN-like sensor networks

Implement and showcase a network architecture similar to LoraWAN in TinyOS (or Contiki). The requirements of this project are:

1. Create a topology with 5 sensor nodes, 2 gateway nodes and one network server node, as illustrated in Figure 1.

2. Each sensor node periodically transmits (random) data (also random), which is received by one or more gateways. Gateways just forward the received data to the network server.

3. Network server keeps track of the data received by gateways, taking care of removing duplicates. An ACK message is sent back to the forwarding gateway, which in turn transmits it to the nodes. If a node does not receive an ACK within a 1 second window, the message is re-transmitted.

4. The network server node should be connected to Node-RED, and periodically transmit data from sensor nodes to Thingspeak (or another service of your choice[2]) through MQTT.

5. Thingspeak (or the cloud service you use) must show at least two charts and one gauge.

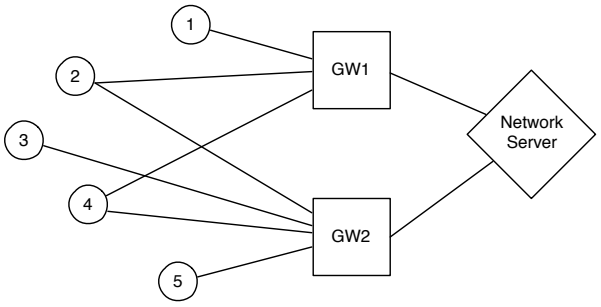

Figure 1: Network topology.

[2]freeboard, thingsboard, etc...