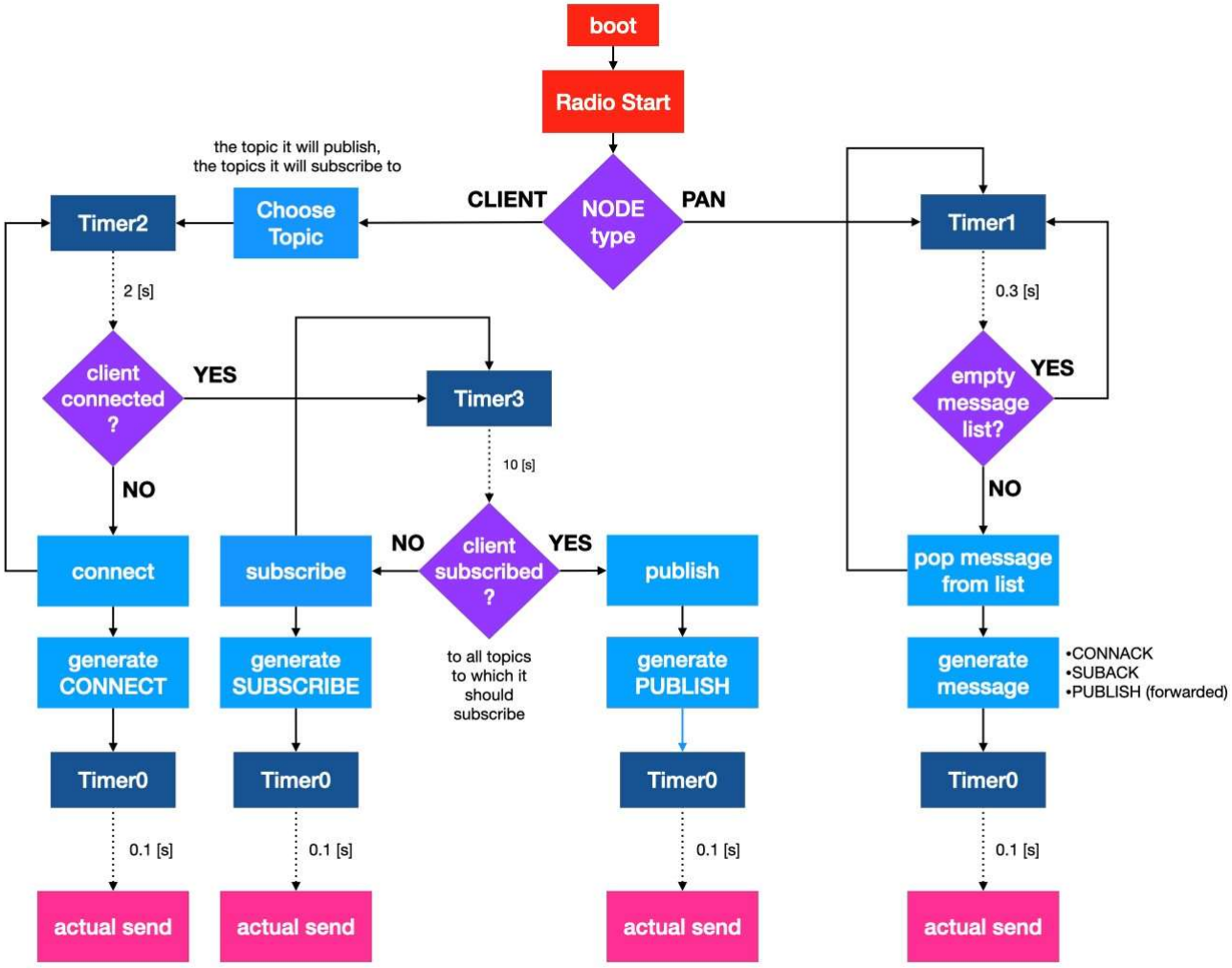
IoT 2023 FINAL PROJECT

(1) Name: Nicolò Pisanu Person Code: 10827469

(2) Name: Marco Viviani Person Code: 10528650

TinyOS



(schematic that represent the algorithm implemented in TinyOS)

BOOT THE APPLICATION

Boot the application. If the booting has been successful turn on the radio communication. If you are the PAN coordinator call Timer1 and initialize the client list. If not, you are a client: choose a topic in which you want to public and the topics you want to subscribe to; then call Timer2.

HANDLING TIMERS

Three timers have been implemented to manage the exchange of messages between the PAN coordinator and the clients.

Timer1 handles the PAN message list. The PAN, to try to keep up with the timing without losing the messages from the clients, saves the messages that have arrived in a list to then be able to send them easily. If the list is empty wait and re-call Timer1. If not, send a message to a CLIENT.

Timer2 handles CLIENT's connection. If you are connected call Timer3 in order to subscribe to a topic. If not re-try to connect to the PAN coordinator.

Lastly, Timer3 handles CLIENT's subscriptions and publications. If the CLIENT is already subscripted to all its chosen topics, try to PUBLISH a message. If not, re-try the subscription.

HANDLING RECEIVED MESSAGES

If the message is correctly received create an answer message depending on the type of message received:

- **CONNECT message received:** you are compulsorily the PAN. If the sender CLIENT is not connected send a CONNACK message;
- **CONNACK message received:** you are compulsorily a CLIENT that correctly received a CONNACK. You have been correctly connected to the PAN;
- **SUBSCRIBE message received:** you are compulsorily the PAN that is receiving a SUBSCRIPTION. If the node is connected and not already subscribed to that topic, answer with a SUBACK;
- **SUBACK message received:** you are compulsorily a CLIENT that correctly received a SUBACK. You are subscribed to the desired topic;
- **PUBLISH message received:** if you are the PAN coordinator forward that message to all the clients subscribed to that topic. Else – if you are a client – you have correctly received the publish message that some CLIENT posted through the PAN coordinator;
  In addition, the PAN sends all the PUBLISH messages it receives to Node-RED via TCP.

Thingspeak

Create a new channel with 3 topics TEMPERATURE, HUMIDITY and LUMINOSITY.

Define a new device object and select the previously created channel.

Now the channel is created.

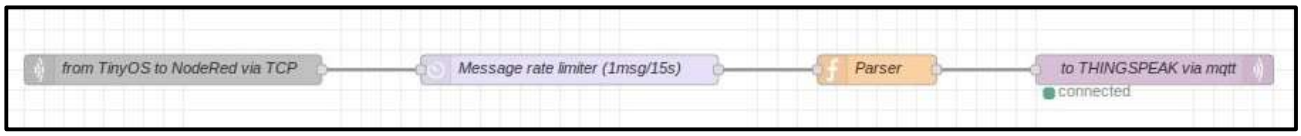
Username: OxQcNjoWDTsPNhAdFDYZCiA

clientId: OxQcNjoWDTsPNhAdFDYZCiA

password: GtO6HokUd101rSuTaJn1J8hp

Channel ID: 2256869

Node-RED



Receive messages from TinyOS via TCP by setting the port (1234) on which the TinyOS code publishes.

Since the messages are all sent almost simultaneously due to the TOSSIM simulation, it was necessary to introduce a rate limiter. It sends a message every 15 seconds.

The “Parse” function node parses the payload of every incoming message from the rate limiter to the topics and encapsulates it creating the message with the correct syntax to send it to Thingspeak.

In the MQTT sender we set “channels/2256869/publish” as topic.

In the “security” and “connection” sections of the node, we must specify also ClientID, username and password.

The name of the server must be the name of the device created in Thingspeak. The port is 1883.

GRAPHS AVAIBLE AT CHANNEL PUBLIC URL: https://thingspeak.com/channels/2256869