Internet of Things

Challenge 04.

***Student**: Marco Bendinelli* ***ID**: 10673478*

***Required Software**: TinyOS, TOSSIM*

Architecture

The topology of the network is composed by two motes that have different behavior and that communicate through a Radio channel with a power of -10 dBm for a better reliability.
The mote 1 has the behavior of a client, so it sends multiple requests to the mote 2. The mote 2, instead, has the behavior of a server, it responds to requests by sending fake sensor readings.
Note that even if not specified, I decided to delay the switching off of the mote 1 by one second to allow the correct reception of the last response (it works only if the channel is not too much noisy). The mote 2, on the other hand, I decided to keep it on because he doesn't know when to stop responding to requests, for that purpose I should implement a particular strategy.

Parameters: X = 9, Y = 73

Mote1: the "client"

When the mote is booted, it activates its radio, and it starts a timer of 1 second period.
When the timer fires, it does the following step:

1. it prepares the payload of the packet by setting the message type (REQ) and by setting the field counter with its local counter (used interface: Packet)
2. it tells a protocol that when it sends this packet, it should use synchronous acknowledgments (used interface: PacketAcknowledgements)
3. it sends the packet to the other mote (used interface: AMSend)

At the end, it checks how many packets it has sent: if X = 9 packets are sent and acknowledged it stops the timer, otherwise it increments the counter.
Note that the specification says explicitly to increment the counter when an ack is received, so it may happen that the client receives the ack but not the response with the reading, due to the noise of the channel.

Mote2: the "server"

When the mote is booted, it activates its radio, and it waits for the requests.
When it receives a request, it reads a reading thanks the fake sensor component and it prepares the payload of the packet by setting the message type (RESP), by setting the field counter with the counter present in the packet just received and by setting the field value with the sensor reading. Then, it tells the protocol that the packet must be acknowledged, and it sends the message to the mote1.

GitHub: https://github.com/MarcoBendinelli/IoT_Polimi