Internet Of Things - Smart Bracelets

STUDENTS:

Aman Gabba 10793117

Maite Johanna Siebel 10811499

Introduction

Smart wearables are gaining ever-increasing popularity among consumers as they provide many opportunities for monitoring and optimizing areas of private life. One example of such devices are smart bracelets, worn at the wrist of the user.
In this project, we design, implement, and test the software prototype for a pair of smart bracelets which are worn by parents and their child to track the child's position and alert the parent when the child goes too far or falls.
This report is structured as follows: In Section 2 we describe the file structure of the project as well as the contents and purpose of each file. In 3, we report the simulation and remarks.

Description of the files

▼ SmartBracelet.h

In this file we define the structures of the two types of messages and a structure for the status sent from the fake sensor. One of the message structure is used for information, sent from child bracelets with status and coordinates and the second one is used by both types of bracelets during pairing phase, to send a broadcast message containing sender's address, bracelet's pairing key and message type to identify whether it's a pairing request or pairing response.

▼ SmartBraceletApp.nc

The file contains the various modules used for the app. Among the components we used we have AMSender, AMReceiver, TimerMilliC, ActiveMessage and FakeSensorC. We also used the following interfaces: Boot, Receive, AMSend, SplitControl, Packet, PacketAcknowledgements, MilliTimerPair, MilliTimerMsg and MilliTimerAlarm.

- ▼ SmartBracelet.nc

  Contains the logic for the application. Whenever a new node (each node is a different bracelet) starts, a key is assigned to it. For simplicity every odd node is a parent and the next even number is it’s child (e.g. “Node 1” and “Node 3” are parents, “Node 2” is child of “Node 1” and “Node 4” is child of “Node 3”). Every parent and child pair have the same key, which is used for pairing.

  The application is divided in two main phases: pairing phase, where the nodes once booted send a pairing request (every 3 seconds in the implementation, but can be easily changed) in broadcast to all the other nodes trying to connect, and the ones receiving the requests check the keys, and in case they match the node sends back a pairing response, thus connecting the two devices and moving towards the next phase; in information phase the child sends to it’s parent the status of the bracelet every 10 seconds (generated randomly by the fake sensor following a given distribution). If the parent bracelet receives an information message with status “Falling” it sends an alarm notifiying the parent, an alarm is also sent in case the parent bracelet doesn’t receive any new information from it’s child for more than 60 seconds.

- ▼ FakeSensorC.nc

  The file contains the interfaces and components used for the fake sensor.

- ▼ FakeSensorP.nc

  The file contains the logic to generate random status for the child bracelet, while the coordinates are completely random, the status is generated according to a given distribution (30% Standing, 30% Walking, 30% Running and 10% Falling).

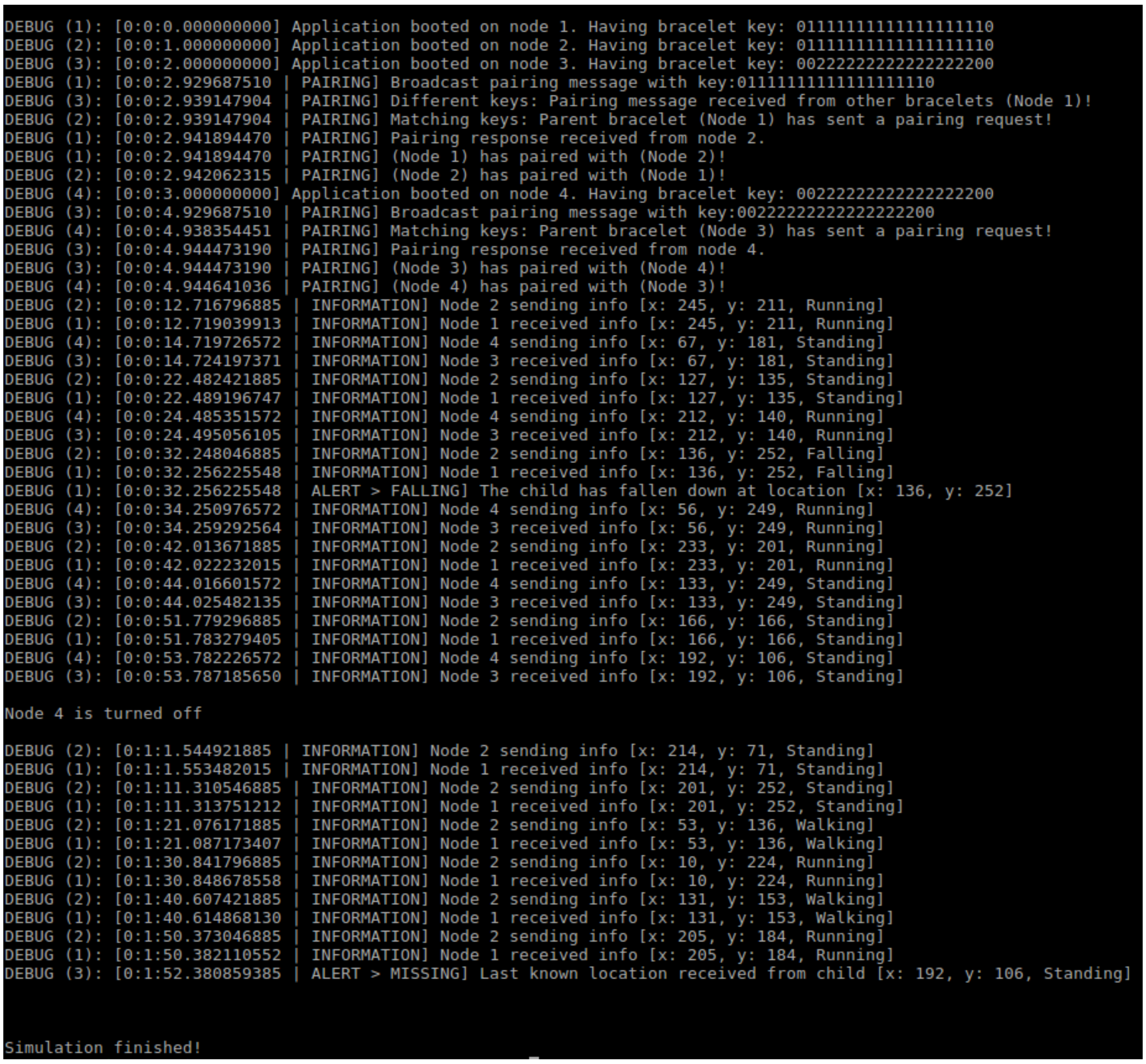
- ▼ RunSimulationScript.py

  The file contains the code to run the TOSSIM simulation, small changes have been made to the original one, such as the number of nodes and a line of code to simulate a child bracelet being too far from the parent by simply turning the node off.

Simulation log

The following image shows messages exchanged between the nodes. As mentioned earlier, there are two pairs, “Node 1” has as it’s child “Node 2” and “Node 3” is parent of “Node 4”. The nodes send the broadcast message every 3 seconds after boot up, in the following example “Node 2” and “Node 4” both pair before being able to actually send the message as they received the pairing request before sending one.

Once the nodes check if the keys match they send back the response, and once the other node receives it they connect and go to the information phase. The example shows both the messages sent and the received ones in the information phase, but the alarms are sent out only when the child is either falling (at 32 seconds) or missing (the last information message “Node 3” received from it’s child “Node 4” was at 53 seconds, and a minute after that a missing alarm is sent, with the last known status).



```
DEBUG (1): [0:0:0.000000000] Application booted on node 1. Having bracelet key: 01111111111111111110
DEBUG (2): [0:0:1.000000000] Application booted on node 2. Having bracelet key: 01111111111111111110
DEBUG (3): [0:0:2.000000000] Application booted on node 3. Having bracelet key: 00222222222222222200
DEBUG (1): [0:0:2.929687510 | PAIRING] Broadcast pairing message with key:01111111111111111110
DEBUG (3): [0:0:2.939147904 | PAIRING] Different keys: Pairing message received from other bracelets (Node 1)!
DEBUG (2): [0:0:2.939147904 | PAIRING] Matching keys: Parent bracelet (Node 1) has sent a pairing request!
DEBUG (1): [0:0:2.941894470 | PAIRING] Pairing response received from node 2.
DEBUG (1): [0:0:2.941894470 | PAIRING] (Node 1) has paired with (Node 2)!
DEBUG (2): [0:0:2.942062315 | PAIRING] (Node 2) has paired with (Node 1)!
DEBUG (4): [0:0:3.000000000] Application booted on node 4. Having bracelet key: 00222222222222222200
DEBUG (3): [0:0:4.929687510 | PAIRING] Broadcast pairing message with key:00222222222222222200
DEBUG (4): [0:0:4.938354451 | PAIRING] Matching keys: Parent bracelet (Node 3) has sent a pairing request!
DEBUG (3): [0:0:4.944473190 | PAIRING] Pairing response received from node 4.
DEBUG (3): [0:0:4.944473190 | PAIRING] (Node 3) has paired with (Node 4)!
DEBUG (4): [0:0:4.944641036 | PAIRING] (Node 4) has paired with (Node 3)!
DEBUG (2): [0:0:12.716796885 | INFORMATION] Node 2 sending info [x: 245, y: 211, Running]
DEBUG (1): [0:0:12.719039913 | INFORMATION] Node 1 received info [x: 245, y: 211, Running]
DEBUG (4): [0:0:14.719726572 | INFORMATION] Node 4 sending info [x: 67, y: 181, Standing]
DEBUG (3): [0:0:14.724197371 | INFORMATION] Node 3 received info [x: 67, y: 181, Standing]
DEBUG (2): [0:0:22.482421885 | INFORMATION] Node 2 sending info [x: 127, y: 135, Standing]
DEBUG (1): [0:0:22.489196747 | INFORMATION] Node 1 received info [x: 127, y: 135, Standing]
DEBUG (4): [0:0:24.485351572 | INFORMATION] Node 4 sending info [x: 212, y: 140, Running]
DEBUG (3): [0:0:24.495056105 | INFORMATION] Node 3 received info [x: 212, y: 140, Running]
DEBUG (2): [0:0:32.248046885 | INFORMATION] Node 2 sending info [x: 136, y: 252, Falling]
DEBUG (1): [0:0:32.256225548 | INFORMATION] Node 1 received info [x: 136, y: 252, Falling]
DEBUG (1): [0:0:32.256225548 | ALERT > FALLING] The child has fallen down at location [x: 136, y: 252]
DEBUG (4): [0:0:34.250976572 | INFORMATION] Node 4 sending info [x: 56, y: 249, Running]
DEBUG (3): [0:0:34.259292564 | INFORMATION] Node 3 received info [x: 56, y: 249, Running]
DEBUG (2): [0:0:42.013671885 | INFORMATION] Node 2 sending info [x: 233, y: 201, Running]
DEBUG (1): [0:0:42.022232015 | INFORMATION] Node 1 received info [x: 233, y: 201, Running]
DEBUG (4): [0:0:44.016601572 | INFORMATION] Node 4 sending info [x: 133, y: 249, Standing]
DEBUG (3): [0:0:44.025482135 | INFORMATION] Node 3 received info [x: 133, y: 249, Standing]
DEBUG (2): [0:0:51.779296885 | INFORMATION] Node 2 sending info [x: 166, y: 166, Standing]
DEBUG (1): [0:0:51.783279405 | INFORMATION] Node 1 received info [x: 166, y: 166, Standing]
DEBUG (4): [0:0:53.782226572 | INFORMATION] Node 4 sending info [x: 192, y: 106, Standing]
DEBUG (3): [0:0:53.787185650 | INFORMATION] Node 3 received info [x: 192, y: 106, Standing]

Node 4 is turned off

DEBUG (2): [0:1:1.544921885 | INFORMATION] Node 2 sending info [x: 214, y: 71, Standing]
DEBUG (1): [0:1:1.553482015 | INFORMATION] Node 1 received info [x: 214, y: 71, Standing]
DEBUG (2): [0:1:11.310546885 | INFORMATION] Node 2 sending info [x: 201, y: 252, Standing]
DEBUG (1): [0:1:11.313751212 | INFORMATION] Node 1 received info [x: 201, y: 252, Standing]
DEBUG (2): [0:1:21.076171885 | INFORMATION] Node 2 sending info [x: 53, y: 136, Walking]
DEBUG (1): [0:1:21.087173407 | INFORMATION] Node 1 received info [x: 53, y: 136, Walking]
DEBUG (2): [0:1:30.841796885 | INFORMATION] Node 2 sending info [x: 10, y: 224, Running]
DEBUG (1): [0:1:30.848678558 | INFORMATION] Node 1 received info [x: 10, y: 224, Running]
DEBUG (2): [0:1:40.607421885 | INFORMATION] Node 2 sending info [x: 131, y: 153, Walking]
DEBUG (1): [0:1:40.614868130 | INFORMATION] Node 1 received info [x: 131, y: 153, Walking]
DEBUG (2): [0:1:50.373046885 | INFORMATION] Node 2 sending info [x: 205, y: 184, Running]
DEBUG (1): [0:1:50.382110552 | INFORMATION] Node 1 received info [x: 205, y: 184, Running]
DEBUG (3): [0:1:52.380859385 | ALERT > MISSING] Last known location received from child [x: 192, y: 106, Standing]



Simulation finished!
```