2 Proposed Projects

The following projects proposal are thought for being implemented in TinyOS, although they may be implemented also with any of the operating systems seen during the lectures (i.e., Contiki or MoteRunner). Therefore, you are free to choose which O.S. to use, even if TinyOS is probably the best and easiest way.

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.

2.1 Project1. Smart Bracelet - up to 8 points

You are requested to design, implement and test a software prototype for a smart bracelet. The bracelet is meant to be worn by both a child and her/his parent to keep track of the child position and trigger alerts when a child goes too far. This kind of bracelets are actually becoming more and more popular, with some commercial available prototypes already on the market.

The operation of the smart bracelet couple is as follows:

1. Pairing phase: at startup the parent's bracelet and the child's bracelet broadcast a 20-char random key, that it used to uniquely couple the two devices. The same random key is pre-loaded at production time on the couple of devices: upon reception of a random key, a device check if the received random key is equal to the stored one and if the check if positive stores the address of the source device in memory. Then, a special message is transmitted (in unicast) to the source device to stop the pairing phase and move to the next step.

2. Operation mode: in this phase, the parent's bracelet listen for messages on the radio and accepts only messages coming from the child's bracelet. The child's bracelet periodically transmit INFO messages (one message every 10 seconds), containing the position X,Y of the child and an estimate of the his/her kinematic status (STANDING, WALKING, RUNNING, FALLING).

3. Alert Mode: upon reception of an INFO message, the parent's bracelet reads the content of the message. If the kinematic status is FALLING, the bracelet sends a FALL alarm, reporting the position X,Y of the children. If the parent's bracelet does not receive any message, after one minute from the last received message a MISSING alarm is sent reporting the last position received.

2.1.1 Requirements

1. Implement the prototype with the O.S. of your choice (including application logic, message formats, etc...). The X,Y coordinates can be random numbers, and the kinematic status should be randomly selected according to the following probability distribution: $P(\text{STANDING}) = P(\text{WALKING}) = P(\text{RUNNING}) = 0.3$, $P(\text{FALLING}) = 0.1$. (up to 50% of the total project points).

2. Simulate your implementation with 2 couples of bracelets at the same time. Your simulation should demonstrate that your code follows the design requirements: if you simulate using TOSSIM, you can emulate that a node goes out of range by turning it off (i.e., calling mote.turnOff() in python). In Cooja, you can just move a node out of the communication range of the other one. (up to 75% of the total project points).

3. Attach your simulation to Tossim-live or Node-red: alarm messages should be transmitted on the serial port and their output should be readable on the terminal or on the Node-Red dashboard (up to 100% of the total project points).