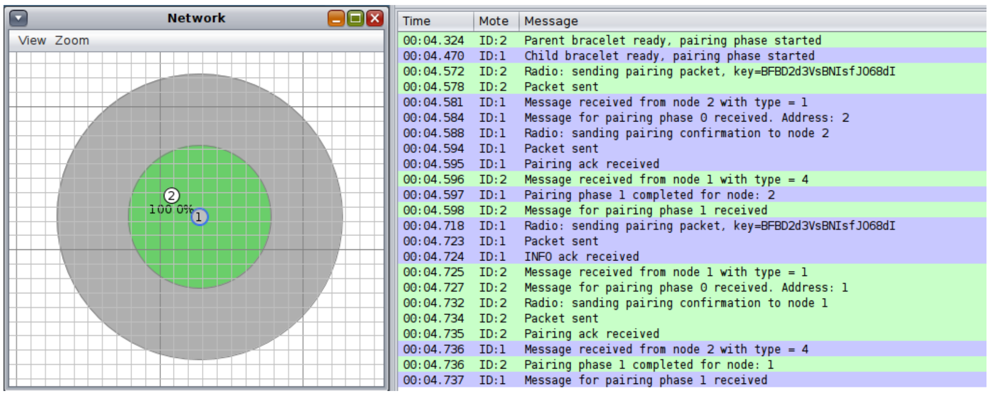
Antonio Guadagno – 10561018
Claudio Alfredo Emanuele – 10682738

The pairing phase

To be able to exchange messages with each other, the parent bracelet and the child bracelet have to be paired.

Everything begins with boot phase, in which we start the radio and a periodic timer of 250 ms for the pairing messages. When the timer fires, each bracelet sends a paring message in broadcast. Each **pairing message** contains the key associated the bracelet which has been randomly chosen starting from the TOS_NODE_ID.

When a pairing confirmation is received, the pairing phase ends and the timer of 250ms is stopped.



```
Time       Mote  Message
00:04.324  ID:2  Parent bracelet ready, pairing phase started
00:04.470  ID:1  Child bracelet ready, pairing phase started
00:04.572  ID:2  Radio: sending pairing packet, key=BFBD2d3VsBNIsfJO68dI
00:04.578  ID:2  Packet sent
00:04.581  ID:1  Message received from node 2 with type = 1
00:04.584  ID:1  Message for pairing phase 0 received. Address: 2
00:04.588  ID:1  Radio: sanding pairing confirmation to node 2
00:04.594  ID:1  Packet sent
00:04.595  ID:1  Pairing ack received
00:04.596  ID:2  Message received from node 1 with type = 4
00:04.597  ID:1  Pairing phase 1 completed for node: 2
00:04.598  ID:2  Message for pairing phase 1 received
00:04.718  ID:1  Radio: sending pairing packet, key=BFBD2d3VsBNIsfJO68dI
00:04.723  ID:1  Packet sent
00:04.724  ID:1  INFO ack received
00:04.725  ID:2  Message received from node 1 with type = 1
00:04.727  ID:2  Message for pairing phase 0 received. Address: 1
00:04.732  ID:2  Radio: sanding pairing confirmation to node 1
00:04.734  ID:2  Packet sent
00:04.735  ID:2  Pairing ack received
00:04.736  ID:1  Message received from node 2 with type = 4
00:04.736  ID:2  Pairing phase 1 completed for node: 1
00:04.737  ID:1  Message for pairing phase 1 received
```

Note that, for a proper pairing, we have specified that the child bracelet and the parent bracelet can be paired only if they have the same key! This is because we want to avoid wrong pairings in those cases in which we have 2 or more pairs of bracelets in the same area. For the sake of simplicity, in the previous picture we show the pairing between a single pair of bracelets, but the same mechanism works also if we consider 2 or more pairs of bracelets as you can see in the simulation.

The operational phase

After the pairing, the child bracelet and the parent bracelet can exchange messages with each other.

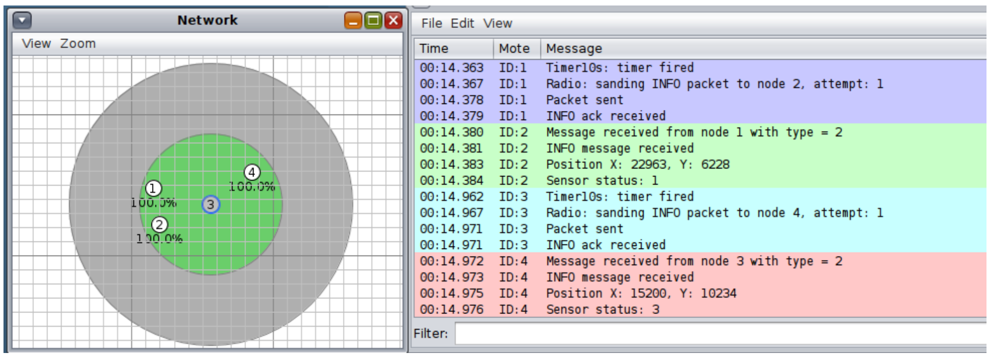
For the operational phase we need 2 timers:

1) A timer of 10s for the child bracelet
2) A timer of 60s for the parent bracelet

When the child timer fires, an info message is sent to the parent. **Info messages** contains the child position (x,y) and the child kinematic status (standing, walking running or falling). Info messages requires ACKs from the parent bracelets and they are retransmitted for a maximum of 3 times before being discarded.

When the parent bracelet receives an info message, it stores the last known child position and, if the kinematic status is “falling”, a **fall alarm** reporting position (x,y) is showed to the parent.

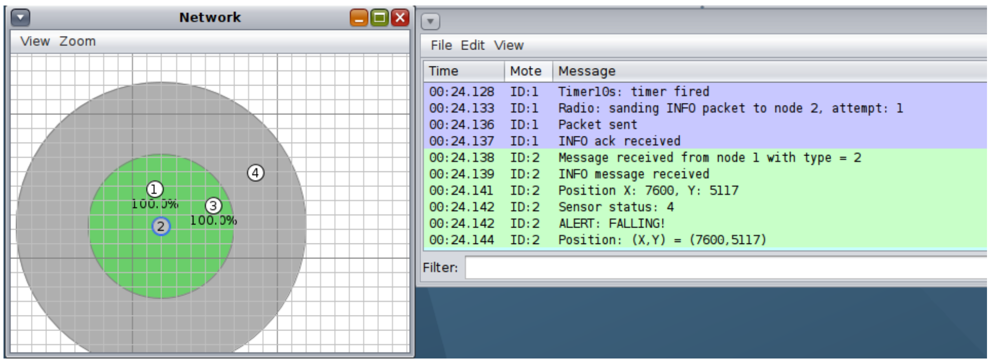
Moreover, when the parent bracelet receives an info message, a one shot timer of 60s starts. If the parent bracelet does not receive an info message from the child bracelet for 60s, a **missing alarm** reporting the last know position of the child is showed to the parent.



```
Time       Mote  Message
00:14.363  ID:1  Timer10s: timer fired
00:14.367  ID:1  Radio: sanding INFO packet to node 2, attempt: 1
00:14.378  ID:1  Packet sent
00:14.379  ID:1  INFO ack received
00:14.380  ID:2  Message received from node 1 with type = 2
00:14.381  ID:2  INFO message received
00:14.383  ID:2  Position X: 22963, Y: 6228
00:14.384  ID:2  Sensor status: 1
00:14.962  ID:3  Timer10s: timer fired
00:14.967  ID:3  Radio: sanding INFO packet to node 4, attempt: 1
00:14.971  ID:3  Packet sent
00:14.971  ID:3  INFO ack received
00:14.972  ID:4  Message received from node 3 with type = 2
00:14.973  ID:4  INFO message received
00:14.975  ID:4  Position X: 15200, Y: 10234
00:14.976  ID:4  Sensor status: 3
```

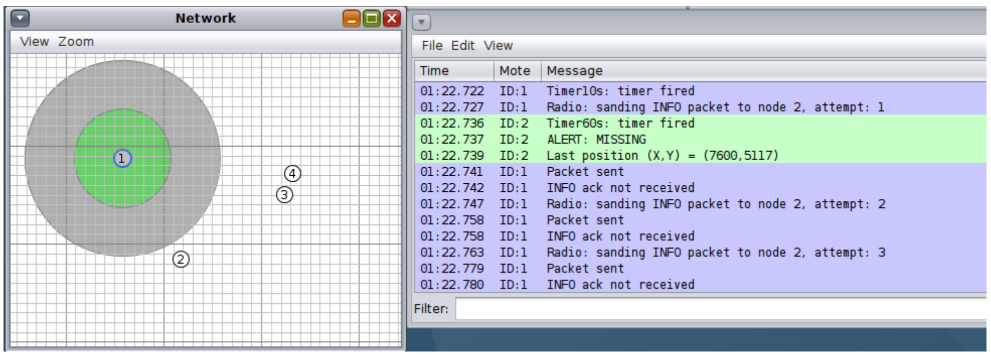
The picture above reports a standard behaviour of the bracelets: we have 2 pairs of bracelets, and the pairing has been successful since messages sent by node 1 are received only by node 2 and messages sent by node 3 are received only by node 4. Info on position and status of the children are regularly exchanged.

“Sensor status” codes: 1 = Standing, 2 = Walking, 3 = Running, 4 = Falling



```
Time       Mote  Message
00:24.128  ID:1  Timer10s: timer fired
00:24.133  ID:1  Radio: sanding INFO packet to node 2, attempt: 1
00:24.136  ID:1  Packet sent
00:24.137  ID:1  INFO ack received
00:24.138  ID:2  Message received from node 1 with type = 2
00:24.139  ID:2  INFO message received
00:24.141  ID:2  Position X: 7600, Y: 5117
00:24.142  ID:2  Sensor status: 4
00:24.142  ID:2  ALERT: FALLING!
00:24.144  ID:2  Position: (X,Y) = (7600,5117)
```

In the picture above you can see what happens when the parent bracelet receives an info message containing “Sensor status: 4” (falling): a **fall alarm (“ALERT: FALLING!”)** is showed together with the position (X,Y) of the child.



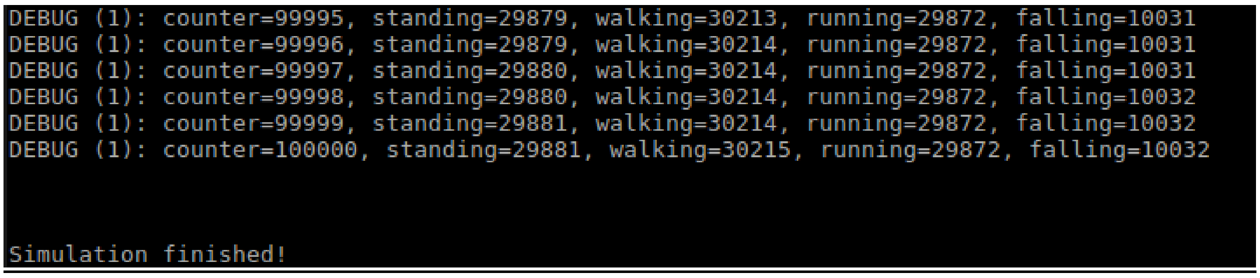
```
Time       Mote  Message
01:22.722  ID:1  Timer10s: timer fired
01:22.727  ID:1  Radio: sanding INFO packet to node 2, attempt: 1
01:22.736  ID:2  Timer60s: timer fired
01:22.737  ID:2  ALERT: MISSING
01:22.739  ID:2  Last position (X,Y) = (7600,5117)
01:22.741  ID:1  Packet sent
01:22.742  ID:1  INFO ack not received
01:22.747  ID:1  Radio: sanding INFO packet to node 2, attempt: 2
01:22.758  ID:1  Packet sent
01:22.758  ID:1  INFO ack not received
01:22.763  ID:1  Radio: sanding INFO packet to node 2, attempt: 3
01:22.779  ID:1  Packet sent
01:22.780  ID:1  INFO ack not received
```

Finally, in this last picture, you can see what happens when a child is out of the range of the parent for more than 60s: after one minute the parent bracelet shows a missing alarm **(“ALERT: MISSING!”)** together with the last known position (X,Y) of the child.

As for the probability distribution of the kinematic status, we want a 30% of standing, a 30% of running, a 30% of walking and a 10% of falling. To do that we generate a random number between 0 and 9 using Random.rand16() % 10. If this number is:

- 0 ,1 or 2 -> msg_status = STANDING
- 3,4 or 5 -> msg_status = WALKING
- 6,7 or 8 -> msg_status = RUNNING
- 9 -> msg_status = FALLING

We have writ a debug script to test the generation of the kinematic status, after 100000 generations we can see that the percentages are correct.



```
DEBUG (1): counter=99995, standing=29879, walking=30213, running=29872, falling=10031
DEBUG (1): counter=99996, standing=29879, walking=30214, running=29872, falling=10031
DEBUG (1): counter=99997, standing=29880, walking=30214, running=29872, falling=10031
DEBUG (1): counter=99998, standing=29880, walking=30214, running=29872, falling=10032
DEBUG (1): counter=99999, standing=29881, walking=30214, running=29872, falling=10032
DEBUG (1): counter=100000, standing=29881, walking=30215, running=29872, falling=10032



Simulation finished!
```