Effort: 8h

**Sensortag development**

***Function activation and deactivation****:* we were able to deactivate the unused functions in the “proper” way, by adding the relevant pre-processor options: EXCLUDE\_OAD, EXCLUDE\_REG, EXCLUDE\_OPT, EXCLUDE\_BAR, EXCLUDE\_HUM, EXCLUDE\_TMP. In the same way we also activated the buzzer, with the option Board\_BUZZER.

***Movement sensor configuration***: first, using the appropriate bitmask in the *mpuconfig* variable, we were able to configure the movement sensor. In particular we set the sensitivity of the accelerometer to +/- 4G and we turned off the magnetometer.

*mpuconfig = 0x1BF*

Additionally, we increased the update rate of the whole sensor to 20Hz (instead of the standard 1Hz).

***I/O service:*** to be able to control the buzzer (and LEDs) remotely via Bluetooth, we modified the I/O service configuration.

*ioMode = 1*

At the moment, this concludes the development on the Sensortags.

**Dead Sensortags**

We did some investigation on why the Sensortags often stopped working after flashing. We found out that, on some computers, flashing by connecting the Debugger module via USB 3.0 would “kill” the Sensortag (it would then only work while powered through the Debugger itself, but not on battery alone). To avoid this, it was enough to flash via a USB 2.0 port instead of 3.0. However, to bring back a “killed” Sensortag it is first necessary to flash the hex file using the Flash Programmer 2 (again, only via USB 2.0).

**Git Lecture**

To bring up to speed the team members with less development experience, we organized an introductory lecture on the use of *git* (via command line).

**Desktop application development**

We completed a first version of the *Alarm Module* and of the *Configuration Module.*