



Aalborg Universitet

AALBORG UNIVERSITY
DENMARK

Introducing Contextual Information to Mobile Phones by External and Embedded Sensors

Grauballe, Anders; Perrucci, Gian Paolo; Fitzek, Frank Hanns Paul

Publication date:
2008

Document Version
Publisher's PDF, also known as Version of record

[Link to publication from Aalborg University](#)

Citation for published version (APA):

Grauballe, A., Perrucci, G. P., & Fitzek, F. (2008). Introducing Contextual Information to Mobile Phones by External and Embedded Sensors. Paper presented at International Workshop on Mobile Device and Urban Sensing - MODUS08, Saint Louis, Missouri, United States.

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- ? Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- ? You may not further distribute the material or use it for any profit-making activity or commercial gain
- ? You may freely distribute the URL identifying the publication in the public portal ?

Take down policy

If you believe that this document breaches copyright please contact us at vbn@aub.aau.dk providing details, and we will remove access to the work immediately and investigate your claim.

Introducing Contextual Information to Mobile Phones by External and Embedded Sensors

Anders Grauballe and Gian Paolo Perrucci and Frank H.P. Fitzek
Department of Electronic Systems
Aalborg University - Denmark
email: {agraubal|gpp|ff}@es.aau.dk

Abstract—This paper introduces the opensensor platform. The idea is to build a wireless sensor platform that is inspired by the idea of open source. The material to build, to program and to teach the opensensor is freely available. The opensensor platform can be used for all kind of wireless sensor network research, but in this paper the focus is on contextual information retrieval for the mobile phone. After introducing and motivating the opensensor platform, some first examples and projects for contextual information retrieval are introduced.

I. GENERAL INTRODUCTION

In a very short period of time mobile phones have penetrated our daily life after their first release in the late 80s. Mobile phones have changed our life. The very first service of mobile phones was the combination of mobility and voice services. Throughout the last decades further services, in addition to voice services, have been added. Nowadays, mobile phones host a large number of services for the user such as Internet access, digital camera, mp3 player and many more. Despite the fact that there are many services in the mobile phone already now and more on the way, mobile phones have nearly no knowledge about the user or its surrounding. Such knowledge, also referred to as cognition or contextual information, however would help to create new services or extend existing ones. One example for the extension of existing services could be that the mobile phone recognizes the location e.g. whether the user is in the bathroom or within an airplane. In such situations, the mobile phone should not accept any phone calls or switch off entirely, respectively.

Parallel with the evolution of the mobile phone, wireless sensor networks (WSN) have attracted a lot of attention. First research works were mainly focus on military applications, but wireless sensors have shown their applicability to civil environments too. A wireless sensor is composed out of a battery, a wireless communication facility and the sensor part, representing parameters such as light, temperature, or distance in electronic values (mostly voltage). There are also some passive sensors such as RFID and UHF devices that do not need a battery and are empowered by the electromagnetic waves from the requesting device. But in this paper we will focus on those sensors with batteries. Those wireless sensors have great capabilities to retrieve contextual information.

The combination of mobile phones with wireless sensors is quite obvious and straight forward. Today there exist two main ways of this combination, namely embedded and external sensors. Embedded sensors are part of the mobile phone and can be accessed through the internal APIs of the mobile phone. Apple is using an accelerometer on the iPhone to retrieve information whether the user is operating the device in portrait or landscape mode. Such accelerometers can be found in many more devices already and used for different kind of applications. Paul Coulton [1] has launched some master pieces using the accelerometer for mobile games on a Nokia N95 device. Before the iPhone was even launched, Nokia came up with their model 5500. This device was intended for joggers counting the food steps. Besides those first applications of embedded sensors, new sensors are on their way. Note, some other sensors are already on the devices but not recognized by everybody as sensors such as the microphone, the camera, the GPS module. The use of the GPS module as a sensor was shown by Christian Søndergaard Jensen [2]. By combing the GPS location of several taxis in the city of Aalborg, Denmark, the traffic flow within the city was derived.

Besides the embedded sensors, external sensors are also used. Apple and Nike offer an external sensor to put into the shoe conveying the number of steps to the iPod. Other sensors are capable to retrieve heart beats or blood pressure and convey that information to a dedicated place such as the watch or the mobile device. It is expected that the variety of external sensors is much larger than those of the embedded ones. Especially the external sensor will lead to new applications in the area of health care, intelligent housing and others. The advantage of omnipresent sensors is that whole measurement campaigns can be carried out, which are more meaningful than instant measurements points. An example is health parameters such as blood pressure or insulin levels.

In the future more external sensors will be on the market such that the mobile device will be surrounded by many sensors with different tasks. Furthermore the architecture of external sensors may change. Besides pure forwarding of the sensed data, a new generation of sensors will come, preprocessing the sensor data before conveying it to the

mobile phone. The combination of external sensors and the mobile phone is a logical step as the mobile phone has sophisticated display functionality for human machine interaction. Furthermore the mobile phone is one dedicated sink where the user expects the information to be collected as given in Figure 1. The figure shows six sensors of two classes. Class one (A to C) may be used for temperature measurements, while Class two (D-F) may be sensing light values. The result of the measurement is made visible on the mobile device. Despite the displaying function, the mobile device (MD) is responsible for data fusion.

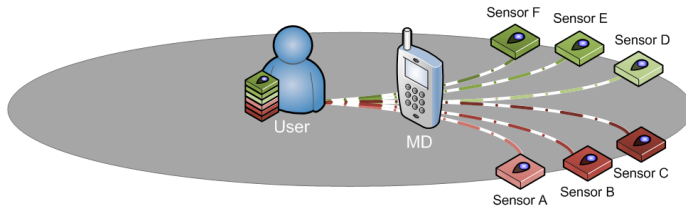


Fig. 1. Two different classes of external sensors conveying information to the user through the mobile phone.

However, embedded or external sensors, both will provide the mobile device with contextual information of and for the user. In the following we will introduce the opensor as one possible collector for such contextual information. By means of several project examples the idea of contextual information retrieval is presented.

II. OPENSOR PLATFORM

In the following we will introduce our opensor platform. We will advocate the need for such a platform and introduce the individual open parts.

A. Motivation for opensor

There are already some sensor platforms on the market. Nevertheless, in collaboration with the Technical University of Berlin, the mobile device group of Aalborg University (AAU) [3] has decided to design their own platform. The main reason was to have as much flexibility on the platform as possible. Flexibility refers to the number of connectable sensor parts as well as the wireless part. As the opensor should be used for ongoing and future research project as well as for teaching purposes, the design of the board was kept as simple as possible. Throughout the evolution of the opensor, we decided to skip the surface-mount devices (SMD) technology and base the board on standard components so that everybody can solder the opensor without any special equipment. Based on standard components, the opensor board is coming at a very low cost, which should enable a lot of researchers to implement their ideas in the field of wireless sensor networking on a real platform. As the mobile device group of AAU has a huge interest in the convergence of wireless sensors with mobile phones, the opensor offers the possibility to communicate with commercial mobile phone.

As most 3rd party solutions for wireless networking are closed solutions, we decided to make the whole opensor concept freely available for everybody [4]. Not just open source, but also open hardware and open teaching material. We hope to inspire other researchers to use our platform and eventually contribute to the project with more software or teaching material.

B. opensor hardware

The opensor platform is given in Figure 2. It is powered up with a 9V battery block. It has several interfaces to connect to the outer world. Two LEDs are on the board. The red LED is showing the operational phase of the opensor and the green LED can be controlled from the DSP and is used for monitoring purposes. The main parts of the opensor are introduced in the following.

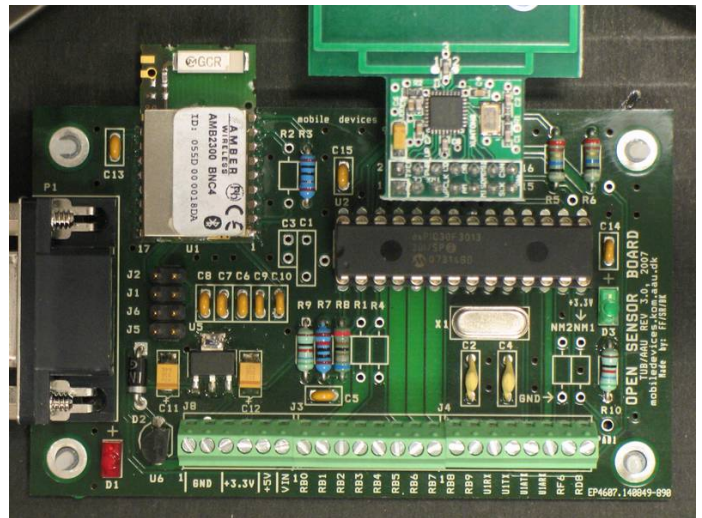


Fig. 2. A picture of the opensor board version 3.0.

1) *DSP*: The core of the opensor is a 16 bit architecture dsPIC30f3013 processor produced by Microchip. It has low power consumption, features 24KB of program memory, 2.048B of RAM and has 28 pins of where 20 can be used as in/output pins. For communication purposes the chip has two UARTS and one SPI port. It is capable of performing Analog-to Digital conversion in 12 bit, 200Ksps for measurement purposes, has three timers and is able to provide pulse-width modulation (PWM) which can be utilized for motor control.

2) *Communication and Programming Interfaces*: The opensor board has up to five interfaces that allow communicating and programming the sensor. While the connection bar, the RS232 and the PICKITPINS are part of the board, the wireless communication interfaces nRF905 and Bluetooth are optional.

a) *RS232*: A RS232 interface can be used to connect to the PC. Prior opensor boards used mini USB for data exchange but also for powering the board while programming on it. The choice for an RS232 was based on the costs related to the mini USB port and its needed additional hardware.

b) *nRF905*: The nRF905 is a radio interface without any link layer or medium access protocol. A basic CSMA/CA protocol is available on the opensensor web page [4]. But the medium access scheme can be changed according to the needs of the project or just for educational purposes. Each packet is broadcasted on the air and can be received by all other nodes. The addressing of the packets is done by the framing as given in [5].

c) *Bluetooth*: As most commercial phones support Bluetooth, the opensensor has the option to add a Bluetooth chip. The current version 3.0 of the opensensor hosts a chip from Amber Wireless. As the Bluetooth chip itself costs more than the whole opensensor platform, it should be evaluated beforehand if the Bluetooth is needed on all sensor boards. In many of our examples, most sensor boards communicate over the nRF905 interface with each other and one dedicated sensor, equipped with Bluetooth connectivity, conveys the results to the mobile phone.

d) *Programmer Pin*: The opensensor can be programmed through the programmer pin where the PICKit2 (see Figure 3) can be connected. The PICKit2 programmer must be connected to the USB interface of the PC and is able to operate in both Windows and Linux environments. Note that the PICKit2



Fig. 3. A picture of the PICKit2 programmer that allows to program the opensensor.

programmer should only be used without the nRF905 module. The design of the board is done in a way that it should not be possible to place both without violating the board.

e) *Connection Bar*: The basic version of the opensensor does not host any real sensor part. But any sensor can be attached to the opensensor easily. Here we explain shortly how to connect a light sensor. The light sensor must be mounted to one of the input pins. It is then possible to use the Analog-to-Digital converter in the opensensor to measure the level of light. The light sensor can be replaced with any other sensor such as for temperature or distance. The software to retrieve the light values can be found on the web page [4].

C. opensensor Software

1) *Programming Environment*: In the following we explain shortly how to use the MPLAB Integrated Development Environment (IDE) for programming the opensensor. Most of the examples on our web page use Windows as operating

system for the PC. Nevertheless in [4] the Linux support for the opensensor project is described as well.

a) *MPLAB*: MPLAB is the programming environment used to develop programs for the opensensor. Both C and Assembler code can be used according to the programmer's skills. It is possible to setup projects and build them using the GUI interface as shown in Figure 4.

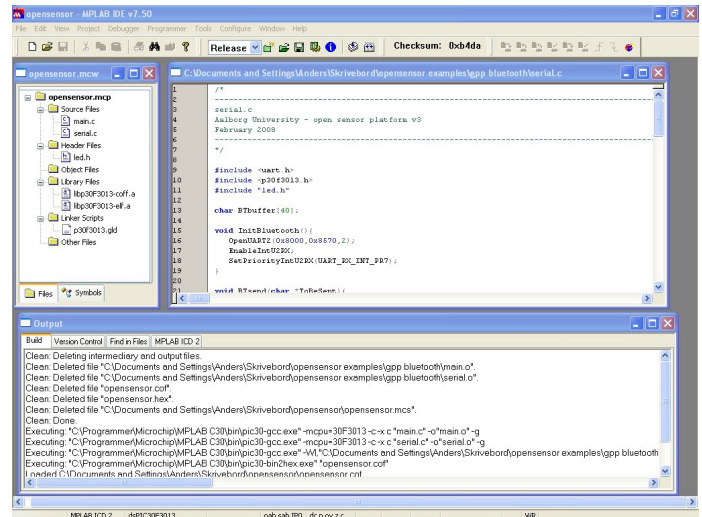


Fig. 4. A screenshot of MPLAB IDE.

b) *Console*: The opensensor can be accessed through the RS232/Bluetooth interface via a terminal program. This makes debugging on the opensensor easier, because simple text commands can test various programmed functions. An example of the console can be seen in Figure 5.

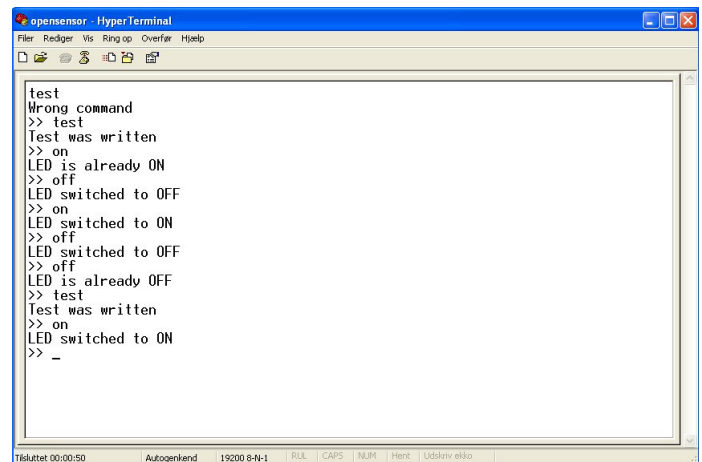


Fig. 5. A screenshot of the Windows Hyper Terminal that allows to send and receiving data to the opensensor through the RS232 or Bluetooth interface for debugging or testing purposes.

2) *Code example*: In Listing 1 a code example for the opensensor is shown. The code is written in C and allows the opensensor to switch on the green LED on the board.


```

#include <p30f3013.h>
#define green_led LATDbits.LATD9
#define green_led_TRIS TRISDbits.TRISD9

_FOSC(CSW_FSCM_OFF&HS);
_FWDT(WDT_OFF);
_FBORPOR(PBOR_OFF&MCLR_EN);
_FGS(CODE_PROT_OFF);

int main( void ){
    green_led_TRIS = 0;
    green_led = 1;
}

```

Listing 1. Code to switch on the green LED on the opensensor board.

D. Mobile phone software

The opensensor platform allows commercial mobile phones to interact with it using Bluetooth. This communication is based on the RFCOMM protocol which is a simple set of transport protocols, made on top of the L2CAP, providing an RS-232 serial port emulation. There are two approaches to describe the interaction between the opensensor and the mobile phone: the *gateway approach* and the *retrieval approach*. In the first one, the opensensor uses the mobile phone as a gateway to send information to the outer world or directly to the user. For example, if a temperature sensor is mounted on the board, the opensensor can send some feedback to the phone when the temperature reaches a certain value. For the *gateway approach* we have created a simple protocol for communication, based on three tags:

- ACTION {DISPLAY, SPEECH, SMS, MMS}
- BODY {any}
- KEYVALUE {any}

When the opensensor needs to communicate with the phone, it sends a message formatted according to this protocol. When the phone receives the message, depending on the ACTION tag, it can perform different actions: display an alert on the screen, send an SMS or an MMS, or speak out the body of the message using the text2speech (functionality available on S60 phones.) An example of a typical message from the opensensor to the phone can be:

```

<ACTION>SMS
<BODY>Warning!!!Temperature 41 C
<KEYVALUE>+123456789

```

On the phone side, an application is running in the background and waiting for information from the opensensor. In Figure 6 some screenshots of a testbed application we have developed are shown.

In the *retrieval approach*, the phone is using the capability of the sensors mounted on the board to retrieve information about the surroundings. The phone needs to run an application able to send some commands to the opensensor via Bluetooth when it needs to get the information. To program applications on mobile phones several programming languages are available. The choice highly depends on the platform, as described in [6]. In the Listing 2 a simple script written in Python for S60 [7], one possible choice for programming on Symbian



(a) The opensensor sends to the phone an SMS action command when the temperature exceeds a certain value. (b) The phone receives the SMS command and sends an SMS to a specified number.

Fig. 6. Example of an application written in Python for S60 that allows the opensensor to use the mobile phone as a gateway to send SMS.

OS platform, is shown. The script allows to open a Bluetooth socket communication with the opensensor and send data to it. This code example can be used to send ON/OFF commands to the opensensor, for example for switching the green LED on and off.

```

import appuifw,e32
from socket import *
BT_addr='00:18:da:00:2b:dc'
class BT:
    def connect(self):
        self.sock=socket(AF_BT,SOCK_STREAM)
        device=bt_discover(BT_addr)
        print "Connecting to opensensor ..."
        self.sock.connect((BT_addr,device[1].values()[0]))
        print "CONNECTED"
    def close(self):
        self.sock.close()
    def transmit(self,data):
        self.sock.send(data)
        print "sent command: "+data
def sendON():
    bt.transmit("ON")
def sendOFF():
    bt.transmit("OFF")
def quit():
    app_lock.signal()
    bt.close()
appuifw.app.exit_key_handler=quit
appuifw.app.title=u"opensensor"
bt=BT()
bt.connect()
appuifw.app.menu=[(u"ON",sendON),(u"OFF",sendOFF)]
app_lock=e32.Ao_lock()
app_lock.wait()

```

Listing 2. Code example of an application written in Python for S60 to send on/off commands to the opensensor.

To guarantee more portability among different mobile phone platforms, Java Mobile can be used for applications development. On the webpage [4] more code snippets for communication between the opensensor and mobile phones supporting Java Mobile can be found.

E. opensensor educational support

Besides the hardware and software also basic teaching material is freely available. The teaching material contains

basic information about the opensensor and how to program on it. The material is available in PowerPoint and pdf format.

III. OPENSSENSOR PROJECTS

In this section we describe shortly some of the projects carried out at the Aalborg University or the Technical University of Berlin.

A. Parksensor

The main idea is to move high priced functionality from the car to the mobile phone and spice those services even up. One example is the presented park sensor idea. A car is equipped with ultra sonic distance sensors which are connected wirelessly with the mobile phone. The mobile phone is placed within the car showing the distances to the nearest objects. Besides the display functionality, the mobile phone can support the driver with audio information (we are using text2speech functionalities of S60 phones). As the service is bundled to the phone it makes no sense to steal anything from the car. The parksensor is described in more detail in [8].



Fig. 7. A picture taken during the tests of the parksensor application. On the screen of the phone, the colored bars indicate the distance measurements coming from the four sensors placed on the car.

B. Mesh Sensor Networks

The purpose of this project was to design and implement a medium access control (MAC) protocol for the ISM module of the opensensor platform, allowing direct communication among the opensensors and proving a solution to overcome possible collisions between packets during transmission. A collision avoidance scheme with acknowledgments and carrier sensing has been designed and implemented to minimize data loss and delay, with a minimal number of retransmissions. To test the reliability of this wireless network and as an application proposal, two mobile phones are connected over a meshed network formed by opensensors. The opensensor used the nRF905 device to communicate among each other, while

the mobile phones used Bluetooth to communicate with the opensensors. In the end it was possible to exchange information among the mobile phones over the meshed architecture.

C. Distributed Storage

Some applications introduce only partially connected sensor networks leading to lower reliability or availability of the measured data. The objective of this project is to develop and analyze a data distribution method which can increase the system reliability and keep the memory consumption low on each device. A Reed-Solomon coding scheme is applied as a solution where a certain number of unavailable devices can be tolerated without jeopardizing the system reliability. A probabilistic system model is derived to describe the distribution and reconstruction of a message from a sensor to the gateway. This model is verified and visualized by means of a simulation implemented in Java. A prototype of a system containing two data devices and one redundant device is implemented to test the model in a real life scenario. Performance evaluation shows that the probability to receive all sensor measurements by using the proposed cooperative data distribution method is higher compared with the non-cooperative case.

IV. CONCLUSION

This paper has introduced the opensensor platform, which follows the concept of open source. Besides the software, also the hardware design and the educational means are freely available. The opensensor can be used for pure wireless sensor network concepts or, as used by the others, for the convergence of wireless sensor networks with mobile phones. Throughout the text, we have motivated the convergence of mobile phones and wireless sensor networks to create better and more innovative services for the user. This convergence will take place by embedded and external wireless sensors and mobile phones.

V. ACKNOWLEDGEMENT

We would like to thank our students of AAU and TUB that have contributed to the opensensor platform in their student projects or in the lectures.

REFERENCES

- [1] *Mobile Phone Programming and its Application to Wireless Networking*. Springer, 2007, ch. Using In-built RFID/NFC, Cameras, and 3D Accelerometers as Mobile Phone Sensors.
- [2] C. S. Jensen and D. Tiesyte, "Gps data management with applications in collective transport," <http://www.cs.aau.dk/TransDB/papers/its.pdf>.
- [3] mobiledevices.kom.aau.dk.
- [4] mobiledevices.kom.aau.dk/opensensor.
- [5] S. Rein, C. Ghmann, and F. Fitzek, *Mobile Phone Programming and its Application to Wireless Networking*. Springer, 2007, ch. Sensor Networks for Distributed Computing.
- [6] F. Fitzek and F. Reichert, Eds., *Mobile Phone Programming and its Application to Wireless Networking*. Springer, 2007.
- [7] J. Scheible and V. Tuulos, *Mobile Python: Rapid Prototyping of Applications on the Mobile Platform*. Wiley, ISBN: 978-0-470-51505-1, 2007.
- [8] J. Rasmussen, P. stergaard, J. Jensen, A. Grauballe, G. Perrucci, B. Kryer, and F. Fitzek, *Mobile Phone Programming and its Application to Wireless Networking*. Springer, 2007, ch. Parking Assistant Application.