"Starter kit for smart buildings"

Pervasive Computing : Selected Topics Project

Aebischer Nadia Department of Informatics
University of Fribourg
1700 Fribourg, Switzerland
nadia.aebischer@unifr.ch

Luyet Gil Department of Informatics
University of Fribourg
1700 Fribourg, Switzerland
gil.luyet@unifr.ch

ABSTRACT
The goal of this report is to describe an easy way to integrate a basic pervasive system into an already built house/building. The system should be cheap and easy enough to integrate and to operate so it could be used by the masses as a starter kit for smart buildings. The report will be based on the data analysis and the context of use of this system. Especially about the central unit managing the whole house.

1. INTRODUCTION
1.1 Smart building description
A smart building is by definition a building enhanced by the technology that is managing it. Generally the technology used is splitted into four parts. The sensors that collect data and allow to have an understanding on what is going on outside or within the house. The second part is the network that allows to spread the data collected by the sensors and to eventually activate the effectors if needed. The third part is the managing unit that allows to gather data and infer knowledge about the environment and thus either show the data to the user or activate effectors to modify the user's environment. The last part is the data visualisation that allows the user to interact with the data. This structure follows the given standard loop of a pervasive system 1.

1.2 Aim of the project
The basic aim of the project is simple. We would like to produce a simple to use and to integrate system that would allows to enhance an already built building to a smart building. This system should be as cheap as possible and should physically alter as less as possible the house. This means it has to be based on cheap and already existing means and be a sort of "do-it-yourself" pack.

2. HOUSE DESCRIPTION
2.1 Context of use
We are going to consider a small building with a dozen of individual apartments. Each apartment would have 4 rooms
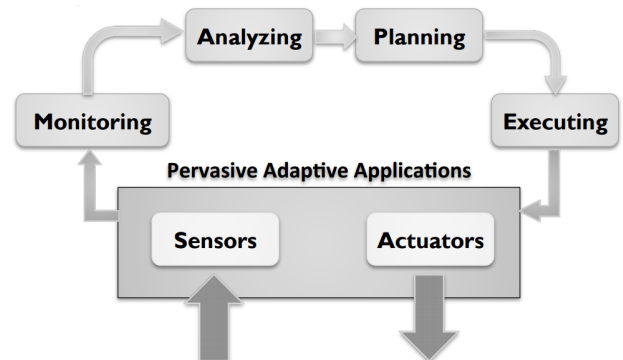

Figure 1: Pervasive system loop

each having 1 light and 1 windows with an electric sunblind. In such a configuration it is usual to split the power supply into so call groups. Each group has his own electrical phase. For a standard 4 room apartment (1 kitchen, 3 rooms and 1 bathroom) we would have 3 or 4 group. One managing the kitchen, one managing the bathroom and 1 or 2 for the other rooms. The building has a central water heater, separated electric meters per apartment and an individual thermostat per apartment. A central electrical panel is available for the whole house. An indoor parking space in common with an electric door that is activated by a remote controller is also considered.

2.2 What could be automated
Sunblinds are opening or closing automatically by checking the sunlight and time. For example if it's sunny the sunblinds will be closing. The same way, at night, they should be closing if we don't want neighbour to take a look. The temperature of the apartment can also be automated. For example we can set the temperature we want to have in the apartment, and if the this temperature is below or above the default one, the system will change the thermostat accordingly. We can also imagine a way to infer if somebody is going to be there soon or not and raise more or less fast the overall temperature. This inference can be computed according to the presence of people in the house by reading the presence sensors. For example if somebody is never present during the weekends then the central unit should over the time learn that fact. The lights can also be automated by detecting the presence of people in a room. RFID tags linked

with cars could be used to open the parking space doors. Of course, these automated behaviors can be stopped by the user if he doesn't want it. To do so, the user can access the central unit via a web interface and change the behavior the way he want.

3. A FEW WORDS ABOUT THE SENSORS ARCHITECTURE

3.1 Architecture of the sensors

The communication between the components are achieved by wifi or over the electrical network in case the wifi is not reachable. Communication over the electrical network can achieve a very fast connection speed between the components, up to 14 Mbits/s [1]. This means that theoretically pre processing of sensors data is not needed but for utility reasons we will preprocess the data. Each sensors is connected to a light computer, typically a Raspberry Pi could work but we could imagine an event less powerful one for small tasks. This computer will preprocess the raw sensor data and play the role of the interface between the sensor and the Central Unit. This interface is needed in case of multiple types of sensors. By changing the sensors one could simply apply a patch to the small computer (even via the Central Unit) and resolve compatibility issues. This interface used here is a simple HTTP based communication via POST with a specific XML-based body very similar to the SOAP protocol. This produce a simple Service-Oriented Architecture (SOA) to access the data.

```
POST address
  <MESSAGE>
    <TYPE>(alert|event|failure|do)</TYPE>
    <FROM>(sensor_id|central)</FROM>
    <CONTENT/>
  </MESSAGE>
```

Where the TYPE denotes an alert, an event or even a failure for example. The FROM and CONTENT parts deals respectively with the author of the message and the actual meaning of the message. A typical proximity sensor message would look like the following :

```
POST central_unit
  <MESSAGE>
    <TYPE>event</TYPE>
    <FROM>sensor_id2345</FROM>
    <CONTENT>distance:'2.34m'</CONTENT/>
  </MESSAGE>
```

A typical message send by the Central Unit to open a door would be:

```
POST address_of_door_effector
  <MESSAGE>
    <TYPE>do</TYPE>
    <FROM>central</FROM>
    <CONTENT>door[0]:'open'</CONTENT/>
  </MESSAGE>
```

3.2 Location

The location of the sensors depend of the type of sensor. Here is the list of the sensors we are going to consider

Luminosity sensor - outside, at each frontage of the house.

Presence sensors - around the lights and the doors.

Temperature sensors - inside/outside.

Energy consumption sensor - after the electric meter.

Water consumption sensor - before the water inlet.

Water temperature sensor - after the water heater.

RFID readers - for cars around the garage doors.

We also consider effectors, the list of them is given there.

Sunblinds effectors - to open and close sunblinds.

Light effectors - to switch the lights on and off.

Thermostat effector - to change the house temperature.

Automatic garage door opener - to be linked with the RFID reader.

This part is not really important for the scope we choose to aim our report but this should be at least mentioned because it list which sensors we are going to consider and receive data from to form a context and also on which effectors the Central Unit can send message to modify the environment.

4. CENTRAL UNIT

The Central Unit is the main part of this architecture and the scope of this report. This Central Unit depends on several topics as its own architecture, the sensors data coming in, the effectors it has to interact with its environment and of cause its standard interface with the user.

4.1 Architecture of central unit

The architecture itself has no specific requirement. We can choose whatever we want, here we might focus on a standard Linux distribution as Ubuntu without any graphical environment as the only interaction we have should be via a web interface discussed later. Furthermore the computation needed is not too high as the preprocessing of the sensor is done on the sensor side. The computer should also be running 24/7 and thus the computer has to have a low consumption. A light computer as a Raspberry Pi might be enough.

4.2 Sensor fusion

The sensors fusion is performed to produce a context. A context is the actual state of an entity such as a room or a person. In this report we need to know exactly which kind of contexts we are going to consider to know exactly which sensor data has to be aggregated to produce the actual state of a room for example. The aggregation itself will be done in such a way that sensors failure could be minimized. A standard way to do this can be done via computing a mean value of the last dozen of incoming values and use this mean as the incoming sensors value. Here are the contexts we are going to considered.

4.2.1 Sensors message synchronization

Has some sensors output can be asynchronous, e.g. if we take the presence sensor it could output each time it detect a presence. This could be solve by using some kind of synchronization protocol similar to the WiFi communication protocol. Each sensor we have is connected to a little computer used for preprocessing of sensor inputs, this allow us to synchronize the data flow by offering time windows management. This kind of management is usually done by either give a time frame by sensor or allow one big time frame to allow each sensor to send periodically during a certain amount of time. The second option will be taken here but we might introduce a little modification. The size and the exact beginning of this time frame is not fixed but we introduce here Ready-To-Send (RTS) messages similar to Wifi collision avoidance protocol. Each time the Central Unit has finished to infer all the contexts it send a message to all the sensors a RTS message to notify them that they can send their data. Such a message would look like the following.

```
POST address_of_sensor
  <MESSAGE>
    <TYPE>event</TYPE>
    <FROM>central</FROM>
    <CONTENT>'RTS'</CONTENT/>
  </MESSAGE>
```

During the time between two RTS messages, sensors are able to aggregate their sensors value to give a mean value of them. As the sensors are now synchronized we need to infer contexts from their data.

4.2.2 Context inference

Inferring contexts depends on the states from several sensors. In this part of the report we list the contexts choosen and how to obtain them. A certain context may depend on several subcontext or states, e.g. the lightning state of a room may depend of the outsite lightning, the state of the lights of the room and the states of the sunblinds.

The lightning state of the room
This context is composed of distinct contexts, the lightning of the room and the outside environment (**"dark" or "lighted"**) and the sunblind states (**"open" or "closed"**). Of course those contexts are inferred by using multiple sensors input, e.g. multiple sunblinds implies multiple sunblinds states that will be summed. But we will not extend the description of such details. The lightning state of the room itself may be described as ("dark" or "natural lightned" or "artificially lightned").

dark if lightning of the room = "dark"

natural lightned if lightning of the room = "lighted" ∧ outside lightning = "lighted" ∧ sunblinds = "open".

artificially lightned if lightning of the room = "lighted" ∧ outside lightning = "dark".

The occupation of the room
In order to determine if the actual occupation of the

room (**"empty" or "occupied"**) we need to merge the inputs of the presence sensors of the room. Each presence sensor can determine if something moves around it.

```
POST central_unit
  <MESSAGE>
    <TYPE>event</TYPE>
    <FROM>sensor_id</FROM>
    <CONTENT>presence:('true'|'false')</CONTENT/>
  </MESSAGE>
```

Those sensors inputs are merge to produce a single state of the room. If S is the set of sensors used then the state is infered by the following equations.

occupied if $\bigvee\limits_{s \in S} s_{state} = True$

empty if $\bigvee\limits_{s \in S} s_{state} = False$

The actual weather state inside and outside
(temperature, lightning and precipitations). This context depends on many different sensors input and merging. The first subcontext to determine how the temperature state outside and for each room is as (**"cold", "comfortable" or "hot"**). Those terms are based on a user-determined threshold. Each subcontext can be infered by using data provided on the Internet or by using outside temperature sensors. If S is the set of sensors used outside or inside then the temperature is determined by the following equation.s

hot|comfort|cold if $\frac{\sum\limits_{s \in S} s_{value}}{|S|} \in [min_{value}, max_{value}]$

The precipitations state (**"rainy" or "dry"**) be infered by using values retrived over the Internet.

Which is the part of the day ?
The actual date and time is enough to infer such an information. The part of the day can be **"morning", "noon", "afternoon", "evening" or "night"**

Is a car waiting at the garage doors ?
If an authorized car approches the RFID reader then the door can be opened. The message send by the RFID reader have the following form.

```
POST central_unit
<MESSAGE>
  <TYPE>event</TYPE>
  <FROM>RFID_reader</FROM>
  <CONTENT>
    presence:('true'|'false'),
    car:('allowed','forbidden')
  </CONTENT/>
</MESSAGE>
```

If a car with a RFID tag approches the RFID reader then the presence field turn to "true". If the tag is authorized then the car field turn to "allowed".

What is the actual consumption of the whole building ?

May be used to display to the webinterface.

[INSERT HERE WHY THOSE CONTEXTS]
As this list is now fully established we have to on which
sensors to use to analyse the context wanted.

Admit now that each context has been computer via the
sensor data. We must now describe how an actual context
combined with new sensors data coming in will react. This
can be done via listening all the possibilities.

Do activate lightning **if** somebody enters the room $\wedge$ the
room is dark.

Do deactivate lightning **if** the room becomes empty $\wedge$ the
room is lightened.

Do reduce heating **if** room is usually empty for a quite long
period $\wedge$ nobody is inside.

Do reduce heating **if** outside temperature $>$ comfort tem-
perature $\pm$ threshold

Do augment heating **if** temperature $<$ comfort temperature
$\pm$ threshold.

Do open doors **if** car is at the parking doors $\wedge$ door closed.

Do close the doors **if** timer for doors is over $\wedge$ no car at the
parking doors.

Do close sunblinds **if** lightning outside $<$ threshold $\wedge$ part
of the day = 'evening $\vee$ night'.

Do open sunblinds **if** sunblinds closed $\wedge$ part of the day !=
'evening $\vee$ night' $\wedge$ someone enters the room.

4.3 Effectors
Automation of control. Inference for temperature calcula-
tion by knowing is somebody usually will come to the house.

4.4 Web-based manual controller
for mobile + desktop + over Internet stats presentations fail-
ure presentations notifications (SMS, e-mail, Facebook, in-
vasive smartphone application) what must be stopped what
must be switched on/off automatically crash of computer (?)
: keep some switches ?

5. CONCLUSION
6. ACKNOWLEDGMENTS