Measuring wireless energy consumption on PDAs and on laptops

Pierre Devevey, Nicolas Lorenzon, Chaibou Tambary
Universitè del la Franche Comtè
Work developed while at DISI, Università di Genova

Abstract

Many authors in the literature develop new algorithms and implement new tools by claiming that these are useful in order to save energy. This report presents actual measures taken in order to evaluate wireless energy consumption on pda and on laptop, and briefly describes existing and newly developed tools to take such measures.

1. Introduction

We describe hereafter a set of tools which we implemented and tested at DISI, Università di Genova as part of our work for Master Thesis.
The laboratory we have been using, SEALab, has many desktops and laptops, and some PDAs. We show in Figure 1 a possible configuration of the Lab.
Our interest is in measuring energy consumption with wireless networks. Our work was mostly in the direction of evaluating multimedia streaming on infrastructured networks, but the portions described after are relevant also for other projects, like IS-MANET, where wireless networking is used as well.

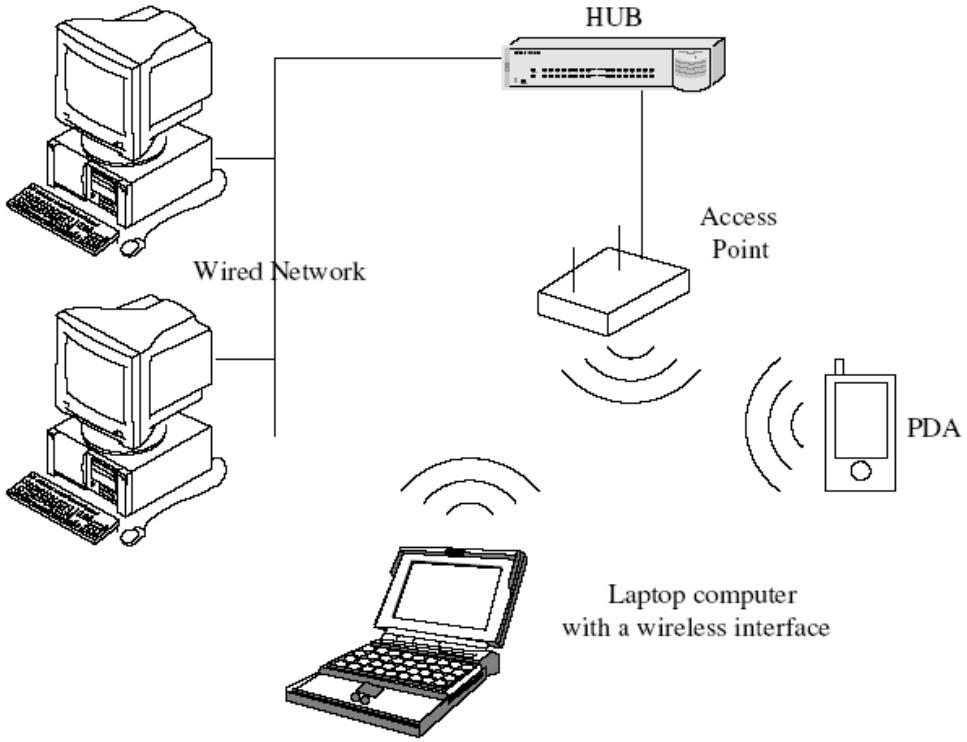

Figure 1. A configuration in SEALab with laptops and PDAs connected in a wireless network.

2. Measurement tools

To take the measures of the energy consumption, we have developed some applications. The first application, monitoring system and network tools, is a program which can take some informations about the network and the system.
The other program, UDP packets sender, is a tool to make tests on the wifi consumption.

2.1 Monitoring system and network tools

Presentation

For the different tests, we must take measures on the network; number of IP, TCP, UDP packets, number of bytes sent and received, the cpu load and the battery load. We have developed an application, to read these values on Linux.
This application allows to record the measures on a file and makes a graphic (Figure 2). It is composed by two parts: a launcher and a main program. The main program can be launched by a command with options.

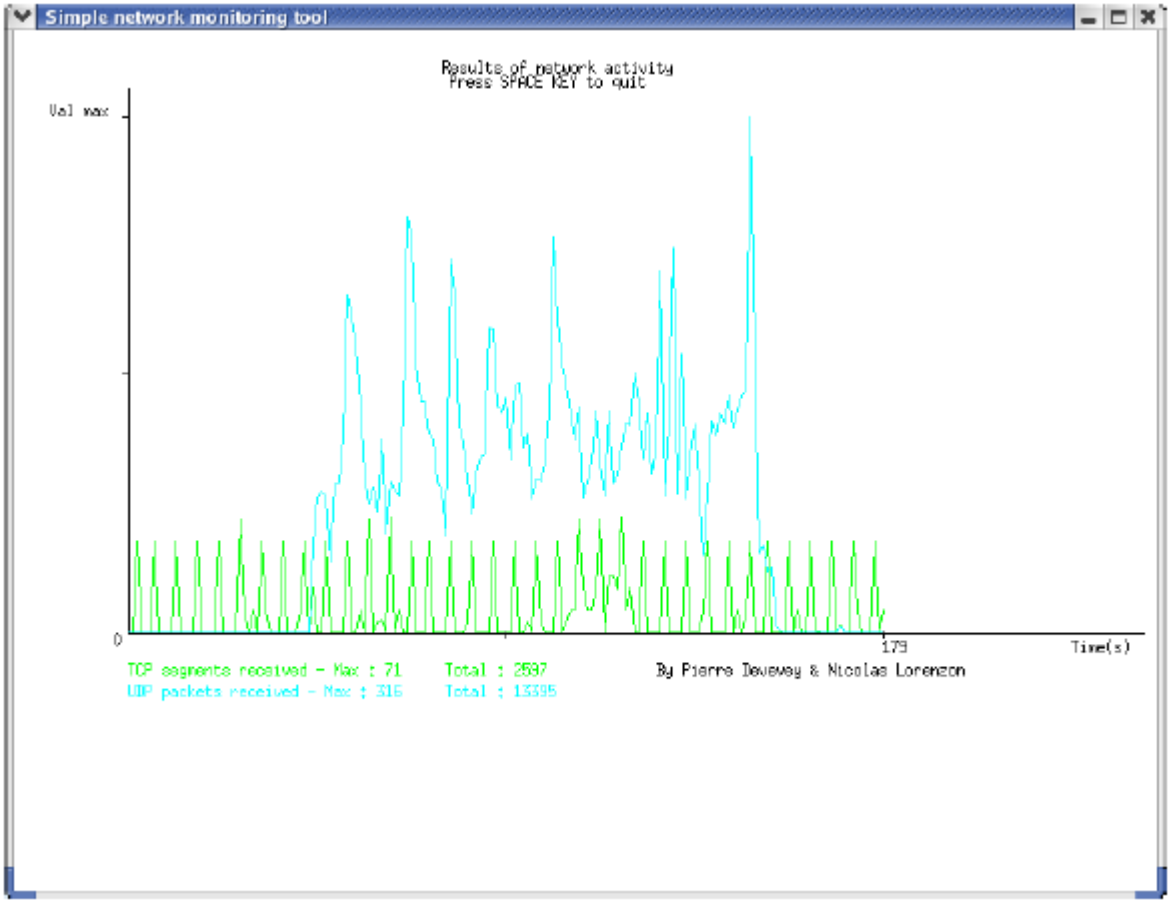

Figure 2. An example of output from the monitoring tool

Another program, more adapted to the PDA on Linux, was developed to take the battery level.

The conception

To take the different measures, we use the ”special file system” in the `/proc` directories. We get the values back to different files like `/proc/net/dev` or thanks to commands like `netstat`. These values are registered on specific files like ”`rIP.dat`” for the received IP packets.
The interface is totally separated from the main program. We use the GTK library to create a command to launch the main program (Figure 3).

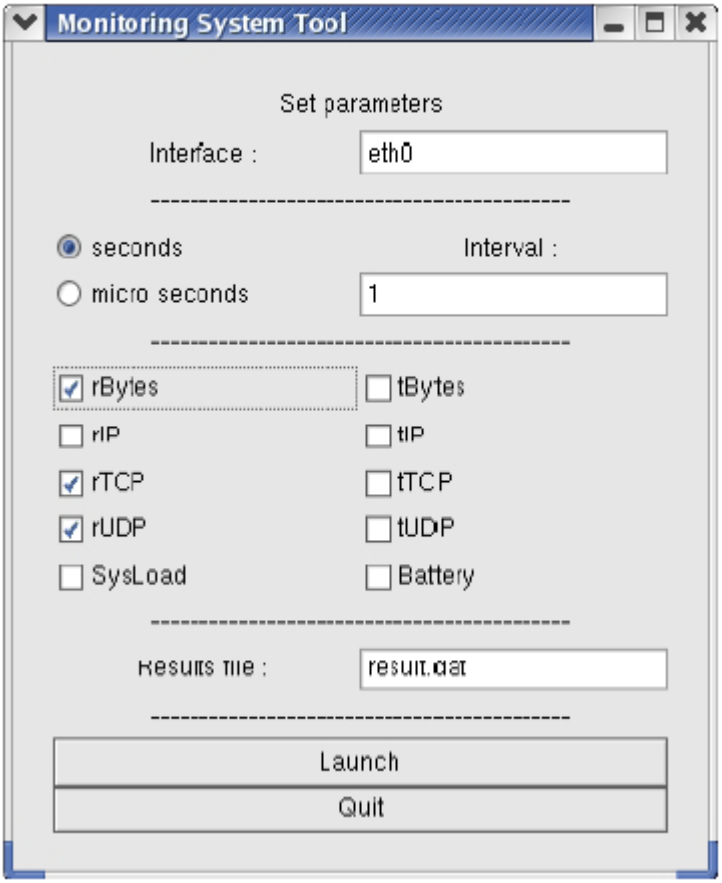

Figure 3. The interface of the measuring tool

• GTK library

GTK was initially created for the graphics program The GIMP, the GIMP toolkit abbreviated GTK+. It's is now one of the two most popular X widget toolkits for creating graphical user interfaces (the other being Qt). GTK+ and Qt have supplanted Motif, which at one time was the most widely-used X widget toolkit.

GTK+ is written in C, although it is designed within an object-oriented paradigm (unlike C++, C does not support object-orientation natively). The toolkit offers bindings to almost all popular programming languages.

The GNOME environment uses GTK+ as a base, which means that programs written for GNOME use GTK+ as their toolkit. GNOME applications are not the only programs using it, though, and any GTK+ program (or GNOME program, for that matter) can run on top of other desktop environments, such as KDE or XFce. The GPE Palmtop Environment is another environment that uses GTK+ as a base.

GTK+ initially contained some utility routines that were not strictly graphics-related, for instance providing such data structures as linked lists and binary trees. This has now been separated into a separate library, Glib, which is regularly used to develop programs that do not have a graphical interface.

After having taken the measures, the main program (graphe) creates a graphic with the measures in function of the time (shown above in Figure 2). We use the Xlib library to draw the graph.

• Xlib library

When users enter input on a terminal, an X server distributes that input to client programs that are on the same system or elsewhere in the network. The X server then returns the requested actions, such as refreshing the screen or starting an application.

The windowing interface is consistent across platforms and remains consistent because the Xlib library controls system calls.The Xlib library is a C language function library that client programs use to communicate with the windowing system. Calls to the client are sent through the Xlib library, and return information passes back through the Xlib library, which translates information to a standard X-Window language.

As a programmer, the application program you create is the client of the client-server relationship; you write the program and the X server gives it hardware independence.

2.2 UDP packets sender

It's interesting to measure the battery consumption on the PDA with only the wireless network device working: to do this, we have developed a small application in python language.

This application sends UDP packets from one point of the network to another. There is no client applications to limit the cpu load. The receiver decomposes the packets until the IP layer, as we can see in Figure 4. The use of python simplifies coding of this application.

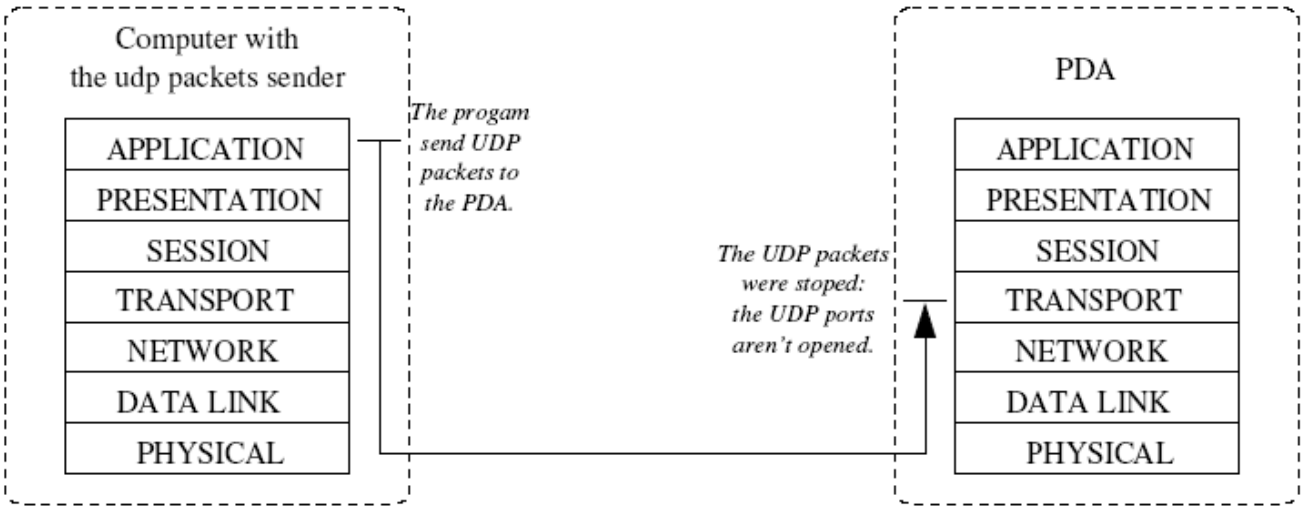

Figure 4. Functions of the UDP packets sender

3. Experiments Result on PDAs

This part explains how we have taken the measures, shows the different experiments, their results and describes a conclusion about the PDA energy consumption over a wireless network.

3.1 How to take measures

In order to take some measures on the battery consumption, there are different ways to make some tests, by software on Windows and Linux, and by voltage measures.

3.1.1 Battery monitor

For Windows Pocket CE 2002, Battery Monitor (Figure 5) allows you to monitor the battery power on PDA device. It has some features like:

• Export measured values to CSV file (can be read with MS Excel, OpenOffice Calc Spreadsheet...).
• Graphics presentation of battery capacity level.
• History of battery capacity in numerical values (date, time in active mode and capacity).

But, this shareware has some limitations:

• Every 5 minutes announcement about unregistered version.
• Automatic turn off application after 30 minutes.
• After reset data will not be refreshed.

The great problem with this shareware is the limitation after 30 minutes. Therefore, most of tests will last for 30 minutes.

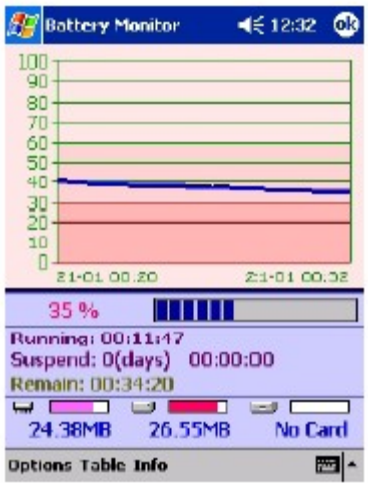

Figure 5. Screenshot of Battery Monitor

3.1.2 Familiar Linux measures

On Familiar Linux, we have developed a program based on the monitoring system and network tools introduced before. This program has no interface and graphical application, it reads only the battery level and the bytes (received and transmit) on the familiar Linux `/proc` directories.

Data is written in a special file. Contrary to the first program, this program can't read the other data like the UDP or IP packets because the familiar Linux `netstat` command is different than the other distributions. To install this program on a PDA with familiar Linux, we must crosscompile the program on a computer and after compilation, we must transfer the compiled file on the PDA with an ssh connection.

3.1.3 Voltage measures

We wish to measure the battery capacity. Voltage and capacity are linked to the power. We can see that with electronics expressions:

$$P = UI \text{ or } P = U.U/R$$

P: Power, U: Voltage, I: Current and R: Resistance.

The current and the resistance don't vary in time, therefore there is only the voltage which varies, as we can see in Figure 6. In this graphic, the voltage curve is in white, the current curve is in blue, the resistance curve is in purple, and the temperature curve is in red.

The graphic corresponds to a AA element (NiMH). We can use the voltage level to compare the battery consumption between two tests.

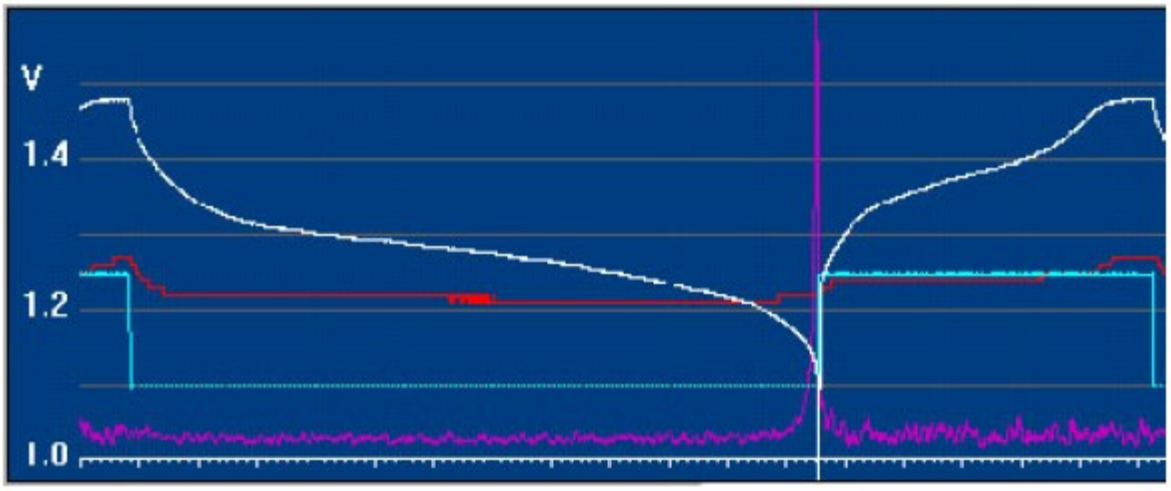

Figure 6. Voltage, current, resistance and temperature curve of battery

To take physical measures, we use a multimeter on the PDA battery between the positive part and the negative part as in Figure 7.

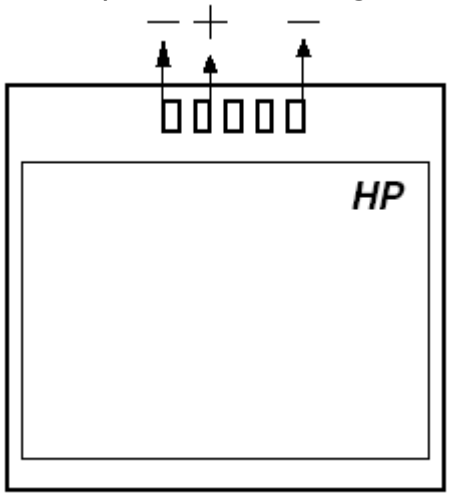

Figure 7. Positioning the multimeter

The great problem with this is the test stop: we must remove the battery. The tests last for 30 minutes so there are only two measures: At the beginning and the end of the tests.

3.2 Different experiments

3.2.1 PDA usual consumption

We first studied the PDA usual consumption to compare the different components, such as the wifi device, and to understand which consumes more. Normally the PDA uses the CPU, the wifi, and the screen (back light). The CPU consumption is weak with respect to the other components, as we can see in Figure 8. And so we concentrated more on the wifi and the screen with back-light.

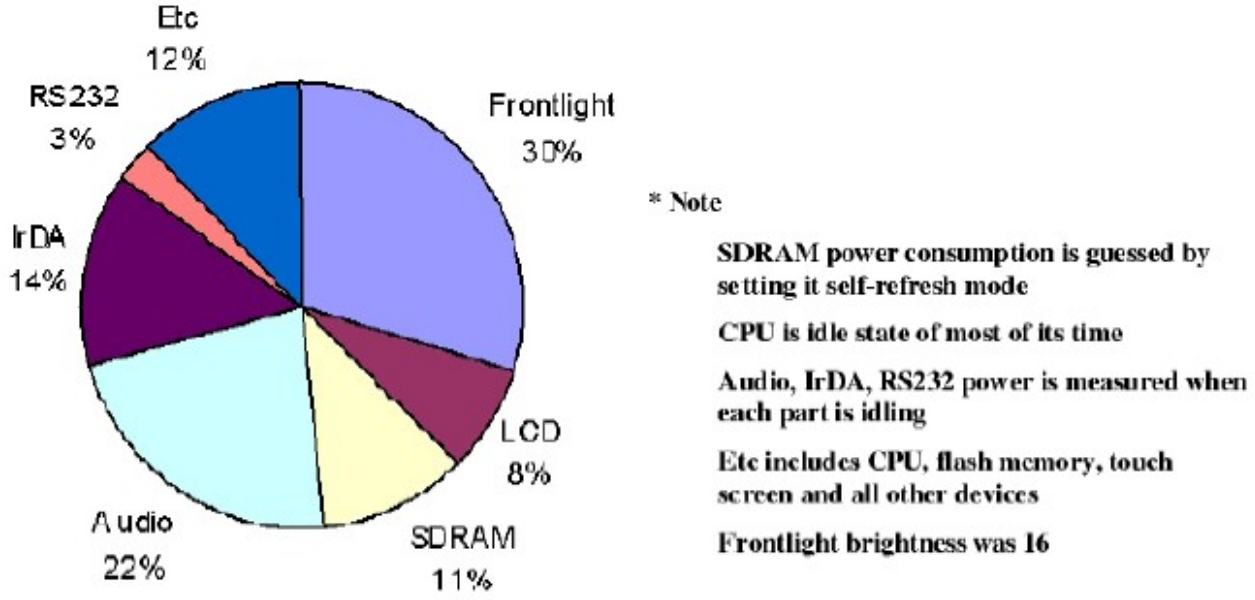

Figure 8. PDA consumption for its main components.

In the first experiment four measures were taken in thirty minutes with Windows Pocket PC 2002. Measures were taken with a software introduced before, Battery Monitor (Figure 9), and with a multimeter (Figure 10).

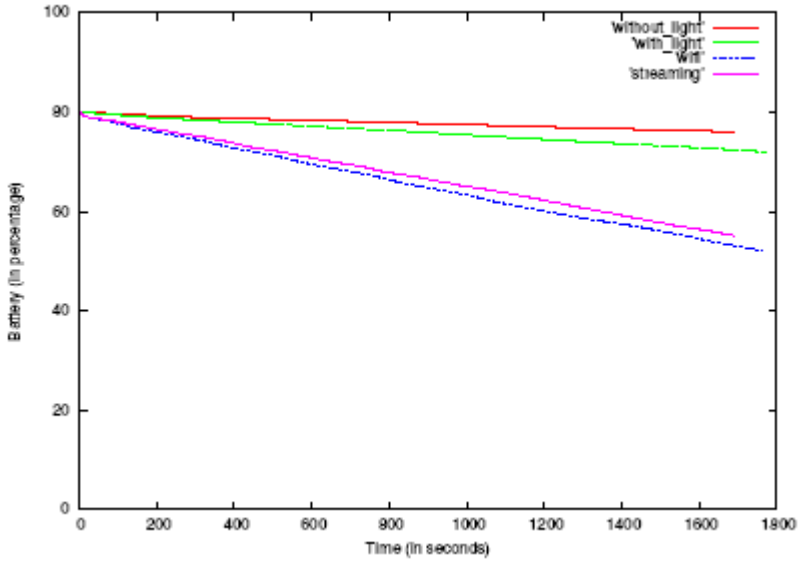

Figure 9. Measures taken by software

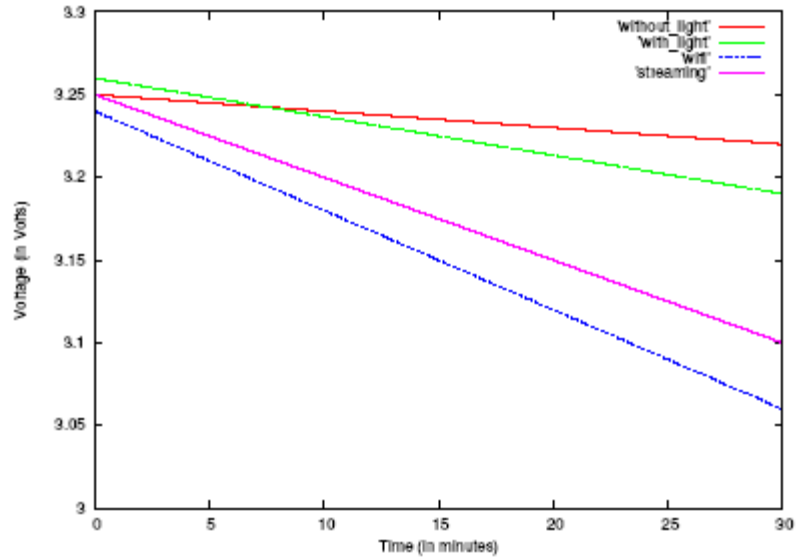

Figure 10. Measures taken with a multimeter

This experiment compares the consumption of the back-light, the wifi component and the use of CPU, by streaming a video as an example of high resource demanding application.
The results of this experiments show many conclusions. First, we can see that the wifi consumes more than the video streaming. This doesn't seem normal because the video streaming uses more of components, like the player and the cpu. But in the wifi experiment, the PDA receives more packets than video streaming. So this comparison shows that the greatest problem in video streaming with PDA is the wifi consumption. Another experiment has shown that it's not possible for a PDA to read a video stream for more than 90 minutes.
We can also see that the back-light consumes much less than the two other components.

3.2.2 Linux and Windows comparison

These experiments compare the two operating systems, Linux and Windows. There is a PDA version for the both, Familiar Linux and Windows Pocket PC. There are other OS available for PDAs within the Open Source and proprietary families, which we could not test.

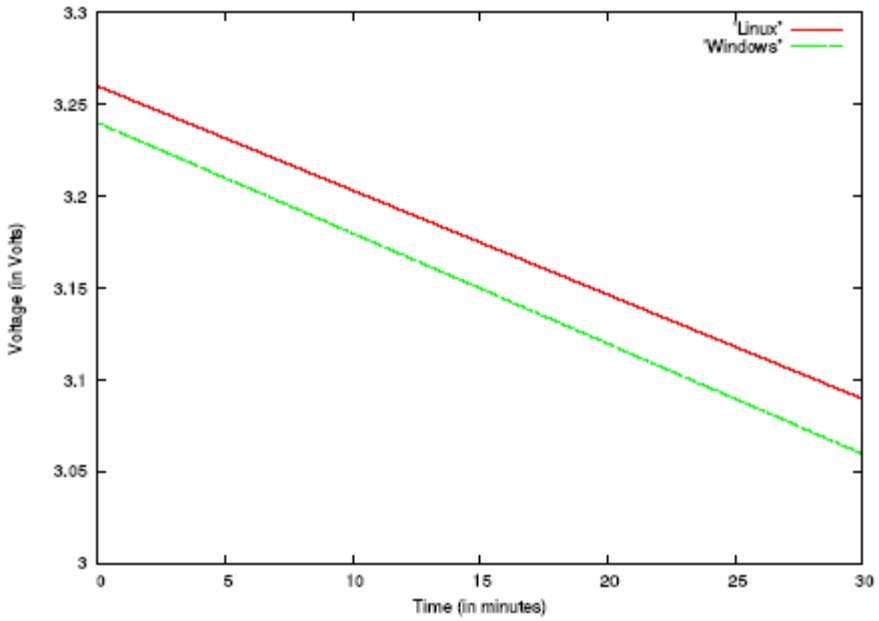

Figure 11. Linux and Windows consumption

Then, the experiments are only with the wifi components, the main component for power consumption. These tests are realized with the UDP packets sender introduce before, and the measures are only with the multimeter (Figure 11). In fact, Familiar Linux and Windows Pocket PC don't manage the battery power level in the same way: The last integrates a backup battery.
The results of this experiments show that the both operating systems have a battery consumption which is almost the same for the wifi component.

4. Energy measurements on a laptop

We made other measures using a laptop as a client, again while streaming a video in order to evaluate a high demanding activity. We used ACPI (Advanced Configuration Performances Interface) to get readings of current voltage level and current drawn from the laptop battery. ACPI can be integrated in the Linux kernel and has a mapping to the `/proc` file system where current battery status can be accessed. ACPI takes measurements directly from the battery. We unplugged the power supply to avoid the constant recharging of the battery .
In order to estimate the energy consumed by the laptop while processing the video stream, we took three times the measurements of the voltage level and current intensity from the battery. We then compute the energy using also the CPU total usage time as follows :

$$E = V\ I\ dT$$

where V is the average of voltage and I is the average of current being drawn from the battery in the time interval dT.
We also used `tcpdump` to trace the downstream TCP connections at the server side and analyzed these traces files using `tcptrace`. `tcptrace` gives several statistics about a given TCP connection, including the overall throughput of the connection. During the streaming we launch from the server the tcpdump command to capture the network activity on the wireless cards and at the end we see if there are any packets that have been lost .
We also used `netstumbler` which is a tool for Windows that allows to detect Wireless Local Area Network (WLAN) using 802.11b, 802.11a and 802.11g. It provides many informations such as the ratio Signal /Noise, the number of access point that are located nearly, as well as their bandwidth, channels, frequency and encryption. During our work we focus on the radio to see if there is some interference and noise that affect the power saving mechanism but the ratio Signal/Noise is still good with all the experiments that we have made.

References

P.Devevey, **Energy consumption analysison mobile devices over wireless network**, Genova 2004.

N.Lorenzon, **Reduction of Wireless Interface Energy Consumption for Video Streaming**, Genova 2004.

C.Tambary, **Energy Saving in Wireless Streaming**, Genova 2005.