Grzegorz Grzelachowski

June 14th, 2011

Report on practice at Poznań Supercomputing and Networking Center.

The practices at PSNC started in the middle of march and lasted for 3 months. The main task was to integrate an Android device with the Vitrall system. It should be able to connect to a server through the Websocket protocol and send information gained from its' sensors that will allow the user to control the rendered model.

I started with some basic issues. The first step was writing an application, which used the device sensors such as accelerometer and magnetic field sensor to calculate the current orientation of the cell phone. The second thing I had to focus on was connecting the Android device to the Websocket server. I used the echo server, which is open for general use. After finishing the application I had to check the performance of the cell phone. The next step was creating an application, which merged Websocket client with the sensor using application.

Below is a list containing brief descriptions of the applications which I mentioned about:

- Sensor_sample – the first application which used sensors to show the current orientation of the device in the global coordinate system; it was only the basic application, but It showed me a few aspects that should I also focus on: rate of sensor events, inaccurate measurements and the power consumption .

- WS_client – the first application that was able to connect to a Websocket server, send and receive text messages. It was the most difficult step during the whole practice, because of the variety of problems I encountered, such as handshake issues.
  In the next stage this application has been modified and used to test the performance of the device.

- WS_test – the test application based on a simple Websocket client. Testing procedure and results are described in the next section.

- Sensor – my last application, which combined Websocket client and sensor using application. It is able to collect the information from the sensors and send the average values to the server with a specified frequency.

The test application was created to check the performance of the number of android devices when using Websocket protocol. It measures the time of sending a specified number of messages and the time of processing them, when they are sent back by the echo server. The tests where run on devices using specially created WiFi network, which eliminated differences between the speed of internet connections. There are also two examples of devices using 3g network. I used two versions of the application, which differed in the way of processing incoming messages, the second one is more complex. All of the results are shown below (values in seconds).

Version 1.

Phone model	OS	500 messages, sending	500 messages, receiving	1000 messages, sending	1000 messages, receiving
Samsung Galaxy S	Android 2.1	0,075	0,388	0,138	0,453
Samsung Galaxy Tab	Android 2.2	0,151	0,239	0,264	0,490
HTC Wildfire	Android 2.2	0,836	0,871	1,758	1,611
HTC Desire	Android 2.2	0,164	0,247	0,269	0,292
HTC Desire HD	Android 2.3	0,266	0,273	0,513	0,464
Samsung I5700	Android 2.1	0,337	0,299	0,884	0,849

Version 2.

Phone model	OS	500 messages, sending	500 messages, receiving	1000 messages, sending	1000 messages, receiving
Samsung Galaxy S	Android 2.1	0,076	0,303	0,131	0,517
Samsung Galaxy Tab	Android 2.2	0,144	0,431	0,268	0,556
HTC Wildfire	Android 2.2	1,191	1,161	2,305	2,284
HTC Desire HD	Android 2.3	0,145	0,249	0,283	0,341
Samsung Galaxy S (3g)	Android 2.1	0,083	0,484	0,155	0,579
Samsung I5700 (3g)	Android 2.1	0,315	0,727	0,711	1,101

As we can see, the performance of the Android based devices is acceptable and will not be a bottleneck in the final application. Even the slowest CPU used in Android cell phones (HTC Wildfire) is able to handle the sort of task in short time. In the final application, the frequency of 25-50 messages per second will be required, which will not be a problem for these smartphones.

The results were presented to Tomasz Kuczyński during consultations, once a week. The final application, Sensor, is able to connect with Vitrall server and enables the user to control the rendered model. There are still many improvements to make, such as supporting multitouch, queuing the users in cases where a number of people try to use Vitrall simultaneously.