REMOTE MONITORING OF A PERMANENT MAGNET SYNCHRONOUS MOTOR USING ANDROID DEVICES

Cristian RECOŞEANU[1], Anca-Simona DEACONU[2], Aurel-Ionuț CHIRILĂ[3], Valentin NĂVRĂPESCU[4], Ioan-Dragoş DEACONU[5], Constantin GHIȚĂ[6]

Currently, the Android operating system is associated in most cases with divertissement type applications. Whether talking about smartphones or tablets, Android applications focus more on offering users ways to spend their free time and less on controlling and monitoring industrial processes. With this idea in mind, this paper discusses concepts regarding the server-client model and how this can be used to create a remote monitor system for the Android operating system. In the presented implementation, the system monitors currents, voltages and temperatures of a permanent magnet synchronous motor. The main advantages and disadvantages of such an approach are outlined.

Keywords: industrial applications, remote monitoring, Matlab plots, Android putingdevices, client-server model, permanent magnet synchronous motor, cloud com

1. Client-server model

All remote monitoring applications are based on the concept of client-server model.

In order to understand how they work, it is necessary to have a basic understanding of this concept.

The client-server model is a computing model that acts as a distributed application which partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters, called clients. [1]

A server machine is also called a “host machine” and is designed to perform one or more server programs in order to share their resources with clients. A client may connect to a server and request its content or the server’s service function. Therefore server machines await incoming requests from clients and when they receive a request they return the desired information (it may be a

[1] Student, Electrical Engineering Faculty, University POLITEHNICA din Bucharest, Romania, e-mail: cristirec@yahoo.com
[2] PhD student, Electrical Engineering Faculty, University POLITEHNICA din Bucharest, Romania
[3] PhD Lect., Electrical Engineering Faculty, University POLITEHNICA din Bucharest, Romania
[4] Prof., Electrical Engineering Faculty, University POLITEHNICA din Bucharest, Romania
[5] PhD Lect., Electrical Engineering Faculty, University POLITEHNICA din Bucharest, Romania
[6] Prof., Electrical Engineering Faculty, University POLITEHNICA din Bucharest, Romania

resource or the result of a processing request). Clients and servers communicate over a computer network, usually on separate hardware. The server provides service simultaneously to one or many clients who initiate requests for such services, as described in Fig. 1.

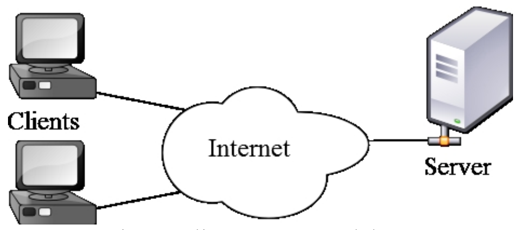


Fig. 1. Client-server model [2]

The client-server model has become one of the central ideas of network computing. Many of the Internet's main application protocols such as HTTP, SMTP, Telnet and DNS use the client-server model. Functions such as email exchange, web access and database access are built on the client-server model.

A few examples of client applications are: web browsers, email clients, chat clients. Some server examples are: web servers, ftp servers, mail servers, file servers, database servers.

One of the latest concepts developed recently in IT is Cloud Computing. Cloud computing is the delivery service of computation, software, data access and storage resources without requiring cloud users to know the location and other details of the computing infrastructure. [3] This concept is also based on the client-server model.

2. Developing the system

The idea of the project was to create a client application for the Android operating system. This application would allow any user with an Android smartphone or tablet and an internet connection to remotely monitor the parameters (currents, voltages and temperatures) of a permanent magnet synchronous motor. The name of the released application is "Industrial Monitor".

In order for this to be achieved, the client has to send a processing request towards a main server and wait for results to be returned. When the client receives the results, the user can view them in the form of Matlab plots covering every parameter presented.

The main server waits for requests from clients. When a request is received, the server launches a client handler for that specific client request. The client handler is a part of the server that makes multiuser communication possible. It can be viewed as a subroutine that launches independently for each client. The client handler solves the request for computation and is responsible for sending the result towards the client. Thus it needs to have data for each parameter presented in order to begin Matlab plot processing. To receive the data, the client handler launches another request for data towards a second server called data measurement server. When the result is returned with the required data, the client handler begins Matlab plot processing and returns these plots to the Android client device. Basically, the main server acts both as a server for the Android client and as a client for the data measurement server.

The data measurement server waits for requests from the main server's client handlers (clients for the data measurement server). When one request is received, the server reads the measurement data stored by the data measurement system in specific files and returns data for every parameter to the client handler in the main server.

The data measurement system is the link towards the physical machine. The device measures the currents (I_1, I_2, I_3), voltages (U_1, U_2, U_3) and temperatures (Motor and Ambient) for the permanent magnet synchronous motor.

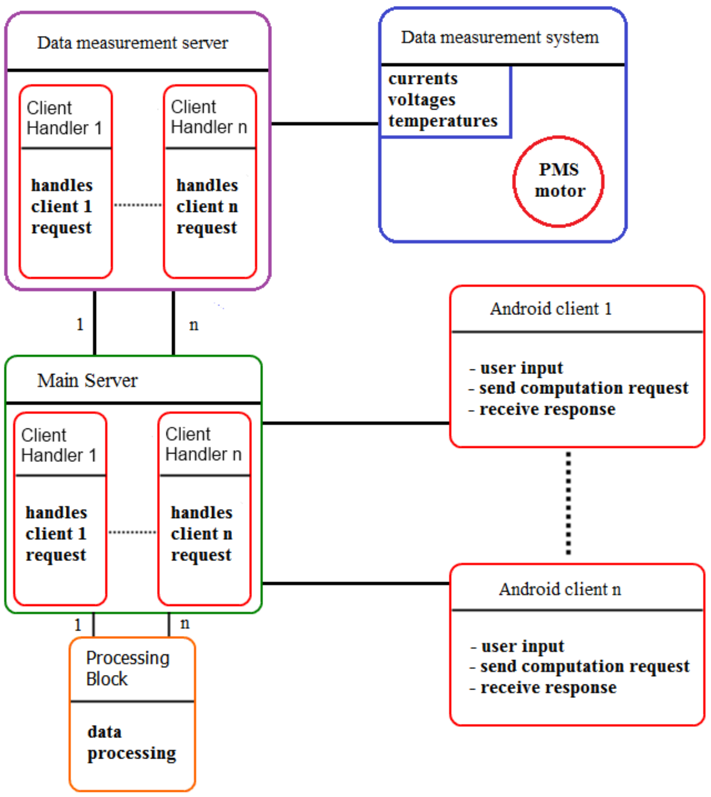


Fig. 2. System design schematic

The Matlab plot processing block was designed using a special Matlab toolbox called "MatlabBuilderJA". This toolbox allows deployment of Matlab code in the form of Java classes. The processing block is fed with the data for each parameter and returns corresponding Matlab plots. This process takes place in each client handler.

The system was designed using more than one server, because there may be several motors to monitor, scattered across different geographical regions. This means that each motor should have its own data measurement server that provides data to the main server. In the current implementation only two servers were needed because there was only one motor to monitor.

3. Necessary programs and programming environments

The servers were coded in Java and the client was created using the Android SDK.

The Matlab plot processing block requires Matlab software.

The main programming environment used was Eclipse because it provides support both for coding Java applications such as the servers and also for developing Android applications.

List of necessary programs:

- Java SDK jdk1.6.0_29 32 bit edition [4];
- Matlab 7.9.0.529 (MatlabBuilderJA toolbox) 32 bit edition [5];
- Matlab Compiler Runtime;
- Eclipse Classic 3.7.1 32 bit edition [6];
- Android SDK r15 [7];
- ADT plugin for Eclipse;

Choosing 32 bit editions for these programs was dictated by the Android SDK which currently is distributed only in the 32 bit variant. Compatibility problems may occur if not working under the same standard.

The operating system used for developing was Windows 7 Professional Service Pack 1 (Build 7601).

4. Testing the system using a simulator and a real Android smartphone device

The Android SDK provides a great simulator tool for testing applications before release. This tool launches directly from the Eclipse environment and can reveal errors and bugs. Eclipse provides a few tools for debugging that trace errors reported by the simulator and reveal the propagation of errors in the application.

The first step in testing the system is starting the servers. In the event that the servers are behind a router, their ports need to be forwarded for use. This can easily be done by configuring the router. When coding the Android client, we need to set its connection port to the exact same port of the main server.

The second step is launching the Android simulator and launching the Android client application. The simulator first installs the application and then runs it. There may be cases where the simulator does not run it after installation. In this situation, the app may be launched from the simulator's applications menu. Starting the simulator may take some time because it is similar to booting an Android smartphone.

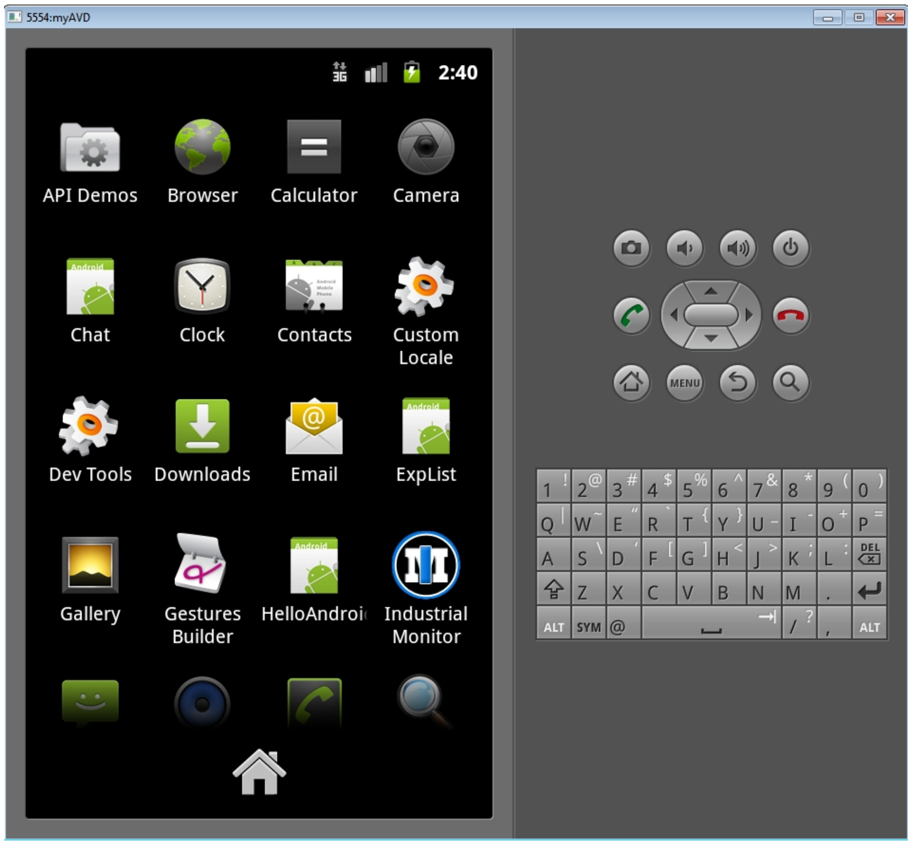
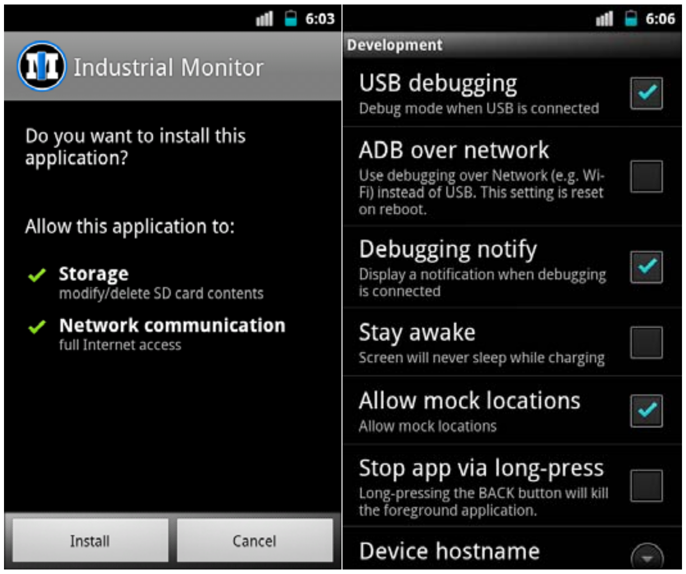


Fig. 3. Android simulator (applications menu)

After the testing of the application on the simulator produced satisfying results, testing on a real smartphone needs to take place. However, it is first necessary to compare the Android version of the simulator to the version of the test device, as they must be operating the same one for optimal compatibility (in this case they both have Android Gingerbread 2.3.7). Also, the resolution used in the simulator needs to be the same as the smartphone's screen resolution.

The next step is to install “Industrial Monitor” on the test smartphone.

For debugging the application, the smartphone needs to have the “USB debugging option” enabled. This can be found in Settings > Applications > Development.



6:03

Industrial Monitor

Do you want to install this application?

Allow this application to:

✓ **Storage**
modify/delete SD card contents

✓ **Network communication**
full Internet access

Install | Cancel

6:06

Development

USB debugging [✓]
Debug mode when USB is connected

ADB over network
Use debugging over Network (e.g. Wi-Fi) instead of USB. This setting is reset on reboot.

Debugging notify [✓]
Display a notification when debugging is connected

Stay awake
Screen will never sleep while charging

Allow mock locations [✓]
Allow mock locations

Stop app via long-press
Long-pressing the BACK button will kill the foreground application.

Device hostname

Fig. 4. Installing and debugging on a real smartphone

Afterwards, the smartphone is connected to a PC and, after making sure that an internet connection is active (usually Wi-Fi), the entire batch of tests may be launched. For testing purposes an LG Optimus Sol (E730) smartphone with an 800x480 screen resolution was used.

5. Industrial Monitor – Android client application

After thoroughly testing the application, it is now in the release candidate phase. This means main bugs and errors are repaired and it is ready to be tested by the users.

In the following section, the general user experience of a first time user will be presented.

First, the user installs the application. After installation, they look for the proper icon and launch the application. On first launch, they are greeted by the main layout of the application (see Fig.5). This has three buttons and a few checkboxes. Being a first time user, they might decide to press the help button marked with the “?” sign. This will redirect the user to the help section depicted in Fig.6. The help section explains the basics steps to get started, what buttons to press and when to do it. At the end, the user is informed they can return to the main layout by pressing the back button on their smartphone.

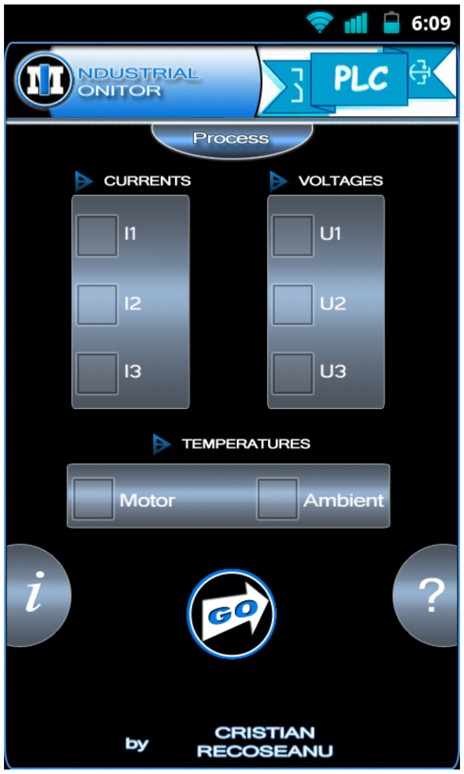
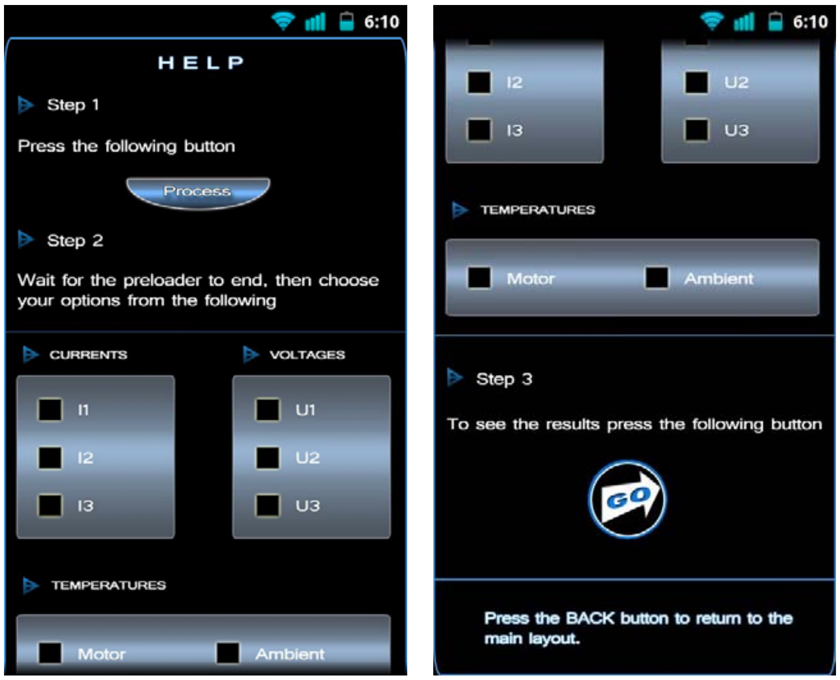


Fig. 5. Industrial Monitor – main layout

After viewing the help section, the user will follow the instructions given. They first press the “Process” button and a pre-loader that states “Loading. Please wait…” pops up as shown in Fig.7. If the client cannot connect to the main server or if the data measurement server is down, the user is informed by a popup message (see Fig.8). After the pre-loader has finished, the user gains control of the checkboxes. They can now choose any of the options for plot viewing. After they make their choice, they press the “GO” button. This button redirects them to the plot layout section of the application. Here the user can view the requested plot for the chosen parameters. They can zoom in and out by using the appropriate button and can also save the plot to the smartphone SD card by touching and holding the plot for a few seconds (as shown in Fig.9).



6:10

HELP

Step 1

Press the following button

Process

Step 2

Wait for the preloader to end, then choose your options from the following

CURRENTS: I1, I2, I3

VOLTAGES: U1, U2, U3

TEMPERATURES: Motor, Ambient

Step 3

To see the results press the following button

GO

Press the BACK button to return to the main layout.

Fig. 6. Industrial Monitor – help section

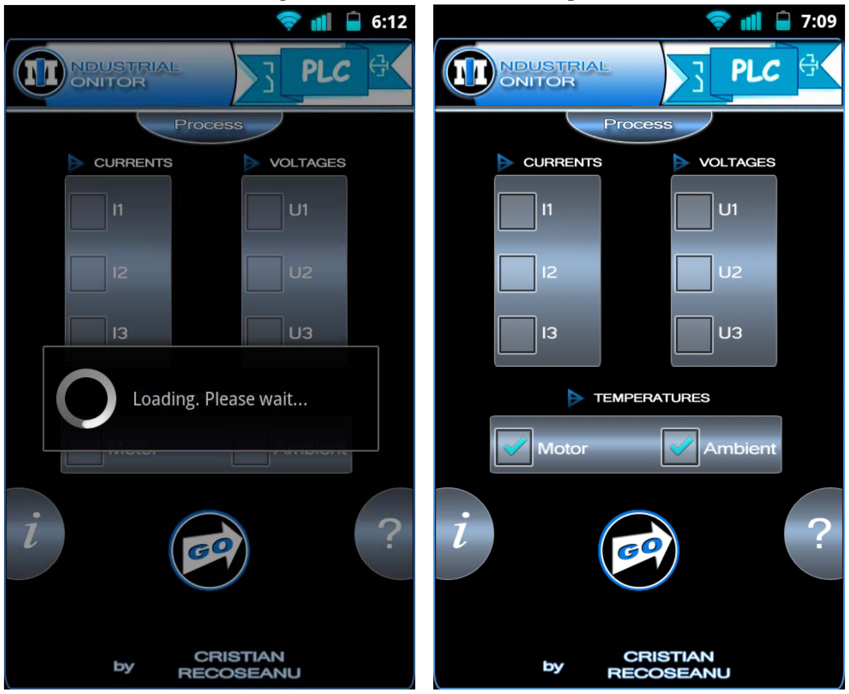
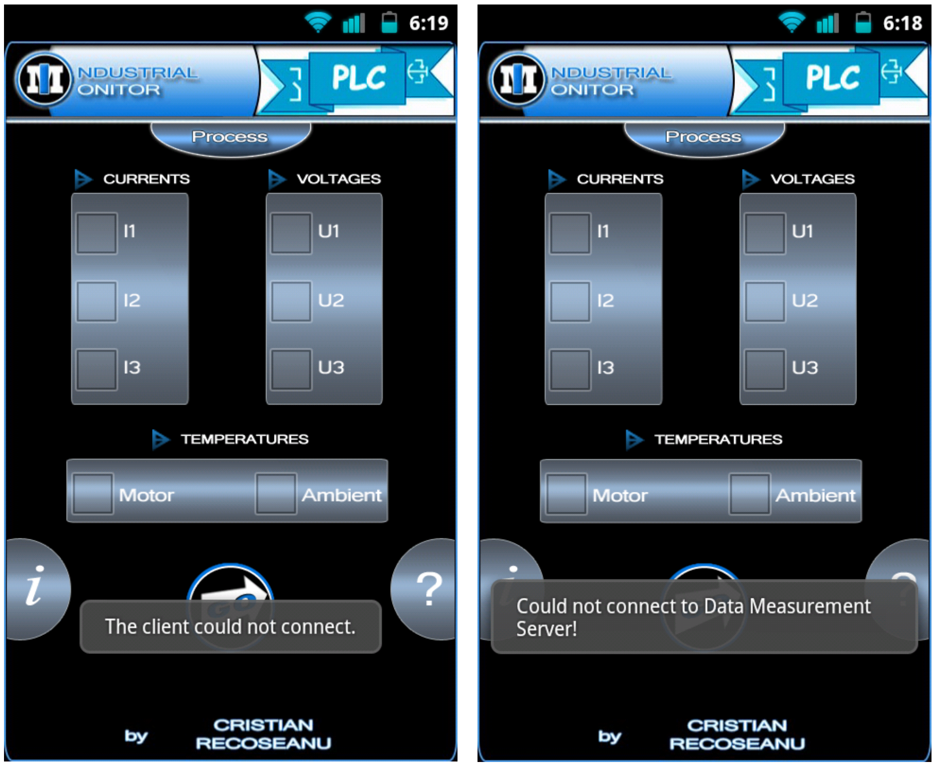


Fig. 7. Industrial Monitor – pre-loader and checkboxes



The client could not connect.

Could not connect to Data Measurement Server!

Fig. 8. Industrial Monitor – popup error messages

If the user wants to return to the main layout, they have two options: first by pressing the back button and second by pressing the menu button and choosing the “Home” option.

When returning to the main layout of the application, they have a few more options when pressing the menu button. They can return to the previously viewed plot by choosing the “Plot” option or they can rate the application on the Google Play store by choosing the “Rate this app” option. The main layout allows the user to choose another parameter for plot viewing or even press the “Process” button again and acquire a new set of measurements.

After the plot viewing process satisfied the user~~’s~~ need~~s~~ for monitoring, they may decide to press the information button marked with the “*i*” letter. This button redirects them to the information section (see Fig.10). This section was designed to explain the construction and purpose of the system in a concise manner. It includes the schematic of the entire system and contact information. At the bottom, the user is again informed that by pressing the back button they will be taken to the main layout. By pressing the back button while being in the main layout, the user is informed that pressing this button again will exit the application (see Fig.11).

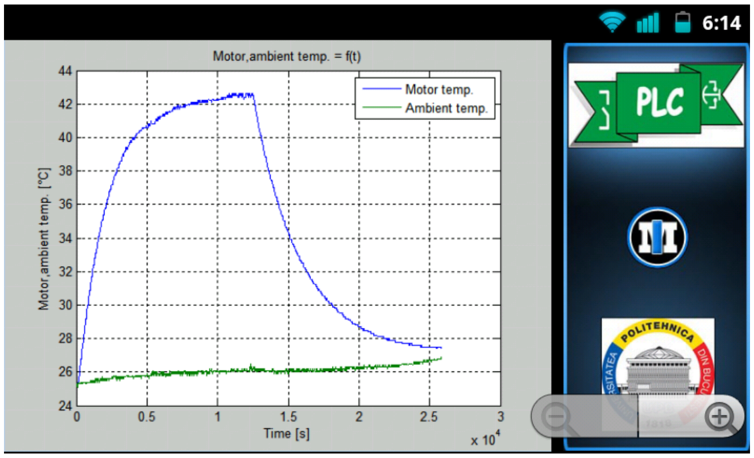
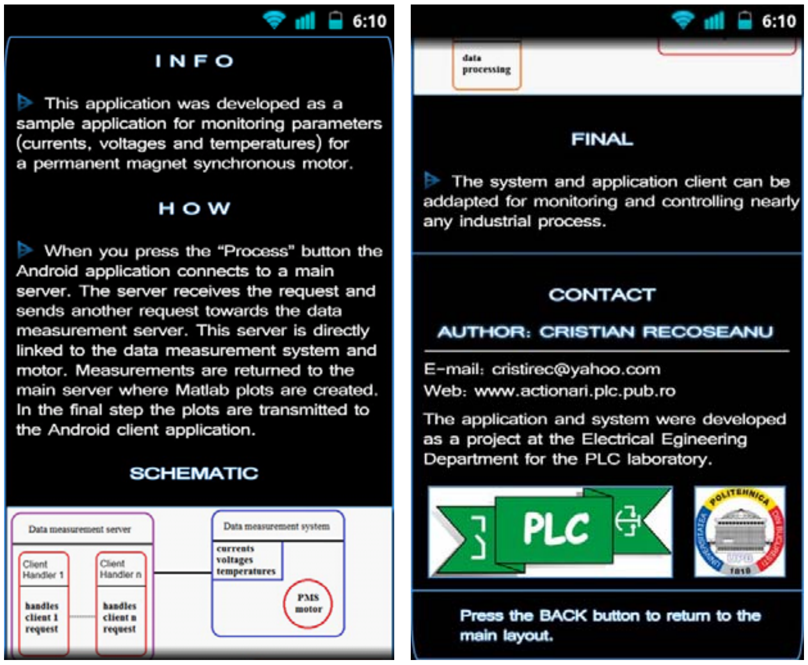


Fig. 9. Industrial Monitor – plot section (zoom features)


6:10

INFO

This application was developed as a sample application for monitoring parameters (currents, voltages and temperatures) for a permanent magnet synchronous motor.

HOW

When you press the "Process" button the Android application connects to a main server. The server receives the request and sends another request towards the data measurement server. This server is directly linked to the data measurement system and motor. Measurements are returned to the main server where Matlab plots are created. In the final step the plots are transmitted to the Android client application.

SCHEMATIC

6:10

FINAL

The system and application client can be addapted for monitoring and controlling nearly any industrial process.

CONTACT

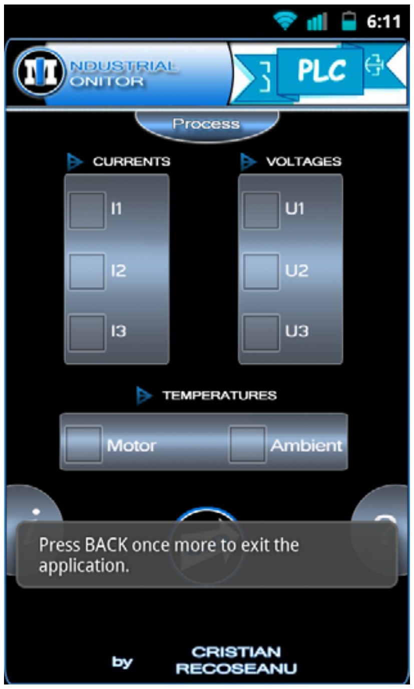
AUTHOR: CRISTIAN RECOSEANU

E-mail: cristirec@yahoo.com
Web: www.actionari.plc.pub.ro

The application and system were developed as a project at the Electrical Egineering Department for the PLC laboratory.

Press the BACK button to return to the main layout.

Fig. 10. Industrial Monitor – information section



Press BACK once more to exit the application.

Fig. 11. Industrial Monitor – exit option

6. System results

The monitor system can be used to study the motor's thermal behaviour under different supplying techniques. In the following section, the experimental results obtained for a 50Hz sinusoidal supply, a 50Hz sin-PWM inverter supply with 0.7 kHz switching frequency and a 50Hz sin-PWM inverter supply with 14.5 kHz switching frequency are analysed. The motor loading is the same for each of the three cases. Temperatures were acquired using a time step of 5 seconds. The ambient temperature was measured to ensure that it remains constant during the test (25°C). This means that the ambient temperature does not influence the heating process. For all the tests, the motor was driving the same load (about half of its rated load).

The motor voltage, current and frequency were measured using the data measurement system. This way it was assured that the voltage and frequency (hence the motor speed) remains constant for the duration of the.

The temperature plots are displayed in the following figures.

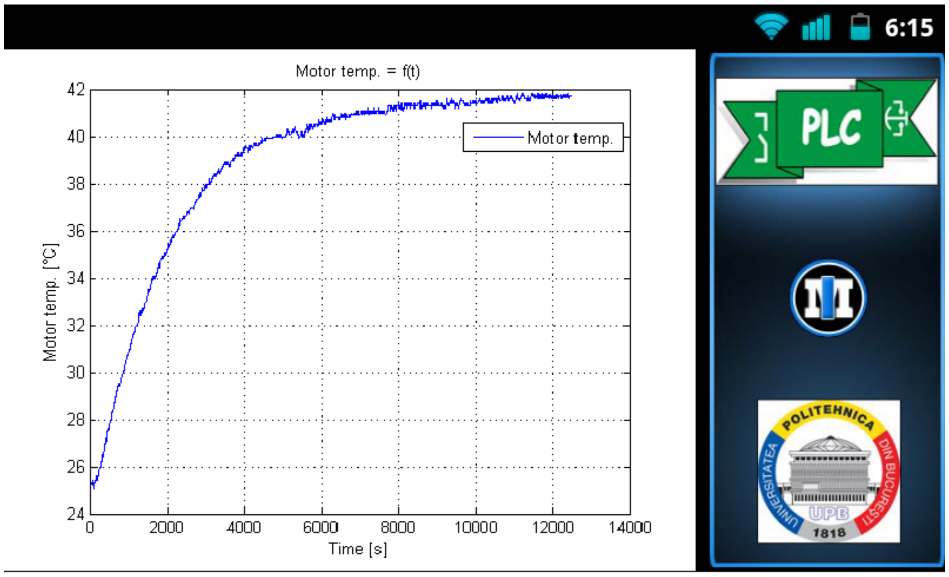


Fig. 12a. Motor temperature - 50Hz sinusoidal supply

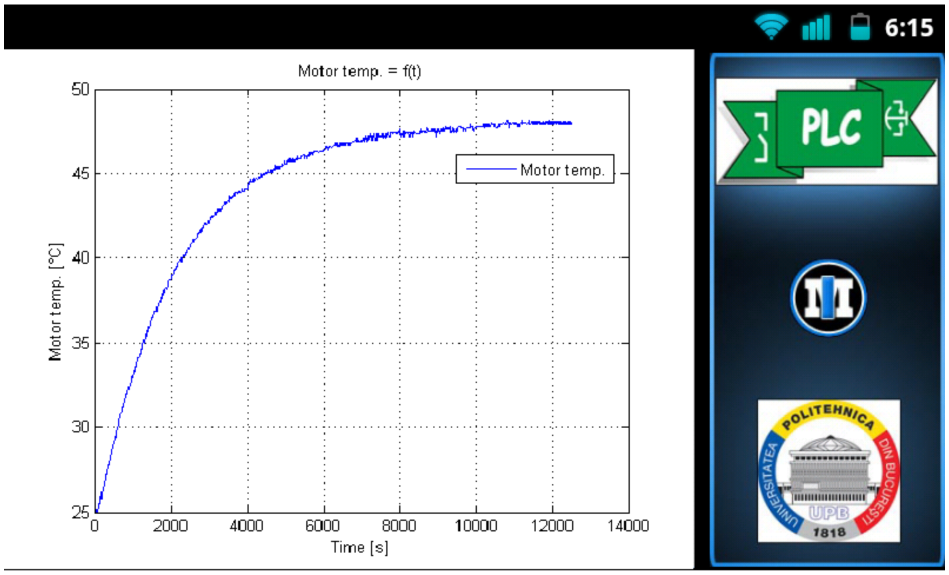


Fig. 12b. Motor temperature - sin-PWM inverter supply (0.7 kHz switching frequency)

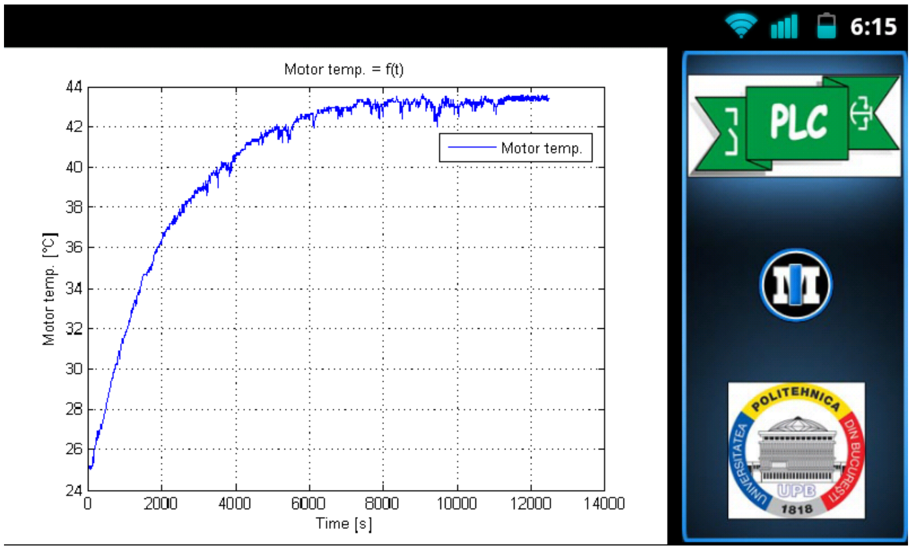


Fig. 12c. Motor temperature - sin-PWM inverter supply (14.5 kHz switching frequency)

The study of the three plots brought to light the following conclusion. In the case of synchronous motors fed by variable frequency converters (PWM type), a temperature increase of the stator armature and windings is observed. The higher temperature is due to the losses generated by the voltage and current harmonics.

For the same motor load, the stator armature steady state temperature is approximately 6.3 °C higher when the stator is fed by a sin-PWM inverter (with 0.7 kHz switching frequency) compared to a sinusoidal supply.

7. Conclusions

The client-server model is probably the most often used concept in computing and presenting information over a network. This concept was adopted in implementing a remote monitor system for the Android platform.

Mobile operating systems and mobile platforms (smartphones, tablets) keep expanding rapidly. 2010 was a key year for the growth of the smartphones market. IDC (International Data Corporation) reported that smartphones outsold PCs for the first time [8]. “Mobile era helps make the right information available to the right person for performing critical business activities regardless of time and location. Mobile applications can be developed in several strategic areas to support employees, customers, and trading partners.” [9]

Creating an application for on-field engineers seemed only logical and provides many advantages compared to classical approaches. The main advantage of the remote system is pretty straightforward – the ability to work from a remote location, when connected to the internet. The user can receive information from the motor and view it by the help of Matlab plots. They don't need to travel to the motor's location anymore. They can also monitor multiple motors scattered across large geographical regions. Such a system can also be implemented in business environments to provide remote access or computation for a specific problem requested by employees. The need for computation no longer means being bound to a desktop computer inside an office. An employee can request computation from their smartphone with internet connection, thus having on-the-spot access to solving a problem. Bottom-line, solving a problem using a remote system will provide increase in mobility for the users.

The second advantage would be in the sense of reducing the necessary processing power. In the remote monitor system, nearly all the processing is done inside the main server. Client systems that connect to the server (Android smartphones or tablets) do not need impressive processing power. Reducing the number of systems with high processing power also means lower power consumption.

Another advantage introduced by the remote system is reducing the costs required for professional software. The main server inside the remote monitor system uses a Matlab processing block. This block was creating using a single Matlab license. Now imagine multiple engineers having to monitor a motor without having the remote system. They would all have to acquire Matlab licenses that may become quite expensive depending on included toolboxes. For a more graphical image, the remote system provides a one-to-many relationship with its users. Many users can access the resources and computation but only one Matlab license is required.

Moreover, such a system can be easily adapted for virtually any industrial process. In the current implementation, the system monitors a permanent magnet synchronous motor. With the right hardware and tools (data measurement system), the system can be adapted for example to monitor a solar energy park.

Of course, as with any artificial system, the remote monitor features some disadvantages. The first obvious one is the dependency on internet access. Any user that requires the information provided by the system needs to have an internet connection. This effect can be diminished by using multiple ways of maintaining a steady connection (for example the servers may use Ethernet connection and 3G as a backup or they may use multiple internet service providers).

Another concern when working over a network connection is data safety. It must be ensured at all times that outgoing and incoming data are not intercepted or altered in any way. The remote monitor system uses custom classes to encode the data transferred between the servers and the clients. These classes are implemented only in the servers and client. The ability to install the client application to the SD card is another potentially risky action. Many developers choose to implement this option because low-end smartphones have a very small internal memory. Giving the option to install the application on the SD card makes the application easier to access, modify and it may be illegally shared.

The monitoring system presented is just a stepping stone for future development. The system is scalable, providing multiple motors scattered across different geographical regions to be monitored and managed by connecting to a single primary node server. Moreover, such a system can be easily adapted for virtually any industrial process. In the current implementation, the system monitors a permanent magnet synchronous motor. With the right hardware and tools (data measurement system), the system can be adapted for example to monitor a solar energy park. By adding more complex features, the system could in the end resemble something similar to the monitoring block of a SCADA system. SCADA system supervises, controls, optimizes and manages generation and transmission systems [10].

The application in its current state is available for download in the Google Play store (previously called Android Market). For further testing, you can download it to your smartphone for free by scanning the following QR code with your phone or by searching for “Industrial Monitor” in the Google Play store.

Industrial Monitor QR code – Google Play Store

8. Acknowledgment

The work has been co-funded by the Sectoral Operational Programme Human Resources Development 2007-2013 of the Romanian Ministry of Labour, Family and Social Protection through the Financial Agreement POSDRU/107/1.5/S/76903 and POSDRU/89/1.5/S/62557. It was also supported by SP1-JTI-CS-2009-02-GRC-03-001 and EDL – UPB.

R E F E R E N C E S

[1]. *Sun Microsystems* - Distributed Application Architecture [online] [quote 21.04.2012]; Available from: URL: http://java.sun.com/developer/Books/jdbc/ch07.pdf

[2]. Wikipedia – Client-server model [online] [quote 21.04.2012]; Available from: URL: http://en.wikipedia.org/wiki/Client–server_model

[3]. NIST - The NIST definition of Cloud Computing [online] [quote 21.04.2012]; Available from: URL: http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf

[4]. Java SE 6 documentation [online] [quote 21.04.2012]; Available from: URL: http://docs.oracle.com/javase/6/docs/index.html

[5]. MathWorks - MatlabBuilderJA toolbox [online] [quote 21.04.2012]; Available from: URL: http://www.mathworks.com/products/javabuilder/

[6]. *Eclipse* - Developing environment resources [online] [quote 21.04.2012]; Available from: URL: http://www.eclipse.org/resources/

[7]. Android - SDK resources [online] [quote 21.04.2012]; Available from: URL: http://developer.android.com/resources/index.html

[8]. GSMARENA.com - IDC report says smartphones outsell computers for the first time [online] [quote 21.04.2012]; Available from: URL: http://www.gsmarena.com/idc_report_says_smartphones_outsell_computers_for_the_first_time-news-2303.php

[9]. *A. Petculescu*, Web Services Applied to Mobile Environments, U.P.B. Scientific Bulletin, Series C – Electrical Engineering, Volume 73, Iss. 3, ISSN 1454-234x, pg. 165-176, 2011, [online] [quote 21.04.2012]; Available from: URL: http://www.scientificbulletin.upb.ro/rev_docs_arhiva/full63574.pdf

[10]. *Nicoleta Arghira, Daniela Hossu, Ioana Făgărăşan, S. S. Iliescu, D. R. Costianu*, Modern SCADA Philosophy in Power System Operation – A Survey, U.P.B. Scientific Bulletin, Series C – Electrical Engineering, Volume 73, Iss. 2, ISSN 1454-234x, pg. 153-166, 2011, [online] [quote 21.04.2012]; Available from: URL: http://www.scientificbulletin.upb.ro/rev_docs_arhiva/full87719.pdf