Open and Flexible Embedded System Applied to Positioning and Telecontrol

Verónica Medina, *Member, IEEE*, Octavio Rivera, David Oviedo, Enrique Dorronzoro, and Isabel Gómez, *Member, IEEE*

***Abstract*—This paper presents the development and testing of an open and flexible embedded system applied to positioning and telecontrol (OFESAPO) for outdoor applications. The system is composed of a control center (CC) and a set of remote terminal units (RTUs); the International Electrotechnical Commission (IEC) 60870-5 series has been chosen for communication among them. This is a standard protocol of real-time telecontrol applications. The CC is a personal computer, and the RTUs are based on open hardware and software. The RTU hardware is an embedded system, i.e., a system-on-chip-type design using field-programmable gate array that has been programmed with the open-core LEON running Linux operating system. For prototyping, the GR-XC3S-1500 board has been used. As there is no open source code available for the IEC standard protocols, an open source code has also been implemented. Hence, both the hardware and the software are open source in OFESAPO. Several tests have been made to show the system's limitations and the suitability for real-time applications. A prototype has also been tested in a real environment, where the real position of two moving RTUs was shown by a CC using Google Map.**

***Index Terms*—Embedded system, field-programmable gate array (FPGA), International Electrotechnical Commission (IEC) standard, open hardware, open software, positioning, telecontrol.**

I. Introduction

POSITIONING technology and mobile communication provide the right environment to make location-based services (LBS) closer to daily life. There are many applications of LBS that make people's or business life easier because of the ability to send positioning information in real time, for instance, the waiting time for a bus in a bus stop. Another application is surveillance, for example, in forest fire fights, not only of firefighters but also of vehicles, or in rescue operations. These kinds of applications use mobile devices (personal digital
assistant (PDA), cell telephone, etc.), controlled devices, a controller device (a computer, a PDA, etc.), and a real-time geographic information system (GIS). The controlled devices are equipped with both a positioning technology and a mobile communication. The controller device can also have the same mobile technology as the controlled devices, but it is not compulsory. The GIS could be implemented in the controller device or outside in a specific server.

There are several positioning systems based on different technologies [1] for the determination of the position of an entity in an outdoor or indoor environment. Outdoor positioning systems determine position using satellite information, known as "Global Positioning System" (GPS), or cellular telephony information, known as "network-based location." Indoor positioning systems use both infrared and radio frequency (RF) to determine position [2]. Each positioning technology has its own location data type, so a system that integrates more than one could use the integrated model presented in [2].

Several mobile communication technologies can be used, like Global System for Mobile (GSM), generic packet radio service (GPRS), or WI-FI, all of them based on cellular configuration, satellite link, or RF. Using one or the other depends on their coverage area, upload and download speed, and roundtrip delay [4].

A real-time positioning system based on GPS, GSM, and GIS (G^3) is presented in [5]. The system is implemented on both a PC-based and a μP-based (C8051 microProcessor, μP) hardware platforms and designed to operate in Microsoft Windows operating system (OS; MS platform) with Microsoft Visual Basic communication mode under MapInfo for GIS. This system is a, LBS but also allows to control the controlled device, i.e., to get some measurements made by its sensors and to receive commands from the controller device. A portable personal positioning system employing System-on-Chip (SoC) technology, i.e., the C8051F021 microcontroller, combined with GPS for positioning and GPRS for mobile communication is presented in [6].

In this paper, a positioning system for telecontrol based on open software and hardware platform is presented, which differs from those in [5] and [6] because of its openness; hence, the whole system is called "open and flexible embedded system applied to positioning and telecontrol" (OFESAPO). Its hardware is based on open core hardware, and it uses a standard telecontrol protocol, i.e., International Electrotechnical Commission (IEC) 60870-5, for communication. This standard protocol is believed to be not applicable for mobile links, but this paper proves it otherwise. Therefore, OFESAPO is the first system that uses IEC-60870-5 outside its original environment. Its architecture can control more remote devices and connect to the Internet, so it is better than the systems presented in [5] and [6]. In addition, in [5], if another kind of measurement has to be done, the implemented protocol has to be changed; however, the

standard protocol used in OFESAPO makes it possible to adapt it to many scenarios depending on the user's needs. Another benefit of OFESAPO, as stated in this paper, is that although a web service for sending data is an option, the amount of extra data required could prevent the operation in real time, so the IEC protocol is better.

This paper is organized as follows: First, the system design for positioning and telecontrol is presented. The software development is described in Sections III and IV. A set of tests in a simulation environment and real environment is shown in Section V. A discussion follows in Section VI, and finally some conclusions are presented.

II. System Design

OFESAPO is composed of "remote terminal units" (RTUs) and a "control center" (CC). The CC is a PC-based system, where the GIS to map in real time is placed. The RTU is an embedded system used to acquire position data to send them to the CC.

The RTU can also send data obtained from the environment via sensors attached to the RTU or receives commands from the CC due to the design criterion flexibility of OFESAPO. That criterion makes OFESAPO similar to a supervisory control and data acquisition (SCADA) one.

The field of SCADA systems has a series of standards specified by the IEC called "IEC 60870-5" for telecontrol equipment and systems. These standards define specific requirements and conditions for data transmission in telecontrol systems, showing the ways to meet those requirements to ensure compatibility between devices from different suppliers. The positioning system presented in this paper complies with these standards, so this makes OFESAPO flexible and adaptable to multiple environments.

To determine the RTU position, the GPS technology is used as it is more flexible than other outdoor positioning systems, so it is both not dependent of the cellular telephony operator and more precise.

As OFESAPO, in its first application, is intended to be used in surveillance applications, like forest fire fight, only the following mobile communication technologies have been considered, as justified in [12]. Both GSM and GPRS are used, where a cellular telephony operator has range. Radio equipment in the UHF/VHF band are used in places where GSM or GPRS is not an option because there is no coverage or it is too expensive.

The next sections show the open hardware used in RTU, the protocol characteristics, and the whole OFESAPO architecture.

A. RTU Hardware Platform

Recent developments in field-programmable gate arrays (FPGAs) have made it possible to implement different architectures resembling a standard workstation (a PC, for example). One of these architectures has led to the specification of LEON [7], a SPARC-based processor, which can be implemented on an XILINX Spartan-3 FPGA. The main advantage of LEON is that it can be developed using standard development tools to save development time. Furthermore, a Linux OS can be installed on LEON, so the embedded system can be used as if it was a standard Linux-based PC [8].

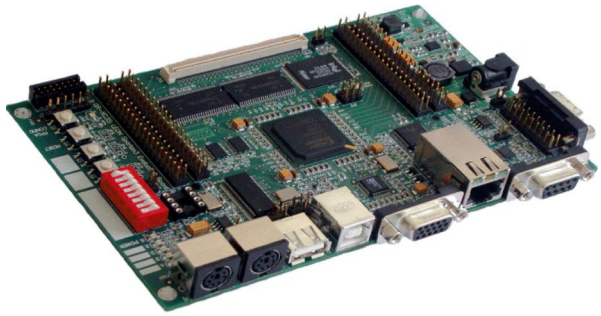


Fig. 1. Development board used for RTU prototype.

The RTU prototype has been developed in the GR-XC3S-1500 board (Fig. 1) by a joint venture between Gaisler Research [7] and Pender Electronic Design [16]. This development board incorporates a Xilinx Spartan3-1500 field programmable gate array, and it is capable of operating standalone from a single +5-V power supply. It also incorporates onboard volatile and nonvolatile memories, together with serial, Ethernet, video, and Universal Serial Bus interfaces, making this board ideal for fast prototyping, evaluation, and development of software for applications based on the LEON microprocessor (open core hardware).

Expansion of a user's peripherals and circuits can be made either using expansion connectors (40 pins) to implement a user-defined mezzanine board or via ribbon cable connections. A specific connector is provided to allow connection to the standard memory bus signals. The RTU prototype uses expansion connectors to connect an 8-GB compact flash card because Linux distribution required more memory than available on chip.

B. IEC 60870-5 Series Overview

The telecontrol protocol stack usually implements the specifications provided by the IEC called "IEC 60870-5" [9]. This series follows the enhanced protocol architecture model, not the ISO standard (open system interconnection (OSI) model). This model only requires three of the seven OSI layers, i.e., application layer, data link layer, and physical layer. The standard is divided into six parts and specifies a suite of protocols for both application and data link layers.

In a typical telecontrol scenario, one station (primary station) called "CC" controls communication with other stations (secondary stations) called "RTUs," so the IEC 60870-5 specification allows real-time telecontrol applications to function. In this sense, the series defines a set of functions called "profiles" that performs standard procedures for telecontrol systems. Not all the implementations perform the same functions, so an application profile must be described depending on the telecontrol system functionality.

The application profile required for OFESAPO only needs the application function initialization and data acquisition by polling.

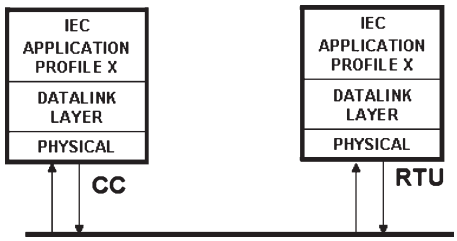


Fig. 2. Permanently connected RTU and CC.

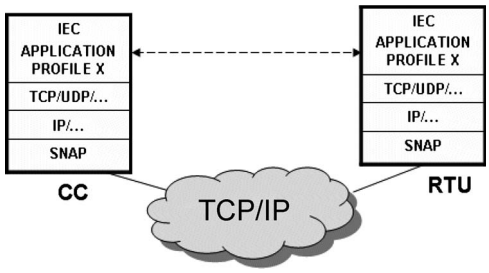


Fig. 3. RTU and CC connected via Internet.

Two different scenarios are possible in the IEC standard, depending on whether the CC and the RTUs are permanently connected, as shown in Fig. 2, or connected via Internet using transmission control protocol (TCP)/IP architecture, as shown in Fig. 3. For the first scenario (Fig. 2), all the protocols described in the standard have to be implemented, including the IEC 60870-5-5 [10] (part 5, application functions), because both CC and RTU share the transmission channel and use the same communication technology. For the second scenario (Fig. 3), only IEC 60870-5-5 [10] and IEC 60870-5-104 [11] have to be implemented because the other layers are available on most OSs (TCP/IP stack, the Internet protocols). In this case, the CC and the RTU can use different communication technologies.

There is no application service data unit (ASDU) [21] to send position data in the IEC standard; it requires a 32-bit accuracy, whereas the ASDUs defined in the IEC standard only have 16-bit precision. All the ASDUs have a field called "type identification" (TI), which is a byte field, to make out with the different types of ASDU. The standard allows the definition of new ASDUs, the only restriction being imposed is that the TI field must be set between 136 and 255. The new ASDU defined is number 201, and it includes all position data required, as shown in Section III-A, so OFESAPO improves the IEC standard.

C. System Architecture

The OFESAPO architecture is shown in Fig. 4. The GPS module and the mobile communication modules are connected to the RTU via RS-232 ports. The RTU is configurable, so it is also possible to include other kinds of measurements made, for instance, from a temperature sensor or others. The OS installed in the RTU is Linux, as explained before, so it is easy to develop high-level open software.

The CC is a Linux PC with a communication technology, mobile or otherwise, and a GIS for mapping.

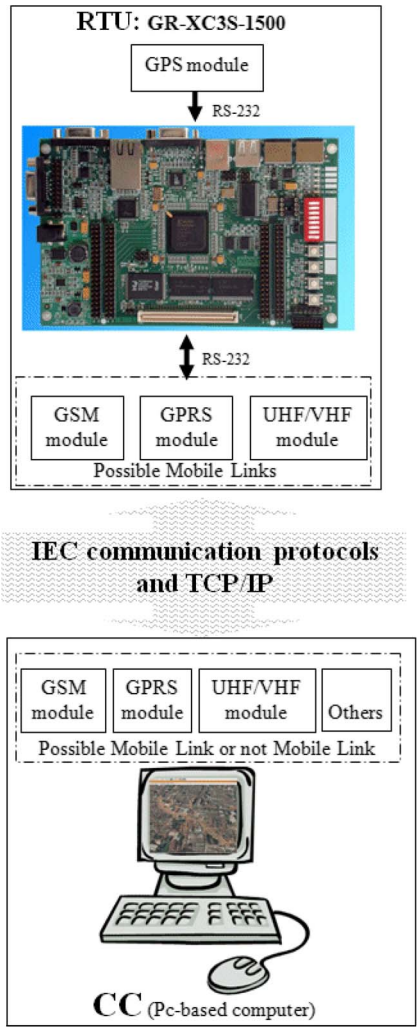


Fig. 4. OFESAPO architecture.

Depending on the mobile communication technology used in the RTU, the required IEC protocols are different, as shown before in Section II-B. If the GSM or the GPRS module is used, then the scenario is similar to that shown in Fig. 3; in this case, the CC is connected to the Internet via the GSM or the GPRS, as the RTUs are connected, or using other communication technologies. However, if the mobile communication is a radio equipment in the UHF/VHF band, the scenario is that shown in Fig. 2, where the CC also requires the same radio equipment.

Hence, the system architecture enables it to be applied in many kinds of LBSs, depending on the user's needs. Because the two possible scenarios require different software designs and tests, this paper only focuses on that shown in Fig. 3. Some researches for the second one are presented in [13] and [14].

III. RTU Software Design

The RTU software is divided into two main processes that run in parallel:

1) Acquisition: This process is in charge of getting data without loss from an external device. In this case, data are geographical coordinates from the GPS module.

Buffer[0]	Buffer[0,0]	$GPGGA	Buffer[1]	Buffer[1,0]	$GPGGA
	Buffer[0,1]	$GPGSA		Buffer[1,1]	$GPGSA
	Buffer[0,2]	$GPGSV		Buffer[1,2]	$GPGSV
	Buffer[0,3]	$GPGSV		Buffer[1,3]	$GPGSV
	Buffer[0,4]	$GPGSV		Buffer[1,4]	$GPGSV
	Buffer[0,5]	$PGRME		Buffer[1,5]	$PGRME
	Buffer[0,6]	$GPGLL		Buffer[1,6]	$GPGLL
	Buffer[0,7]	$GPVTG		Buffer[1,7]	$GPVTG
	Buffer[0,8]	$PGRMV		Buffer[1,8]	$PGRMV
	Buffer[0,9]	$PGRMF		Buffer[1,9]	$PGRMF
	Buffer[0,10]	$PGRMB		Buffer[1,10]	$PGRMB
	Buffer[0,11]	$PGRMM		Buffer[1,11]	$PGRMM
	Buffer[0,12]	$PGRMC		Buffer[1,12]	$PGRMC
Buffer[2]	Buffer[2,0]	$GPGGA	Buffer[3]	Buffer[3,0]	$GPGGA
	Buffer[2,1]	$GPGSA		Buffer[3,1]	$GPGSA
	Buffer[2,2]	$GPGSV		Buffer[3,2]	$GPGSV
	Buffer[2,3]	$GPGSV		Buffer[3,3]	$GPGSV
	Buffer[2,4]	$GPGSV		Buffer[3,4]	$GPGSV
	Buffer[2,5]	$PGRME		Buffer[3,5]	$PGRME
	Buffer[2,6]	$GPGLL		Buffer[3,6]	$GPGLL
	Buffer[2,7]	$GPVTG		Buffer[3,7]	$GPVTG
	Buffer[2,8]	$PGRMV		Buffer[3,8]	$PGRMV
	Buffer[2,9]	$PGRMF		Buffer[3,9]	$PGRMF
	Buffer[2,10]	$PGRMB		Buffer[3,10]	$PGRMB
	Buffer[2,11]	$PGRMM		Buffer[3,11]	$PGRMM
	Buffer[2,12]	$PGRMC		Buffer[3,12]	$PGRMC

Fig. 5. Buffer array. This figure shows the four buffers that have the GPS information. There are 13 rows, each one containing specific data that are labeled adequately. There is an index that points to the right buffer to be written. For instance, if the index is set to 1, the Buffer[0] contains the last information acquired from the GPS, so the transmission control process reads this buffer, and the acquisition process writes in Buffer[1]. Only the $GPGGA positioning information label is required to know the exact position of the RTU, so this is the only information sent as required by the CC in the new ASDU defined. The other labels are saved for future application of OFESAPO.

2) Transmission Control: This process is in charge of RTU initialization and data transmission to the CC according to the IEC protocol.

A. Acquisition Process

The GPS module is connected to the RTU via an RS-232 port, as shown in Fig. 4. The transmission speed of the RS-232 port is set at 9600 b/s, which is the maximum speed that can be configured, as data are lost at lower speeds.

The GPS position data are sent from the GPS as a text string of 13 lines. Each line represents different kinds of data, some defined by The National Marine Electronics Association and others by the GPS manufacture; in the case of OFESAPO, it is Garmin [23]. Each string line is labeled to interpret correctly the data that are received by the GPS.

The GPS receives a new position data, i.e., the thirteen lines, every second, and they are saved in a buffer because one RTU only sends data to the CC when it is polled. Therefore, the transmission control process has to read this buffer when the CC polls the RTU to get the information required in the position ASDU. As it is well known, there is a problem when two or more processes share a variable. One process can be writing the variable when the other is reading it, and then the read variable is incorrect. In the literature, there are some solutions to this problem. The typical one is using semaphores, but in this case, they are not a solution because the acquisition process is blocked when the transmission control process is reading the variable, so the acquisition process could not write.

The implemented solution consists of a buffer array (Fig. 5), where a buffer is composed of thirteen rows, each one containing the corresponding GPS line that it is labeled adequately for correct interpretation. These buffers are in a shared memory so that both processes can access them. There is also a shared index to the buffer array. This index indicates where the acquisition process writes the data in a specific buffer. To avoid reading and writing simultaneously the same buffer and data loss, the transmission control process always reads the previously indexed buffer. As both processes shared the index variable, some mismatches could appear. To prevent such occurrences, four buffers have been used, although three buffers should have been enough. This way, it is impossible that the acquisition process writes the same buffer that the transmission control process reads.

There is also another problem because each minute the GPS sends a status line, i.e., the Garmin proprietary information, which causes loss of synchronization. This scenario is detected because the last line of the buffer does not have the right label, i.e., buffer[X,12] is not $PGRMC. If this occurs, then the index does not increase, so the buffer is written twice to solve the problem.

Selection of application functions from IEC 60870-5-5	Initialization	**User Process**
Selection of ASDUs from IEC 60870-5-104		**Application Layer**
APCI (Application Protocol Control Information)		
Selection of TCP/IP protocol suite		**Transport Layer**
		Network Layer
		Link Layer
		Physical Layer

Fig. 6. IEC 60870-5-104. Protocol structure.

B. Transmission Control Process

This process is in charge of RTU initialization and sending to the CC the requested data that have been previously polled.

Before any data exchange can take place, a previous initialization is required. The RTU initialization consists of setting local variables to their initial state and listening in a passive mode to a connection request sent by the CC. Once connection between the CC and the RTU has been established, i.e., a TCP connection, the transmission control process behaves as described in the IEC 60870-5-104 and IEC 60870-5-5 specifications. The protocol structure is shown in Fig. 6; the IEC protocol uses the services provided by the transport layer implemented by the TCP protocol in the TCP/IP architecture.

IV. CC Software Design

The CC is also divided into two processes that also run in parallel:

1) Transmission control: This process is required to get both the CC and the RTU to an initial state and to get different data types, as described in Section III-B; from the RTUs, according to a preestablished configuration, the application profile is predefined.
2) Geographic data process: This process gets the collected data from an RTU and writes them to an extensible markup language (XML) file (GPS eXchange (GPX) schema [15]), which is sent by streaming to a remote or a

```
<!-- latitude and longitude -->
<wpt lat="eee.dddddd" lon="eee.dddddd">

    <!-- elevation -->
    <ele>eeee.dddddd</ele>

    <!--  year-month-dayThour:minutes:secondsZ of Data -->
    <time>aaaa-mm-ddThh:mm:ssZ</time>

    <!-- Data extension -->
    <extensions>
        <ext_data>
            <name>NAME_DISP</name>
            <id>ID_DISP</id>
            <value>VALUE</value>
        </ext_data>
    </extensions>

</wpt>
<!-- end -->
```

Fig. 7. XML file in GPX scheme. This scheme includes waypoint, routes, and track. In this example, only a waypoint is shown ($\langle$wpt$\rangle \ldots \langle/$wpt$\rangle$).

local web server. This server represents the position data received in an interactive map using the Google Maps application programming interface (API).

A. Transmission Control Process

This process sets the general configuration parameters as the working directories, the number of RTUs to be controlled, and the connection parameters (IP address, TCP port, and physical medium) of each RTU as well as others. It also boots and allocates memory for the different resources (log files, data structured, ASDUs, etc.).

Depending on the configuration parameters, the CC checks that all the configured RTUs are able to establish a TCP connection—each RTU is always previously initialized in listening passive mode; once the connection is established, the initialization process ends.

Another task made in this process is getting position data from RTU, for doing so, each configured RTU is polled periodically by the CC, as described in the IEC standard.

B. Geographic Data Process

The collected position data received from an RTU are processed according to the GPX scheme and saved in an XML file. This XML file is sent by streaming to a web service in a remote or a local machine, making the system more flexible. This web service uses the received information to represent in real time the geographic position of the RTU through the Google Maps API.

The scheme GPX is a lightweight XML data format for the interchange of GPS data (waypoints, routes, and tracks) between applications and web service on the Internet. The GPX defines a common set of data tags for describing GPS and geographic data in XML, as shown in Fig. 7.

A question one might raise is that if the RTU has the TCP/IP stack running over it, why not use the web service in the RTU instead of using the IEC 670870-5 protocol to send position data. There are three answers to this question. The first answer

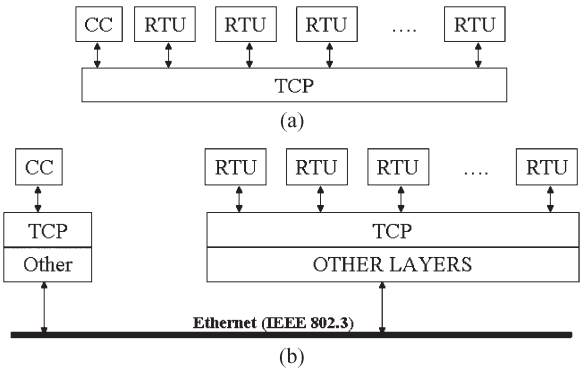


Fig. 8. Test environment.

is that the web service requires a lot of additional information (not only position data); if the link channel has a low speed, then the time required to send it could prevent it from using OFESAPO in real time, as shown in Section V-A. Further, if the RTU has also any kind of sensors, then it is possible to send their value associated to the actual position of the RTU. In the IEC standard, there are predefined ASDUs for different kinds of parameters (e.g., analog counter, digital counter, sensor measurement, etc.). The second answer is that there are some transmission channels that cannot work with the TCP/IP stack, as justified in [12], and require the development of specific protocol, as explained in [13]. The last answer is that with the IEC standard, the RTU can be applied to applications other than positioning, for instance, a typical RTU in a power utility.

V. System Testing

A set of tests has been made to check that OFESAPO works correctly and can be adapted to different scenarios. Some tests have been made in a real environment, where there were some moving RTUs sending their position data to the CC that showed them in Google Map in real time. Others have been made in a simulation environment to determine the system limitations.

As OFESAPO is intended to be applied in surveillance applications, the time requirements for real time are from 5 to 10 s, but if the system could guarantee a time from 1 to 5 s, then it could be applied in control applications, as shown in [5].

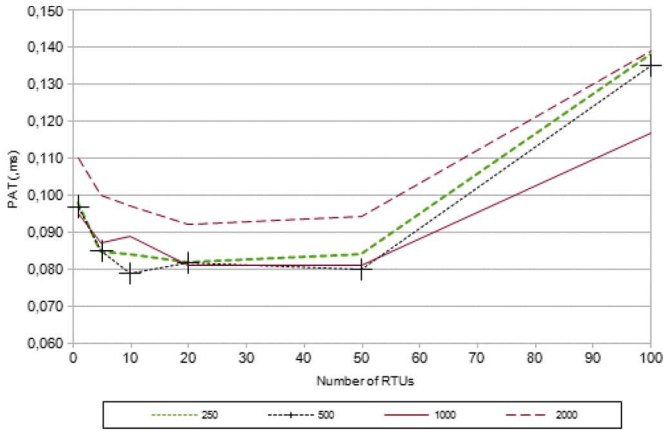
The following sections describe the two sets of tests.

A. Simulation Testing

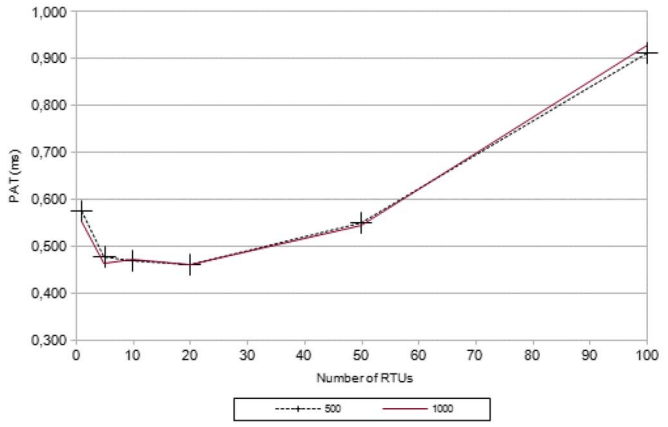
Two scenarios focusing on the protocol performance have been tested to measure the response time of OFESAPO.

Protocol engineering techniques have been used to measure the OFESAPO performance, as they are described in [22]. One of the measurements made consisted of having $n + 1$ processes, all executed in a single computer that behaves like n RTUs and one of them that behaves like a CC. All processes use TCP services for communication, as required in IEC-60870-5-104 [Fig. 8(a)]. In this case, the transmission channel was simulated.

The other measurements have been like that explained earlier, but in this case, the transmission channel was not simulated. The process that behaves like the CC was in one computer, and the n process that behaves like an RTU in



(a)



(b)

Fig. 9. PAT and number of RTUs. These two graphics show how the PAT changes, depending on the number of RTUs using (a) one or (b) two computers, for a total number of 100 polls. The waiting time between the polling of all RTUs and the next polling has also been changed between 250 and 2000 ms in the same computer, and only 500 and 1000 in different computers. The differences between tests in one or two computers are due to transmission delays, in this case a Fastethernet link (100 Mb/s). (a) CC and RTUs in the same computer. (b) CC and RTUs in a different computer.

the other [Fig. 8(b)]. The IEEE 802.3 link (Fastethernet) has been tested to check the OFESAPO behavior using a real communication channel.

Fig. 9. shows how the poll answer time (PAT) changes, depending on the number of RTUs, using one (a) or two computers (b), for a total number of 100 polls, and by simulating different waiting times between two polls. As shown in Fig. 9, if OFESAPO is used to control up to 100 RTUs (a very high number, the PAT), the mean time required to receive an answer from all the RTUs is about 0.14 and 0.9 ms for the simulated channel and for Ethernet, respectively. This is sufficient for the two possible real-time applications, i.e., surveillance and telecontrol, as the minimum time required for the last one is 1 s. The real transmission channel tested does not introduce much delay, the transmission speed is 100 Mb/s, and the transmission delay is almost 0, so the previous claim also depends on the channel characteristics. If the channel operates at a lower speed and introduces a higher delay, then the PAT parameter increases its value, and the number of RTUs has to be reduced. For instance, for ten RTUs, PAT is about 0.1 ms [Fig. 9(a)] with a simulated transmission channel. Hence, to use OFESAPO in real time using a real transmission channel, the transmission delay must be bounded between 0.1 ms and 1 s.

Another important aspect to consider is the number of bytes sent, i.e., the protocol data unit (PDU) size, because it limits the amount of data to be sent. OFESAPO PDU requires only 35 B, but the IEC 60870-5-104 allows a maximum of 255 B. If OFESAPO PDU is increased, then all parameters measured, as expected, increase linearly, but for a lower-speed channel, this increment should prevent real-time operations. For instance, if the web service had been used from the RTUs and the CC instead of the IEC protocol, each position datum would require more than nine times the amount bytes sent in OFESAPO PDU. Hence, it takes longer to send the data, and the requirement of 1 s could not be fulfilled.

B. Field Testing

The RTU prototype, i.e., LEON, has also been tested only with two of the three transmission channels available in the RTU, viz., GSM and GPRS (see Fig. 4), because RF does not work with TCP/IP, as justified in [12].

A GSM/GPRS modem has been used, specifically a Wavecom Fastrack M1306B [17]. This modem behaves as a standard AT-command modem via an RS232 port, so the modem is connected to the RTU this way. The device's specification allows data transmission of up to 14.400 b/s for GSM and 115.200 b/s for GPRS, but this feature is dependent on the GSM/GPRS operator used, so it might not always be available. In this test with a Spanish mobile telephony operator, GSM operates up to 9600 b/s, and GPRS operates up to 38.400 b/s.

The transmission delay measured (time to send data + time for processing + link delay) is below 750 ms for GPRS and below 1000 ms for GSM, which is adequate for surveillance. The CC can poll simultaneously all the RTUs using GPRS, so the delay is 764 ms (750 ms + 0.14 of PAT, as explained in Section V-A). However, each RTU has to be individually polled in GSM, so the number of RTUs is limited in this technology to ten in surveillance applications and five in telecontrol applications.

For field testing, the following scenario has been made. A CC has been connected to the Internet via a Fastethernet link to show in Google Map two RTUs that were moving in real time. One of them was the RTU based on LEON, and the other was a laptop; the same source code has been used in both cases. The Google Map interface in the CC with the two moving RTUs is shown in Fig. 10. The CC was placed in a laboratory at our department facilities, while each RTU was placed in a different car. This solution guarantees larger and easier to appreciate position changes with respect to those achievable by moving the RTUs at walking speed. The transmission channel used was GPRS for both RTUs. The results were as expected; OFESAPO worked correctly. A video with the test can be downloaded in [18].

VI. Discussion

OFESAPO is a system whose first application has been related to position, but it is also open to other applications.

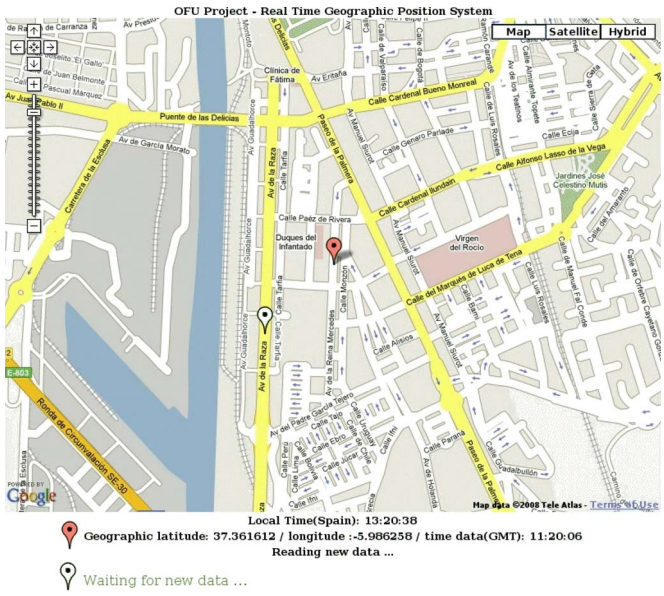


Fig. 10. OFESAPO test. Point red and point white are the two moving RTUs connected to the Internet via a GPRS channel, one RTU with LEON hardware and the other RTU with a laptop both controlled by a CC that was placed in our laboratory.

As presented in this paper, its response time is sufficient for surveillance and telecontrol. Therefore, OFESAPO is a real-time telecontrol system that can be used as a communication interface among a CC, a controller unit, and one-to-many RTUs (controlled units). The CC can receive measurement data from any of the RTUs and generate the appropriate command to control the RTUs. The IEC standard defines how to send all kinds of measurements (digital counter, analog counter, analog sensor, etc.) and how to generate any command. Therefore, OFESAPO is open to any telecontrol application. As presented here, OFESAPO can control RTUs that are connected to the Internet or directly connected via a multipoint link, so the telecontrol architecture can be expanded all over the world, and the RTUs can be placed anywhere provided with an Internet link. As the CC sends the information received from RTUs to a web service, it allows having different levels of hierarchy, for instance, there could be a CC in a town that sends the information received from RTUs to a CC in a city and so on. Another important point of OFESAPO is that the same source code used in the embedded system (LEON) is used in a laptop or PC, so all the software development time is reduced, and it is spent only once.

One study in [5] describes a system like OFESAPO, but OFESAPO improves on its limitations, which are as follows:

1) Only two units are allowed, i.e., the CC and the RTU.
2) The software developed in the RTU cannot be reused as it is microcontroller dependent.
3) It does not allow RTUs connected to the Internet. It requires a point-to-point link, and only GSM has been tested.
4) No standard protocol has been used. A protocol has been developed specifically for this application.
5) If further measurements are required, then the protocol has to be changed.
6) The GIS used requires a license.

VII. Conclusion

This paper has traced the OFESAPO development through real-time positioning and telecontrol systems. Although real-time positioning and telecontrol systems are also shown in [5] and [6], OFESAPO differs from them in that it is based on both open hardware and software, and it allows more RTUs to be controlled. The hardware in OFESAPO RTU is an embedded system, i.e., an SoC-type design using FPGA that has been programmed with the open core LEON with a Linux OS running over it. For communication and control between a CC and RTUs, the IEC 60870-5 standard over TCP/IP has been used. All the software has been developed and tested in a PC with Linux OS using standard tools, so there is no requirement to use specific software design tools for embedded systems, as shown in [19] and [20]. This software has run without any problem in the RTU based in LEON.

For GIS, the Google Map API has been used, which is an easy way to embed Google Map in any web page with JavaScript. The Maps API is a free service, available for any web site that is free to consumers, so no license is required, unlike the GIS used in Lin et al.'s paper [5]. Moreover, as the communication protocol is not one designed specifically, OFESAPO could be applied to other scenarios and not just related to position; it is a more flexible system.

The IEC standard protocol was not developed originally for use in mobile communication, but this paper shows that it is possible to apply it in other kinds of links too. The open source software of the IEC standard protocol has also been developed as there was none until now. Furthermore, a new ASDU (number 201) has been defined to send position data in the IEC standard protocol, so the standard has also been improved.

Tests show that OFESAPO could be used in real time for control applications (time requirements between 1 and 5 s) and surveillance applications (time requirements between 1 and 10 s).

Although a web service (http protocol and Internet standard) for communication among RTUs and a CC could be an option, as shown in this paper, it increases more than nine times the number of bytes required to be sent for position, so if the link is low speed and high delay, it could prevent using OFESAPO in real time. Further, if the RTU also has any sensor or actuator, it is possible to send their value describing the actual position of the RTU and to receive command from the CC.

Another problem is that there are some transmission channels that cannot work with the TCP/IP stack [12], no web service could be used, and the development of a specific protocol is required [13]. Therefore, the IEC protocol is an option because it has a data link layer protocol. Finally, using the IEC standard protocol makes it possible to use OFESAPO not only for position applications but also in telecontrol applications, like in power utilities.

REFERENCES

[1] L. E. Martínez Gens and M. U. de las Heras, "Tecnologías de Localización y Posicionamiento para Servicios Basados en Localización (LBS)," *Bit*, no. 154, pp. 68–70, 2006.

[2] S. Han, H. Lim, and J. Lee, "An efficient localization scheme for a differential-driving mobile robot based on RFID system," *IEEE Trans. Ind. Electron.*, vol. 54, no. 6, pp. 3362–3369, Dec. 2007.

[3] O-H. Choi, J. Kim, J.-E. Lim, and D.-K. Baik, "A design of location information management system in positioning systems," in *Proc. Int. Conf. Convergence Inf. Technol.*, Gyeongju, Korea, 2007, pp. 114–120.

[4] H. Saarnisaari, "Some design aspects of mobile local positioning sys-tems," in *Proc. PLANS*, 2004, pp. 300–309.

[5] C. E. Lin, C.-C. Li, A.-S. Hou, and C.-C. Wu, "A real-time remote control architecture using mobile communication," *IEEE Trans. Instrum. Meas.*, vol. 52, no. 4, pp. 997–1003, Aug. 2003.

[6] G. Yongqiang, F. Kangling, and N. Zedong, "Design of portable personal positioning system based on SoC technology," in *Proc. 8th ICEMI*, 2007, pp. 1-707–1-711.

[7] *GRLIB/LEON3 Manual*. [Online]. Available: http://www.gaisler.com

[8] A. Muñoz, E. Ostúa, P. Ruiz, M. J. Bellido, J. Viejo, A. Millan, J. Juan, and D. Guerrero, "Un ejemplo de implementación de una distribución Linux en un SoC basado en hardware Linux," in *Proc. Actas de las IV JCRA*, Sep. 2007, pp. 85–92.

[9] *Telecontrol Equipment and Systems. Part 5: Transmission Protocols*, IEC-60870-5, six parts. First Edition, Apr. 1992. [Online]. Available: http://www.iec.ch

[10] *Telecontrol Equipment and Systems. Part 5: Transmission Protocols. Section 5: Basic Application Functions*, IEC-60870-5-5. First Edition, Jun. 1995. [Online]. Available: http://www.iec.ch

[11] *Telecontrol Equipment and Systems. Part 5: Section 104: Network Access for IEC-60870-5-101 Using Standard Transport Profiles*, IEC-60870-5-104. First Edition, Dec. 2000. [Online]. Available: http://www.iec.ch

[12] J. Benjumea, V. Medina, I. Gómez, E. Dorronzoro, G. Sánchez, and S. Martín, "Choosing the right protocol stack for an open and flexible remote unit," in *Proc. IEEE ISIE*, Cambridge, U.K., 2008, pp. 1668–1673.

[13] E. D. Zubiete, I. M. Gómez González, A. V. Medina Rodríguez, J. B. Mondéjar, G. Sánchez Antón, and S. M. Guillén, "Abstract imple-menting IEC 60870-5 data link layer for an open and flexible remote unit," in *Proc. 34th IEEE IECON*, Orlando, FL, 2008, p. 268.

[14] E. D. Zubiete, I. M. Gómez González, A. V. Medina Rodríguez, G. Sánchez Antón, and S. M. Guillén, "Implementing IEC 60870-5 data link layer for an open and flexible remote unit," in *Proc. 34th IEEE IECON*, Orlando, FL, 2008, pp. 2471–2476.

[15] define the GPX schema. [Online]. Available: http://www.topografix.com/

[16] [Online]. Available: http://www.pender.ch/index.shtml

[17] Wavecom Fastrack 1306M User Manual. [Online]. Available: http://www. sendsms.cn/download/Fastrack_M1306B_User_Guide_rev003.pdf

[18] [Online]. Available: http://matrix.dte.us.es/videos/

[19] H. O. Almedia, L. Dias da Silva, E. Oliveira, and A. Perkusich, "A formal approach for component based embedded software modelling and analysis," in *Proc. IEEE ISIE*, Dubronik, Croatia, 2005, pp. 1337–1342.

[20] C. Petitjean, J. P. Lauffenburger, J. M. Perronne, and M. Basset, "A unified software architecture for embedded systems," in *Proc. IEEE ISIE*, Ajaccio, France, 2004, vol. 1, pp. 565–570.

[21] V. Medina, I. Gómez, D. Oviedo, E. Dorronzoro, S. Martín, J. Benjumea, and G. Sánchez, "IEC-60870-5 application layer over TCP/IP for an open and flexible remote unit," in *Proc. IEEE ISIE*, Seoul, Korea, 2009, pp. 420–425.

[22] V. Medina, I. Gómez, J. Luque, and S. Martín, "ESTELLE:A method to analyze automatically the performance of telecontrol protocol in SCADA systems," *IEEE Trans. Power Del.*, vol. 17, no. 3, pp. 712–717, Jul. 2002.

[23] [Online]. Available: http://www.garmin.com/es/