8th International Conference on DEVELOPMENT AND APPLICATION SYSTEMS
Suceava, Romania, May 25 – 27, 2006

RFID GATES FOR ACCESS CONTROL

Vasile GĂITAN [1], Cătălin SÎICU [2], Alexandru LARIONESCU [3], Valentin POPA[4]
[1,2,3,4] *"Stefan cel Mare" University of Suceava*
str.Universitatii nr.13, RO-720229 Suceava
[1]*gaitan@eed.usv.ro,* [2]*siicu@eed.usv.ro,* [3]*lari@eed.usv.ro,* [4]*valentin@eed.usv.ro*

Abstract. *This paper proposes an implementation of access control gates in a distributed administration system, based on RFID technologies (RFID tags). Within this administration system, we emphasize the independent stations controlling the access through different gates, as well as the architectures of the involved hardware and software part.*
Keywords: *access control, RFID gates, RFID tag, OPC, IEEE 1745.*

Introduction

The nowadays security requirements shows an accentuate growing trend, coming up from various domains, starting even from the simplest software and/or hardware systems, arriving to the big branches of industry, banks, transports etc. A noticeable example is the requirement to implement a rigorously yet flexible checking process for boundaries access points; e.g., to limit or deny the access based on the person's license, or to easily monitor and manage the items' inputs and outputs in storehouses or through industrial halls' gates.
Before proceeding to the proposed solution, let's remind the structure of an integrated system for the enterprise processes information management, with an emphasis on the production process and the matters control. The industry continuously evolution brings up new smart field technologies, targeting the industrial processes management. The "field" technologies [1] include various electrical, hydraulic and pneumatic appliances, forming the basic level for the processes management. The information pertaining to the process status and its control are transferred toward various process management levels through analog and digital signals entities. This information is in stochastic form at the process management lower levels and tends to be increasingly determinist as it is passed up to the higher levels; besides, the reacting time grows simultaneously. The conventional hierarchy about the processes management is depicted in figure 1.

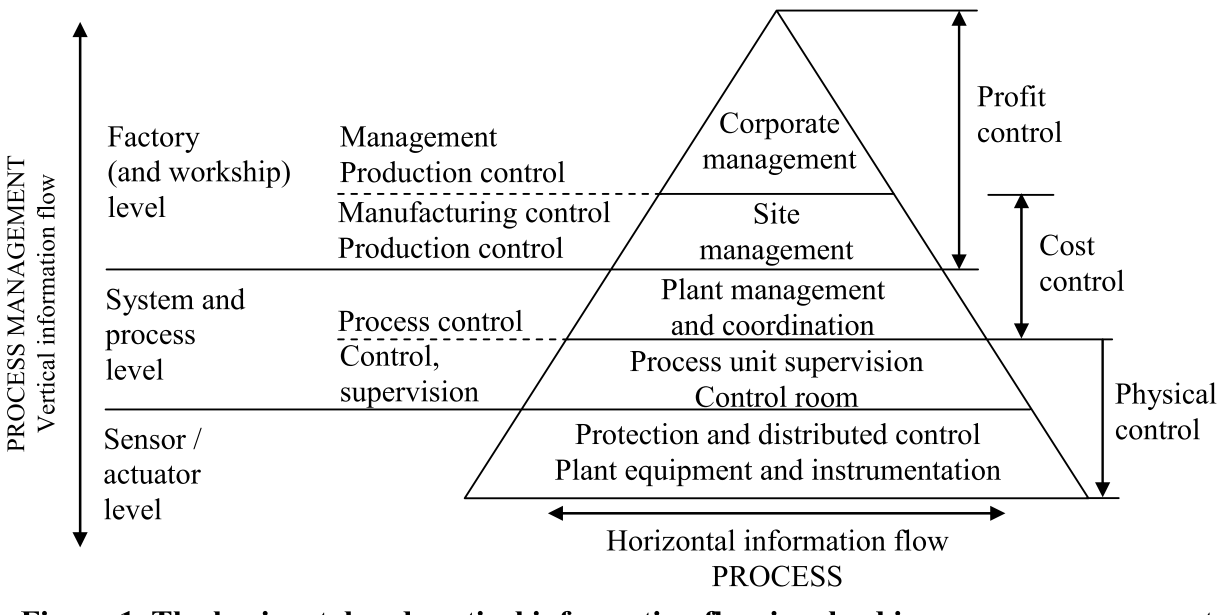


Figure 1. The horizontal and vertical information flow involved in process management.

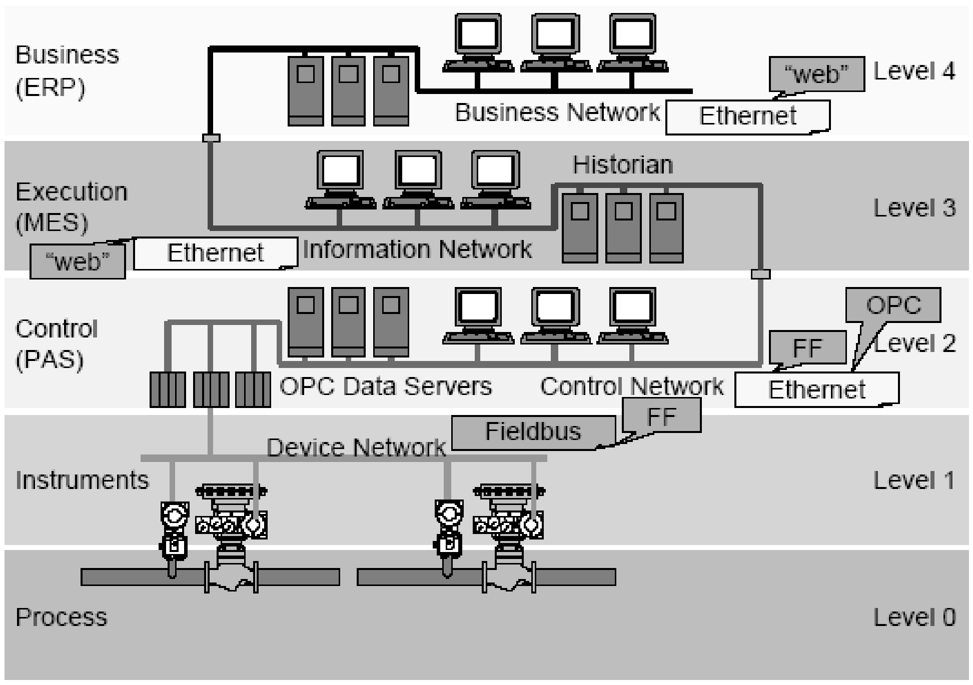


Figure 2. Hierarchical levels, communication media, network protocols.

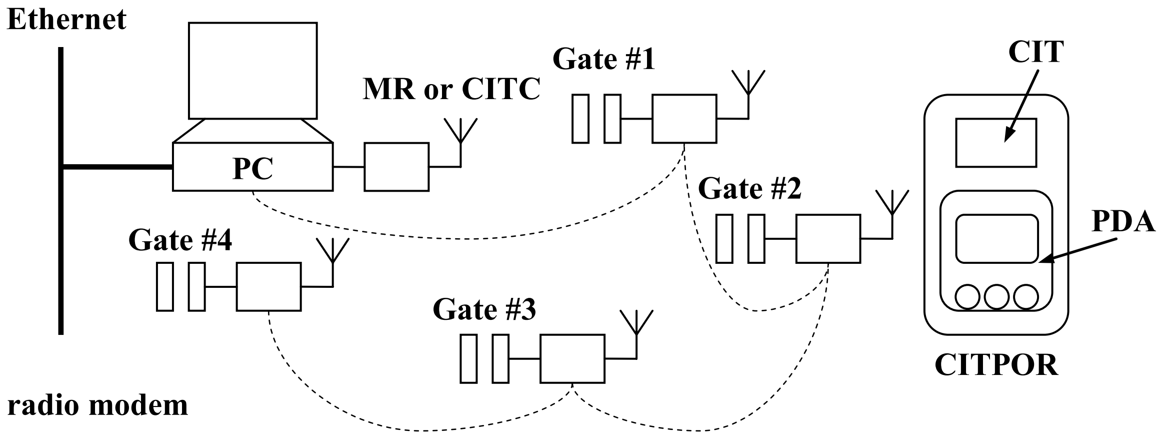
The study [2] presents an example of five levels architecture, with details regarding the applications places, from the user of Foundation Fieldbus H1 and HSE point of view. The figure 2 depicts this architecture.

The Foundation Fieldbus is a compact architecture (a framework) for information integration, including a field bus, a host network (often based on the Ethernet standard) and a high level aiming to supply the data to wired OPC applications. Those three technologies (H1, HSE and OPC) are complementary to each other, they don't compete within their scopes, and present some mutual excluded features; they operate on distinctive levels of the control system architecture. The field networks are employed in field level instrumentation. The Ethernet is the stem (the base network) for the automation systems, but it can be employed by the higher levels of enterprise. The OPC technology acts in software to software communications, as long as the field networks and Ethernet aim hardware to hardware communications.

The architecture of the integrated system for the information management

A possible solution for the access control, as a system, is described by this paper. The system aims to control the access within boundaries of various persons, goods or any other items. It uses RFID (transponders) in the identification procedure. The main clients of those systems could be located in: companies (staff and workforce access, tally management), banks (restricted access inside high risky locations, valuables exploitation security), public institutions (scheduled access within buildings, rooms or compartments), appliances or goods storehouses (input and output management), transports (easy check in or payment, statistical studies), etc.

In the hierarchical model of enterprises [3], the information flows in a horizontal and vertical manner, at the same time. Thus, the proposed system should allow the connection to an industrial network (wired or wireless) in order to acquire the horizontal information by using the transponders technology; the same time, it should accomplish the vertical functionalities, as the information need to arrive to the upper levels of the enterprise. As the needs can grow with new check points, the system should be as modular as possible, easily expandable. Accordingly with those requirements, we propose the architecture presented in figure 3.



MR – radio modem

CITC – transponder reader / writer

CITPOR – portable transponder reader / writer

PDA – Personal Digital Assistant

Figure 3. The integrated system structure for information and process management.

The system contains a host computer, having the role to collect the arriving data from fixed embedded systems CITC (by the way of wired or wireless communication), to create a local database and to promote (export) these data toward the enterprise upper levels for any subsequent processing. At the enterprise's lower levels, many such systems can be found; they can be interconnected. The gates represent the points where the transponders, attached to different items, are read along the manufacturing process. These transponders are like antennas, whose size depends on the requested reading distance. The CITC devices are not moveable stations, allowing the read and write procedures on the transponders.

The hardware architecture implies the software architecture, composed as: a PC software component, a PDA software module, and multiple CITCs respectively.

From the communication point of view, the system presents a master-slave behavior, where the master is an IBM compatible PC, and the CITC embedded devices play the slave role. A transponder reader is incorporated in each station. The computer – devices communication is achieved through a RS-485 serial link.

The software components implemented on the PC side will accomplish some essential tasks: exchanging information with CICT stations using particular algorithms; acquiring data from other important manufacturers; software to software communication through OPC standard (Ole for Process Control), allowing the data transfer toward enterprise upper levels; creating PC format databases with transponders related information; creating the corresponding device profile for remote controlled transponders data.

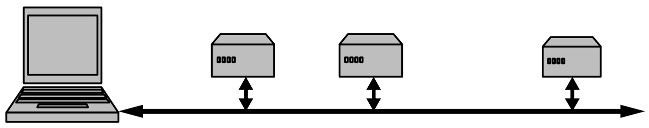


Figure 4. Bidirectional flow of data between CITC stations and PC.

Below are detailed the CICT embedded devices and the PC software modules.

The description of the access control stations

The access control stations (CITC) are constructed around the µPSD 3234A microcontroller, which is produced by ST Microelectronics. This microcontroller has the 8032 core, providing the MCS51 microcontrollers family characteristics. In addition, on chip there are two FLASH memory blocks, the main one having 256 KB and the secondary having only 32 KB, and an 8 KB SRAM memory block, as specified in [4]. The content of the SRAM

memory can be preserved if there is an external battery connected to it. The on-chip FLASH memory can be mapped either in the code memory space or in the data memory space. The microcontroller can be programmed through a JTAG interface [5].
The two serial ports of the microcontroller are used for the communication with the RFID reader and the PC. The serial communication with the RFID reader is based on the RS-232 standard and uses the communication protocol specified in the RFID reader's technical specifications [6]. It uses the first serial port of the µPSD 3234A microcontroller. The second serial port, used for the communication with the PC, allows the station to be connected in a local network based on the RS-485 serial communication standard. The protocol used for this serial communication is based on ISO 1745 standard [8]. The two serial ports have different baud rates because they use different timers to generate the communication speed.

The station provides 4 digital inputs, which can be used to connect external sensors (as an open door sensor, a proximity sensor, a smoke sensor etc.) and 8 digital outputs that can be used to connect different execution or alarming devices, such as electromagnetic door lock, alarm speaker or signaling LEDs. The signaling LEDs connected to the station can indicate a number of situations, such as: access denied, access granted, RFID tag present, RFID tag read or write etc.
For storing the prevalent or non-prevalent data logs, the station has two serial FLASH memories with the capacity of 1 MB each. The prevalent data log memory stores the prevalent events, such as the ones produced by the execution of a command received from the PC, the presence of a RFID tag or the generation of an alarm. The non-prevalent data log memory stores the non-prevalent events, such as the content of an RFID tag which entered the RFID reader's reading field.

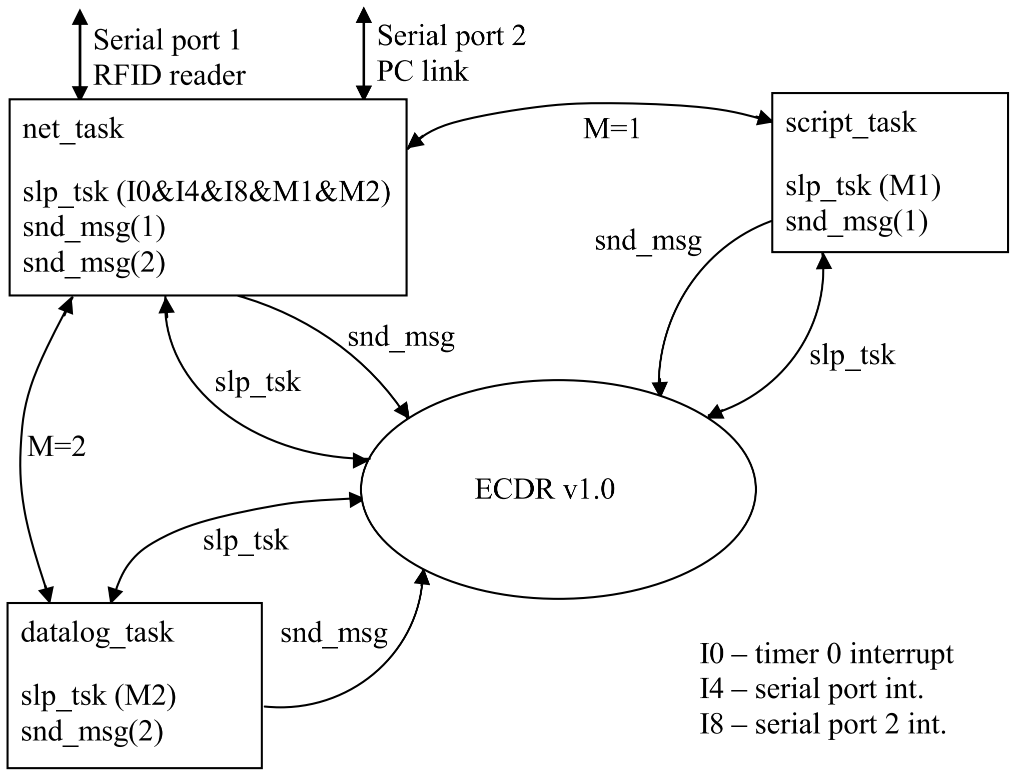


Figure 5. The kernel and the three tasks.

The software implemented on the station is based on a real-time kernel developed by Genpro S.R.L. Suceava, and has three distinctive tasks. The first task, which has the higher priority, realizes the communication with the RFID reader and the PC. Also, this task executes the commands received from the PC, reads the content of the RFID tags and authorizes the access, and generates events and alarms, if necessary. The second task executes a script that was read from the RFID tag, if it exists. The script is composed from a group of commands, which can determine the execution of some operations on the content of de RFID tag, such as incrementing or decrementing the value of a field, setting the value of a field or verifying if a condition is true. After the execution of the script, if there are some modifications made on the content of the RFID tag (some fields were changed), they will be written back on the RFID tag. The third task has the lower priority in the system and it handles the data logs. This task writes the prevalent and non-prevalent data logs in the external serial FLASH memories, searches an event in those memories, reads the events from the FLASH memories and send them to the first task, from where they will be send to the PC. The communication is always initiated by the PC. The stations are connected in a local RS-485 network. Each station has a unique address on the network. From the PC we can send different commands to the stations or we can interrogate the stations if there are new events.

The general structure of a command is presented in Table 1. The field *ID_Command* (1 byte) identifies the type of the command (e.g. setting the time on the station, changing the station's address etc.). The next field, *AutoNumber* (2 bytes), is used to identify the answer received from the station after the execution of the command. The last field, *Command_Content*, differs from one command to another. There are two types of commands: those which determine the station to confirm receiving them and those which are executed immediately and return an answer to the PC. If the command was not received correctly or it doesn't exist, the station will respond with an error code. There was adopted the following convention regarding the commands: the most significant bit (7th) of the *ID_Command* field is 0 for all the commands. Also, the 6th bit is reset (0) for the commands which determine the station to confirm receiving them, and set (1) for the commands which are executed totally or partially immediately and generate an answer for the PC.

Table 1. The structure of a command

Command field	Field type	Description
ID_Command	BYTE	The identifier of the command
AutoNumber	WORD	Number used to identify the answer of the execution of the command
Command_Content	BSTR	The command's parameters, specific for each command

Table 2. The structure of an prevalent event

Event field	Field type	Description
ID_Event	BYTE	The identifier of the event
AutoNumber	WORD	Number that identifies the command received from PC
Date	BYTE [8]	The date when the event occured
Specific_Info	BYTE [8]	Aditional infos, specific for the event

Table 3. The structure of an non-prevalent event

Event field	Field type	Description
ID_Event	BYTE	The identifier of the event
Date	BYTE[8]	The date when the event occured
TransponderSerialNo	BYTE[8]	The unique serial number of the RFID tag
Length	WORD	The length of the TransponderContent field
TransponderContent	BSTR	The content of the RFID tag

In Table 2 is presented the structure of a prevalent event. The *ID_Event* field identifies the type of the event and has the dimension of 1 byte. The *AutoNumber* field (2 bytes) will contain the AutoNumber value of the command received from the PC, if the event was generated by the execution of that command. The *Date* field (8 bytes) contains the date (year, month, day, hour, minutes and seconds) when the event occurred. The last field, *Specific_Info* (8 bytes), contains information specific to the event type. If the event was generated by the authorization of an RFID tag, this field will contain the unique serial number of that RFID tag.
Besides the prevalent events, which are sent to the PC when it interrogates the stations, it may occur non-prevalent events. They are determined by reading and memorizing the content of an RFID tag, are larger than the prevalent events and they are transmitted to the PC only when a "read the non-prevalent data log" command is received. The structure of a non-prevalent event is presented in Table 3. The *ID_Event* field (1 byte) identifies the type of the event. The *Date* field (8 bytes) is similar to the *Date* field from the structure of the prevalent event and contains the date when the event occurred. The *TransponderSerialNo* field (8 bytes) contains the unique serial number of the RFID tag that was read. The *Length* field (2 bytes) will contain the length of the last field – *TransponderContent*, which is the content of the RFID tag.

PC software components

The access control stations are hardware devices being able to operate completely autonomous; they do not require exchanging information among them, nor any human operator intervention. Any generated data or events are stored in various ways within internal memories, and the actions that a station should do (e.g. access granting or denying) are normally previously established, or at least do not require a permanent data exchange with the master computer. However, a computer would allow an easy management of all events or data, an expanded running flexibility and an efficient exploitation of all functionalities offered by the station.
The PC applications can interact with the stations, and they run under Microsoft Windows operating system, due to his popularity. Aiming to simplify the applications development process, it has been decided to follow the domain international directions. Thus, the PC software modules has been built in conformance with the OPC standard (Ole for Process Control), an industrial standard conceived initially by Microsoft, then granted to an independent organization for future management and development.

The OPC standard consist in a series of weak standards specifications, it define only the required interfaces through which the applications can exchange not only data, but functionalities too. More precisely, a procedure can be exposed and called through such "universal" interface, regardless the language used to implement it.

The base model employed for those modules features is COM/DCOM [7], a public standard also issued from Microsoft; COM/DCOM acts at application level of the ISO OSI networking model; as a consequence, distributed functions calls can be easily accomplished, even toward remote machines, as the underlying details are supported by the operating system using the existing protocols on the lower levels.

The market already offers various applications employing those technologies in a unitary form. Among them, there are the OPC servers which collect the data and expose them to other clients. However, those servers act only as frameworks; they require the implementations of some particular modules, easily attachable to the framework, allowing the devices integration from the logical point of view. The figure 6 presents such system architecture.
The OPC server and the additional applications will interact with physical devices through the software modules named "Hardware device profile" and "Communication component".

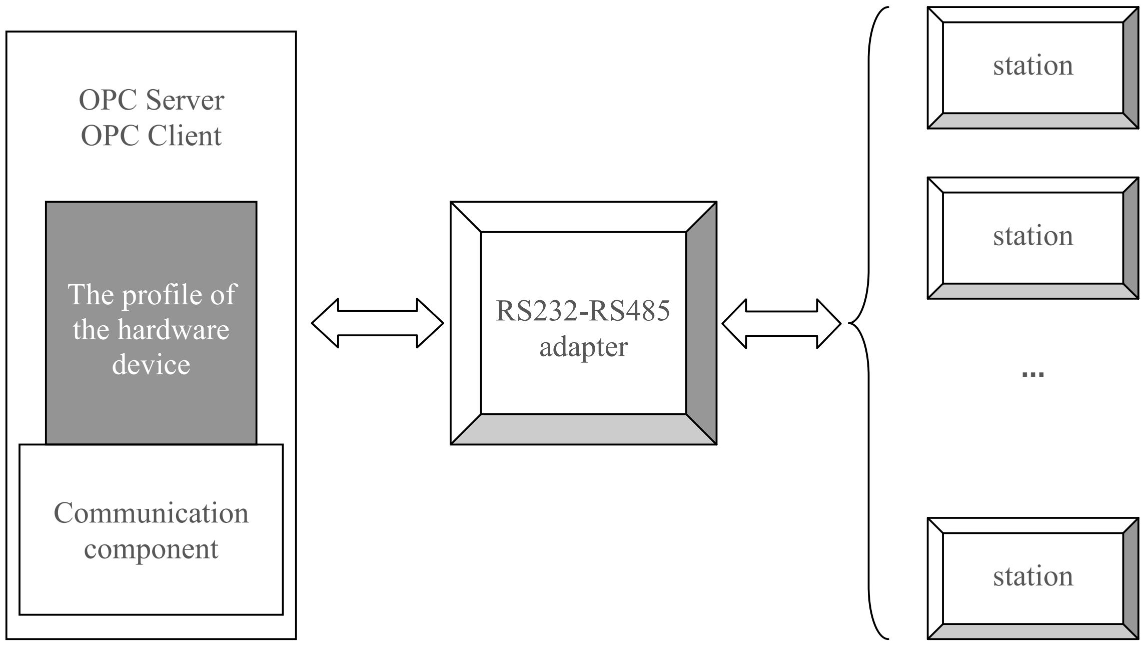


Figure 6. The logical view of data flow in a distributed architecture.

Communication component

This module commits a transparent data exchange between two main entities of the proposed architecture: the OPC server and its client applications, on one side, and hardware devices on the other side (through the RS232/RS485 adapter).
The module implements, with inherent particularities, the data link level of the OSI model. Thus, here are implemented: the data physical transmission towards the adapter (the communication derive from ISO 1745 standard [8]), the data reception, the error detection (bytes loss or bits erroneously received) and the retries of data frames incorrectly arrived.
This module assists the software to software communication by using the OPC standard, this way allowing the link with enterprise upper levels.
The communication component is built as a multithreaded application, enabling the data transfer simplification (as the data flow is more clearly understandable by any user/programmer whishing to follow or modify the data structures) It enables also the increase of the application responsiveness regarding the incoming commands from the devices or from the upper software level (e.g. from hardware device profile). To achieve this goal, two different queues have been implemented, allowing an independent bidirectional "messages" transfer and ensuring the sequencing of the commands and answers. This way, a data flow control can be pursued, with significant gain in flexibility, needed for an eventual future project growth.

Hardware device profile

It is the module implementing the interface with upper levels (PC applications), exposing the hardware device functionalities from the logical point of view. If the communication component is instantiated once in the server, this time, there is a profile corresponding to each hardware device.
The module acts at the network layer from OSI model: it handles the devices' addressing; it checks the transmission accuracy (with transmission retries if needed); it handles the vary errors in a unitary manner.
Aiming to ease the programming experience and the system responsiveness there are multiple

messaging queues. Thus it allows the serialization and the avoiding of dead times - the client applications do not need to wait for a message's answer before sending a new one. Consequently, a future target could be the messages prioritizing, easily supported by those queue techniques.

The module presents some basic debugging functions, helping the client application's programmer to check the communication messages formatting, signaling every discrepancy. Additionally, there are some profiling functionalities (with the communication component contribution) allowing to the profile's users to obtain statistical information regarding the messages transfer speed, execution and processing times, in order to tweak the performances.

The commands (messages) format was kept as simple, flexible and general as possible, to be relatively simply used and extended by the client applications, allowing also to minimize the impact of the hardware device growth.

Conclusions

The system presented in this article is modularly, having support for an easy physical and logical expansion. With the regard of distributed systems concepts, these modules can be used in any other projects respecting the chosen communication model. This project can be deployed as didactical system too. The system is a first step toward multiple integrations.

References

[1] Koskinen, Pentti (2000) *Decision-making process on field technology for process management*, Department of Electrical Engineering, University of Oulu, FIN-90014 University of Oulu, Finland.
[2] Jonas Berge February 18-19, (2004), *Fieldbus Foundation's open, integrated architecture for information integration*, New Orleans, Louisiana
[3] Vasile GAITAN (2002) *Rețele Industriale Locale. Nivelul Fizic.* MATRIX ROM BUCUREŞTI, ISBN: 973-685-354-3
[4] * * * - *uPSD 323x Data Sheet*
[5] www.keil.com, *Embedded Development Tools*
[6] FEIG Electronic – *OBID i-scan Standard Reader User Manual*
[7] http://www.microsoft.com
[8] IEEE - *International Standard ISO 1745*
[9] Alexandru LARIONESCU, Cătălin SÎICU, Vasile GĂITAN (2005) – *Sistem distribuit de gestiune pentru controlul accesului bazat pe tehnologii RFID (SDGAT)*, Seminar Sisteme Distribuite, Suceava.