HÖGSKOLAN SKÖVDE

Institution for communication and information
Final year project in computer science 20p
Level C
Spring term 2007

Solutions towards domotic interoperability

The contribution of the OPC Standard

Jorge Serrano Betored

HS-EA-DVA-2005-001

Solutions towards domotic interoperability

This final year project has been submitted by Jorge Serrano Betored to the University of Skövde, as a dissertation towards the degree of Bachelor of Science (B.Sc.) in the School of Humanities and Informatics.
The project has been supervised by Anders Dahlbom.

21th May 2007

I hereby certify that all material in this dissertation which is not my own work has been identified and that no work is included for which a degree has already been conferred on me.

Signature: _____

Solutions towards domotic interoperability

Jorge Serrano Betored

Abstract

This report presents the existence of a set of problems making the growth of the domotic field more difficult. They are mainly the lack of a common communication standard among devices and the existence of a proprietary market, where each provider focuses on developing its own devices, protocols and interfaces. There isn't a convergence criterion in order to overcome this problem by the main domotic providers. Several studies try to overcome this problem by applying different strategies. This study analyses the main strategies followed in that field, concluding with a model that combines them. The model is based on the use of OPC and web services.

Keyword: BAS, OPC, Ambient intelligence, integration, interoperation

Acknowledgments

The outcome of this research wouldn’t have been possible without the help and feedback provided by my supervisor Anders Dahlbom and my examiner Björn Olsson. I also have on my mind the exchange student community, which became my family during the academic year 2006-2007 in the beautiful frame that was Skövde.

Table of contents

1. Introduction

The information and communication technologies spread across our lives to make our everyday tasks easier and to increase the quality of our existence in every domain. As in other domains, also in home environments many appliances are rapidly becoming computationally enabled. Domotics is the science that applies computer and robot technologies to domestic appliances. Domotic systems are based on the use of three kinds of devices, sensors, actuators and controllers, which are connected by some network (called a *control network)*. By acting together and following some *protocol*, they can behave in an intelligent way. Controlling indoor climate, provide choices for comfort, energy conservation and security systems all fall under the domotics umbrella (Mateos et al., 2002).

Domotics is a booming market with applications in regular houses and in other kinds of larger buildings such as hotels, office buildings, etc. A wide range of terms have been found in the literature referring to the domotic field; according to some sources two subfields can be distinguished. These terms are on the first hand home automation, and on the other hand intelligent buildings, building automation and Building Automation Systems (BAS). Some researches focus their studies in one subfield, proposing specific solutions. On the contrary, some other authors consider domotics as an only field and their studies can be applied for both kinds of installations. Home automation focuses on the use of computer technologies within the domestic home field, while intelligent buildings, building automation and BAS take into account the use of the same technologies for automating office buildings (Thomas and Soleimani-Mohseni, 2007).

The reasons for using computer technologies are similar in both cases, but not identical. The purpose of the first subfield is to provide comfort to the residents by automating some daily tasks they normally have to carry out. The purpose of the second subfield is to reduce the operating cost of the building while maintaining the desired environment for the occupants. BAS saves energy by widening temperature ranges and reducing lighting in unoccupied spaces. BAS also reduces costs for electricity by shedding loads when electricity is higher-priced (Intelligent building, 2007).

Although the purposes of these two subfields are different, they are both handled by the same kind of technology. Even if they utilize the same technology, an installation in a building normally will be more complex in terms of the surface covered by the network connecting the sensors, actuators and controllers. Also the types of some sensors and actuators are different, and of course, the logic stored in the controllers is more complex for BAS, since the number of variables it has to manage (related for example to the temperature in each room) is larger.

All the research depicted in this paper can be applied to both subfields, but the examples shown will focus on the domestic field. To have a more pragmatic point of view, let us list the main types of devices that can be installed in a control network. First, the most used kinds of **sensors** are:

- Occupancy: detect if there is someone inside a room. Used for security and safe energy applications. An image can be seen in figure 1.
- Temperature: register the current temperature inside/outside the house.
- Lighting: measure the level of light inside a room or outside the house.
- Sound: measure the noise level. Security applications.

- Contact: register if a door or a window is closed. Security and safe energy applications.
- Gas detector: detects a possible gas escape.
- Fire detector: detect fire inside the house. An image can be seen in figure 1.
- Humidity/water detector. The detector is directly located over the floor. In case there is a broken pipe in the house, it will change its output state.

Other inputs like button sets or switches can also be considered as sensors. Secondly, the most representative kinds of **actuators** are:

- Alarms: normally located outside the house for security applications.
- Electrical valves. To control (open and close) the incoming flow of water and gas from the pipes. An image can be seen in figure 1.
- Light controllers, acting as central nodes connecting several light outputs, such as binary lights (the ones that can only be turned on and off) and dimmable lights (which allow regulation of intensity).
- Operators for sun blinds, roller shutters and Venetian blinds.
- Garden irrigation controller.
- Air conditioning controller.
- Heating controller.

Figure 1: occupancy sensor, fire detector and electric valve actuator.

Finally the third kind of device that can be installed in a control network are controllers; they don't have to be installed in every domotic system. It depends on the architecture of the system. We can distinguish centralized and distributed architectures (Domotics, 2007). In a centralized architecture there is a controller connected to the network that will act as an intelligent unit, holding all the logic of the system. It's responsible for all the communications between sensors and actuators. Normally, controllers can be accessed through a serial interface from user interface software, in

order to program all the applications controllers are going to manage. A controller can be seen in figure 2. On the other hand, in a distributed architecture there isn't a central node (controller) storing the logic of the system. In this case, sensors and actuators have to interact and to take decisions by themselves. Anyway, the network can either be accessed by connecting the bus to user interface software through a USB or serial interface. In this way the applications they are holding can be programmed.

Figure 2: controller for X10 standard.

By the use of all these devices a large number of applications can be executed within the automated house. Some examples are presented. Firstly, for security applications, we can easily improve the security by communicating a fire sensor and some actuators. In case the fire sensor detects smoke, it will send a first datagram through the control network towards the gas valve, which will be closed. After that, a second datagram will be sent from the fire detector towards the alarm, which will be turned on, after having received and analysed the information within the datagram. This example presents a distributed architecture.

Secondly, for safe energy applications we can develop an easy interaction between devices related to lighting services. Let's suppose that one of the persons living in the automated house is inside the kitchen during a sunny morning. Although there is enough natural light outside, the electric lights of the kitchen are switched on, while the window blinder is closed. An occupancy sensor will register that there is somebody inside the room; it will interact with the light controller and the window blinder actuator, in order to know their states. They will answer with their current states. After that the occupancy sensor will ask an external light sensor if there is light outside the house (where this sensor is installed). After the affirmative answer the light controller will receive a command to switch off the lights of the kitchen, and the window blinder actuator will be activated. The system shown in this example has a distributed architecture. The devices interact directly with each other, without the use of a central controller holding the intelligence of the system.

Finally for comfort applications, efficient systems for controlling the temperature can be implemented thanks to the control network. A temperature sensor will permanently measure the temperature inside the house. In case the temperature registered by the temperature sensor has a high level, the air conditioning controller will receive a command to switch on and reach a determined level. The levels of the temperature can be pre-programmed by the user, depending for example on the time zone.

Even though the possibilities offered by domotics are wide, the current state of the spread of domotics shows an irregular panorama. While the installation of automation systems in office buildings is successful, the introduction of this technology in home environments can't be considered a general trend. The reason behind that fact is probably the high cost of the devices. The initial outlay for the installation in regular houses is bigger than the economic profits the users will obtain by saving energy. However in building applications the investment is profitable. Nowadays home installations can only be afforded by people with a high economical level, although developers consider that in the near future it will become a general trend and the price of the installations will considerably decrease.

During the last years the number of companies taking part in this market has increased. The most important companies have joined their efforts to create organizations that work in developing and following specific technologies. This market is growing and the companies need to invest large amounts of money into research. We are in front of a potential field for computer science professionals.

Following the above description, we know about the existence of a set of organizations developing 'specific technologies' each of them equipped with different communication protocols and middleware. The set of devices, the types of network linking them, and the communication protocol within each network define a domotic standard. North America, Europe and Japan are the three main areas oriented to different wiring technologies and protocols, and the picture gets even more confused when looking at a Nation-wide level (Pellegrino et al., 2006). The different standards are very irregularly distributed around the world. While standards like X10, LonTalk, CEBus and Smart House are well spread in North America, KNX has a stronger influence in Europe and HBS is the most successful technology in Japan.

All of these standards aim to realize intelligent home or building environments. However home appliances and devices belonging to different developers are nowadays completely isolated from each other, creating the main obstacle to the domotic market growth. Other factors slowing down the spread of domotics are the price of the devices and the lack of user-friendly interfaces, but it's essential to provide an interoperability framework for the growth of the field (Miori et al., 2006). The development of an interoperability and communication model between those standards will contribute to the expansion of some technologies through markets where they initially don't have a representative presence, and to the general growth of the field.

Reading the previous paragraph we have a clear idea that the communications between devices belonging to different standards are initially not possible. These standards don't have a sufficient degree of interoperability. This problem is the point of departure of this project.

2. Background

This chapter will present relevant background material for this work. We will start in chapter 2.1 with a more in depth study of domotics and the notion of a digital home. The several networks within that and devices linking them are detailed. In chapter 2.2 the interoperability problem is again presented in detail; after that, the several branches followed by researchers with the purpose of solving this problem are presented.

2.1. Digital Home

The convergence of the communications, the informatics and the entertainment thanks to the broadband networks is a consolidated global trend. Traditionally there has been a clear division between voice services, data services and TV or image services; however, during the last years this division has started to disappear. In parallel, new multipurpose networks have been designed and implemented, where services including voice, data, audio, video and control are merged (Hintze, 2007).

Another important fact with technological and social connotations has been the Internet expansion. The Internet has become an important information source and also has become the center of the development of new markets and service sources.

This change in the way to have access to the information is the main factor behind a series of deep social transformations. The new technological media availability causes changes in the way of executing processes. This evolution is defined as the information society, where each person, organization or company not only has access to their own information, but on the contrary they have a large capability to have access to the information generated by other people/organizations, and they become an information source for others. This concept can also be applied for having access to services provided by other organizations/companies, and in the same way, provide services that can be accessed by clients distributed all around the world (example in figure 3).

The Digital Home is the materialization of this services convergence idea. Entertainment, communication, home digital management, infrastructure and equipment services are all linked by the Home Networking (Telefónica, 2003).

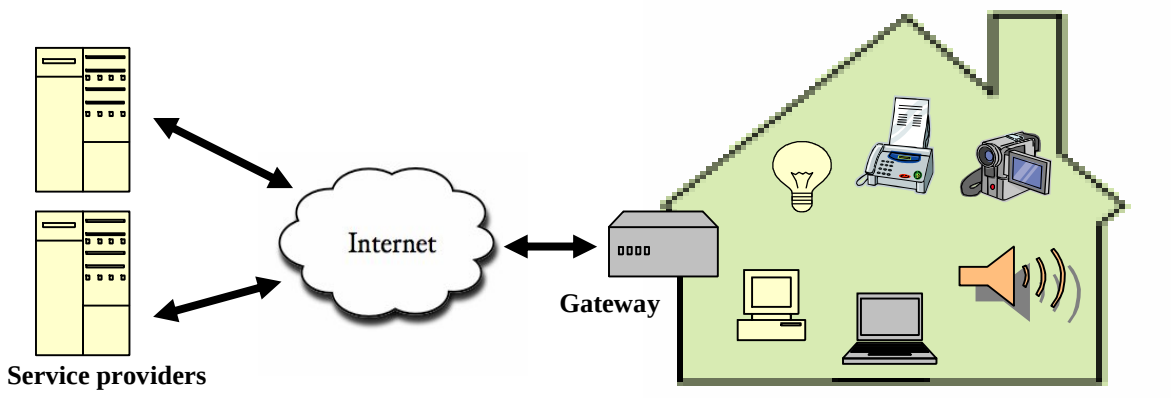

Figure 3: several external services are used from the Home Networking.

The Home Networking concept embraces several physical networks, elements and equipment necessary to allow the access to external services from inside the home. They are:

- A **broadband connection to the Internet** (ADSL, Cable, etc), indispensable for having access to services such as Video on Demand, remote music player and services requiring a permanent connection (always on) such as remote vigilance.
- A **residential gateway**. This device is responsible for connecting the three networks introduced in the next points with the Internet. It needs to guarantee the security in the communications from outside to inside the home. Furthermore, it is normally remotely configurable.
- A **Data Network** to allow the connection between PCs, printers, scanners, cameras and IP telephony– based on the voice over IP protocol (VoIP). Through this network it's possible to share computer resources (files, programs). It also allows Internet access from all the devices connected to the same one and it also allows phone communication. Basically it's a LAN installed across the several rooms of the house, which communications are based on the TCP/IP protocol.
- A **Multimedia Network** for connecting devices such as TV, VCR, DVD, HiFi, game consoles, security cameras and other electric appliances such as new washing and drying machines. This connection allows management and distribution of audio and video over the entire house. Electronic appliances can be accessed and programmed from outside the house or from some of the computers connected to the Data Network. The communications in this network are mainly based on two standards, Universal Plug and Play (UPnP) and JINI, a Java standard developed by Sun Microsystems for the construction of distributed systems.
- A **Control Network** allowing home automation by connecting sensors, actuators and controllers (not all control networks need the installation of a controller). Devices installed in this network and the communication protocol used to exchange information between them follow one specific standard. The main standards used for that task are explained in the last paragraph of this section (Telefónica, 2003).

Although a single standard would be strongly desirable, we should rather get used to the coexistence of multiple communication interfaces (Pellegrino et al., 2006). It is not possible to interconnect and allow communications within the same control network between devices following different standards, since their communications are based on different protocols, and sometimes, they use different transmission media. Neither is it possible to communicate directly with devices belonging to the Control Network and other devices belonging to the Multimedia Network. Furthermore, even if we consider a particular Home Networking linking a Data Network, a Multimedia Network and two Control Networks through the residential gateway, communication between the two Control Networks is not possible if they don't follow the same standard (e.g. they both are KNX installations). This restriction constitutes a big limitation when developing home services, as the services are limited to one specific standard. More complex applications (services) could be developed by the use of a set of devices belonging to different networks, that is to say, following different standards (Pellegrino et al., 2006).

The main standards are X10, CEBus, Lonworks, Smart House, KNX and HBS. Normally these standards are supported by an organization, composed by a group of companies which merge their efforts to create a common framework. They research, improve and promote their common standard; later each of these companies develops devices/software compliant with the common specification of the organization. There are also some companies which use their own standards and technologies, which are not related with any other standard. This work is centered upon the most successful standards, which follow common specifications, since they take the bigger part of the market. Briefly the three main standards that participate in the market are:

X10. This standard was developed in 1976. Its main characteristic is that it doesn't need a specific bus in order to transfer commands between devices; it uses the feed cable. This fact is an advantage (simpler installation), but it's also a drawback, because the signal uses a quite noisy communication medium (X10 knowledge base, 2007). It's an open protocol which is used by a large number of companies. It is orientated toward the final user with simple and cheap devices, although without a high quality.

KNX. This standard is promoted by the KNX Association (KNX, 2007). It is the most successful solution in Europe, due to its reliability, although their devices are highly priced. It is a building automation fieldbus that focuses on the energy management of electric installations, the demand side management, the environment control and safety. A KNX system can be installed in all types of buildings and monitors and controls various environmental procedures and functionalities (Kolokotsa et al., 2006). The KNX association comes from the merger of three large European groups (EIB, HS and Batibus) that previously supported their own standards. The KNX standard inherits most of its characteristics from EIB. A description of each group follows:

European Installation Bus (EIB), which was the most used bus system. It was developed by the EIBA the biggest consortium of European companies. Its main characteristic is that there isn't a central node controlling communications. It is a decentralized set of sensors and actuators; each one can take its own decisions and manage its own communications. The system requires the installation of a bus in order to feed and communicate devices. It is a reliable solution but it's quite expensive.

HS, supported and financed by European public institutions. It is an open standard. It supports different kinds of buses.

Batibus, oriented to technical and security control of buildings. It uses a twist pair bus and supports several net architectures.

Lonworks. This is an open standard with a strong influence in America. It's a distributed system that can use several ways of communication. The devices are programmed in a language called NeuronC and the name of the communication protocol is LonTalk. It uses a proprietary technology called LNS (LonWorks Network System). Each supported operation of the LNS can be performed locally, through DCOM-based clients and remotely through IP-based clients (Kapsalis et al., 2003). Implementations usually include an Internet server that allows remote control (Echelon Corporation, 2007).

2.2 Solutions for the interoperability problem

The previously stated interoperability problem between heterogeneous home networks will be the center of this project. The aim is to configure a system composed by several device networks (each one following specific technologies) that can exchange data and cooperate with each other to constitute a working, technology-independent home system. The set of networks includes data network(s), multimedia network(s) and control network(s). The resulting system will be able to offer services to other systems through the Internet, and to use services stored at other physical locations.

In the next sub-chapters several research branches looking for solutions to achieve this goal are presented.

2.2.1. Gateways

A possible approach for interoperability could be represented by a hardware/software module (gateway) between each pair of home networks. These gateways must have an interface towards both the networks they link and provide a translation mechanism between the two domotic protocols. Some devices of these characteristics have been developed in several research projects; one of them is the experimental bridge LonWorks/UPnP (Chemishkian and Lund, 2004).

The main problem of this first solution is that it's insufficiently scalable because the number of the required gateways grows too fast regarding the number of networks to be connected. For that reason a more suitable solution requires to build up a common infrastructure in order to link every network protocol. Additionally it becomes indispensable to develop a universal communication paradigm to be used in that common infrastructure.

2.2.2. Service-Oriented Architecture (SOA)

In order to overcome the limits of one-to-one protocol bridges and to deploy home network services based on different standards a large number of studies propose to follow a different way to find the solution.

During the last years, a new computing paradigm called Service-Oriented Architecture (SOA) has emerged (Knorr and Venezia, 2007). This paradigm presents systems and the components in systems like sets of services offered by them. The most representative implementations of this kind of architecture are Web Services and Universal Plug & Play (UPnP).

In a SOA, the nodes within a network can have access to the resources of other nodes in the network, considering them independent services. The way of accessing these services is standardized. Most SOA implementations are based on the use of Web Services but there are some implementations based on different service technologies.

Web Services are language and platform independent and could be accepted and adopted as the universal standard for the application-to-application communication paradigm by everyone. The use of the Web Services paradigm assures the building of a natural and native distributed architecture. By means of using this technology, the home can become an Internet node, so it could use all the services available offered by other nodes spread through the Internet and it could also provide new services to other nodes. Web Services are the emerging standard on which the communication between Internet applications will be based.

Contrary to Object-Oriented Architectures, SOAs are composed by low coupling and highly operable services. In order to support the communication between them, these services are based on a formal definition platform and are programming language independent (Weiss et al., 2007).

As explained before, the residential gateway is the point where the different networks converge. Each one of these networks has its own communication mechanisms and protocols. In order to make this convergence possible a large abstraction capacity is required. This abstraction demands a high degree of independence regarding the technologies and standards employed in the implementation of the system.

The availability of a modelling language allowing the description of a domotic installation helps to improve this abstraction capacity. In the same way a modelling language can provide an easier design process when developing a system.

Every modelling language using XML files as repository is called Modeling Markup Language (MoML). MoML is a successful technology thanks to the simplicity and the expressiveness of XML (Lee and Neuendorffer, 2000).

For modelling a domotic installation, the first requirement is a list of devices. After that, each device has to be described by using an XML file. This file describes the services offered by this device. Each service is composed by a set of actions. A second file contains the description of the digital home. In this file the home is divided into rooms; each room contains devices belonging to pre-defined types. Finally the interaction between the devices will be defined. An example of an XML file describing a device is shown in figure 4.

Several studies have implemented different solutions (implementations) for the interoperability problem by using MoML and Web Services. The implementations are very different since they describe systems and the services (MoML structure) in different ways. Also the ways to distribute and access the services are different. Some other researches focus on more specific implementations, like OSGi (Open Service Gateway initiative) and SENDA (*Services and Networks for Domotics Applications*). Specifications for implementing residential gateways like OSGi and SENDA have chosen to implement a SOA.

OSGi is defined as a standard, non-proprietary, software component framework for manufacturers, service providers and developers that could act as Home Residential Gateway. The aim behind OSGi is to define and to promote an open standard for connecting the services offered in Wide Area Networks (WAN), Local Area Networks (LAN) and Local Operating Network (LON). In this way the connection between intelligent devices in the house and services offered through the Internet will be possible (OSGi Alliance, 2007).

Although it is non-proprietary and it could solve the interoperability issue between domotic standards, OSGi is completely based on the Java platform and this fact may certainly represent a “political obstacle” to its definitive affirmation in the domotic context.

SENDA is an industrial specification, alternative to OSGi, for the development of heterogeneous domotic networks. An infrastructure to provide services and communicational networks based on CORBA (Common Object Request Broker Architecture) (SENDA, 2007).

```
<?xml version="1.0"?>
<root>
    <name>LuzBinaria</name>
    <description>Binary light</description>
    <serviceList>
        <service>
            <name>Power</name>
            <description>Service for switch on/off the device</description>
            <actionList>
                <action>
                    <name>getPower</name>
                    <argumentList>
                        <argument>
                            <name>Power</name>
                            <relatedStateVariable>Power</relatedStateVariable>
                            <direction>out</direction>
                        </argument>
                    </argumentList>
                </action>
                <action>
                    <name>setPower</name>
                    <argumentList>
                        <argument>
                            <name>newPower</name>
                            <relatedStateVariable>Power</relatedStateVariable>
                            <direction>in</direction>
                        </argument>
                    </argumentList>
                </action>
            </actionList>
            <serviceStateTable>
                <stateVariable sendEvents="no">
                    <name>Power</name>
                    <dataType>boolean</dataType>
                    <labelUnit>on/off</labelUnit>
                </stateVariable>
            </serviceStateTable>
        </service>
    </serviceList>
</root>
```

Figure 4: XML file describing a binary light device.

2.2.3. OLE for Process Control (OPC)

An innovative approach to solve the problem comes from the world of industrial process control. OPC, a standard created for that field, can help to have a different point of view of the problem.

In order to realize integration and interoperation among these different subsystems developers have to make various BAS software systems communicate effectively. Middleware technologies, including DDE, COM/DCOM, OPC, CORBA and JAVA/RMI have been developed to establish the communication among applications (Wang et al, 2007). DDE has gradually been phased out for its poor performance. Although CORBA is a robust object communication protocol with

features such as object activation, stateful and stateless request etc., it has quite a few drawbacks. For instance, CORBA uses binary encoding for data transmission so it is assumed that both the receiver and the sender have full knowledge of the message context since no meta-data information is encoded. Although this approach offers better performance than XML where an amount of overhead is involved, it makes it hard for intermediaries to process the message. In addition, CORBA relies on the Internet Inter-ORB protocol (IIOP), which is firewall-unfriendly (Kapsalis et al., 2003). RMI can be seen in some mobile applications for Intelligent Buildings. Probably due to the broad use in the automation industries and the popularity of the Windows platform, some manufacturers have applied OPC technology to BAS integration. It seems that OPC has broader use than the other middleware technologies (Wang et al., 2007). For that reason the study of this technology is carried out during this chapter.

In order to provide an in depth discussion of the OPC technology, a few concepts and definitions are first needed. The OPC technology stems from the industrial field where process control is defined as a statistics and engineering discipline that deals with architectures, mechanisms, and algorithms for controlling the output of a specific process. In this report it will refer to any industrial process. Some kinds of devices are used to obtain data and to execute actions in the plant, and they constitute the physical basis of this discipline. We define programmable logic controllers (PLCs) as microprocessors used to manage industrial processes, such as machine control into assembly line. Contrary to user PCs, they are designed to work under adverse temperature or dirty conditions. They are different from any other kind of computer since they are connected to a different type of inputs and outputs. For example a PLC can read if an interrupter has been pushed, the value of a temperature sensor and even an artificial vision device record. They can also control output devices like electrical or pneumatic engines, hydraulic cylinders, LEDs, relays, etc. They are considered real time systems since the outputs (connected to actuators) change in function of the value of the inputs (connected to sensors).

Now that we have all the concepts and definitions in place, let us continue by examining OPC in more detail. OPC (OLE for Process Control) is a communication standard in the control and supervision process field. The specification was developed in 1996 by an industrial automation industry task force. They constituted an organization called OPC Foundation (The OPC Foundation, 2007).

Deepening in the definition “communicational Standard in the control and supervision process” we can consider it as a real time data communication standard between control devices manufactured by different manufacturers.

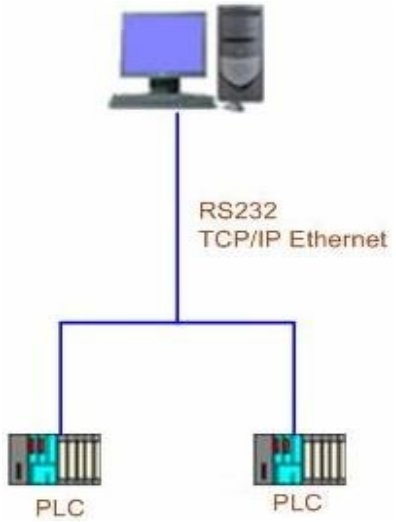

Figure 5: A PC controlling two PLCs.

Control devices are considered hardware tools such as PLCs, which are able to gather information from industrial processes being executed in an industrial plant. They can receive information from manufacturing robot sensors, infrared cameras, temperature sensors, pressure sensors, etc.

These devices are able to carry out measures, execute some simple processing with that data, and finally send this information through any kind of network towards some

Human Machine Interface (HMI) – normally computers - that will store this data. Later this computer can execute some application to analyze the data, such as comparing it with other input values from other PLCs, and later take a decision and communicate with some output PLC, which will transmit some action to an actuator. Another application in the computer can be to have a historical record of the measured values from the input PLC. A scheme is shown in figure 5.

Traditionally the PLC manufacturing companies were responsible for the design and implementation of the high or medium level control software and the communication protocols linking its devices and the HMI executing those control programs. This means that, every time a production company decided to buy some hardware for some of its plants, they were under obligation to buy the associated control software and drivers from the same hardware provider. Furthermore, subsequent expansions in the production plant meant to buy the new hardware and software from the company which developed the previous system, in order to avoid incompatibilities. At that time we were in front of a "proprietary systems" market. According to Hong and Jianhua (2006) the system requires an open, standard, and real-time communication mechanism to exchange information between different layers, since in networked industrial information systems (especially in a large scale) there are several small independent automation systems provided by different vendors.

The release of the OPC standard meant a revolution in this field. It meant that the previously existing constraint of writing down a driver for each pair control-software control-device disappeared. Instead, once an OPC driver has been programmed for a certain device, any application can have access to the device through this new driver. In this way production companies gain a large freedom when they have to choose a certain technology for their plants (Hong and Jianhua, 2006).

Since that moment a certain production company can buy some PLCs (for input applications) from a first provider, some other PLCs (for output applications) from a second provider, and later buy some SW application from a third provider. The system will work without incompatibilities, whenever these three companies follow the OPC standard. Architecturally the OPC Server will act as a medium layer, gathering data from the PLCs and providing data to the application software, located in a higher layer. The OPC server can be accessed by any OPC client; when an external client accesses a server it receives the values of the items according to the configured update time of the data. The communication with the client can be synchronous or asynchronous. The first one is used with the control signal because it is necessary to transfer the data in a given time instant. The asynchronous one is only used for the data to be exchanged (Alves et al., 2005).

The OPC standard supports several data sources sending information to a unique *OPC Server*, where several application programs will have access (since they fulfill the OPC specification). We will call these programs OPC Clients. They can be located in the same computer or they can be distributed over several PCs. In figure 6 a single PC hosts both a client and the server. The communication will always take place between a client and a server. A client can communicate with several servers, and several clients can access one server. It is therefore possible for an application to access data coming from several different fieldbus systems by using different OPC servers (Rüping et al, 1997).

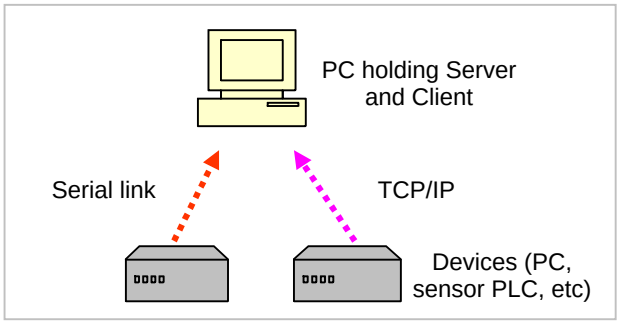

Figure 6: OPC Server and OPC Client working over the same PC.

In the case the clients are distributed in several computers, they can get access to the OPC Server through TCP/IP connectivity, supported by the OPC standard. It represents an open door to the distributed systems world, which performs one of the more important features of this technology. A scheme is shown in figure 7.

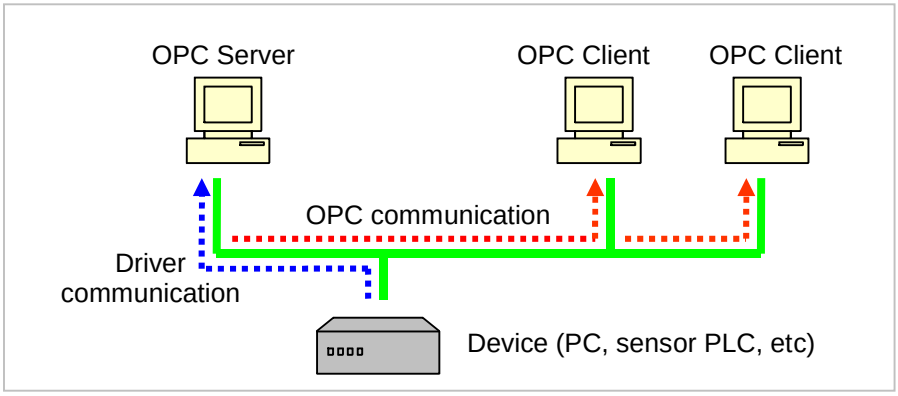

Figure 7: TCP/IP connectivity, supported by OPC standard.

Technologically speaking OPC is based on OLE (Object Linking and embedding), COM (Component object model) and DCOM (Distributed component object model) technologies. All of them (meaning OLE, COM and DCOM) were developed by Microsoft and are used in the Windows O.S. family. More specifically COM and DCOM are used for the real time data exchange between the control devices and the software application (OPC Server) and OLE is used for communication between the OPC Server and OPC Client(s). According to Alves et al. (2005) this emerging technology based on DCOM has gained wide acceptance and became a “de facto” standard in the process industry for communications among devices. Currently there are hundreds of commercial products available that will provide connectivity to every major control system on the market. OPC Servers are available for many systems and protocols, including ModBus, BADnet, LonWorks, Johnson Controls and others. Furthermore, there are hundreds of OPC enabled client applications to choose from including HMIs, visualization and reporting tools, preventive and predictive maintenance packages, lighting controls, security applications and many more (Hong and Jianhua, 2006). Kapsalis et al (2002b) strengthen this point of view when they state “OPC is the dominant standard interface between different automation system vendors, enabling the connection of control networks with data/enterprise networks in a seamless and standard way”.

The OPC implementation defines a set of standard objects, interfaces and methods, which are used in process control and manufacturing automation applications. With the use of this set of standards interoperability is facilitated. OPC reduces the development of several drivers to a single one. Encapsulating a device, an OPC server interface allows external access to the data acquired by the driver of this device to several heterogeneous clients (Alves et al., 2005). The OPC standard was designed as a link between Windows based applications and control process applications (HW and SW). It's an open standard that constitutes a consistent data access method from high level applications to production plant control devices.

As previously explained, before the release of the OPC standard, each time a SW packet needed data access to control devices it was necessary to implement a driver to allow this communication Software-Hardware. Hence, we can conclude that the purpose of the OPC standard is to define a common interface, once it has been written, it can be reutilized by several applications. Once an OPC Server has been programmed for a specific device, it can be used by any application acting as OPC Client.

Although the OPC Standard was initially designed for process control applications, it can be used for domotic purposes. We can see a certain degree of parallelism between an industrial production plant and an automated house/building. Both types of installations are composed by a network connecting a set of devices. These devices have the task of measuring certain parameters or to carry out some actions. They communicate with each other to execute some complex process, following the rules stored in some 'intelligent unit'. Obviously the kind of devices, the kind of network, and the place where that 'intelligence' is located are different in those two cases, but the basic behavior within those installations is the same. This statement is supported firstly by Kapsalis et al. (2002a) who expose the common necessity in industrial and building management systems to integrate distributed real-time applications across control networks, data networks and the Internet, since those two kinds of systems share the necessity to associate real-time applications with time-critical control data. Kapsalis et al. (2002b) go further and state the existing need both in the industrial field and the building/home management sector to integrate different and diverse applications. As a result of this integration, the user is provided with advanced high added value services that enable the connection to and utilization of applications and systems around the world and the subsequent access to their real time data. Kapsalis et al. (2002b) state that research in one field can easily be applied to the other field, since the above need is common in both fields. In this context, the end users, being a service provider for the industry or the building respectively, may have access to and control of specific system data for reasons of enterprise level applications such as system monitoring, performance evaluation, operational status checking, maintenance and service scheduling.

The current proprietary systems frame supposes that each domotic standard has its own evolution. However, several standards have adopted the OPC Specification to achieve interconnection and distributed aims. For example KNX and Lonworks have developed their own OPC Server for controlling their networks. The aim behind the development of these OPC Servers is to take advantage of the existing OPC standard to link domotic networks (following the same standard) in a simple way. It is also possible to develop software packages (acting as OPC Clients) allowing easy access to runtime services such as group addressing and device checks. In this context, high level software tools (OPC Clients) used to control an installation through the OPC technology are normally called visualization or control programs.

The biggest advantage this technology provides to the domotic field is to have access to several networks or devices (following the same standard) from only one visualization program. Furthermore, due to the tool's compatibility, it's possible to access and control any device or net from virtually any standard process visualization toolkit or environment on Windows platforms.

3. Problem description

In the background of this report the concept of the Digital Home has been introduced. It's the materialization of the services convergence idea. The physical tool supporting this concept is the Home Networking. The Home Networking is composed by a set of networks linking several types of devices. Specifically the Data Networks link devices such as computers, printers, VoIP phones and cameras; the Multimedia Networks link services like TVs, DVDs and HiFis; and finally the Control Networks link domotic devices following a specific standard.

All these networks converge in the residential gateway, the linkage point between the home environment and the Internet. A scheme is shown in figure 8.

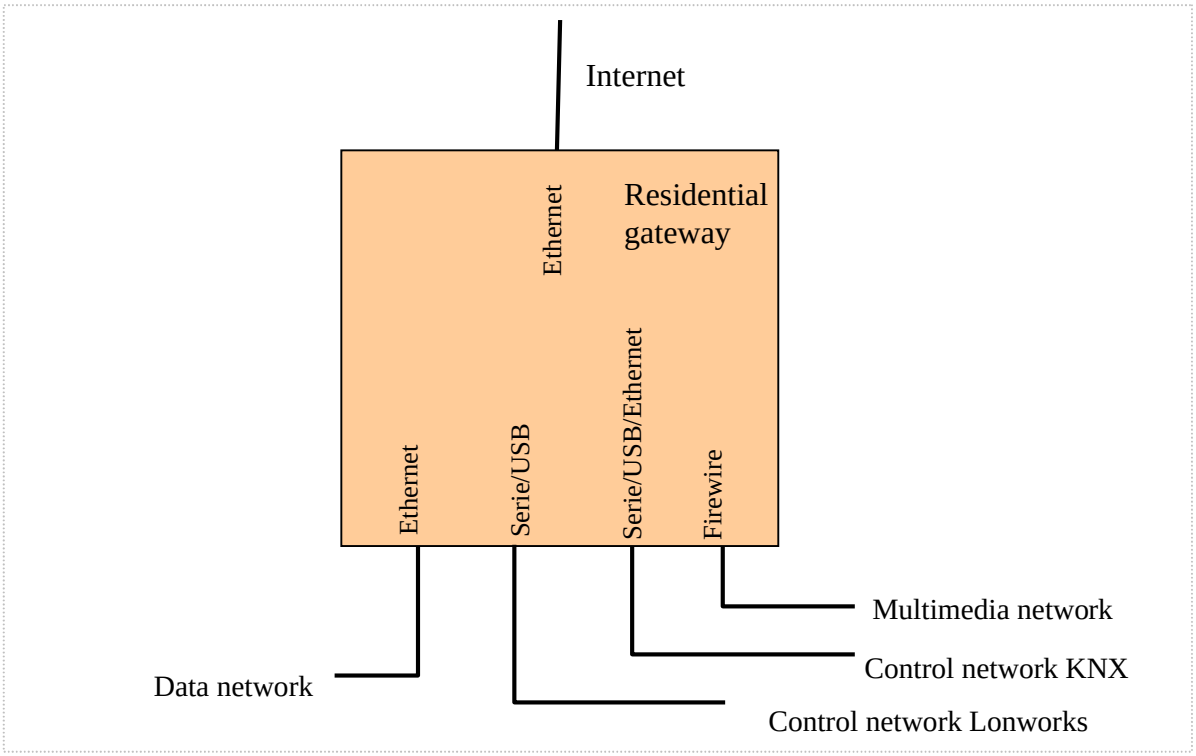

Figure 8: Basic schema of a residential gateway.

The Control Network has been the most important network from the domotic point of view, since it connects the 'pure' domotic devices. After the intercommunication among heterogeneous devices within local area systems, integration between geographically distributed BASs has to be taken into account too. In this field, the current existence of a proprietary systems market has been shown. As each of these alternatives is better suited in a given context, they are still far from converging into a unique solution, even because of strong commercial influences (Pellegrino et al., 2006). The lack of a common language allowing the communication between heterogeneous devices is the main obstacle behind the growth of the domotic field. The target behind a lot of research is to design a tool that permits the development of services using devices distributed in several networks (meaning different control networks or multimedia networks).

The ways to try to achieve this solution are very varied in the research community. Initially researchers tried to develop one-to-one standards-translator bridges to overcome the lack of a common language. Later some of the studies focused on the use of existing technologies based on SOA, like the OSGi specification. Other studies tried to implement solutions by the development of Web Services and the description of systems through MoML files.

The next step in this evolution can be the use of the industrial standard OPC. Even if the reasons behind the existing implementations of this technology by different standards (KNX, LonWorks) are not to solve the interoperability problem, it's an open door to continue the research. The existing implementations only try to obtain benefits from applying the OPC specification to a particular domotic standard. Mainly these benefits are to interconnect networks and to easily develop visualization programs.

On the contrary, as explained in section 2.2.3, the OPC specification was used in the industrial world for solving the previously existing interoperability problem in that field. Due to the conceptual parallelism between industrial and domotic fields (Kapsalis et al., 2002b), some questions can be raised: Which are the similarities between the industrial automation and the domotic field? Can OPC be the solution to the interoperability problem in the domotic field? Have the main domotic consortiums tried to use this technology to achieve the solution? Is it possible to reach the solution with this technology, that is to say, is there any application (OPC Client) linking two different domotic networks which are controlled by their respective OPC Servers?

Hence, the aim of the project can be defined as "Study how a solution for the interoperability problem can be reached by comparing or combining several methods (residential gateways, web services, OPC), taking into account the requirements of a global solution and determine if the OPC specification can be used to reach a solution which improves the outcomes of previous methods".

In order to achieve this aim, a set of objectives have been identified.

1) Determine what requirements should fulfil a global solution. Exhibit some scenario (example) gathering these requirements.
2) Study the contribution the OPC standard can mean for this field. For that reason a theoretical study will be necessary, comparing the different methods behind this subject (residential gateways, web services, OPC), determining if the requirements stated in objective 1 can be reached.
3) Finally to have a more pragmatic point of view, study what can be gained by the use of this technology. It will be exposed the procedure necessary to interconnect different networks following a specific technology (KNX), before and after the use of OPC. It involves the study of two existing implementations, and the reader will get a specific view about the physical implementation of this technology, after all the theoretical description preceding it.

4 Method

This chapter describes the methods that can be used to solve the problem defined in chapter 3. This chapter compares different approaches to achieve the objectives and describes the chosen approach.

A method is a procedure that gives a description on how to tackle a problem and may be used to collect, process and to summarize information to acquire knowledge in a certain subject field (Olsson, 2002). A more global point of view is given when analyzing the several techniques that can be applied in research. There are, according to Andersen (1998), two main forms of approaches within scientific research. These are generally characterized into either quantitative or qualitative approaches. These approaches differ in the way collected data is processed and analyzed. The quantitative approach refers to research that uses statistical analytic methods. There is a clear guideline on how to put research into practise and it makes frequent use of statistics, mathematics and arithmetic formulas. In contrast, the central point in the qualitative approach is to create a deeper comprehension of the problem area. Qualitative methods do not use numerical data, since this type of data cannot meet the main purpose of this approach, that is, to exemplify and achieve a deeper understanding.

For this project, an approach that is mainly qualitative will be used, since the purpose of this work is to interpret and understand the evolution in the research applying different solutions to solve the interoperability problem in heterogeneous domotic systems.

4.1 Objective1: Determine requirements for a global solution

For this first objective it will be necessary to establish a scenario showing a complex heterogeneous system, in order to easily explain to the reader the requirements that should fulfil a global solution. For that reason some of the concepts tackled in the background of this report will be detailed in the context. Following this approach comparative criteria can be formulated; these criteria must therefore be clear and present valid arguments that can later be compared against the results of the next objectives, as an indication of whether or not the different solutions exposed in them are achieving the requirements.

4.2 Objective2: Contribution of the OPC standard to domotics

This second objective can be summarized as: Can this standard be used for solving the interoperability problem between heterogeneous devices? Which are the differences between the outcomes of other solutions (Gateways and Web Services) and the outcome obtained by applying the OPC standard?

One approach could be to implement a new solution. It will involve to have access to at least two different OPC Servers (for example KNX and Lonworks), and develop a new software, acting as OPC Client, controlling both servers, that means, both networks. This could be used to design test cases and analyze the results from those tests. However, the drawbacks of this approach are that there is no access to both

OPC Servers, and the University of Skövde neither has the specific hardware that is necessary in order to implement the system. Furthermore the time required for implementing these solutions exceeds the available time for this project.

The second approach, and the one that is selected for this part of the objective, is to undertake a literature analysis. According to Berndtsson et al. (2002), a literature analysis is defined as a systematic examination of a problem, by means of an analysis of published sources, undertaken with a specific purpose in mind. To succeed in that objective a careful interpretation is required and systematic analysis of each individual source. The outcome of this analysis will describe the advantages and drawbacks of the OPC solutions compared with the previous ones (gateways and web services). For achieving this aim it will necessary to deepen in scientific articles dealing with this issue. This aim is to determine if the OPC specification can fulfil the interoperability problem between heterogeneous domotic systems.

4.3 Objective3: Applying OPC among KNX networks

One approach could be to conduct some interviews with someone responsible for the design of OPC Servers within domotic companies. Although a deeper knowledge of each implementation could be reached, we should be sceptical about the objectivity of their arguments, since as has been stated by several researches, commercial interest is the main problem behind the growth of the field. The providers that tackle the concept of interoperability conceive it as the possibilities to link several networks fulfilling a specific standard. The aim pursued by this research is not the same, therefore other approaches have to be considered.

The method chosen for this objective is the case study. Berndtsson et al. (2002) defines a case study as an in-depth exploration of a phenomenon in its natural setting, which allows you to undertake a detailed examination of the problem. The case study is normally centred on a single case, and may be undertaken within a specific unit (organisation, department, product, etc.). For this objective the centre of the study will be the KNX OPC Server. The reasons for choosing KNX are the relation availability of technical information since the KNX association organizes and supports the scientific research within this field; furthermore this company has shown its interest in the OPC standard since they develop solutions based on this technology. This case study will be used to show the differences between the method used previously to interconnect several networks and the new method provided by this OPC Server. The different types of systems that can be established with both implementations will be described. In this way the reader will have a clearer point of view of the objectives followed by those implementations, after all the theoretical description in preceding objectives.

5 Results

The current chapter contains the results obtained after applying the chosen methods to the objectives identified in chapter 3. A large research effort has been necessary to obtain an in-depth description of some technological issues. Sub-chapter 5.1 starts describing the requirements for a global solution. The contribution of OPC to the domotic field is presented in sub-chapter 5.2. Finally in sub-chapter 5.3 the study focuses on the advantages obtained by the OPC standard when it's implemented in KNX systems.

5.1 Objective1: Determine requirements for a global solution

In chapter 2 and 3 the problems within the domotic field, regarding the interoperation among heterogeneous devices were shown. The current panorama shows a scenario where proprietary protocols still dominate the market. In this section the requirements for a global solution are presented. Two different application fields within the domotic discipline should be taken into account: the integration over local area systems and the integration among geographically distributed systems.

Firstly, if we think about a local area installation, of course it becomes essential to integrate building automation systems among products of different companies. An installation could ideally be provided by several control networks, each one managed by a different protocol. Communication among nodes of different networks becomes more and more important in order to improve the global usage of the infrastructure. A system that enables a simple and fast communication between these nodes provides more possibilities to find efficient solutions (Rüping et al., 1997). As was introduced in chapter 2, several approaches try to overcome this problem by different means. This communication among sub-systems has to be distributive, flexible and scalable; furthermore new sub-systems should be easy to add to this middleware system and therefore the system has to be flexible and scalable (Wang et al., 2007).

Secondly, if intercommunication among geographically distributed BASs is taken into account, there are two important aspects to consider. According to Miori et al. (2006) it is important to construct a common infrastructure in order to link every home subnetwork protocol; in addition, it is essential to develop a universal communication paradigm to be used by this infrastructure. Wang et al. (2007) refer to the current state in this field when they state that the currently used integration software systems usually realize integration and interoperation of BAS in the LAN. However, these solutions haven't truly achieved the integration based on Internet. To achieve this aim domotic systems have to implement powerful communication and integration capabilities on the Internet. BASs have to be seen as part of a much larger information system. At present, most domotic standards allow end-users to access and control an individual BAS remotely, by the use of web browsers. This functionality is still far from real intercommunication**.**

Ideally each installation should become a node on the Internet, providing and using services from and to other installations. An integrated BAS should communicate with other Internet applications, i.e., providing remote monitoring and controlling or getting data from other Internet applications. For that reason, the next question is: How to integrate BASs with the Internet? Some approaches also try to overcome this issue, but a high level of abstraction has to be considered to reach a global solution that fulfils local and based-on-Internet requirements.

The requirements in both sub-fields (meaning local area and geographically distributed systems) should be seen as part of a common aim. According to Wang et al. (2007) there is a pressing need to develop a middleware solution that combines Web Services and BAS software to realize the integration and interoperation of BAS on the Internet.

When the concept of intercommunication is used, an important fact is that the interoperability between devices does not mean that a particular device belonging to a home middleware must offer exactly the same functionality as an equivalent device belonging to another home middleware. Communication does not need to be focused on improving the intrinsic capabilities that each middleware provides (Miori et al., 2006). For example X10 will not be able to manage a complex UPnP device (e.g. media player) as well as UPnP itself can do.

In order to show what all these requirements would imply in a real system, an example gathering them is presented. As explained above, we should consider a global system, as a result of linking several local domotic installations through the Internet. Each installation would act as an Internet node, interacting with other local installations (providing and using services). Furthermore, other (non domotic) Internet applications could be accessed from the local installations. Figure 9 shows a schema of this model. Subsequently the functionality of the whole system would be further improved, and complex applications managing the different local systems could be implemented. If we focus on each local domotic system, it can be installed in an office building, an apartment, a hospital, etc. A local system could be provided by several control networks, where each is managed by a different protocol. These control networks would interact, transferring information between heterogeneous devices. The result would be a system acting as if a sole standard was under use. The models proposed by several researchers in different papers show systems that aim to fulfil those requirements; however the strategies followed in order to achieve them are diverse.

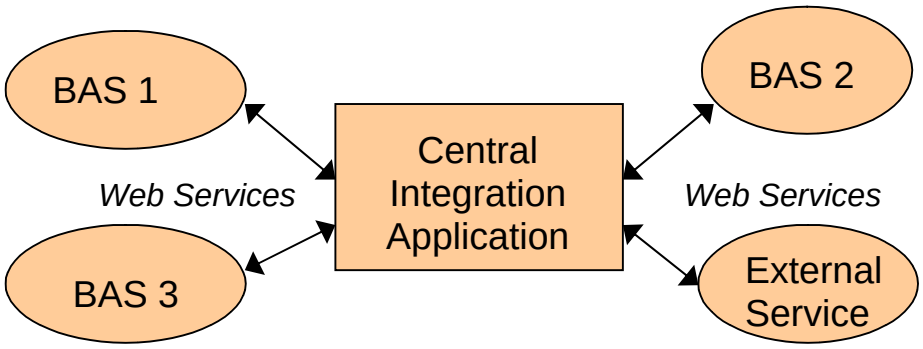

Figure 9: Integration of BAS and external services over the Internet. Based on the diagram presented in Wang et al. (2007).

5.2 Objective2: Contribution of the OPC standard to domotics

After a large literature analysis, the main articles dealing with this subject have been located. The first step was to try to classify them. Basically, a general trend can be identified or "evolution" in the research if the chronological order of the articles is taken into account. It is also possible to categorize them depending on the strategy they follow to achieve the solution. In most of the cases, the articles dealing with this field are

composed by a description of the existing problems within the field, and later, they propose a solution to some of these problems; for that solution they first state its theoretical basis, and later normally a model or an implementation is shown.

Wang (2007) describes the current scenario as follows: “Great progress has been made on standard communication protocols of BAS. However, proprietary protocols still dominate the current BAS market even today mainly due to business reasons”. In order to solve this situation several approaches were shown in chapter 2. This chapter is divided into sections, each one deepening in a particular approach. In this chapter is presented a comparison between the different approaches, showing the general evolution in the field and finally evaluating the performance of the OPC specification. In order to get a more pragmatic view, each section contains the description of a model implemented or designed in some of the most significant papers within its field; those descriptions are followed by some comments evaluating their performance. Chapter 5.2 is organized as follows. Section 5.2.1 studies the performance of the residential gateways approach. Section 5.2.2 shows the main advantages and drawbacks of the OPC approach, when that handles a system by itself. The framework of web services is presented in section 5.2.3. Finally section 5.2.4 analyses the performance of the approach combining OPC and Web Services.

5.2.1 Residential Gateways

A popular way to integrate the products using various protocols is to employ the hardware gateway method. The hardware gateway converts a protocol to another protocol by mapping data points from one protocol to another protocol.

One of the most significant models implemented within this field is the domotic house gateway (DHG) designed by Pellegrino et al. (2006). The authors consider that technologies that are well established for common PCs are being transferred to numerous devices and simple computer systems can be used to bridge the interconnection gap among intelligent environments. They consider that the intelligence provided by BASs is generally basic, unless enhanced by using a more complex and versatile device, such as a computer. Therefore it becomes desirable to use a “neutral” device capable of interfacing all these parts. In their proposed architecture DHG is used as central node, and they state that domotic systems usually include a control device which permits management of the devices connected to the BAS, so these control devices act as intermediate points between a particular BAS and the DHG. Any device may communicate with the DHG using its preferred protocol, as the information exchange is handled by a specific driver for each type of device. They also state that device drivers in the DHG are responsible for translating low level or hardware states and activities of the devices into events in the DHG. Therefore, it is necessary to develop specific drivers for each type of device.

The proposed model has two main drawbacks. Firstly, the authors defend that every BAS connected to the DHG need a control device managing all the devices connected to the BAS. As was explained in chapter 1, we can distinguish centralized and distributed architectures in BASs. In centralized architectures this control device (controller) is available, acting as an intelligent unit, holding all the logic of the system. On the contrary, in distributed architectures there is not a central node (controller) storing the logic of the system. In this case, sensors and actuators have to interact and take decisions by themselves; for that reason distributed architecture BASs can not be used in their model.

Secondly, the authors resort to install drivers on the central node (DHG). This method, according to Wang et al. (2007) will lead to too many drivers in one host; it implies that it's a centralized method and it's difficult to add new devices. Furthermore the authors underestimate the difficulties in developing such a number of drivers (one driver for each device). It's an unrealistic requirement since deep knowledge and long time is required for that task.

To conclude this section, the development of the hardware gateway requires significant effort and developers need to understand the technical details of the two protocols for conversion. Considering the large number of protocols in existence today (mainly proprietary protocols) and protocols-specific orientation of gateways, the development of gateways is very costly. Much effort needs to be put into configuring the gateway to map the data points correctly. This makes gateways expensive. The gateway also slows down the response due to the time required for conversion. Furthermore, one can hardly program and configure a controller through a gateway (Wang et al., 2007).

5.2.2 OPC

Traditionally, each software or application developer was required to write a custom interface, or server/driver, to exchange data with hardware field devices. OPC eliminates this requirement by defining a common interface that permits this work to be done once, and then easily reused by HMI and custom applications. The OPC drivers and BAS software with OPC interface can be distributed on different computers, which makes it a distributed system. New components/devices with OPC interface/driver can be added with little difficulty for the unified interface (Wang et al., 2007).

After this protocol description (it was further described in chapter 2) it is time to analyze the reasons behind the success of OPC. As explained before, a lot of OPC components (generally called OPC Server due to its Client/Server architecture) based on OPC Specification have been implemented and have been probed in thousands of applications over several years. There are a lot of factors that contribute to the success of OPC. According to Hong and Jianhua (2006) the detailed reasons why it is so successful can be divided into three areas:

1. Definition of OPC Component. The first base of the good definition is the suitable granularity applied by domain experts to divide the complex process of industrial automation fields into several independent key issues (above mentioned Data Acquisition, Alarm&Events, etc.). These key issues are solved by creating corresponding application standard interfaces. The set of OPC specifications form an integrated architecture solution.

Secondly it has well defined interfaces. Each one of the key issues defines an interface or a set of interfaces. Each interface focuses on exposing core functionality through a set of methods based on the desired functionality. It also specifies meticulously each parameter and the range of values it can take.

Thirdly each OPC component only deals with its special issue. All details are encapsulated in each individual OPC Interface. This minimizes the dependency between components and implies that each OPC Component is characterized by a high cohesion and a low coupling regarding the other components.

Finally as a fourth feature, the definition of an OPC component is based on a fixed platform, since all of them are built under Microsoft's COM/DCOM framework.

Following the commercially successful Windows operating system open doors to a lot of advanced technologies.

2. Implementation of OPC Component. Firstly the implementation is considered well documented. OPC specification gives an all-around definition in order to define the behavior of the interfaces. Secondly the OPC Foundation provides a reference implementation framework in source code style for C or C++ programmers to implement OPC custom interfaces. This sample code provides a starting point for companies to create their own components. Thirdly the OPC specification provides a reference design model defining a series of interaction processes, data organized styles, data exchanging models and object models in detail. Finally, as a quality proof of the resultant implementations, the OPC foundation provides the OPC certification test. By means of the use of 'OPC compliance test tool' and a series of test cases, developers can determine what characteristics of the OPC specification are supported by their products; either can determine and verify whether those features are implemented correctly.

3. Usage of OPC Component. Firstly the OPC components are easy to use in practice, thanks to their features: independent, well-documented, certified, plug and play, updateable, etc. Using friendly-environment features such as monitor the run status of the plant or browse interface showing enabled devices in the bus reduces the OPC users' integration difficulties when deploying OPC-based systems. Secondly the usage of OPC components is characterized by component deliverable styles. The deliverable for most OPC specifications is a high performance custom interface. The generic, working automation "wrapper" module, which can be used by any vendor to turn a custom interface component into an automation component is developed by the OPC Foundation. Automation wrappers extend the applied scope of OPC.

The third main characteristic of the use of OPC components is the availability of multiple platforms. Using a single "XML-wrapper" can connect multiple existing COMbased OPC components to the Internet. This method makes it possible for the OPC to be used in any user desired operating systems, not limited to Windows. The release of the XML-DA Specification is the only present OPC component carrying out this feature. For the near future this characteristic has to be extended to all the components. This issue is one of the objectives pursued by the new OPC Unified Architecture.

Even though the previous description shows an excellent panorama and the OPC specification is considered as 'de facto' standard in the industrial field, many articles do not hesitate to expose its drawbacks. A large summary of them follows at next paragraph.

OPC uses COM/DCOM as the core technology for the software interface. Therefore, when an OPC client on a computer connects to an OPC server located on another computer, the DCOM security must be configured correctly. Many installers experienced this requirement as a problem. As a result, DCOM security is often disabled, leading to severe security risks. Obviously, it gets even more risky when using an OPC server over the Internet. As OPC DCOM interaction over the Internet would result in severe security problems, it is not practical to use it over the Internet (Wang et al., 2007). According to Kapsalis et al. (2002a), although OPC clients provide considerably advanced functionality, they work usually as ActiveX controls. They can also be used over the Internet, by embedding them in Web pages in order to be viewed by ActiveX-enabled browsers (e.g., MS Internet Explorer). Under this configuration, the communication between ActiveX clients and OPC servers is based entirely on DCOM,

which may work over the Internet but is a platform-dependent solution, and furthermore, it has problems with firewalls and TCP ports.

Kapsalis et al. (2003) describes in detail this issue assuring that the OPC foundation, a major standardization organization, has published a multitude of standards related to the integration of industrial type data. These standards describe and enable the seamless integration of control networks with LAN. However, several inherent features of DCOM make it not the ideal technology under all cases. First of all, it is a platform-depending technology, since it is best supported on Microsoft platforms. The second limitation comes from the fact that this protocol was not built with the Internet in mind. This means it doesn't support inherently the access of objects across Internet, resulting in problems when such an attempt is made. The third limitation is that the types of communication messages generated by the COM protocol are very complex and would have trouble being carried over any medium other than they were specifically built for. Sending COM messages over the Internet poses particular difficulties, mainly due to the presence of firewalls.

Kapsalis et al. (2002b) conclude that the two main drawbacks of a DCOM implementation are its platform dependencies, since DCOM and its associated components are best supported on Microsoft platforms, as well as its security problems caused by a lack of firewall friendliness, especially when a system is accessed via the Internet. A solution to these DCOM drawbacks would be to provide accessibility through the Internet, depending entirely and solely on the HTTP protocol. This solution fully exploits the functionality of OPC Servers, without compromises relevant to platform-independence and promoting, in addition, cost-effectiveness, firewall-friendliness and scalability.

XML technology has been used to fill these gaps in a new development of OPC technology. The newest progress is the new OPC XML-DA standard. In this new standard, OPC allows manufacturers to process data which can be accessed via the Internet. In this case, the OPC server is configured as a Web service. However, only OPC DA has been extended with XML capability, the remaining of batch specifications of OPC have not been extended yet (Wang et al., 2007). According to Kapsalis et al. (2003) OPC XML-DA is the first step towards the direction of fully integrating the industrial automation data at the enterprise level. OPC XML-DA is an interface based on XML and SOAP. The data exposed is the same as in the older, DCOM-based OPC-DA. Since XML and SOAP are the fundamental parts of Web Services, OPC-XML actually describes a Web Service. This standard overcomes the two major drawbacks of the DCOM architecture, namely the platform dependence, since it operates on Microsoft platforms, and the firewall unfriendliness.

Another problem of using OPC DCOM servers is that they cannot be accessed from non-Windows systems, OPC DCOM clients cannot communicate with non-Windows systems. Therefore, they cannot be used on other platforms. For that reason a "bridge" or other software of this kind is needed when communicating between various platforms (Wang et al., 2007).

For tracking and monitoring software in BAS, it is important that all data are received by the application without interruption. When an application experiences a bad connection or even gets disconnected, OPC standard is not able to automatically try to reconnect. This problem of OPC in BAS applications needs to be addressed in future development (Wang et al., 2007).

As will be detailed in chapter 5.2.4, there are problems to transfer information from the server to the client when the client doesn't ask for that information. These problems occur when an information exchange between an Internet Server controlling a LAN installation (acting as server) and an external server handling a multi-BAS installation (acting as Client) is implemented by OPC (over a TCP/IP network). Here we have to distinguish between the different kinds of data that can be transferred by OPC; it can be classified depending on the different protocols the OPC specification holds:

- OPC-DA (Data Access).- The original, allows real time data exchange between servers and clients.
- OPC HDA (Historical Data Access).- Historical access to OPC data.
- OPC-AE (Alarms & Events).- It supplies alarms and events notification.
- OPC B (Batch).- Used to manage discontinuous processes.
- OPC DX (Data eXchange).- It supplies interoperability between servers.
- OPC S (Security).- It specifies rules for clients to have access to servers.
- OPC XML-DA (XML Data Access).- It mixes the OPC-XML (eXtensible Markup Language) and OPC-DA.
- OPC CD (Complex Data).- It Allows servers to describe complex data type by means of binary structures and XML documents.

We can mainly distinguish between, on the first hand synchronous data (e.g. used by the OPC DA) where regular information is asked by the client to the server (controlling a certain installation) to display the current state of the systems to the user. The client receives the values of the items according to the configured update time of the data (Alves et al., 2005). On the other hand there is asynchronous data (e.g. used by OPC AE); some functions need the server to send data automatically to the client in case certain events or change of value of some variables occur. Within the context presented in the previous paragraph there is no problem to communicate the client and the server with synchronous (e.g. OPC DA) data types. On the contrary, this communication is much more difficult when asynchronous (e.g. OPC AE) data type communication is required, because the Client is initially not ready to ask for that information; firstly, the server has to inform the client that this kind of data has to be transferred. It can be a problem to implement this communication model with Web Servers. As exposed by Wang et al. (2007) "Server initiated callbacks are not possible; the client has always to poll the server for new data".

OPC XML DA is Web services based, and could be used for communication over the Internet. However, other OPC specifications do not extend to XML yet. If the fact that other specifications like AE are also required in a domotic system is considered, we can conclude that nowadays the communication among BASs over TCP/IP (Internet) can not be conferred on OPC; for that reason XML/Web Services has to be used for the communication over the Internet.

Regarding future versions, in January 2004 the OPC Foundation formed a workgroup with the target of designing a new architecture that provides a viable interoperability framework for the next 10 years and beyond. The first part of the new "Unified Architecture" was released in June 2006. One of the most innovative features is that OPC clients programming is supported under Java, Microsoft .NET and C languages, removing the previous Windows platforms constraint. Furthermore UA combines the existing OPC interface functionality with new technologies like XML and web services. For that reason, if the outcome of this new architecture is successful,

maybe in a near future the OPC specification will be enough to overcome the communication between BASs over Internet.

5.2.3 Web Services

There is a trend to integrate BASs and other enterprise applications, e.g., Management Information System (MIS), Enterprise Resource Planning(ERP). To realize this aim, a promising technology in IB integration is XML and Web Services technology. Wang et al. (2007) claimed "Finally, using TCP/IP connections, protocols like XML will dominate the future of interoperability among embedded devices—even in building automation".

Miori et al. (2006) state that the biggest advantage of Web Services is that they are language and platform independent, and could be accepted and adopted as the universal standard for the application-to-application communication paradigm by everyone. The use of the Web Services paradigm as the basis of the solution, also assures the building of a natural and native distributed architecture.

Craton and Robin (2002) appealed to construct information models based on Web Service. They also claimed that "it might be possible to expose information as XML at a building-controller level; it would not be practical to do so at a zone or unitary controller level." The Continental Automated Buildings Association (CABA) has formed a committee (CABA XML/Web Services Guideline Committee) which "will address the application of new communication standards for Web-based communications such as XML, SOAP and Web services within building control systems", now this committee formed as a technical committee, namely, oBIX (OASIS Open Building Information eXchange Technical Committee), at the Organization for the Advancement of Structured Information Standards (OASIS) (Continental Automated Building Asociation, 2007).

A relevant architectural solution in this field is the DomoNet Architecture designed by Miori et al. (2006), which is depicted in figure 10. It is based on a Service Oriented Architecture (SOA) model, in which the services present the functionalities offered by devices. It essentially consists, on the first hand, of a network connecting applications gateways called TechManagers (TMs) –one per each middleware they want to interconnect, and on the other hand, Web Services, each one handling a different device typology, such as lighting devices. The Web Services can be installed on one or more web servers reachable from the TMs and act as a service container handling a single device typology. The TMs are applications installed on computers connected to both a BAS and the DomoNet network. The TMs are responsible for translating commands between the protocol used in the BASs and the one used in DomoNet. Every TM will register itself to all the DeviceWSs available on the home network. Through these DeviceWSs, a TM will be able to operate devices in other BASs just knowing their offered services. TMs can control "real devices", which are devices available in the BAS connected to the TM, and "virtual devices" which are devices belonging to other BASs registered to DomoNet. Clients (devices) in the BAS will see "virtual devices" as normal appliances that can be controlled using the specific protocol of the BAS. Commands towards these devices will be intercepted by the TM, translated into the protocol used in DomoNet (XML) and forwarded to the right DeviceWS. A DeviceWS exposes all the devices of the same typology and maintain the services they offer in a useful data structure (XML files), aside from which BAS they belong to. They give a uniform view of the entire system to all the BASs. Another task is correctly routing messages between TMs.

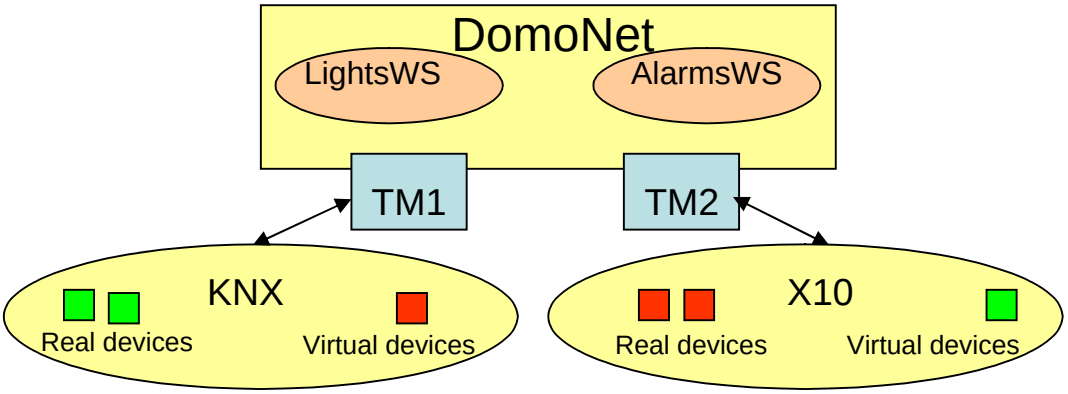

Figure 10: Schema of the proposed model by Miori et al. (2006).

Even if the proposed approach is probably the most complete within this field, making an excellent use of Web Services and XML, the model has again the weak point of the gateways (TMs) translating between specific domotic standards and the "DomoNet language" (XML). The authors resort to installing drivers (for translating specific domotic standards to XML) on the TMs. This method underestimates the difficulty in developing a large number of drivers (One driver for each BAS). It's an unrealistic requirement since a large knowledge and time is required for that task.

Using the Web Service technology, BAS systems from different vendors, even on different platforms, can easily be integrated. A central application controlling a whole system could access different BASs via Web Services, and non-BAS systems could be easily integrated as well. For example, a weather bureau could offer a Web service that allows a building automation system to automatically retrieve temperature forecast data for use by various control algorithms. Similarly, the building automation system itself could offer a Web service that allows a tenant's accounting system to obtain up-to-the-minute figures on energy consumption. However, since the SOAP request/response is enveloped in XML format, it will be too complex to be used in the communication of field level control in some situation and will increase the need to the processor power and additional response time. Therefore, it is not suitable for field level up to now since the overhead level of traffic. With the development of microcomputer technologies and the usage of more and more powerful embedded systems in the market, Web Services may push its applications to field device level (Wang et al., 2007).

5.2.4 Combination of OPC/Web Services

In previous chapters advantages and drawbacks of OPC and Web Services have been shown separately. However, the combination of these two approaches can be beneficial for the domotic and industrial field. In a modern complex, usually multiple sub-systems from various manufacturers are installed. Sub-systems are supervised via their own BAS software systems. These BAS software systems from different manufacturers are based on different hardware platforms and operating systems, providing different communication interfaces. In order to realize integration and interoperation among these different sub-systems, developers have to make various BAS systems communicate effectively. Middleware technologies have been developed to establish the communication among applications. Some of them are DDE, COM/DCOM, OPC, CORBA and JAVA/RMI. Probably due to the broad use in the automation industries and the popularity of Windows platforms, some developers have

applied OPC technology on BAS integration. It seems OPC has broader use than other middlewares.

The requirements for “information integration” are now much broader than those in the past. In particular the broad acceptance and even lowering cost of Ethernet/Internet/XML/Web Services communications is finding its way into the intelligent building industry. This implies that the requirements for a global solution can not be focused uniquely within local systems, they have to be focused on interconnecting them over the Internet.

Considering that most OS platforms of BASs are Windows platforms and OPC has more applications than other traditional middleware technologies, Wang et al. (2007) present a middleware solution which is based on combining OPC and Web Services. They state that the advantage of this combination is their mutual complement that OPC and Web Service have different characteristics and application environments. The advantage of OPC is that it offers a good security level in LAN, not over the Internet. It is also a remarkable fact that OPC can achieve better communication performance in intranet, compared with Web Services. Furthermore the OPC interfaces for several mainstream protocols can be accessed. On the other hand Web Services can provide better security and flexible integration –compared with OPC- over the Internet.

The most representative model implemented within this field was designed by Wang et al. (2007). After a theoretical description, the authors implement some experiments to test the functionality of the system. A schema can be seen in figure 11.

In the proposed model, in LAN, OPC is employed to integrate different BASs. The authors state that generally an OPC interface can be provided by three ways in BAS. One way is through the OPC server interface provided by third-party BAS management software. The second way is the standalone OPC driver (server) related to the controllers. The third way is to transfer other protocols, such as DDE, to OPC. The OPC servers may be distributed on different computers. The OPC client components are responsible for communicating with OPC servers. In this figure, the authors define the Building Management System (BMS) based on LAN as a fully-functional BAS software of LAN version which provides building management functions on LAN mainly based on OPC technologies. It functions as a HMI front end. It provides DCOM interfaces of real time data access and historical data access, too.

These DCOM interfaces are wrapped as Web Services in another server called Web Services Server that provides public Web Services with Internet access. Its main task is not page storage and access. Instead, it is used as http protocol parser since the transportation method for Web Services adopted here is the http protocol. It converts the COM/DCOM interfaces to Web Services interfaces. Users can easily develop their own applications so as to access the Web Services to supervise and control the BAS system via Internet.

The main access point to the complex multi-building system is called “building management web server”. This server links the several independent LAN-installations with external services (through their Web Services Servers) and acts as access point for users. Kits of Active Service Pages (asp) and dynamic link library (.dll) files are deployed within the Building Management Web Server to communicate with the Web Services Servers. These files provide user access interfaces (web pages) to users. Users can manage, monitor and control the connected BASs by browsing web pages. This is a multi-tier architecture.

At this point, an important fact is the communication between the servers. The communication transportation of Web Services is based on http or some other Internet protocol. This communication process belongs to the so-called "Pull" technology. That is, only the client can initiate the process to get information from the server, the server cannot send messages to the client automatically. In the BAS communications, most functions except the alarm and event function are those clients request data from servers (control and monitoring). In the services provided by the Web Services server, most of them can be realized by this "pull" means. The clients (i.e., "building management web server") submit services requests to the Web Services server, the Server will execute the specific services and return the result to the clients. However, some functions need servers to send data automatically in case of certain events, such as Alarm/Event (A/E) and Change of Value (COV) event. When the events come out, the client should be notified immediately if it has subscribed to those events. One way is to refresh the web page quickly so as to keep updated, i.e., "pull" continuously, but this will result in serious network traffic problems. The authors solve this problem by applying a different strategy. They configure the "building management web server" as a Web Services provider, but its services only include the services to deal with A/E and COV events. In this way the Web Service Server can "pull" the "building management web server" to inform it about these events, and later the "building management web server" pulls the specific Web Service Server to obtain this information.

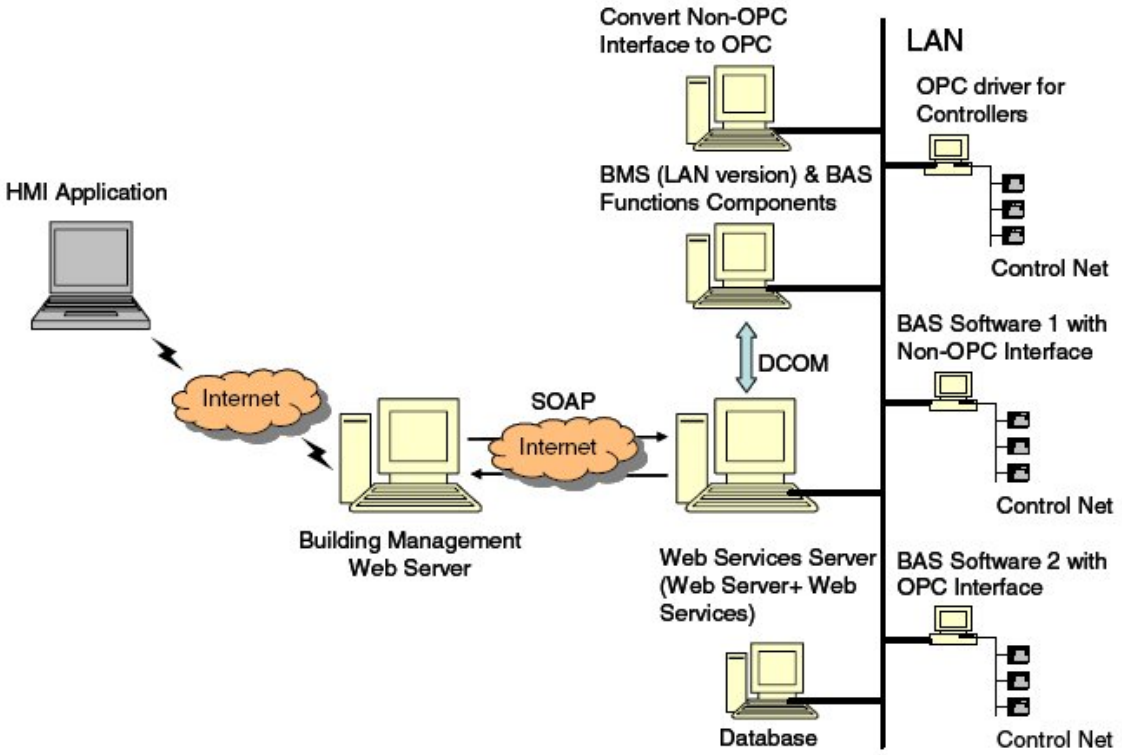

Figure 11: Schema of the multi-building model proposed by Wang et al. (2007).

5.3 Objective3: Applying OPC among KNX networks

As explained above the method used for this objective is the case study. It will describe the different procedures (before and after the release of OPC) followed to interconnect several geographically distributed control networks, creating a global system – all of them following a common standard. The chosen Standard for this study

is KNX, supported by the Konnex association. The reason of this choice is the availability of technical information describing the protocols, networks, hardware and software that is implemented by this standard. It is due to the effort taken by the KNX association to organize and support the scientific research within this field. Another point is the availability of a software tool called KNX OPC Server, which can be used to link several KNX networks; in this way a comparison between the manner the distributed networks were linked before the use of OPC and later, can be established.

Firstly, the study focuses on the procedures followed for that task before the use of the KNX OPC Server. Basically the intercommunication among distributed control networks was achieved by means of the use of a protocol called KNXNet/IP. The aim of this protocol is to interconnect KNX control networks through Ethernet networks in a transparent way. It is based on the use of a device (IP Router) acting as gateway between a control network and a computer. The IP Router acts as server, sharing all the information flowing by the control network with the computer, which acts as client.

The documentation about the KNXnet/IP protocol is available in Volume 3: System Specification Part 8: KNXNet/IP of the documentation provided by the Konnex association through CDs for scientific partners. This description is split in the next chapters: Chapter 1. Introduction to KNXNet/IP. The different parts of the KNXNet/IP protocol are described, next to a description of the offered services and the requirements a device has to fulfil to be able to manage this protocol. Chapter 2 describes the basic format of the KNXNet/IP packets, the different services offered by the protocol and how it manages the communication between a server and a client. Chapter 3 describes the variables required by a device to work as a KNXNet/IP server and be acceded by the high level software called ETS. It also describes the format of the packets used to configure the devices. Chapter 4 describes the tunnelling procedures among devices, the different operational modes as well as the communication protocol and the format of the packages. Chapter 5 describes how to route KNXNet/IP packages over the Ethernet network.

As a summary, the documentation states that the KNXNet/IP protocol can mainly achieve the following actions:

- Configure KNXNet/IP devices.
- Monitor the KNXNet/IP network.
- Allow the communication of devices though the server.
- Link several KNX networks, like if they were a single one.

The method used to link several KNX networks it is based on the use of the Internet Protocol (IP). A schema is shown in figure 12. The fist important point in this schema is the IP Router, managing each KNX network. They are responsible for determining which information flowing within the KNX network has an external destination (another KNX network), encapsulate that data within UDP packets, in the space destined to data (encapsulated as an KNXNet/IP protocol frame), and send these packets to a client PC. After that, the next responsible for the communication is the PC acting as a client, which is responsible for receiving the UDP packets from the IP router and to forward them trough the Internet towards the client PC linked to the IP Router managing the destination KNX network. The data encapsulated within the first UDP packet (KNXNet/IP frame) can be encapsulated in a TCP or UDP packet when it is forwarded from the first client to the second one through the Internet. It will depend on the quality and speed of the connection available between them. Finally the second

client will unpack the KNXNet/IP frame and encapsulate it again in a new UDP packet that will be sent towards the IP Router managing the destination KNX network.

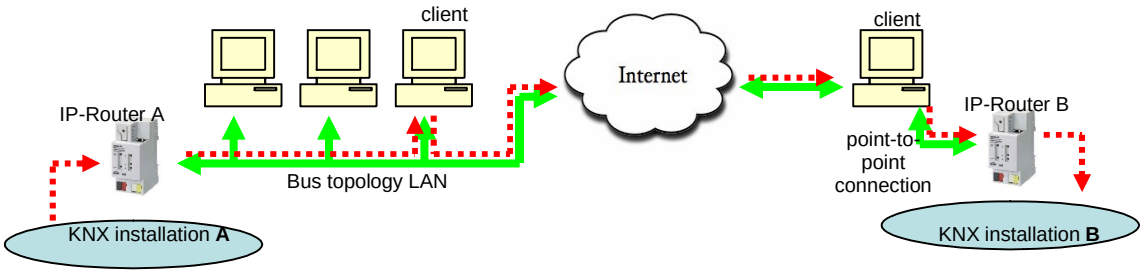

Figure 12: Schema of the model implemented by KNXNet/IP.

5.3.1 KNXNet/IP frame

After having depicted the general running of the system, the description focuses on how the information which describes the status of a KNX installation is organized within KNXNet/IP frames; it is also important to classify the types of frames that can be transmitted. A KNXNet/IP frame is encapsulated in a TCP or UDP packet, in the space destined to data. KNXNet/IP frames are composed by:

- Header.
  - Header size (8 bits).
  - Protocol version (8 bits).
  - Type of KNXNet/IP service (16 bits). Indicates the "kind" of packet, the "funcionality".
  - Size of the header + size of the body (16 bits).
  - Configure other devices within the KNX Network (ascribe directions, groups,etc.).

0..... ...7	8..... 15
Header Length	Protocol version
Service identifier	
Header Length + Body Size	

Figure 13: KNXNet/IP frame.

- Body.
  - With variable format and length depending on the kind of service.

5.3.2 Tunneling frames

As can be seen in the previous section, the body of the frame has a different format and length depending on the kind of service. Let us focus on the tunneling frame; it is the most important kind of frame since it allows a device to send and receive data to/from another device located in a different control network, reachable by a Ethernet network.

There are 2 main frame formats within the tunneling frame, the send/reception frames and the acknowledgement frames (ACK)

- TUNNELING REQUEST, with service identifier 0420h.
- TUNNELING ACK, with service identifier 0421h.

Furthermore, there are specific frames for the connection with the KNXNet/IP server, but let us focus on the reception/send and ACK frames, since they are the most important in order to understand the communication. A graphical description is shown in figures 14 and 15.

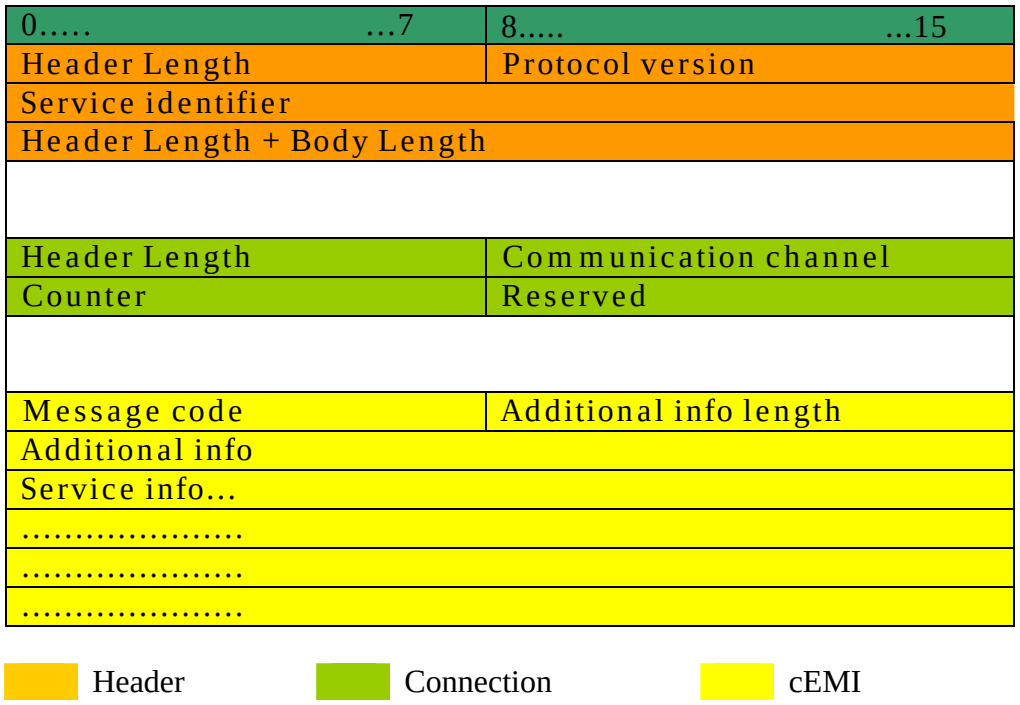

Figure 14: KNXNet/IP REQUEST frame.

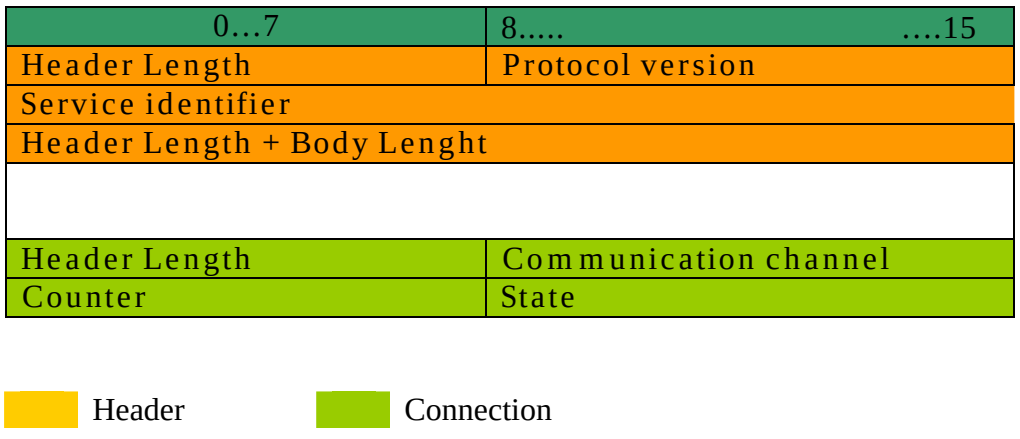

Figure 15: KNXNet/IP ACK frame.

cEMI package. When sending or receiving a request frame through the Ethernet network, the information concerning group destination address, which device created and sent this package and the data is sent, are encapsulated in a package called cEMI

(common External Message Interface). An example with a complete frame sent by the IP-Router is shown in figure 16.

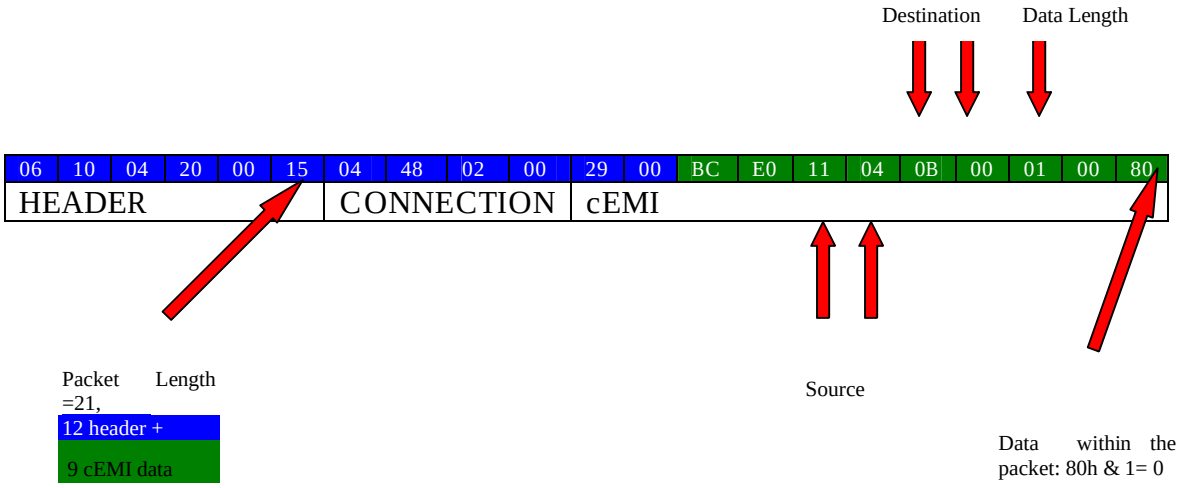

Figure 16: KNXNet/IP frame.

5.3.3 IP Router N146

The device allowing the IP communication between the KNX network and the PC is the IP Router N146, manufactured by Siemens, member of the KNX association. The IP router is directly plugged to the KNX network; furthermore it is provided with a RJ45 connector, which allows communication with the client PC through Ethernet or point-to-point connection. This device can be configured in a static or dynamic way. In the static way, an IP direction is manually assigned to the device; in order to assign this direction it is necessary to use a high level software tool called ETS. On the other hand, if the dynamic way is chosen, it is necessary to have a DHCP (Dynamic Host Configuration Protocol) server within the network where the N146 is connected. Regarding the connection between the IP Router and the computer acting as client it is normally implemented by a point-to-point connection with twist pair cable.

Figure 17: IP Router.

Once the IP Router has been configured, it is possible to connect with a client by using the KNXNet/IP protocol. As a facility, the device includes LEDs which inform when data is being received/sent from/to the KNX control network it is connected to.

This device does not allow the connection of several clients (meaning the computer interacting with the IP Router) simultaneously. That means that in case the IP Router is connected to a bus topology LAN, only one of the computers within this LAN can act as client.

Another point is that the device needs an electrical feed (12V). Finally it has to be taken into account that the communication with the client PC is performed through a UDP connection, using the port 3671.

5.3.4 KNX OPC Server

The second part of this chapter concerns the procedures followed to interconnect domotic networks that are distributed geographically, with the use of the OPC standard. In order to establish a comparison with the previous sections of this objective, this study is focussed on the use of the OPC technology over the KNX standard. In this case, it will study how to establish an OPC server controlling a KNX installation, and what performance can be obtained by the user.

If we start thinking about the physical installation, the first important issue is to determine the kind of media of the network linking the devices. The KNX standard includes several communication media. Firstly, the TP-1 (Twisted pair, type 1), with a defined bitrate of 9600 bits/s; it is the most common medium in most of KNX installations. Secondly, the power line, that can be defined as PL-110, (Power-line, 110 kHz) with a bitrate of 1200 bit/s or as PL-132, (Power-line, 132 kHz) with a bitrate of 2400 bit/s. Thirdly, the RF (Radio frequency on 868 MHz) with a bitrate of 38.4 kbits/s. Finally, Ethernet (KNXNet/IP); this widespread communication medium can be used in conjunction with the “KNXNet/IP” specifications, which allow tunneling of KNX frames encapsulated in IP frames; however this last medium is not advisable for any kind of installation, on the contrary its use is focused on intercommunication aspects, as described in chapter 5.3.1.

The next step when establishing the system is the communication between the network and an external PC acting as client. The available methods for KNX network access are USB, EIA-232 interface and KNXNet/IP. For the use of the first ones (USB and EIA-323) it is necessary to install a gateway hardware which is responsible for parsing the internal KNX telegrams among devices to the interface they manage (USB or EIA-232). The last one is based on the use of the IP Router described above: its advantage is that it is hardware independent and the cEMI message format covers all media; on the other hand the price of this option exceeds the price of the previous ones.

After this interface between the KNX network and an external PC, the next element to study is the KNX OPC Server, which is installed in that external PC linked to the KNX network. This tool is described (KNX, 2007) as a powerful tool for linking KNX networks, with an easy management and visualization through an international, system-wide industry standard communication protocol - OPC. The result is that for a wide range of software packages (OPC clients) the KNX OPC Server allows easy access to KNX runtime services such as group addressing and device checks. In this context high level software tools used to control an industrial plant through the OPC technology are normally called visualization or control programs.

The biggest advantage this technology provides is to have access to several networks, machines or devices from only one visualization program. Furthermore, due

to the tool's compatibility, it is possible to access and control any device or net from virtually any standard process visualization toolkit or environment on Windows platforms. It is also described from a technological point of view as an OPC Server layer put on top of a high level software tool developed by the same company, called *eteC Falcon*, that is described as a high level component to have access to KNX networks. Falcon is a DCOM-based 32-bit access library for Windows. It offers a comfortable API for all aspects to access or manage buses and devices.

Regarding how the KNX OPC Server is implemented, it entirely relies on the Data Access (OPC-DA) part of the OPC Specification. It does not support Alarms & Events (OPC-AE). According to Hong and Jianhua (2006) OPC DA provide specifications for communicating real-time data from data acquisition devices to display and interface devices. It also supports multi-tier implementations. The OPC Data Access interface can be used to talk from the OPC client 'down' to the devices installed in the network as well as from the devices 'up' to the higher level OPC client application. As was described in chapter 5.2.2 OPC DA is based on a synchronous exchange of information. That data is asked by the Client to the Server (controlling a certain installation) to display the current state of the systems to the user. The client receives the values of the items according to the configured update time of the data. The communication can also take place in the other direction (from the client to the server).

Finally we should take into account a distinction. With the previous stated use of an IP interface to KNX networks (IP Router), the KNX OPC-Server supports also KNXNet/IP connections. It is important not to confuse this with the IP capabilities of OPC itself since also through the OPC connections IP can be used (between clients and servers). For clarifying all the information presented in this chapter, figure 18 presents a system composed of an OPC client managing several OPC servers. All of them (meaning the servers and the client) are distributed over the Internet. The OPC client manages 2 KNX installations and a LonWorks installation through their OPC Servers.

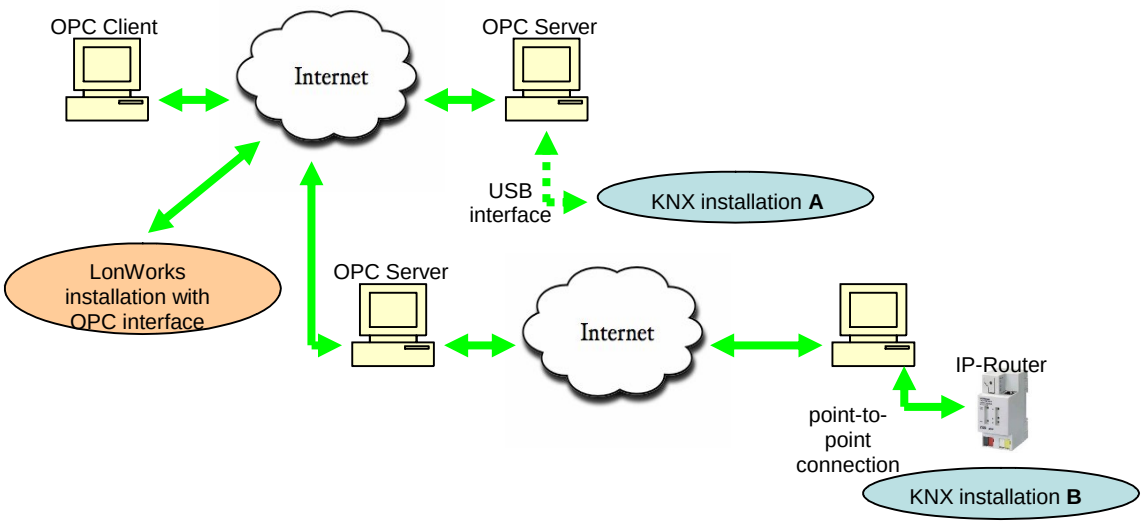

Figure 18: OPC client handling heterogeneous domotic installations.

As last point of this chapter, regarding the information provided by the domotic standard companies about the interoperability concept, some information can be found in their web sites. Generally, the existence of strong commercial influences can be perceived. Vendors do not refer to other vendors or companies on their websites. There

is a general trend to show only their own product selection. Furthermore, when the interoperability issue is mentioned, it is normally referred to the interoperation between products developed by different companies following the same standard (e.g. KNX). Even when the technical description of OPC servers controlling specific installations are analyzed, no references to the possible development of an OPC client controlling OPC Servers that manage heterogeneous installations can be found.

To have a more clear idea, let us focus on a particular case. The KNX association in its website states that in order to profit to the full extend of the use of a Home and Building Control system, it is indispensable that products of different manufacturers ("multi-vendor interworking") and products of different application fields ("cross-discipline interworking") interwork. According to their description the advantages of such interworking are:

i. It enhances the OEM market amongst KNX solution providers.

ii. It gives easy market access for niche products (commonly made by smaller companies).

iii. It allows setting up a common market infrastructure for KNX compatible products (e.g. training measures).

This example shows clearly the trend to focus the interoperability issue exclusively on the products developed by each group. The strong commercial influences imply a counterproductive fact for the growth of the domotic discipline.

6 Conclusions

After have undertaken a large research process, this chapter evaluates whether the purposed aim has been achieved by the results shown in chapter 5. The conclusions chapter ends this report. It's structured in 2 sub-chapters. Chapter 6.1 establishes some final conclusions that summarize and analyze the most important information depicted in chapter 5. Chapter 6.2 identifies some future work for continuing the research in this field.

6.1 Final Conclusions

This report has shown the existence of a set of problems making difficult the domotic field growth. They are mainly the lack of a common communication standard among devices and the existence of a proprietary market, where each provider focuses on developing its own devices, protocols and interfaces. There is not a convergence criterion in order to overcome this problem by the main domotic providers. In order to overcome this situation the aim of the project was defined as determining if a solution that solves this problem can be achieved by comparing or combining several previously used methods.

This aim has been reached by following 3 objectives. In the first objective, before undertaking any kind of study of the methods, it is essential to determine the requirements of a global solution that completely solves the problem, that is to say, how should be a solution that solves the problem? This first objective was reached by undertaking a literature analysis that identified the requirements stated by several researchers. Those requirements where classified in two layers, local and Internet area, and shows the different perspective that have to be applied to design systems depending on their distribution.

The second objective puts these requirements into context by analysing the different methods that can be used to reach a solution. The method was again a literature analysis that examines the models proposed by previous studies. Several studies try to overcome this problem by applying different strategies. The aim is to design and implement an overall system linking several heterogeneous installations. The methods that have been used for that task have been the residential gateway, web services and the use of the OPC standard. These methods have not been able to overcome the problem when they have been applied separately. The main problems associated with each method are:

- Residential gateways convert a protocol to another protocol by mapping data points from one protocol to another protocol. This method resorts to installing drivers on a central node and leads to too many drivers in one host; it implies it's a centralized method and it's difficult to add new devices. Furthermore the development of the hardware gateway requires a significant effort and developers need to understand the technical details of the two protocols for conversion. Considering the large number of protocols in existence today (mainly proprietary protocols) and the protocols-specific character of gateways, the development of gateways is very costly.
- The biggest advantage of Web Services is that they are language and platform independent and should be used for extending the interoperability over the Internet. However, since the SOAP

request/response is enveloped in XML format, it will be too complex to be used in the communication of field level control.

- The OPC standards describe and enable the seamless integration of control networks with LAN. However, several inherent features of DCOM make it not the ideal technology under all cases. This protocol was not built with the Internet in mind. This means it inherently does not support the access of objects across Internet, resulting in problems when such an attempt is made.

For all those reasons it becomes essential to design a new strategy, which should be able to achieve the proposed aim. A combination between the features presented by the previous methods can lead to a solution. The combination between the OPC standard and web services has been described by some studies as the best one to implement a system that reaches all the required features.

Specifically, this solution presents a system divided into two layers. The OPC standard is used to link several different field buses in the local area. Later, web services communicate every local installation over the Internet; furthermore a central node handles the overall logic of the system, and acts as access point for user interfaces.

Finally the third objective was a case study that tried to give a pragmatic point of view of the benefits obtained by a domotic developer (KNX) by using the OPC technology. That case study showed the commercial influences within the market. Each particular developer uses the depicted technologies to provide 'interworking' between their own systems; however, they don't seem to be interested in achieving interoperability with the products of their competitors.

After this summary it becomes necessary to discuss how the results fit in the wider context of the subject area as a whole. For that reason a set of questions should be answered. Firstly, the results can be conceived as a contribution to the research within the domotic field, since several studies with different orientations have been examined, and a set of common requirements has been extracted. Subsequently, the proposed models have also been compared, identifying their weak and strong points. By combining them, a solution that fulfils the aim of the project has been found. Secondly, the findings of this report are in line with other related work, they confirm the existing framework in the field. Thirdly, the results are a contribution of a theoretical nature, but they are the first step for subsequent real-world applications; the proposed model should be implemented and tested in future studies, and if their outcomes are positive, it could contribute to the real interoperation among heterogeneous domotic systems. Finally the results can be beneficial from other related research areas, such as the industrial field. In general, some scenarios related with communication among products following proprietary protocols, can obtain new outcomes by applying the techniques depicted in this report.

6.2 Future Work

Firstly it would be necessary to strengthen the described model by the development of more practical implementations. Basically, the current studies have a theoretical point of view. Some of them include implementations but they are quite often based on the use of OPC Server simulators. It would be essential to implement a system linking at least two local sub-systems, each one controlled by an OPC client

handling heterogeneous installations (at least two) through their own OPC servers. The results of the experiment could determine the real performance of the systems within the local sub-systems and in the overall system. In case the results are successful, it would also be important to determine the consequences this new technique that provides interoperability can imply for the commercial frame. Some questions arise: Will the domotic providers face the interoperability issue? Or on the contrary, will they continue with the current policy? If they do not face it, will third-party companies be able to implement OPC Clients managing specific OPC Servers? Or will the providers avoid that? How can a domotic provider obtain more benefits, by continuing to develope their own line of products or by cooperating with other providers?

Finally it can be established the necessity to determine whether the future OPC Unified Architecture release will overcome the interoperability problem by itself or not. OPC UA programming is supported under Java, Microsoft .NET and C languages, removing the previous Window platforms constraint. Furthermore UA combines the existing OPC interface functionality with new technologies like XML and web services. For that reason, the OPC standard could be enough to overcome the communication between BASs over the Internet.

References

Alves Santos, R., J.E. Normey-Rico, A. Merino Gomez, L.F. Acebes Arconada, C. de Prada Moraga. (2005) OPC based distributed real time simulation of complex continuous process. *Simulation Modelling Practice and Theory*. **13**, pp. 525-549.

Andersen, I. (1998) Den uppenbara verkligheten: Val av samhällsvetenskaplig metod. *Lund: Studentlitteratur*.

Berndtsson M., J. Hansson, B. Olsson and B. Lundell. 2002. Planning and implementing your final year project. *London: Springer*.

Chemishkian,S., J. Lund. 2004. Experimental Bridge LonWorks/UPnP. *CCNC Conference.*

Craton, E., D. Robin (2002). Information Model: The Key to Integration. *Automated Buildings*[online]. Available from the from World Wide Web: <http://www.AutomatedBuildings.com>

Domotics. 2007. [online]. [Accessed 5th March 2007]. Available from World Wide Web: <http://en.wikipedia.org/wiki/Domotics >

Echelon Corporation, LonWorks developer [online] [Accessed 10th March 2007]. Available from World Wide Web: < http://www.echelon.com >

Hintze, C. 2007. Reconnecting the connected home. *Electronic Design*, **55**(2), pp.36-39.

Hong, X., W.Jianhua (2006). Using standard components in automation industry: A study on OPC Specification. *Computer Standards and interfaces*, **28**, pp.368-395.

Intelligent building. 2007. [online]. [Accessed 8th March 2007]. Available from World Wide Web: < http://en.wikipedia.org/wiki/Intelligent_building>

Kapsalis V., K. Charatsis, M. Georgoudakis, G. Papadopoulos. 2003. Architecture for Web-based services integration. *IEEE*. 0-7803-7906-3/03

Kapsalis V., S. Koubias, G. Papadopoulos. 2002a. OPC-SMS: a wireless gateway to OPC-based data sources. *Computer Standards & Interfaces*, **24,** pp. 437–451.

Kapsalis V., K. Charatsis, A.P. Kalogeras, G.Papadopoulos. 2002b. Web Gateway: A platform for Industry Services over Internet. *IEEE*. 0-7803-7369-3/02

Knorr, E., P. Venezia. 2007. SOA. *Computer Architecture*, **29**(1), pp.16-18.

KNX Association [online] [Accessed 10th March 2007]. Available from World Wide Web: < http://www.konnex.org >

Koloktsa, D., G. Saridakis, A. Poiliezos, G.S. Stavrakis (2005) Design and installation of an advanced EIB fuzzy indoor comfort controller using Matlab. *Energy and Buildings*, **38**, pp. 1084-1092.

Lee, E. A., S. Neuendorffer. 2000. MoML — A Modeling Markup Language in XML. *Technical Memorandum ERL/UCB M 00/12*

Mateos, F., V. M. Gonzdlez, R. Pot, M. Garcia, R. Olaiz (2006). Design And Development Of An Automatic Small-Scale House. *31st ASEE/IEEE Frontiers in Education Conference T3C-1*.

Miori, V., L. Tarrini, M. Manca, G. Tolomeo (2006). An Open Standard Solution for Domotic Interoperability. *IEEE Transactions on Consumer Electronics*, **52**(1), pp.97-104.

Olsson, M. (2002) Data Warehouse – An Outlook of Current Usage of External Data. *Master Thesis HS-IDA-EA-02-407*, Department of Computer Science, University of Skövde.

OSGi Alliance [online] [Accessed 13th March 2007]. Available from World Wide Web: <http://www.osgi.org/ >

Pellegrino, P., D. Bonino, F. Corno. 2006. Domotic House Gateway. *21st Annual Symposium on Applied Computing, Dijon, France.*

Rüping, S. H. Klugmann, K.-H. Gerdes, S. Mirbach. A modular OPC-Server connecting different Fieldbussystems and Internet Java Applets. *Citeseer*.

SENDA [online] [Accessed 14th March 2007]. Available from World Wide Web: <http://arco.inf-cr.uclm.es/ >

Telefóncia. 2003. *Libro blanco de las comunicaciones y el hogar digital*. [online]. [Accessed 16th March 2007]. Available from World Wide Web: <http://www.telefonica.es/sociedaddelainformacion/html/publicaciones_libroblanco.shtml>

The Continental Automated Buildings Association [online] [accessed 6th May 2007]. Available from World Wide Web: < http://www.caba.org/index.html >

The OPC Foundation [online] [Accessed 14th February 2007]. Available from World Wide Web: < http://www.opcfoundation.org >

Thomas, B., M. Soleimani-Mohseni. 2007. Artificial neural network models for indoor temperature prediction: investigations in two buildings. *Neural Comput & Applic*, **16**, pp.81–89.

Wang, S., Z. Xu, J. Cao, J. Zhang (2007) A middleware for web service-enabled integration and interoperation of intelligent building systems. Automation in Construction, **16**, 112 – 121.

Weiss, M., B. Esfandiari, Y. Luo. 2006. Towards a classification of web service feature interactions. *Computer Networks*, **51**, pp.359–381.

X10 Knowledge base. 2007. [online]. [Accessed 10th March 2007]. Available from World Wide Web: < http://kbase.x10.com/wiki/Main_Page >