Alexander T.E.I. of Thessaloniki
School of Technological Applications
Department of Informatics

Advancing Building Management System technologies for energy saving with Information and Communication Technologies

Bachelor's Thesis in Informatics performed at NEC Laboratories Europe

by
Stefanos Tragalos
(08/3322)

Thesis Supervisors

Mischa Schmidt (NEC)
Dr. Periklis Chatzimisios (ATEITHE)

Heidelberg - Thessaloniki
March, 2012

Foreword

As an integral part of the undergraduate studies of the Department of Informatics at the Alexander Technological Institute of Thessaloniki, the development of a bachelor's thesis is assigned to the students before their graduation. The current thesis has the title "Advancing Building Management System technologies for energy saving with Information and Communication Technologies" and is the result of the past months of interesting and exciting work at NEC Laboratories Europe in Heidelberg, Germany on the European Seventh Framework Programme CAMPUS21 project (EU FP7, Control and Automation Management of Buildings and Public Spaces). The supervisors of the current thesis are Mischa Schmidt, Senior Researcher at NEC Laboratories Europe and Dr. Periklis Chatzimisios, Assistant Professor at Alexander T.E.I. of Thessaloniki.

This thesis analyses the existing technologies on Building Automation and Building Management Systems, and distinguishes their benefits and disadvantages. Furthermore, it evaluates their ability to provide functionalities that could improve the BMS performance on energy saving and identifies possible ways of reusing the existing building automation technologies in a more efficient way or optimising them by using various Information and Communication Technologies.

The first step was a research on the existing BMS technologies. Moreover, it was vital to identify their problems and the possible solutions to them. For this purpose, some demo sites were available through the CAMPUS 21 project.

In summary, this thesis focuses on the research as well as on possible improvements of the existing technologies. Furthermore, it introduces the remote optimisation of the energy consumption through monitoring the buildings and evaluating the concept using also external data, such as weather forecasts, energy provider's pricing, etcetera.

Abstract

As the level of automation on building infrastructure was increased rapidly the last few decades, the need of effortless management of building automation services led to the development of the first Building Management Systems (BMS). The “BMS” term was introduced in the early 1970s, when for the first time complex electronic devices were able to operate the devices of the building infrastructure and retain data of their operation. Today's BMS are computer-based control systems that provide monitoring and management of the automation devices installed in a building infrastructure.

This thesis focuses on defining the technologies used by the BMS in our days as well as identifying their advantages and disadvantages. Moreover, the main scope is to define an ideal implementation of a BMS that could provide improved management and flexibility of the existing building automation, giving the opportunity to achieve optimised energy performance and power saving. This implementation has to overcome the existing technological barriers which limit today's BMS flexibility and extensibility and introduce novel ideas on advancing the BMS technologies.

By extracting requirements from theoretical and existing use cases, the state of the art of an implementation is presented and the key innovative device is defined. In addition, decisions on the ideal technologies that could be used for its development are made and justified.

Finally, a prototype of the prior art architecture is developed, together with a virtual building automation field which implements some virtual devices for testing and demonstration purposes. A simple scenario is demonstrated and its results are analysed.

Acknowledgement

Foremost, I am heartily thankful to my supervisors Mischa Schmidt and Dr. Periklis Chatzimisios for the continuous support, guidance, patience, understanding and most importantly the amicable relationship they offered me.

Especially I would like to thank Mischa Schmidt for creating a very friendly working environment during the 6 months of my internship at NEC Laboratories Europe (NLE) at Heidelberg and for all his wise advices that I found really helpful. The last 6 months by his guidance I manage to see things from a different point of view, much more mature and smart. His mentorship was paramount in providing a well rounded experience consistent my long-term career goals.

Moreover, I deeply thank Dr. Periklis Chatzimisios for being always there guiding and helping me and other colleagues during our studies on A.T.E.I.THE. Many times he gave me the filling that he is not only a professor but also a friend. This thesis wouldn't have been possible if he didn't support and help me when it was needed. His confidence on me helped me to continue trying my best on what I did.

I also want to thank Dr Hans-Jörg Kolbem, Manager at NEC Laboratories Europe at Heidelberg, for providing uncountable wise advices though very interesting discussions during my internship.

Furthermore, I would like to thank Surendran Palanimuthu, who was placed to the office next to mine for the last 6 months and has been a real friend. Working at NEC was much more funny and enjoyable with colleagues like him.

Finally, I would like to thank all the students and stuff that I met and cooperate with. This is an experience that I will never forget.

Stefanos Tragalos

Contents

Index of Tables

Index of Figures

Abbreviations and Acronyms

6LowPAN	IPv6 over Low power Wireless Personal Area Networks
AHU	Air Handling Unit
ANSI	American National Standards Institute
APDU	Application Protocol Data Unit
API	Application Programming Interface
ARCNET	Attached Resource Computer Network
ASHRAE	American Society of Heating, Refrigerating and Air Conditioning Engineers
B-AAC	BACnet Advanced Application Controller
B-ASC	BACnet Application Specific Controller
B-BC	BACnet Building Controller
B-OWS	BACnet Operator Workstation
BACnet	Building Automation and Control Network
BEMS	Building Energy Management System
BIM	Building Information Modeling
BMS	Building Management System
BTL	BACnet Testing Laboratory
CABA	Continental Automated Buildings Association
CAMPUS21	Control and Automation Management of Buildings and Public Spaces
CHW	Chilled Water
CoAP	Constrained Application Protocol
COM	Component object model
CSMA/CA	Carrier Sense Multiple Access/Collision Avoidance
DA	Data Access
DCOM	Distributed component object model

EIA	Electronic Industries Alliance
EIB	European Installation Bus
ETH	Ethernet, IEEE 802.3
FIFA	Fédération Internationale de Football Association
FP7	The Seventh Framework Programme
gbXML	Green Building XML
GUI	Graphical User Interface
HDA	Historical Data Access
HTTP	Hypertext Transfer Protocol
HVAC	Heating Ventilating and Air Conditioning
IAI	International Alliance for Interoperability
IAs	Interoperability Areas
IBC	Intelligent Building Controller, Intelligent Proxy's Operating System
ICT	Information and Communication Technologies
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IFC	Industry Foundation Classes
IP	Internet Protocol
ISO	International Organization for Standardization
IT	Information Technology
JSON	JavaScript Object Notation
KNX	Konnex
LAN	Local Area Network
LON	Local Operating Network
LPHW	Low Pressure Hot Water
NLE	NEC Laboratories Europe
NPDU	Network Protocol Data Unit
OASIS	Organization for the Advancement of

	Structured Information Standards
oBIX	Open Building Information Exchange
OLE	Object Linking and Embedding
OPC	OLE for Process Control
OSGi	Open Services Gateway Initiative
OSI	Open Systems Interconnection
PICS	Protocol Implementation Conformance Statement
PTP	Point-to-point
R&D	Research and Development
RESTful	Representational State Transfer
SCADA	Supervisory Control and Data Acquisition
SDK	Software Development Kit
SNVT	Standard Network Variable Type
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
TCP	Transmission Control Protocol
UA	Unified Architecture
UCC	University College Cork
UDP	User Datagram Protocol
UML	Unified Modeling Language
UpnP \ PnP	Universal \ Plug and Play
VIP	Very Important Person
W3C	World Wide Web Consortium
WLAN	Wireless Local Area Network
XML	Extensible Markup Language

Chapter 1

1 Overview

1.1 Introduction

The level of automation in residential and commercial buildings has increased rapidly the last few decades. The need of comfort and effortless management of the building services like heating, ventilation, lighting etcetera, in addition with the benefits on energy managing and saving were the main reasons for developing Building Management Systems (BMS).

In early years of building management, manpower was needed for controlling the systems. For example, it was common to manually, with the use of a handle, increase or decrease the heating. However, the “BMS” term was introduced in the early 1970s, when the use of complex electronic devices that are capable of retaining data, for management and controlling purposes of the building infrastructure functions such as lighting and heating. Nowadays, a BMS is a computer-based control system installed in buildings that provides monitoring and management of the mechanical and electrical equipment in an automated way. Furthermore, it gives the facility manager the ability to have a central control and monitoring of all the systems that are installed and integrated to the BMS. A BMS consists of software and hardware that that are responsible for these tasks.

In building automation the distributed system can be depicted using a three layer model (as shown in Figure 1.1). Field layer, which is responsible for measurements, position and metering and includes the sensors and actuators. Automation layer is responsible for measurement, control and regulation processes and includes controllers and gateways. Finally, the Management layer, where monitoring, energy management and operation management functions are being performed.

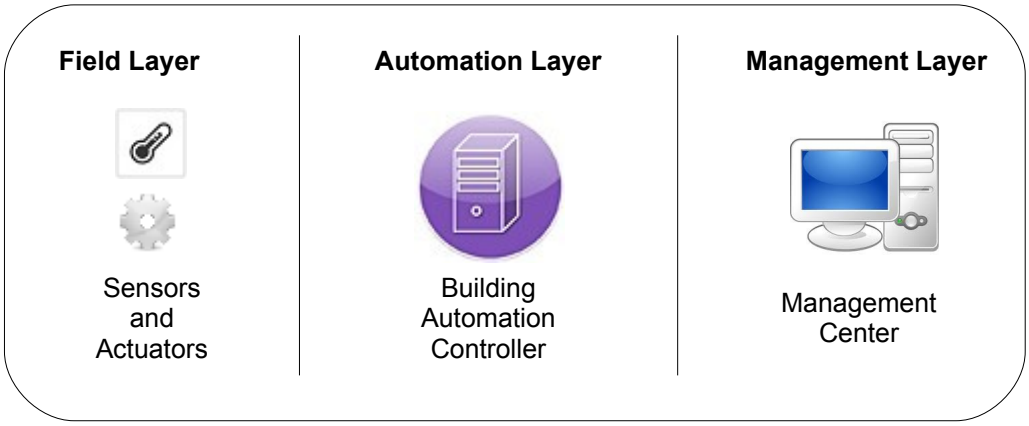

Figure 1.1: The Three Layer Model of Building Automation

In each level, different technologies and protocols are used in order to complete simple or more complex tasks. This thesis focuses on a research of the existing protocols and technologies, their benefits and disadvantages. Moreover, it introduces new technologies and some more efficient solutions on building management and energy saving.

1.2 Scope

Initially, a literature study was carried out about building automation and building management. Furthermore, an extensive study and research on the existing protocols and standards on the global market had to be made. The next step was to identify the technologies used in the existing infrastructure and distinguish the possibilities of improvements on the BMS in the existing use cases of the demo sites in the European project CAMPUS21 (Control and Automation Management of Buildings and Public Spaces, FP7), in which I was involved during my internship at NEC Laboratories Europe (NLE), without vital changes in the infrastructure. After these initial studies, a middle-ware software for the BMS had to be designed that would provide interoperability between the different systems and devices used, so centralised optimisation functions and algorithms could run in one or more cloud servers and return to the BMS recommendations so the energy management and consumption could be improved by the knowledge of external data. Also, use of some virtual devices through an emulation of a BMS, written in

Java, was vital for better understanding of the protocols, the technologies and some use cases. As a result, deeper theoretical and practical knowledge of the software architecture was gained.

1.3 Purpose

The primary purpose of this thesis is to study and analyse the existing BMS technologies and distinguish improvements that could be made in order to design the “ideal” BMS. Following the study and theoretical knowledge gained on the building automation, the next goal is the design of the middle-ware software that would provide interoperability, between the existing devices and sub-systems of different manufacturers or generations, in a transparent way and would give the ability to the hole facility to be operated as one complete system. Moreover, a device called “Intelligent Proxy” is defined and decisions on protocols, data models and functions of the BMS are made and justified so an ideal implementation of a BMS could be developed later on.

1.4 Outline

The rest of this thesis is organized as follows. In Chapter 2, the theoretical background of the existing building automation technologies is provided, including the most commonly used building automation protocols around the globe and data models that could be used for the purposes of a BMS. Chapter 3 provides the state of the art of a device defined as “Intelligent Proxy” that could meet the requirements of advancing the BMS technologies, which were extracted from theoretical and existing use cases. Further on, in Chapter 4, the implementation of a prototype for this device's operating system and a virtual building automation field are analysed, along with a demonstration scenario. Finally, Chapter 5 concludes the thesis, providing results as well as interesting relative topics for further research and development.

Chapter 2

2 Building Automation Technologies

2.1 Introduction

Since many different technologies are used in order to achieve the goals of a BMS, in this chapter we discuss communication protocols and data models used in building automation. Moreover, as it is known, the interoperability and extensibility on the BMS are few of the vital needs of the market and concerns the development of a building automation product, hence we are going to present standardised technologies.

By providing standardised solutions, we gain advantages on compatibility, interoperability, repeatability, quality and of course on the development cost.

2.2 Building Protocol Standards

2.2.1 Introduction

In building automation industry, the need of communication between the devices is vital in order to complete the main tasks of a BMS and, thus, several communication protocols were developed. As the level of automation in buildings increased and the services provided by the BMS changed through the years, these protocols had to be updated or replaced. Different manufacturers developed a range of protocols, and as a result to that there was no interoperability between the devices of different manufacturers or even the newer devices with the older of the same manufacturer. This created the need for the development of a standardized way of communication between the devices. Moreover, manufacturer and platform independent protocols were developed, the most commonly used in the global market are going to be presented in this chapter.

2.2.2 Building Automation and Control Network (BACnet)

BACnet (Building Automation and Control Network) is a standardized data communication protocol developed by ASHRAE (American Society of Heating, Refrigerating and Air Conditioning Engineers) for use in building automation to enable devices and systems to exchange information. BACnet was developed to cover the need for a standardized data communication protocol that would allow devices and systems of various vendors and manufacturers to communicate, and its development focused on providing interoperability. BACnet is world-wide the most commonly used protocol in building automation and it is an international standard (ISO 16484-5 standardized on 2003). Hence, heating, ventilating, air conditioning, lighting, security, fire and other building automation systems are able to communicate with each other by using BACnet and its standardized methods for presenting, requesting, interpreting, and transporting information [3],[4].

2.2.2.1 Basic Principles and Interoperability Areas

The main goal of BACnet is to provide interoperability. Interoperability is the ability of devices from different vendors to cooperate together without faults, so they could exchange information in order to accomplish complex or simple building automation services in a computerized BMS. Furthermore, the compatibility does not refer only on the interconnection of devices from a variety of manufacturers but also of devices from the same manufacturer from different generations. BACnet therefore ensures the expandability of any BMS installation and gives the designer the comfort of using devices from a large variety of vendors. Finally, in order to achieve this,the building automation functions are divided in five Interoperability Areas (IAs):

- Data Sharing

- Alarm and Event Management
- Scheduling
- Trending
- Device and Network Management

As is already known, we can depict the distributed BMS using the three layer building automation model (Management level, Automation level and Field level). Field level is responsible for measurements, positioning and metering. Automation level is responsible for measurement, control and regulation processes. Furthermore, monitoring, energy management and operation management functions belong to Management level. BACnet can be used at all levels of this building automation model, but is most commonly used in the management functions. Even though there is a BACnet implementation for the lower levels, BACnet solutions are mostly used as superordinate management systems in larger installations which use other protocols in field and automation level, like LonWorks and KNX that are going to be presented later in the current chapter [3].

The BACnet protocol defines the structure of the messages and information that is transmitted from one device or system to another. The different types of messages are the following:

- Binary input or output values (e.g., on / off states)
- Analogue input or output values (e.g., analogue temperatures)
- Software binary and analogue input and output values (e.g., temperature readings in °C)
- Schedule information (e.g., scheduled metering)
- Alarm and event information (e.g., fire alarms)
- Files (e.g., for devices configuration)
- Control logic

2.2.2.2 The BACnet Communication Architecture

BACnet was developed based on the Open System Interconnection (OSI) 7-layers model, which defines the architecture of communication systems. *Table* 2.1 shows the BACnet layers and the corresponding OSI layers.

Table 2.1: BACnet layers compared to OSI layers

<table>
<tr><th colspan="5">BACnet Layers</th><th>OSI Model Layers</th></tr>
<tr><td colspan="5">BACnet Application Layer</td><td>Application</td></tr>
<tr><td colspan="5">BACnet Network Layer</td><td>Network</td></tr>
<tr><td colspan="2">ISO 8802-2 (IEEE 802.2) Type 1</td><td>MS/TP</td><td>PTP</td><td rowspan="2">LonTalk</td><td>Data Link</td></tr>
<tr><td>Ethernet (802.3)</td><td>ARCNET</td><td>EIA</td><td>EIA 232</td><td>Physical</td></tr>
</table>

On the lower level communication layers (first and second layer according to OSI reference model, ISO 7498), messages can be transferred through a variety of protocols, such as Ethernet, MS/TP, LONTALK, ARCNET or even Point-to-point connections over a common telephone line. Wireless communication is also a choice by using Wireless Local Area Networks (WLANs). On the network layer (third layer according to OSI model), the relay of messages between similar or different networks is taken care of in a way that the used lower layer technologies and protocols in these networks are transparent, hence the devices are able to communicate without knowing the implemented technologies. “BACnet supports Internet Protocol (IP) and therefore enables global networks to be interconnected (BACnet/IP protocol). IP is a network layer protocol and is responsible for routing packets through a complex network” [3].

Above the network layer (in OSI model) is the transport layer. In this layer, BACnet uses User Datagram Protocol (UDP), which belongs to the Internet Protocol family.

UDP is a very simple protocol that is used for transporting datagrams without acknowledgement or guaranteed delivery. For this reason, the BACnet application layer is responsible for controlling retransmission of a packet. Finally, the application layer is responsible for the object types, services and procedures involved (these concepts are explained in the following paragraphs) in any communication of a BACnet implementation.

BACnet devices store information, such as the temperature reading from a sensor or a control command, in an object-oriented approach. Objects in BACnet are abstract data structures in which information is stored as object properties. Objects have readable (R) and/or writeable (W) properties which can be optional or mandatory for each object type. Developers are free to define additional non-standard object types or properties if required.

2.2.2.3 BACnet Objects

Devices using the BACnet protocol are exchanging and storing information like calendar events and sensor readings. Hence, this information has to be accessed by the BMS designers, facility managers, tenants etcetera (for different reasons such as system upgrades, management and monitoring of the current BMS). A simple way for this to be achieved is the representation of data in an object-oriented approach.

The information exchanged by BACnet compatible devices are stored in to objects in a form of object properties. Moreover, properties can be defined as required and readable (R), required\readable\writeable (W) and optional (O). BACnet objects are the base for the BMS functionality. The BACnet standard defines objects for simple or complex tasks, like sensor readings and calendar events. Furthermore, BACnet as an open protocol, gives developers the possibility to define their own objects for specific tasks.

2.2.2.4 Provided Services

For communication between devices, BACnet defines services that can be used to

access, create and manipulate objects. Those services are divided into five groups:

- Object Access Services
- Alarm and Event Services
- Remote Device Management Services
- File Access Services
- Virtual Terminal Services

Each group has a list of supported services. Object Access services are based on the access and manipulation of the properties of BACnet objects. Alarm and Event services are used for error messages, alerts (e.g. change of value reporting) or even simple measurements. Remote Device Management services are used for device management and administration (e.g. device deactivation). File Access services are used for specific applications that required file readings and writings. Finally, Virtual Terminal services are used for “TELNET style” communication with the devices, that provides the ability for administrators to sent direct text-based commands to the devices (e.g. device configuration).

2.2.2.5 BACnet Operator Workstations and other device profiles

Each device can be categorised by the functions it is providing into some device profiles. One of the most important and commonly used is the Operator Workstation. On the operator point of view now, “BACnet Operator Workstation (B-OWS) is the user interface for the system. While it is primarily used for the operation of a system, the B-OWS may also be used for configuration activities that are beyond the scope of BACnet standard. The B-OWS is not intended for the direct digital control of systems” [3] but it does enable some functions in the five IAs defined above. “A B-OWS GUI, configuration and programming tools and additional tools for integrating disparate networks vary depending on the manufacturer. Unfortunately, there is no standardized development environment

available (as it exists in other protocols). As a result, the practical use of BACnet is strongly influenced by proprietary tools and, to a certain extent, limits the operation of components from different manufacturers" [3]. Furthermore, BACnet describes additional device profiles depending on the functionality and the most important are listed on the *Table* 2.2 presented bellow.

Table 2.2: BACnet common device profiles

Profile name	Description
Building Controller (B-BC)	*A programmable automation device capable of carrying out a variety of building automation and control tasks*
Advanced Application Controller (B-AAC)	*A control device with limited resources compared to a B-BC, intended for specific applications that do not require that much programming*
Application Specific Controller (B-ASC)	*A controller, programmed by the manufacturer and the user can only change the parameters.*
Smart Actuators and Smart Sensors	*Simple devices that can communicate using BACnet and then only send values as electrical signals upon request.*
Routers	*Devices responsible for interconnecting networks that have the same or different network technology.*
Gateways to Other Systems	*Devices allowing BACnet networks to connect with other networks such as KNX and LONWORKS.*

2.2.2.6 Protocol Implementation Conformance and Certification of BACnet Devices

All devices conforming to the BACnet standard must conform precisely to the requirements. To prove that a device conforms to the protocol, the manufacturer has to supply a Protocol Implementation Conformance Statement (PICS) that identifies all the portions of BACnet that are implemented by the device.

The BACnet Testing Laboratory (BTL) is an independent institution which was established by BACnet international to support compliance testing and interoperability testing for BACnet products. Once a product has passed all the tests it is awarded the BTL mark.

In order to integrate BACnet devices with the ICT world Gateway products are available which translate messages between BACnet and BACnet/IP and BACnet/WS. Both allow common IT systems and applications to access BACnet protocol data.

2.2.3 Konnex (KNX)

KNX, formerly known as the European Installation Bus (EIB), is a building automation communication technology used to provide exchange communication between devices such as sensors, actuators, controllers, operator workstations etcetera. KNX technology was designed to be used in electrical installations for accomplishing automated functions and processes in building management systems. KNX is recognised as an International Standard (ISO/IEC 14543-3) as well as European Standard (CENELEC EN 50090 and CEN EN 13321-1) and Chinese Standard (GB/Z 20965). This makes KNX a worldwide open standard for building automation. Open, in this context means that devices from different manufacturers can communicate with each other over KNX [3], [5].

2.2.3.1 KNX Benefits

Comfort, convenience, increased security and energy saving are some of the

benefits by using KNX and the main reasons of using building automation and building management systems on both residential and commercial buildings. In order to derive its benefits, KNX compatible devices have to be installed in the building infrastructure.

2.2.3.2 KNX Installation Architecture

Products that support KNX technology can be separated into four basic categories:

- System components (e.g. power supply units, accumulators, line couplers, backbone couplers, line repeaters)
- Sensors (e.g. temperature sensors, motion detectors)
- Actuators (e.g. switch actuators)
- Other devices such as logical components and control panels

The topology of a KNX building automation system is based on the structure of common building installations similar with the tree topology having the following hierarchical structure:

- Nodes (N) are assigned to a line (L)
- Several lines are connected via a main line (ML) and form an area (A)
- Several areas are connected with each other via the backbone line (BL)

An area represents, for example, the first floor of a building. On each first floor's corridor are the lines to which the bus devices in the neighbouring rooms are connected. It is common to smaller BMS to have just few nodes that can be assigned to one single area and to the one single line. A device must be assigned to a line and an area [3].

2.2.3.3 KNX Communication Architecture

When a communication between two or more KNX compatible devices is held, the

information has to form a data frame and then be transmitted it over a network.

On the lower communication layers (first and second layer according to OSI reference model, ISO 7498), information between the devices can be exchanged by using a variety of protocols, such as:

- Twisted-pair cable (KNX.TP)
- Power line (KNX.PL)
- Radio frequency (KNX.RF)
- Fibre-optic cable

Various transmission media can be integrated in a way that older implementations are compatible with newer ones requiring only few changes.

On higher layers (compared with the OSI model), KNX devices exchange information using frames. There are two types of frames: data frames and acknowledgement frames. A data frame (e.g. sensor readings) can be sent as a response of polling, by a schedule request or periodically (interval). Then all the receivers of a data frame confirm that they received the frame by transmitting an acknowledgement frame. If the sender transmits a frame to a device located on another line, the messages has to pass from a device called coupler, which connects the lines and confirms the receipt of the frame. KNX technology uses Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA) as media access control protocol to avoid and resolve collisions that may occur by the usage of the common access medium. CSMA/CA is also responsible to decide which device is allowed to send its frame first (bus arbitration). In order to execute a function, the application program of at least one sensor must exchange data (in the form of data frames) with the application program of at least one actuator. To do this, the sending and receiving applications use a specific number of communication objects. A communication object, is a way of representation of the exchanging information and is also physically located in an area of memory of every KNX device’s micro-controller which is used for communicating with applications in

other devices.

2.2.4 Local Operation Networks (LonWorks)

LonWorks is defined as a networking platform specifically created to address the needs of control applications and enables intelligent devices to communicate with each other over a locally operated control network (LON stands for Local Operating Network). The platform is built based on a protocol developed by the American company Echelon Corporation. LonWorks is an open networking solution for building automation and control networks, and it is designed in such a way that it can be used in centralized building automation management systems as well as in decentralized building control components. LonWorks is a standardized bus system (ANSI/CEA-709.1-B and ISO/IEC DIS 14908) [3],[6],[7].

2.2.4.1 LonWorks in Building Management Systems

LonWorks technology is widely used in building automation world wide and in the USA is the market leader. The main tasks achieved by LonWorks are sensor measurements, infrastructure monitoring and control on the local installed components of the BMS, through functions and profiles that provide a variety of comfort solutions for each controlled area of the building. For instance, there is a wide range of devices on the market implementing LonWorks technology for managing the integrated to BMS systems. Some systems that could be possibly operated utilizing LonWorks are listed below:

- HVAC (Heating, Ventilation and Air Conditioning)
- Lighting control
- Shade/blind control
- Security control

2.2.4.2 LonWorks Benefits

The main building automation benefits are known to be convenience and superior comfort to the residents of a building. Furthermore, LonWorks devices are used to both residential and commercial new building infrastructures and are increasingly replacing the conventional existing electrical infrastructures.

By using LonWorks technology different devices can communicate allowing components to interact and exchange data in order accomplish simple or complex tasks by using automated functions. In a simple example a smoke detector could invoke an air conditioning system to switch off and instruct windows to open to extract the smoke. To sum up, using LonWorks in BMS we gain benefits such as:

- Comfort and convenience
- Efficiency through energy management functions
- Flexibility through the reprogramming options available
- Security

2.2.4.3 The LonMark Association

Interoperability, the ability of devices from different manufacturers or generations to be able to exchange information with each other, is more than important to be provided by a device so the LonMark Interoperability Association [6] is responsible for defining a device's basic functionality and minimum requirements as well as the standard network variable types. If a device is awarded with the LonMark interoperability certification, it is assured to be interoperable with other LonMark certified devices on its basic functionality.

2.2.4.4 Basic Type of Messages Exchanged on LonWorks Systems

Every device is programmable so it can provide an application interface which is developed, configured and customized by the manufacturers and vendors of the device. This application interface defines the device functionality, hence the messages that it could receive from or transmit to the network. The functionality of

the device can be a combination of many simple functions and the messages that could be sent or received as defined by LonWorks are:

- Bit input and output
- Byte input and output
- Serial input and output
- Slope input
- Period and pulse count input
- Frequency output

2.2.4.5 LonTalk Protocol

LonTalk is the protocol that defines how LonWorks compatible devices are programmed to function and how they communicate with each other as nodes in a network for completing a task. Each device has a transceiver (short for transmitter and receiver) that allows it to connect to a network and communicate. The transceiver provides a physical communication interface between a LonWorks device and a LonWorks network. There are different transceivers for each medium such as:

- Twisted pair (most commonly used medium)
- Power line
- Radio frequency
- Fibre optics

As the LonTalk protocol defines, nodes (devices) that have equal access rights to the multiple access medium, use the p-persistent Carrier Sense Multiple Access protocol to compete for transmitting. A physical connection between the LON nodes is required for the communication to be accomplished. The LON nodes exchange the actual data using network variables, a standardized way to

represent data in order to provide interoperability. The LonMark Interoperability Association has defined a set of rules so the different network variables could be standardized (known as Standard Network Variable Types, SNVTs). The definitions listed below are defined for the most commonly used network variables:

- Area of application
- Name of the network variable type
- The variable's configuration
- The total length in bytes
- Value range, degree of precision and units

2.2.5 OLE for Process Control (OPC)

OLE for Process Control (OPC), which stands for Object Linking and Embedding (OLE) for Process Control, is a standard specification which was released in August 1996. This standard specifies the exchange of real-time data between sensors, instruments, controllers, software systems, and notification devices in a way that interoperability could be provided.

It has been specified by collaboration of a number of leading worldwide automation suppliers working in cooperation with Microsoft, for industrial automation functions. Moreover, its creation lead to the creation of the OPC foundation, which is responsible for maintaining this standard. Originally it was based on Microsoft's OLE COM (component object model) and DCOM (distributed component object model) technologies, and the specification defined a standard set of objects, interfaces and methods that could be used in process control and manufacturing automation applications to facilitate interoperability. The COM/DCOM technologies provided the framework for software products to be developed.

Since its initial release, some new standards have been added and the names of the existing changed. On 2012, OPC has already a series of standards specifications, with a total of 10 current and emerging specifications as shown in *Table* 2.3 [8]. Currently there are hundreds of OPC Data Access servers and clients in the global industry market.

Table 2.3: OPC Standards Specifications

Standard Specification	Description
OPC Data Access (DA)	*The original specification, used to transfer real-time data between devices. Data Access 3 specification is now a Release Candidate, and it extends earlier versions while improving the browsing capabilities and incorporating XML-DA Schema.*
OPC Alarms and Events	*Provides alarm and event notifications on demand.*
OPC Batch	*Provides interfaces for the exchange of equipment capabilities and current operating conditions.*
OPC Data eXchange	*This specification is about server-to-server communication across a network. It provides interoperability and adds remote configuration, diagnostic and monitoring/managing services.*
OPC Historical Data Access (HDA)	*OPC Historical Data Access provides access to data already stored (in contrast to OPC-DA, which provides real-time data).*
OPC Security	*OPC Security specifies how to control client access to these servers in order to protect this sensitive information and to guard against unauthorized modification of process parameters.*
OPC XML-DA	*Provides flexible, consistent rules and formats for exposing plant floor data using XML, cooperating with Microsoft and others on SOAP and Web Services.*

Standard Specification	Description
OPC Complex Data	*A companion specification to Data Access and XML-DA that allows servers to expose and describe more complicated data types such as binary structures and XML documents.*
OPC Commands	*A new set of interfaces that allow OPC clients and servers to identify, send and monitor control commands which execute on a device.*
OPC Unified Architecture (UA)	*A new set of specifications that are not based on Microsoft COM and provides standards based cross-platform capability.*

The first OPC standard released on 1996 (originally called simply the OPC Specification), is now renamed as the OPC Data Access Specification, and its 3rd edition is currently a release candidate. Moreover, while OPC originally stood for "OLE for Process Control", the OPC Foundation states that OPC is no longer an acronym and the technology is simply known as “OPC”. One of the reasons behind this, is that while OPC is heavily used in the process industries, it is also widely used in discrete manufacturing. Hence, OPC is known for more than just its applications within process control [17]. Finally, later in the current chapter we are going to focus and analyse further OPC Data Access (DA) and OPC Unified Architecture (UA), which are commonly used as technologies in building automation, for the communication of the devices and servers in building management systems.

2.2.5.1 OPC Data Access (DA)

The OPC Data Access was the first specification of the series and was originally named as the OPC specification. Later on, it was renamed to OPC Data Access, and with its name it states its main functionality, to retrieve data. This standard specifies a mechanism for real-time data exchange between sensors, instruments,

controllers, software systems, notification devices and numerous other data sources. OPC DA can also be used to enable a software application to write data to a control system. This enables OPC DA applications to handle supervisory control. Thus, OPC DA is widely used for Supervisory Control and Data Acquisition (SCADA) scenarios.

OPC DA enables the exchange of real-time data, meaning it queries the current values. To get the older values, applications must use OPC HDA (Historical Data Access) to query data bases or other data sources [18].

2.2.6 OPC Unified Architecture (UA)

OPC Unified Architecture (OPC UA) is the evolution of OPC specification from the OPC Foundation. The Unified Architecture (UA) is the next generation OPC standard that provides a cohesive, secure and reliable cross platform framework for access to real time and historical data and events. The first version of the OPC UA was released in 2006 after some years of development and prototyping and with the experience from the development of OPC DA and the other OPC specifications [8]. The target of OPC UA is to provide a standardized and platform independent communications model. The previous OPC specifications based on COM technologies are currently outdated and do not cover the needs of the market, for standards providing interoperability, platform independence and high security.

2.2.6.1 OPC UA Benefits

By implementing OPC UA and using OPC UA compatible products there are many advantages, some are further discussed in the following paragraphs and the most important are listed below:

- Service oriented architecture for communication
- Interoperability (with Certification)

- Single solution for enterprise and embedded systems
- Improved security with a standard security model
- High performance communication protocols (Binary TCP based, SOAP/XML etc.)
- Reduced development cost (with existing OPC UA SDKs)
- Flexible object oriented information models
- Code reuse (with UA methods and programs)

2.2.6.2 OPC UA Interoperability Certification

Interoperability was one of the main reasons for developing OPC UA. Multi-vendor systems, devices and software must be able to communicate and “understand” each other's messages and commands. This is ensured by OPC certification programs that include self-certification, interoperability workshops and 3rd party testing by independent interoperability certification test labs. A product can get two levels of OPC certification, the Self-Tested and the Compliance-Certified. The Self-Tested level is gained by a process where OPC vendors, manufacturers and developers verify the correct operation of their products by passing a series of test software developed by the OPC foundation. In addition to that, Compliance-Certified level is awarded by the Independent Certification Test Lab, where the staff of the lab verify the correct operation of the products. OPC UA and OPC COM specifications are able to be certified in this labs and products are awarded with a Certification logo which allows end users to reduce system integration costs by choosing OPC Certified products.

Figure 2.1: OPC Certification

2.2.6.3 OPC UA Development Architecture

OPC UA was developed based on a technology known as Service Oriented Architecture (SOA), in addition backwards compatibility is provided for previous generations OPC products. Implementing SOA, means that the applications should provide well defined functionalities, in a way that discrete pieces of code could be reused for different purposes. SOA methodology, lead to a better structured software development with discrete functions and data structures that can be reused for different purposes.

The original specification was only about processing data and different specifications were needed for other services to be provided. Further on, based on the SOA technology, the OPC foundation decided that a single specification should provide a set of services that would have access to the data providing services like:

- Alarms and events
- Historical Data
- Data Access
- Batch data
- And more

2.2.6.4 OPC UA Communication Architecture

The OPC UA specification defines a binary TCP based protocol which provides

good performance with minimal overhead. The most commonly used communication protocol in enterprise solutions is SOAP/XML. OPC UA provides the ability to pre-encode messages with UA Binary before encapsulating them in a SOAP/XML (see Figure 2.2) compatible message which improves performance up to 10 times when compared to the same message sent using XML. This function offers benefits even when using SOAP/XML with formats such as the proposed W3C Efficient XML because it reduces the complexity and size of the XML before it is sent on the transmission medium [8].

Figure 2.2: Pre-encoding Messages into UA Binary

The OPC UA architecture is designed to be platform independent, in contrast to its ancestor which was based to Microsoft technologies and Windows operating systems . This means developers are free to use the languages and operating systems of their choice to develop OPC UA applications. Furthermore, developers do not have to use the SOAP/XML over HTTP approach but they could use a different one of their choice. This platform independent feature is also valuable to Windows users because it will allow applications to migrate to the next generations of Microsoft communication technologies in needed. In addition to this, vendors of OPC UA products will have options if a particular communication technology turns out to have technical problems, and they will be able to replace it.

2.2.6.5 OPC UA Information Modeling

Furthermore, today's buildings have already a variety of building automation systems implemented utilizing different networking technologies. Therefore, interoperability between devices using different technologies is one of the most important goals of OPC. Therefore, the specification includes an abstract data and information model. OPC UA enables mapping the information of many other

existing protocols (for example BACnet) into the OPC information model, and as a result this information can be accessed by OPC UA clients in a standard and well defined way. More details on this information model are going to be presented later on in Section 2.3".

2.2.7 Open Building Information Xchange (oBIX)

oBIX is a specification standard for RESTful (Representational State Transfer) Web Services-based mechanisms for building management systems. oBIX is being developed by the Organization for the Advancement of Structured Information Standards (OASIS) and its version 1.0 was completed in December of 2006. It is defined as a web services interface because it can provide only monitoring of data without having to allow control of the embedded building automation integrated systems. This interface provides communication between applications and building automation systems as well as it provides communication between the installed devices and automation systems.

2.2.7.1 The Idea for Developing oBIX

The reason for developing oBIX was to provide a publicly available web services interface specification that can be used to obtain data with a simple and secure mechanism from building automation devices and controllers, as well as from full entities of BMS. oBIX also provides data exchange between the installed automation systems and applications. In addition, implementation guidelines are developed to facilitate the development of products that use the web service interface [9]. The oBIX Committee has been formed in 2003 as the CABA XML/Web Services Guideline Committee. Since then oBIX has been transferred to a Technical Committee at the Organization for the Advancement of Structured Information Standards (OASIS) [10].

2.2.7.2 oBIX Benefits

The protocols that already exist in the market for building automation and industrial

device communication (e.g. BACnet, LonTalk, KNX etc.) are able to be used over TCP/IP networks. These communication protocols, face problems with routers, firewalls, security, and compatibility with other network applications. Moreover, the bigger challenge is that the market is split between several largely incompatible protocols.

The oBIX committee claims the integration of existing mostly non-IP based protocols with access via Web Services [9]. Because oBIX installation is integrated in the infrastructure, it aims to enable mechanical and electrical control systems to provide continuous monitoring of operational status and performance. Furthermore, events are created in case of problems and trends that lead to system analysis or need of human interaction.

2.2.8 IETF 6LoWPAN with IETF CoAP

Finally, an IETF specification known as 6LoWPAN [15] (IPv6 over Low power Wireless Personal Area Networks) which provides encapsulation and header compression mechanisms that allow IPv6 packets to be sent to and received from IEEE 802.15.4 based networks. The main reason for development of this specification was to enable resource constrained devices such as sensor nodes to reduce the IPv6 packet header overhead. Together with the IETF Constrained Application Protocol (CoAP) specification it will enable developers to realize RESTful communication paradigms using IPv6 on wireless sensor devices. Currently, CoAP is not yet standardised, and is published as an internet draft. In more details, CoAP is HTTP adapted for Machine-to-Machine communication pattern using UDP as transport protocol, which implied on one hand that for reliable communication a transaction mechanism needed to be added to CoAP but on the other hand allows unicast and multicast message exchange.[16]

2.3 Building Information Models

2.3.1 Introduction

After the communication was achieved, the data that were exchanged between different devices and applications had to be modelled in a standardized way that different vendor's applications could communicate and cooperate by exchanging information between them for different purposes. The process of generating and managing building data during its life cycle is called Building Information Modeling (BIM). BIM involves representing the raw data collected from the BMS (such as metering, alerts, messages etcetera) as objects, in order to facilitate exchange and interoperability of information. Furthermore, the two most commonly used data models in the building industry are presented in this chapter.

2.3.2 OPC UA Information Model

The OPC UA architecture provides also an data model that allows developers to create reusable components and provide multi-vendor interoperability. For the creation of this data mode, the OPC foundation collaborated with other organizations which had experience in data modelling in order to create the ideal information model for the global market. The abstract approach of OPC information modelling allows it to be used for generic data exchange in many applications. The base principals of OPC UA specification on information modelling are listed below:

- Object oriented definitions of type include hierarchies and inheritance
- Type information can be accessed the same way as data.
- The same information can be presented in different ways, depending on usage and the network nodes that require it. Full meshed network topology is also possible.
- The architecture is extensible in many ways, due to hierarchies and

references between nodes.

- The information model is adjustable to the current information model of any system.
- Server side modelling indicating less requirements from the clients

2.3.3 Industry Foundation Classes (IFC)

The buildingSMART organization (formerly International Alliance for Interoperability, IAI) developed a common object oriented data schema in order to represent and exchange data between multi-vendor software applications providing interoperability [11]. The data schema comprises information covering the many disciplines that contribute to a building throughout its life cycle: from conception, through design, construction and operation to refurbishment or demolition [16].

Industry Foundation Classes (IFC) are the main buildingSMART data model standard. The IFC format is registered by ISO as ISO/PAS 16739 on 2005 and is in the process of becoming an official International Standard ISO/IS 16739 [12]. The IFC2x Platform Specification provides an information model, written in EXPRESS (ISO 10303-11:1994), for sharing data in the construction and facilities management industries. The IFC specification defines how different applications are exchanging data relative with building information and focuses on the life-cycle of construction facilities, including all the phases as they are identified by generic process protocols for the construction and facilities management industries, such as:

- Demonstrating the need
- Conception of the need
- Outline feasibility
- Substantive feasibility study and outline financial authority

- Outline conceptual design
- Full conceptual design
- Coordinated design
- and more

IFC, aims at supporting the different disciplines that participate in the project life cycle, such as [12]:

- building services (HVAC - heating, ventilation, air conditioning, cooling, electrical),
- building automation
- facilities management
- and others.

The IFC data model is an object oriented model that separates the object identifications and the associated properties. It facilitates interoperability in the building industry, and is a commonly used format for Building Information Modelling (BIM). IFC defines multiple file formats, supporting various encodings of the same underlying data (IFC-SPF as text format defined by ISO 10303-21, IFC-XML as XML format defined by ISO 10303-28, IFC-ZIP as ZIP compressed format consisting of an embedded IFC-SPF file).

IFC defines an EXPRESS (data model language defined as ISO 10303-11 [13]) based entity-relationship model consisting of several hundred entities organized into an object-based inheritance hierarchy.

2.3.3.1 IFC Creation and Maintanance

IFC development started on 1994, when Autodesk invoked and formed an industry consortium in order to create the ideal set of (object oriented) classes to represent a building information model. In the formation of the Industry Alliance of Interoperability, twelve US companies joined and initially released IFC 1.0 on

1997, and since then I have been improved and maintained by this consortium. Finally, in 2005 the Alliance was reformed and renamed into buildingSMART. The version releases are shown in Figure 2.3.

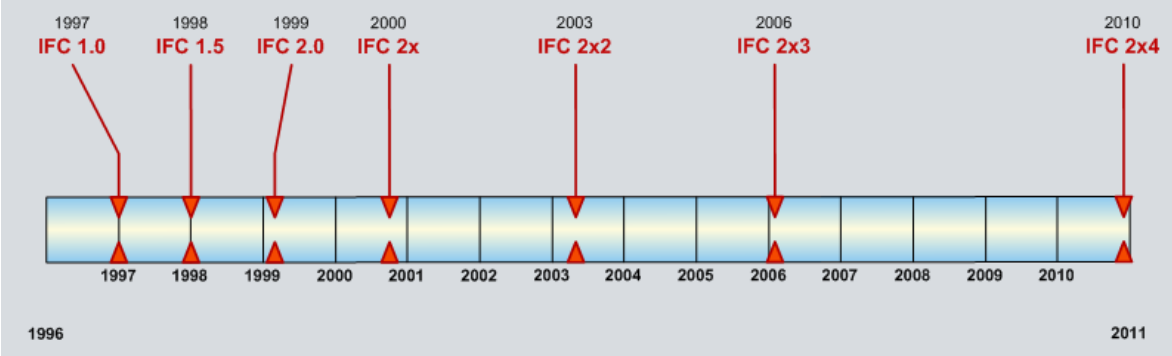

Figure 2.3: IFC Versions Release Dates

The current stable release (February 2012) is IFC2x Edition 3, which was published in July 2007. The IFC2x Edition 4 status is Release Candidate 3, review which started on 17 of October 2011 and is expected to close on 31 of March 2012 when the final release is expected also to be published.

The 4th Edition of IFC2x is expected provide several extensions of IFC in building, building service and structural areas, enhancements of geometry and other resource components, and numerous quality improvements.

2.3.4 Green Building XML (gbXML)

The Green Building XML schema, also known as “gbXML”, was developed to facilitate a common interoperability model integrating many design and development tools used in the building industry [14]. It was designed and developed by Green Building Studio and was funded by the California Energy Commission. The gbXML open schema aims to enable the transfer of building properties stored in 3D building information models (BIM) to engineering analysis tools. In addition, with the development of integration modules inside major engineering analysis tools, gbXML has become the de facto industry standard schema.

In June of 2000, the gbXML schema was first submitted for inclusion in aecXML (an industry standard launched by Bentley Systems in the summer of 1999). As of January 12, 2012, version 5.0 of the gbXML schema is released in the market.

2.4 Conclusions

As the level of automation increased in building infrastructure over the last few decades, the need of data exchange led to the development of a variety of protocols. While improvements on the building automation were made and the first building management systems appeared, protocol faults and incapability to provide and support several building services had as a result of developing even more protocols. Furthermore, interoperability was raised as the most important issue. Each manufacturer or vendor supported its own protocols and specifications and as a result devices and software from different vendors or even from different generations where not able to cooperate.

The solution was the development of an open protocol that was going to be adapted globally from the majority of the market. Some attempts were made by different organizations that led to the standards that were presented in this chapter. Each protocol had some advantages or disadvantages compared with the others, but again the most important issue was the share of the global market, as most of the protocols could cover the needs of building automation.

The domination of a single protocol globally is not clear, and that led to the creation of devices (in a way routers and gateways) that could translate messages from one protocol to an other (only the most commonly used protocols where supported). Thus, the building automation designers and manufacturers are able to choose between few protocols without facing the problem of extensibility and compatibility of their infrastructure with new products.

Furthermore, data exchanged between devices and applications faced the

problem of interoperability and a standardised way for representation of the building informations and data had to be developed. Few solutions finally dominated the market. Again, efforts are made for the “translation” from one data model to an other. A comparison could be made between the three most used approaches, for example compared to IFC, gbXML has some limitations in geometry support, it is however easier and more flexible to extend, while IFC is more comprehensive model, still with limited complexity (OPC UA data model is not yet published and used in the market). But again, it is not obvious which is “better”.

To sum up, a decision for the ideal protocol and data model is not easy to be made neither a prediction for which of them is going to lead the market. Thus, the developers and designers of BMS and building automation products have to choose between a limited number of options on the technologies that their products are going to support with only criteria their own special needs depending on each technology advantage (e.g. compatibility with existing infrastructure).

Chapter 3

3 BMS Intelligent Proxy Design

3.1 Introduction

The difficulty of choosing a specific technology to be supported from a building management system and the variety of solutions, had as a result many incompatible and independent subsystems to be installed in a building automation infrastructure. This created the need of a device that would provide a “bridge” between heterogeneous and independently operating building management subsystems, in order to interconnect them and create a centralised managed monitoring and control platform that will enable an aggregated energy saving profile for the entire building infrastructure.

In the current chapter, the state of the art of a theoretical Intelligent Proxy device, responsible for “bridging” the existing or future subsystems, is going to be presented and analysed. The requirements for the design of this devise and the feasibility of its functionalities are knowledge gained from real world's problems research that has been done in NEC Laboratories Europe for the European project CAMPUS21 (Control and Automation Management of Buildings and Public Spaces, FP7).

The introduction to a prior art design and the theoretical approach of the basic functionalities that a BMS Intelligent Proxy should provide are the main topic of this chapter. Furthermore, the decisions made in this state of the art, are going to be justified with examples from the real world problems on the building automation global market, along with some examples provided by the demonstrator sites of the European CAMPUS21 project. More details about this project can be found on the published documents and on the projects web page [19]. Finally, the results of a research on innovative methodologies that could improve building management services and theoretical solutions for problems on today's generation BMS will be

also presented.

3.2 The CAMPUS 21 Project

CAMPUS 21 stands for Control and Automation Management of Buildings and Public Spaces in the 21st Century, and it is a European project under the topic of Energy Saving. NEC, a leading frontier enterprise in Telecommunications, is one of the participants, which are Universities and enterprises from Ireland, Austria, Spain and Germany.

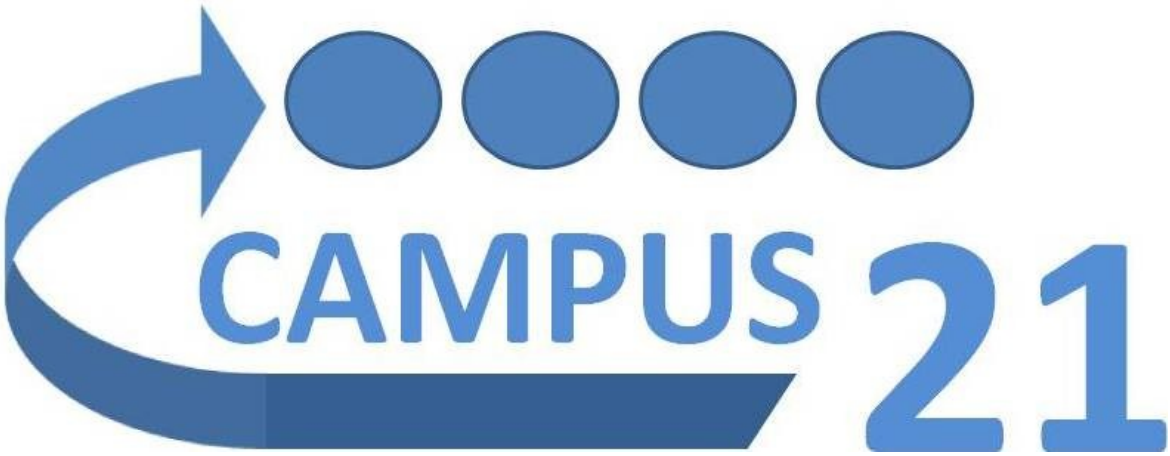

Figure 3.1: CAMPUS 21 Logo

CAMPUS 21 develops, deploys, and tests a Hardware-Software-Platform for the integration of existing ICT subsystems supporting energy, building, and security systems management. CAMPUS 21 target is to use cutting-edge technologies in order to provide innovations, such as:

- integration concepts for energy management systems including the related middle-ware components
- methodologies for intelligent and optimised control of building automation and BMS services
- algorithms and tools to support load-balancing between renewable micro-generation storage systems, and energy consuming devices in buildings

and public spaces

- new business models for integrated energy management and the underpinning novel procurements schemes
- the development of Performance Metrics and a holistic Evaluation Concept for Systems Integration

In addition, external data, such as weather forecasts, could be used in order to calibrate the building infrastructure functionality profile for a more optimal energy performance.

3.2.1 The CAMPUS 21 Demonstration Sites

CAMPUS 21 uses demonstration sites existing building infrastructures and it aims on up-scaling and expanding the building management systems scope from single building infrastructure to campus scale. Three sites are used for research and validation of energy and cost savings, a university campus, a multi-purpose sports arena and an indoor sports complex [16],[19].

3.2.1.1 The University Campus

University College Cork (UCC) is located in Ireland and is a third level institution with 120 educational, research and sports buildings varying in construction year from 1849 to 2009. UCC is composed by a main Campus Centre and several, distributed buildings close to the main campus location. The infrastructure is spread over approximately 330.000 m^2 excluding sports fields.

The main campus heating system is assembled of 3 gas fired steam boilers that is distributed across the site and is converted to Low Pressure Hot Water (LPHW) in various buildings using steam to LPHW heat exchangers that are all controlled by a BMS. Some buildings are not fed from the steam main and have local gas-fired LPHW boilers and pumping systems, again controlled by the BMS. Where cooling is required, Chilled Water (CHW) systems are deployed using air-cooled chillers and various types of chilled emitters. There are also a few air-conditioning

subsystems across the campus, some of them are controlled by the BMS and some are controlled by a local controller. UCC includes also off campus buildings which are a mix of domestic type dwellings that have simple boiler (gas) and radiator heating systems and no BMS control to buildings that have full HVAC systems with BMS control.

3.2.1.2 The Sports Arena

The Commerzbank-Arena is a sports stadium in Frankfurt, Germany. The stadium was inaugurated in 1925 and its original name was Waldstadion (which means "Forest Stadium" in German) and it was renamed in 2005. The stadium has been upgraded several times since it was first opened to public and the most recent upgrade included remodelling and redevelopment of the hole stadium into a football-only stadium for the 2005 FIFA Confederations Cup and 2006 FIFA World Cup. With a capacity of 51,500 occupants, it is among the ten largest football stadiums in Germany.

The sports complex, which is owned by the city of Frankfurt, includes not only the actual stadium but also other sports facilities such as:

- a swimming pool
- tennis fields
- a beach volleyball court
- a winter sports hall
- an integrated garage with 1800 parking spaces
- over 70 VIP lounges (for almost 1.000 guests)
- a business club lounge (of 2.800 m^2)

The arena has also its own railway station, “Frankfurt Stadion”, on the national German rail network. Moreover, the hole infrastructure is spread over a gross floor area of approximately 110.000 m².

As it is expected, the energy consumption of a sports arena is not insignificant. Consumption could be optimised in many different use cases, for example BMS controlled HVAC systems that are installed to all VIP lounges, and provide heating an air-conditioning and the garage space lighting could be calibrated.

3.2.1.3 The Indoor Sports Center

The Huerta del Rey sports center, located at 9th Joaquin Velasco Martin Road, Valladolid, Spain, is an indoor sports center that was inaugurated in 1975 with a capacity of 7000 occupants. This sports center in its infrastructure of total 9100m^2 includes:

- an indoor gym
- an indoor swimming pool
- 8 outdoor tennis courts
- 2 outdoor basketball courts
- a handball sports hall (3500 occupants)
- a snack bar
- some offices

On the sport center's infrastructure it includes a HVAC (Heating Ventilation and Air-conditioning) which is assembled by one AHU (Air Handling Unit), a gas boiler system, heat pumps, and a solar thermal power plant. Altogether, these sub-systems provide dehumidification and heating to the sports center.

3.3 Prior Art Architecture

In this section, the basic requirements of the “ideal intelligent proxy” and its main functionalities are going to be presented. In addition, high level UML (Unified Modeling Language) Use case diagrams, Class diagrams and sequence (interaction) diagrams are going to be demonstrated and explained. Hardware

requirements are not going to be discussed, as there is no limitation in a theoretical device and the feasibility of the functionalities is based on existing hardware.

The information provided in this sub-section of the thesis are based on research of today's real worlds building automation and building management systems and they are personal opinion of the thesis author for the ideal implementation of an intelligent proxy for a BMS infrastructure.

3.3.1 Intelligent Proxy Requirements

The main goal for the development of the proxy is to provide bridging of the existing or future sub-systems of a building automation infrastructure in a way that they could managed and monitored by a centralised control platform that will enable an aggregated energy saving profile for the entire building infrastructure. Following the vital requirements of a BMS, there are additional requirements which were extracted from use case examples which are presented below. The use cases are based on real world needs and CAMPUS 21 project.

3.3.1.1 Use Case #1 Building Monitoring and Control

The need of monitoring the building is important, as by monitoring each BMS user could calculate the cost based on stored measurements, get notified for malfunction of a device or even calibrate his lease's energy consumption. This use case can be separate by the two different points of view, the facility managers one and the end user/tenant.

Table 3.1: Use Case #1 Extracted Requirements

Requirements	Description
Graphical User Interface \ Login	GUI based on user credentials, enabling: view of historical data graphs, device management and future state calculations (e.g. cost)
Database	For historical data and user

Requirements	Description
	authentication
Configuration Functions	For predefined device behaviour or current state change (e.g. change the heating temperature or close a window)
Communication Interface	For accessing the BMS over the network

3.3.1.2 Use Case #2 Event Based Actions

The actions made automatically by the BMS triggered by an event (e.g. high temperature reading or motion detection) .

Table 3.2: Use Case #2 Extracted Requirements

Requirements	Description
Security Functions	Security policies that define and handle security violations depending on the facility manager settings.
BMS Automation and Intelligence	Intelligence to the BMS in order to handle events based on the hard-coded or real-time calculated automated functions

3.3.1.3 Use Case #3 Integration of Existing Sub-Systems

Various sub-systems may operate at the building infrastructure as discrete systems, without the ability to communicate or to be centrally managed as one complete BMS. Interconnection of these sub-systems in order to be managed as one holistic entity is important. Furthermore, in case of different technologies implementation, due to multi-vendor or different generation of devices, needs to be considered and interoperability must be provided.

Table 3.3: Use Case #3 Extracted Requirements

Requirements	Description
Device Adaptation	Devices based on different technologies have to be interconnect and operate providing a homogeneous environment feeling.

3.3.1.4 Use Case #4 External Data Sources Interaction

The use of external data sources is one of the basic innovations aimed by this thesis. Forecasts, energy pricing or even energy consumption calibration could lead to a more optimal energy saving profile. An API must be provided, in order these external sources and cloud servers to gain access to the BMS and get associated with the optimisation of the energy plan. Recommendations for more optimal operation could also be given in a form of a GUI pop-up to the facility manager based on novel and innovative algorithms that run in cloud servers and are processing the BMS data.

Table 3.4: Use Case #4 Extracted Requirements

Requirements	Description
Application Programming Interface	An interface that would provide access to external sources to the BMS functionalities, in order to participate on its calibration.
Communication Interface	The CI, should enable access to the API over networks.
GUI	The functionality to provide notifications to the facility manager based on recommendations from external sources.

3.3.1.5 Use Case #5 Scheduled Actions or Actions with Intervals

Several actions (e.g. measurements) need to be executed based on regular

intervals a calendar should be kept, in order to trigger those actions execution based on the system time. Moreover, in case of many actions needed to be a executed at the same time, a queue should be implemented providing also the ability of priority preferences.

Table 3.5: Use Case #5 Extracted Requirements

Requirements	Description
Action \ Process Scheduler	An entity that would be responsible for holding the queue for each and every action to be done by the BMS. Implementation of priorities and alarms should be considered

3.3.1.6 Use Case #6 System Energy Consumption Auto-Calibration

Based on an interval or an event (e.g. weekly or when external data received) the system should dynamically re-configure the operational profile by processing the historical data, external data or even current values (e.g. shut down the heating of a room when there is no motion detected, but if based to historical data there is motion once every 30 minutes and heating is not needed do not turn on again the heating system if motion is detected for less than xx minutes). This functionality is an innovation for developing “smart” BMS with the ability to learn from historical events and get auto-calibrated. Providing real time dynamic optimisation of energy management could have as a result an optimal building performance with outstanding energy saving values.

Table 3.6: Use Case #6 Extracted Requirements

Requirements	Description
BMS Intelligence And Automation	Innovative implementation of the BMS operating system in order to provide evolutionary algorithms for real time profile calibration
Database	For historical data, that could be used as “knowledge” that could help

Requirements	Description
	calibrating the system operation
API - Communication Interface	Access to external sources to the system for providing data
Scheduler	To organize the actions that have to take place in order to optimize the building performance

3.3.1.7 BMS Basic Functionality Requirements

It is obvious that the proxy has a central role in the building management, and it could be called as “the heart of the BMS”. Some of the basic functionalities that it could offer in order to accomplish an optimal energy consumption management are listed in the *Table 3.7* below, and the most important are going to be further explained and discussed later on in this chapter.

Table 3.7: The Basic Functionalities of an Ideal BMS Proxy

Proxy Functionality	Description
Device Abstraction / Adaptation	*All devices, controllers and sub-systems connected to the proxy should be operated in a generic way and the protocols and technologies used by them should be transparent.*
Database Access	*The proxy should provide database access in order to provide historical data functionalities (e.g. Past energy consumption). Optimisation of the operating profile could also be provided by processing the historical data.*
External Information/Recommendation API	*An API providing access to the system by external data sources and that could participate on the energy management optimisation.*
Action Scheduling / Calendar	*Actions and commands to the devices should be triggered based on a calendar (or an interval).*
Event/Alarm Management	*Automated activities based on alarms or events created by the building*

Proxy Functionality	Description
	automation devices (e.g. Pop up to the administrator for device malfunction, shut down the heating system in case of fire)
Security and Observation	*Security operations and building status information to the facility manager should be available (e.g. motion detection, door opening notification)*
Central Management of the Installed Building Automation	*The reason that the proxy is called "the heart of the BMS". The main intelligence of the management profile, which includes automated tasks, event management and energy consumption calibration and optimisation.*
Graphical User Interface	*An graphical interface that would provide to the facility manager full access to all the BMS functionalities and control. Furthermore, it can provide limited access to the end-user. (e.g. energy consumption monitoring and individual HVAC control for each tenant)*

3.3.2 Intelligent Proxy State Of The Art

A high level design of an implementation of the proxy, in order to meet the requirements that were described earlier in this chapter in an abstract approach is presented below (Figure 3.2). In this figure the main functionalities are painted as discrete entities (rectangles) and arrows show the interaction with the devices of the building infrastructure by using any of the supported protocols and with the external entities (e.g. users and cloud servers). The data stored to the database, sent to the cloud or presented to the users (end users, facility managers etcetera) should be in the form of one data model, chosen to be used by all the proxy's components (and external entities if data exchange is needed), so the technologies used bellow the management layer could totally transparent all the entities above the Device-Adaptation (protocols "x" are protocols used in building automation and supported by the intelligent proxy) and data exported from the

BMS could be at a well-defined standardized form.

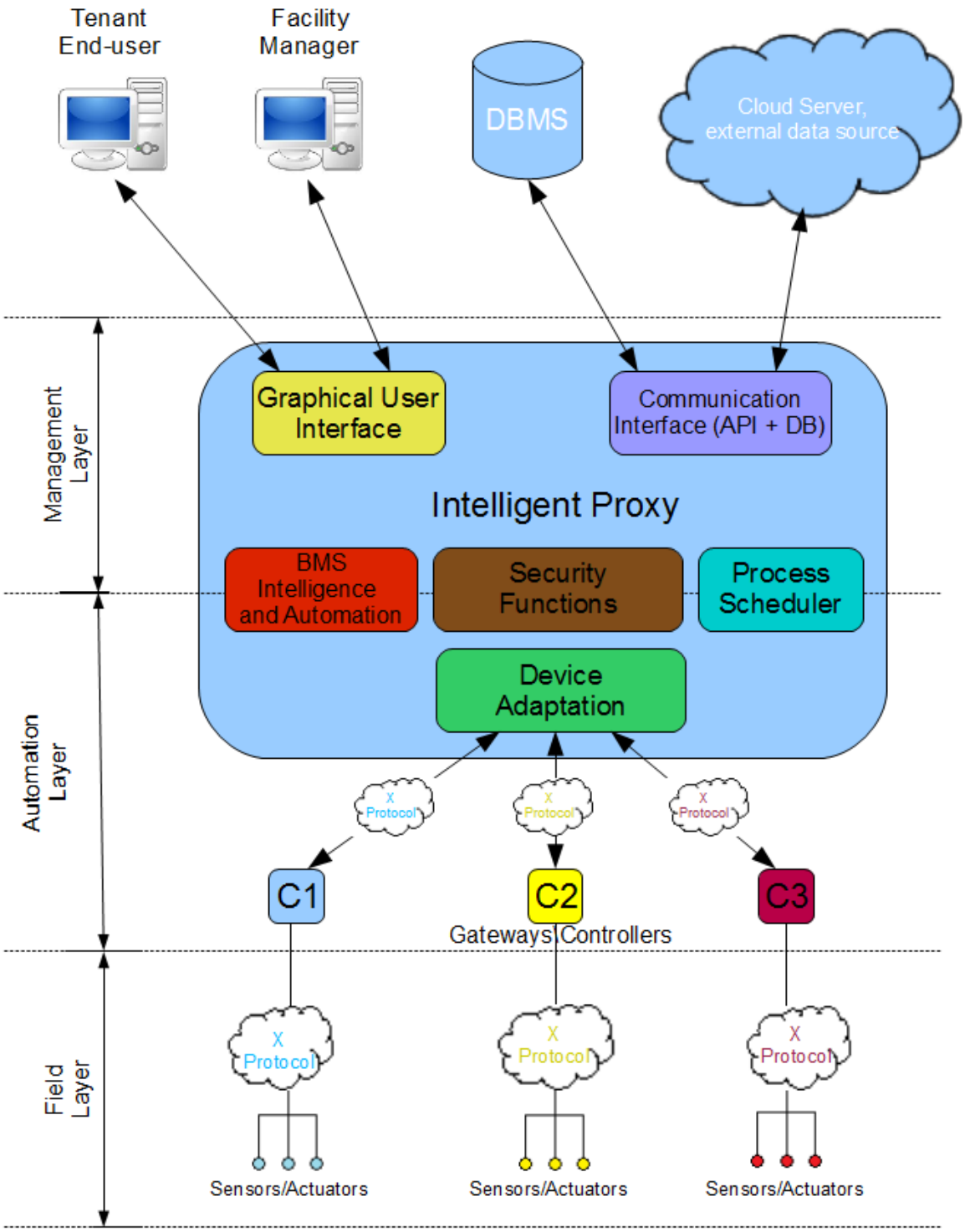

Figure 3.2: Intelligent Proxy High Level Design

The C1, C2, C3 are controllers or gateways operating in the automation layer providing an interconnection to the field where the sensors, actuators and other devices are connected. It is common for these devices to have an Ethernet interface, and in many occasions they are used only for translating and transmitting the filed messages over networks with TCP/IP implementation (gateways). Furthermore, these devices could provide some intelligence and some low level automation functionalities such as handling of alarms (controllers).

To sum up, in the current chapter a high level design of the intelligent proxy is presented and it's main functionalities are going to be defined as entities.

3.3.2.1 Device Adaptation Entity

The device adaptation entity, provides the protocol support for the devices in the building infrastructure. Its main functionality is to present the devices to the system (proxy) as generic devices, so the protocol/technology used would be totally transparent not only to the users (e.g. tenants and facility managers) but also to the BMS designer. In addition, interoperability is achieved by the Device Adaptation entity, as devices from different vendors or generations (supporting various technologies) can now cooperate and provide a holistic functionality of the building automation.

On a software approach the Device Adaptation entity could be presented as an interface for handling all the devices and their data. Hence, there is no need for software developers for building management systems to develop technology-based or protocol-based applications but they can develop reusable generic applications that through the Device Adaptation entity, can now manipulate and operate the building automation hardware.

3.3.2.2 Graphical User Interface Entity

The GUI entity exists to provide access to the BMS information and functionalities (depending on each user's privileges). The GUI could be a web-interface hosted on the proxy, so each user could login with his credentials and have access to

specific functionalities. For example, the facility manager could use the GUI to configure the corridor lighting system or to monitor the building infrastructure and check if maintenance is needed for any device in the building infrastructure (by error notifications or offline devices). Furthermore, in case tenants exist, they could login to check their lease's current state (temperature, humidity etcetera), their current energy consumption or even visualised past data (graphs from data stored in a database) that could help them manage their leased area's energy consumption and cost.

3.3.2.3 Communication Interface

The Communication Interfaces (could also be called as communication center) are presented as the entity that enables and controls the proxy's communications with other entities (outside the proxy), such as data sources or even other proxies, that are internal, located into the building infrastructure, or external (e.g. cloud servers that provide forecasts or back-up databases). This entity is also responsible for storing the BMS data and configurations such as:

- Sensor readings (e.g. measurements and states)
- Operation log (e.g. past commands and events)
- Building infrastructure information (e.g. the building design that could be visualised through the GUI and each device location and extended information)
- Operation profiles (e.g. different energy management profiles for various occasions)

Moreover, the Communication Interface provides an API (application programming interface) in order to provide access to the functionalities of the proxy to external applications and sources. Hence, by using the provided API software running on external devices (e.g. cloud servers) could interact with the proxy and trigger some of its functions. This also enables the many proxies that control different sub-

systems or buildings to interconnect and be operated as one holistic system providing a centralised control of buildings and creating “building neighbourhoods”.

3.3.2.4 Process Scheduler Entity

The Process Scheduler's functionality is similar to a calendar, it has to inform the proxy about processes that have to be executed by the BMS. In addition, this entity is responsible to keep track of priorities and intervals of all processes and it could be visualised as a queue for better understanding.

3.3.2.5 BMS Intelligence and Automation Entity

This entity is the “heart” of the BMS. The BMS Intelligence and Automation entity is responsible for the vital functionalities of the BMS and the interconnection of all the other entities. In more detail, this entity takes the important decisions for the BMS operation, handles the events, executes all functions, and provides the intelligence to the system. The term “Intelligence” is defined by all the automated actions done without human interaction taking into account some predefined operation profile after a variety of events (such as Scheduler event, alarms or even high temperature readings from a sensor). Intelligence is the evolution of automation which used just to execute functions on specific events, without taking into consideration any other data.

To sum up, intelligence is the ability of the BMS taking decisions about which actions will be executed by processing many factors (e.g. historical data, past human interactions, external data, current readings etcetera) in order to provide an optimal performance of the building management. Hence, in cases that there are not a lot of devices and factors that could lead into different actions, intelligence could be described as automation as the most of the executed actions would be “hard-coded”.

3.3.2.6 Security Functions Entity

This entity could be described as an extension to the BMS Intelligence and

Automation. It's main functionality is to provide a policy to the BMS, so several actions could be executed automatically in case of security violations (defined by rules of the policy, based for example on readings of motion sensors), such as notifications for human interaction and triggering of devices functionalities (e.g. phone notification to the security and activation of the lighting systems). Moreover, it could provide also an extension to the GUI where monitoring of the infrastructure for security reasons would be offered (e.g. IP/cameras could be connected to the BMS).

To conclude, these functionalities are treated as a separate entity based on their importance, even though they could be omitted to the entities that where defined earlier in this chapter.

3.3.2.7 High Level Class Diagram of the Proxy

The main objective of this sub-section is to provide a more detailed design of the proxy from a software point of view, by presenting class diagrams that will provide a better understanding of its functionality. Diagrams showing the interaction of the entities for completing BMS tasks, based on today's building automation needs, will be also demonstrated and analysed further on. In addition, decisions on the technologies that could be used for the development of a real proxy implementation will be also presented, explained and justified.

A high level class diagram (Figure 3.3) of the proxy's vital classes is shown. Some of the basic functions are presented in the figure in order to make clear the use of each class and its main functionality. It is obvious that the BMSIntelligenceAndAutomation class has the main role on the proxy. This class implements the configuration of the proxy's operation, and defines the functionalities provided by the proxy, which are the entities defined previously in this chapter. Also, a tree style hierarchical structure could be implemented by defining siblings, children and ancestors of the proxy. Each command or action to be made by the proxy is stored to the scheduler's queue (in case of higher priorities the actions get in the top of the queue), which is responsible for triggering

the doAction() method. Finally, the main intelligence and automation of the BMS services is implemented in the doAction() method, which is responsible for executing all the actions of the building automation by taking into account the policies, automation and intelligence supported by the proxy. The doAction() method is responsible to call the operations provided in the Device-Adaptation class, such as readings, data input to the devices, actuation of devices etcetera.

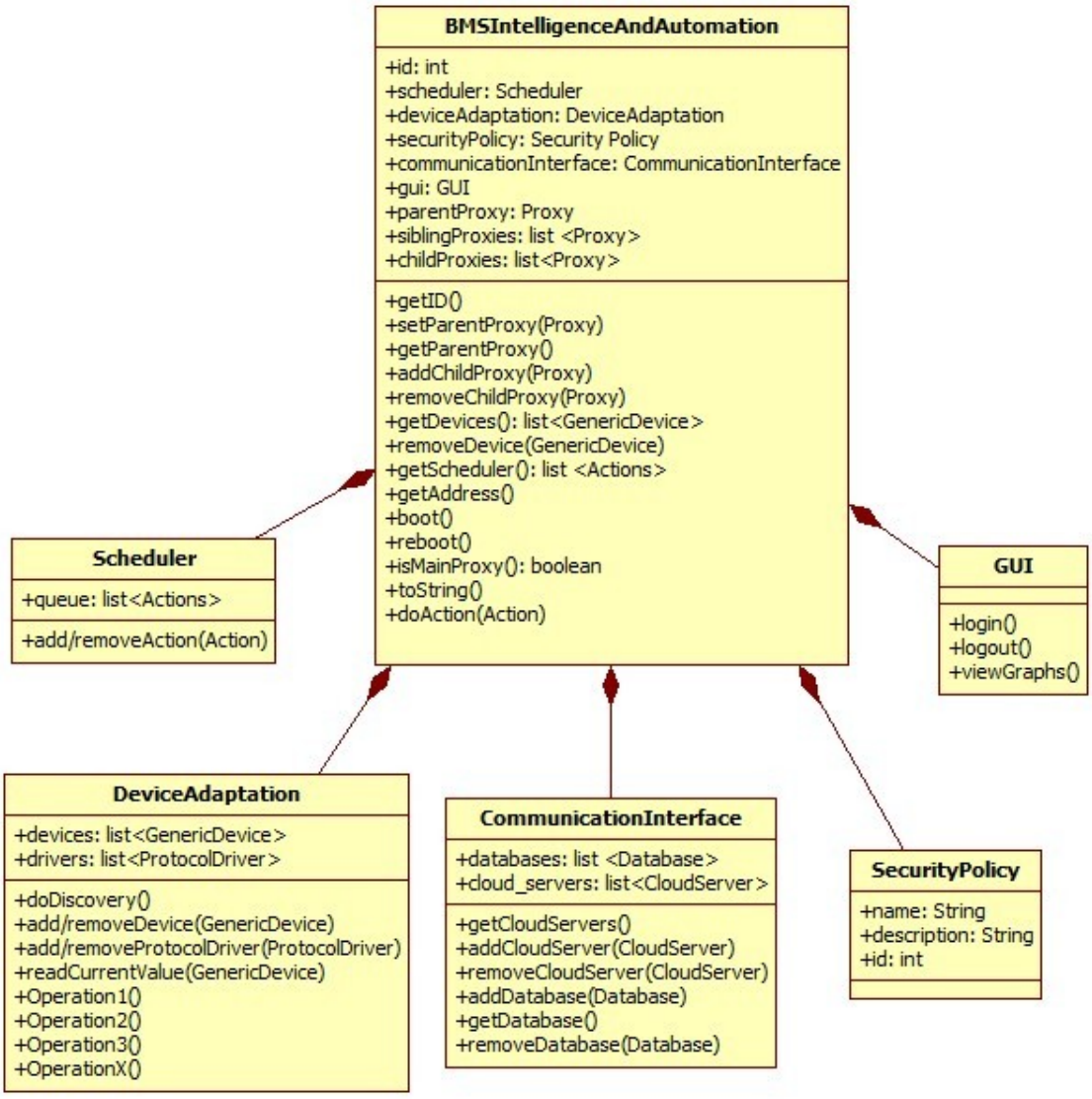

Figure 3.3: High Level Class Diagram

Following, the Device-Adaptation translates each operation into the protocol that is

supported by the device related to the action and sends the command over a network. Depending on the profile implemented, the doAction() may forward data to the Communication-Interface which is responsible to provide feedback to the external sources (in case of the action being result of an external entity's interaction to the BMS) or store data into a database if needed (e.g. proxy operation logging and measurements storing as historical data). Details on the design of the GUI are not of high importance at the current stage, as the GUI is only responsible to present the functionalities provided and their results (e.g. graphs).

3.3.3 Decisions on the Ideal Technologies for a Proxy Implementation

Coming after the description of the vital services that have to be provided by the proxy in order to cover the requirements of a BMS, decisions on the ideal technologies to be used such as programming language, communication protocols and data model have to be made and justified. The outcome of the previous chapter was that there are currently few technologies that could cover the needs of a BMS with only some minor (not vital) differences between them. Hence, decisions on the ideal technologies such as field, automation and management protocol, data model and programming language are not easy to be made based on advantages or disadvantages. In this sub-section an ideal implementation of a BMS and it's proxy is presented further on, from the point of view of this thesis author based on knowledge and experience gained on research and studies made at NEC laboratories.

3.3.3.1 The Middle-ware Operating System

One of the main goals of this thesis, after defining the term of the intelligent proxy, is the design of the software running on the proxy. The development of this software should be an evolution of the existing BMS software, designed from scratch and introducing innovations. Existing BMS were developed based on legacy building automation sub-systems. This technology/protocol-based

approach lead to a very limited compatibility and support of devices and further improvements or extensions to the BMS. It is common for software to have a name, so from now on the operating system of the proxy will be called Intelligent Building Controller (IBC).

3.3.3.2 IBC Architecture and Development

The IBC leverages on the SOA (Service Oriented Architecture), which by definition is a set of principles and methodologies for software development in the form of interoperable services. This means that the functionality of the operating system is split into well defined software components that can be reused for different purposes. The different services provide a well defined interface to access them and to communicate with other services. This provides flexibility to further extension of the BMS and the ability to up-scale the system from a Building Management System into a Neighbourhood Management System.

By implementing the SOA concept, data from the system are exchanged over the network in a well defined form such as XML or JSON (which are commonly used for interfacing SOA services).The benefits of implementing software with the Service Oriented Architecture are modularity, isolation, flexibility, loose coupling, and interoperability, between heterogeneous systems communication.

The software is based on the OSGi (Open Services Gateway Initiative) Knopflerfish framework. OSGi is a service platform for Java that implements a complete and dynamic component model (in this case separate services). The key advantage of software developed on an OSGi approach is that services are totally discrete having the ability of being installed, started, stopped, updated and uninstalled without requiring a system reboot. This makes easier also the use of PnP-like technologies to be implemented.

3.3.3.3 IBC Data Model

The Data Model selected to organize and represent the BMS data is IFC, which is a common data schema that allows to hold and exchange data between

heterogeneous software applications and devices. Since the IFC2x release, IFC provides also an XML schema that makes data exchange even easier. Moreover, data stored in the BMS databases, such as building information (e.g. the position of each device and the building architecture), device descriptions and sensor readings are stored in the IFC data model format. Furthermore, the interfaces provided by the IBC, use IFC definitions to exchange data.

The main benefit of using IFC, is interoperability that is provided by using a well defined and standardised data model. More information about IFC is provided in Chapter 2 and in the bibliography.

3.3.3.4 IBC Management and Automation Layer

In this implementation of the BMS, the Automation layer is almost omitted, as the functionalities of the BMS are not based on automated functions but on dynamic data and intelligence provided by the Management layer. Hence, the IBC is responsible for most of the services provided by the BMS and automation controllers are mostly used as gateways/routers to the devices that do not have TCP/IP interfaces. The Device-Adaptation entity exist in order to provide support for a variety of protocols.

Moreover, it is suggested to the building infrastructure designers to uses devices supporting the BACnet/IP protocol, as it seems to be the market leading technology with an active community supporting it and a variety of devices and applications implementing it. Finally, there are several gateways/controllers for various field layer protocols in the market that translate these protocol messages to BACnet and versa.

3.3.3.5 IBC Field Layer Protocol

The implementation described in this thesis provides interoperability between multi-vendor devices and devices from different generations. This means that devices using any of the most commonly used technologies (described in Chapter 2), should be supported even thought it is suggested to building designers and

facility managers to prefer BACnet implementations which seem to be the market leading products with an active community supporting them and a variety of existing compatible applications.

3.4 Conclusions

The variety of technologies and protocols used in building automation and BMS had as a result many heterogeneous sub-systems to operate in the same building infrastructure. The need of a centralised management and control of these sub-systems made the use of a bridging device essential. While trying to advance the BMS technologies the definition of this device was necessary. From further research, the idea of creating a more intelligent device arose, this device is defined as an intelligent proxy.

The main responsibilities of the intelligent proxy are to provide device adaptation, an API for the BMS and last but not least the BMS intelligence, meaning the ability to take decisions for each and every action made by the BMS.

Moreover, several dilemmas about the ideal design of this device, such as the software development architecture in which it should be based and the supported protocols appeared and were resolved. Hardware was not taken into consideration, as many different implementations meet the requirements. Finally, considering the opportunity of creating an optimal implementation from scratch, the ideal technologies to be used were defined based on the research done by this thesis author. As a result, the software is based on the Service Oriented Architecture and the programming language to be used is Java (OSGi Knopflerfish framework), the data model is IFC and the main suggested protocol is BACnet (a variety of other protocols is also supported by the proxy).

To conclude, a middle-ware device and it's software architecture were defined in order to provide interoperability between the existing or future sub-systems of a building automation infrastructure providing the ability to be centrally managed and operated in a form of a complete BMS.

Chapter 4

4 Pre-Implementation Model

4.1 Introduction

Following the research on the building automation and management technologies, the definition and design of the intelligent proxy was presented. This design defined the basic functionalities of the proxy. The main requirement was to provide an integration of the existing building automation technologies and sub-systems, in order to be used as one holistic system providing a homogeneous monitoring and control platform. In this chapter the implementation of a prototype is presented and a demonstration scenario is used for better understanding.

The development started by creating a pre-implementation model. This model is divided into two parts. The first one provides a virtual building automation field implementation of some building automation devices and the second one implements a virtual (software) controller, which conducts data exchange with the virtual devices cultivating a base for the actual proxy's operating system (IBC) to be developed later on.

Moreover, the use of this model was vital for deeper understanding of the building automation operations and technologies and it was developed alongside with the research. Outcomes from this testing environment helped forming the final proxy design and architecture.

4.2 The Scenario

The scenario of this model consists of a controller that would be able to connect via one or multiple interfaces into one or multiple networks and discover the operating virtual (or real if they exist) devices and then keep a reference (network address, port, id etcetera) of each device discovered. Further on, using these

references, the controller should provide functionalities of polling each device's data such as current state, current value, device information or any other data available by the device.

The next step, after simple communications are enabled between the controller's interface and the devices, is to cover the basic requirements of the intelligent proxy state of the art. First and most important the device adaptation entity has to be created, so devices from various technologies could be supported, and their management (e.g. data polling from a user or command transmitting by the Intelligence) outside of the Device-Adaptation software entity should be generic and all devices of the same type (e.g. temperature sensors) would be treated by the Intelligence and Automation entity the same way, without the need of considering their individual technology.

Further on, as some functionalities are already operational, the need of data output as a feedback of the operation leads to the creation of a GUI. For the current development stage of the model, a simple “command line” output platform is efficient and covers our needs, so the output of the Knopflerfish platform is used to print on the monitor the feedback data. Moreover, those data are temporary stored in the memory and can stored further on in a database.

Finally, some simple automation is added, so device discovery and data polling is done with intervals. In addition, configurations for both parts of the model are read from configuration files at booting time or during the platform operation.

4.3 The Virtual Field Part

Firstly, for testing reasons of the model some building automation devices were needed. The most flexible implementation was to implement virtual devices in software, so there would be no limits on their quantity, type or functionalities. In addition, this virtual field would be better to be hosted at a different machine than the proxy's development one in order to enable network traffic generation and monitoring for a more realistic model. Once more, for reasons of flexibility, virtual

machines were used, Oracle Virtual Box tool was used for their creation and management. The building automation protocol used by the Virtual Field for those tests was BACnet/IP.

4.3.1 The BACnet/IP Protocol Framework

Since Java is the main language of this model's development, a framework written in Java implementing the BACnet protocol stack had to be used. An Open Source implementation would provide deeper knowledge on the BACnet protocol and would give the opportunity for adjustments of the code for the needs of the testing. BACnet4J was chosen, an open source framework developed and maintained by Serotonin Software which is mainly used by the Mango M2M platform, one of the worlds most popular open source Machine-to-Machine software.

4.3.2 The Field Concept in More Details

From the developer's point of view, the main functionality of the field was to virtualize several building automation devices over a network (we also used multiple network interfaces during the tests). In addition, the functionality to configure these devices and their supported services based to configuration data read from a file is provided. Hence, testing flexibility on the quantity and the type of the devices is achieved.

Furthermore, a thread is responsible to change the values of the created objects (e.g. temperature measurement) in regular intervals, with a pseudo-random algorithm that generates values inside logical limits to provide a more realistic testing environment.

4.4 The Proxy's Operating System Prototype

Consequential to the field development, the skeleton of the IBC was created and some of the basic requirements were implemented. The first task was to enable communications with the field level and to provide a discovery for building

automation devices over the network (BACnet devices at the current stage of the modelling). The Device Adaptation support was the next step, mapping protocol based devices to generic data-model based ones, that are going to be handled by the proxy's entities. Extended information about the devices, such as the device type and it's position in the infrastructure, were provided to the model by a database. Furthermore, a simple GUI had to be used for the modelling procedure, printing the operation feedback.

4.4.1 The Pre-Implementation Model with More Details

In this sub-section, technical details from the developer's point of view are going to be provided for the implementation of the IBC skeleton. As it was explained in Chapter 3, the IBC design is based on the SOA, which by definition means that the system development is focused on service-orientation. Hence, the best choice for implementation is the OSGi technology, which is a platform that provides a set of specifications that define a dynamic component system. These specifications enable applications to be composed of many different reusable components (Services). In particular, the Knopflerfish framework was used for the development.

The reasons for development of this part of the software was for testing the building automation technologies, gaining deeper theoretical and practical knowledge over the protocols and finally improve the IBC design and provide a skeleton for the IBC to be developed later on.

The operations provided at the current stage by the software are a controller that does a discovery to predefined networks for building automation devices, using the broadcast address of the network and transmitting a Who-is request. After the Who-is request is received, the devices have to reply with an I-am message providing basic informations of the devices which are kept by the controller as “contact details” of the devices. Further on, the controller can transmit data directly to each device (without using the broadcast address of the network). The controller could request more data from a device or just send an operation

command. In addition, the devices discovered, are mapped to a generic form of their type, based on configuration provided by a database.

Moreover, for demonstration reasons, the data of each device and readings of their values are printed to the Knopflerfish GUI. These operations are repeated in regular intervals.

4.5 A Brief Demonstration

In this sub-section, a brief demonstration of one simple scenario using the pre-implementation model defined in this Chapter is presented. In addition, network traffic captured with the “Wireshark Network Analyzer” tool is explained and some screen-shots of the output are also analysed.

4.5.1 The Demonstration Concept

In this demonstration scenario, the Virtual Field level consists of 3 devices, and 4 BACnet objects (2 inputs and 2 outputs) registered on each of these devices. Moreover, in an interval of 5 seconds the object values of the devices are changed with a pseudo-random algorithm. On the other hand, the controller does a discovery to the network, gets a reply from each device and prints their data and objects into the Knopflerfish console. In addition, following a preconfigured database, it maps the devices into generic ones based on the device type.

Table 4.1: Devices Used In the Demonstration

Object Identifier (ID)	IP:Port	Object List	Description
Device 5	192.168.56.101:47809	Analog Output 0	Co2 Sensor
Device 6	192.168.56.101:47811	Binary Input 0	Light Sensor
Device 7	192.168.56.101:47812	Analog Input 0 Binary Output 0	Temperature Sensor
Device 1234	192.168.56.1:47808	Null	Controller's BACnet Interface

The devices on the virtual field are initialized by data read from a file, providing the device ids, names and listening ports. By default, the BACnet UDP port is 47808, but due to the implementation of more than one device under the same IP address each device must use a different port, as shown in Table 4.1. The objects are also defined and initialized by the configuration file. In addition, the database provides the relation between the objects and the devices and the virtual field software registers the objects to the devices.

Following, the controller sends the Who-is request to the network (to all the ports used by the field devices). Then, each device replies with an I-Am message which includes a brief description of the device (e.g. IP address, id, port and the services that are supported). The captured network packets are shown in Figure 4.1.

Source	Destination	Protocol	Length	Info
192.168.56.1	192.168.56.255	BACnet-APDU	54	Unconfirmed-REQ who-Is
192.168.56.1	192.168.56.255	BACnet-APDU	54	Unconfirmed-REQ who-Is
192.168.56.1	192.168.56.255	BACnet-APDU	54	Unconfirmed-REQ who-Is
192.168.56.1	192.168.56.255	BACnet-APDU	54	Unconfirmed-REQ who-Is
192.168.56.101	192.168.56.255	BACnet-APDU	62	Unconfirmed-REQ i-Am device,5
192.168.56.101	192.168.56.255	BACnet-APDU	62	Unconfirmed-REQ i-Am device,6
192.168.56.101	192.168.56.255	BACnet-APDU	62	Unconfirmed-REQ i-Am device,7

Figure 4.1: Who-Is Request

Meanwhile, the user gets feedback on the actions that are made by the controller, in a command line console provided by the Knopflerfish framework (Figure 4.2).

```
[stdout] Bundle Started...
[stdout] Send Broadcast WhoIsRequest() to --> 192.168.56.255
[stdout] Waiting for iAm to get Recieved
[stdout] iAmReceived!
[stdout] iAmReceived!
[stdout] iAmReceived!
[stdout]  waited for iAmReceived: 5001 ms
[stdout] 3 devices where found!
[stdout]
```

Figure 4.2: Controller Output: Network Discovery

New objects representing the generic devices with the data extracted from the I-Am replies are now created by the device adaptation entity and are stored temporarily in the controllers memory (or a file if needed). Now, “references” to the devices exist on the controller and by user prompt additional information about a

specific device could be queried over the network. Furthermore, the user could send a command to the device or remotely configure it if he has the correct privileges (e.g. facility manager), but in this demonstration only the data request and their printing on the users monitor are going to be presented. Thus, following the device “mapping”, the controller requests from the devices their extended information, including their objects data and current value. Figure 4.3 shows the network traffic captured with Wireshark.

Source	Destination	Protocol	Length	Info
192.168.56.1	192.168.56.101	BACnet-APDU	59	Confirmed-REQ readProperty[0] device,7
192.168.56.101	192.168.56.1	BACnet-APDU	68	Complex-ACK readProperty[0] device,7
192.168.56.1	192.168.56.101	BACnet-APDU	65	Confirmed-REQ readPropertyMultiple[1]
192.168.56.101	192.168.56.1	BACnet-APDU	90	Complex-ACK readPropertyMultiple[1]
192.168.56.1	192.168.56.101	BACnet-APDU	59	Confirmed-REQ readProperty[2] device,7
192.168.56.101	192.168.56.1	BACnet-APDU	70	Complex-ACK readProperty[2] device,7
192.168.56.1	192.168.56.101	BACnet-APDU	79	Confirmed-REQ readPropertyMultiple[3]
192.168.56.101	192.168.56.1	BACnet-APDU	1089	Complex-ACK readPropertyMultiple[3]

Figure 4.3: Extented Device Information Request

The BACnet/IP protocol stack as shown in Figure 4.4 to provide better understanding of the data encapsulation, and the way that data travel through the system before getting transmitted over the network. As it is presented, the data from the BACnet application are encapsulated into a UDP packet, that is encapsulated into an IP packed later on and finally medium-based headers are also added before the packet is transmitted. In this case, the MAC protocol is Ethernet, the most commonly used protocol on local area networks.

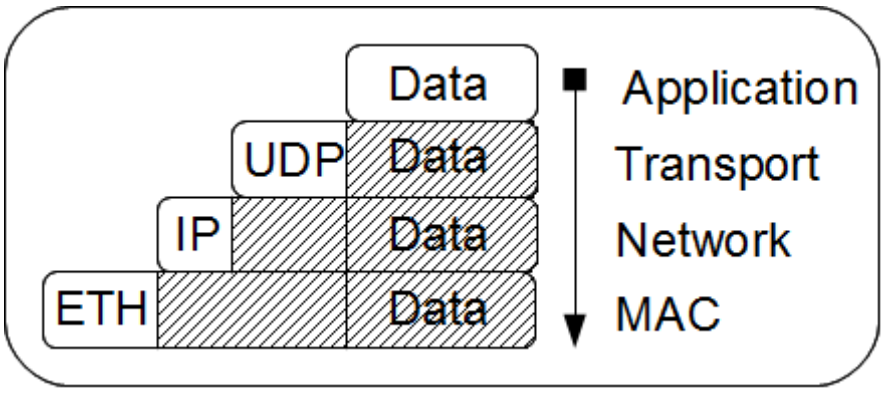

Figure 4.4: BACnet Protocol Stack

Now that the encapsulation process is clear, the structure of a packet transmitted

over the network, captured with the Wireshark tool is presented below (Figure 4.5).

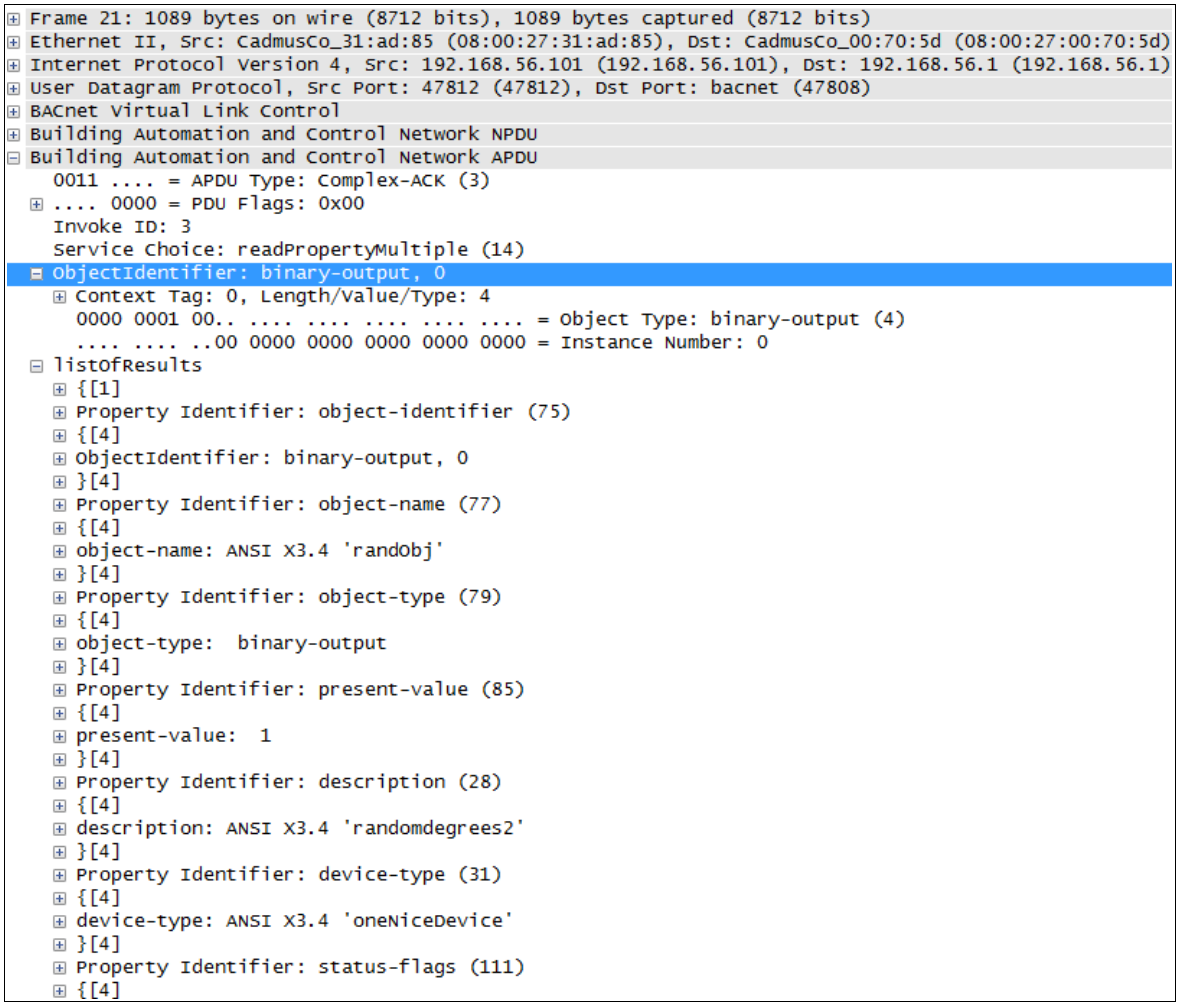

Figure 4.5: BACnet/IP Transmitted Packet

If a BACnet protocol packet is analysed, for example the packet in Figure 4.5, an internal structure is revealed. BACnet implements a Virtual Link Control, which provides some extra information to BACnet devices in order to achieve better routing via BACnet routers and gateways. In addition, more informations about the version number, priority etcetera of the message are included in the NPDU, and finally the “raw” requested data are inside the APDU, containing extended information about the device and its objects (depending on the request).

The last step of this demonstration is to present the data that was requested by the user and fetched by the controller to the user interface. The data captured by

the network interface are forwarded to the operating system, and the Java listener on the UDP port 47808 forwards them to the controllers software which finally prints them. A screen shot of the printing is shown in Figure 4.6

```
Printing Device ---> RemoteDevice(instanceNumber=7, address=Address(networkNumber=0, macAddress=[c0,a8,38,65,ba,c4]
[stdout] Start read properties
[stdout] Properties read done in 12 ms
[stdout]        Device 7
[stdout]                Protocol revision = 0
[stdout]                Database revision = 0
[stdout]                Active COV subscriptions = []
[stdout]                Model name = thePerfectName3
[stdout]                Object identifier = Device 7
[stdout]                Object list = [Analog Input 0, Binary Output 0]
[stdout]                Number of APDU retries = 2
[stdout]                Max APDU length accepted = 1476
[stdout]                Object type = Device
[stdout]                Object name = BACnet device
[stdout]                Segmentation supported = both
[stdout]                Protocol object types supported = [true, true, true, true, true, true, true, true, true, tr
[stdout]                Protocol services supported = [false, true, true, false, false, true, false, false, false,
[stdout]                Protocol version = 1
[stdout]                Vendor identifier = 236
[stdout]                Vendor name = Serotonin Software Technologies, Inc.
[stdout]                System status = 0
[stdout]                Local date = WEDNESDAY MARCH 21, 2012
[stdout]                Local time = 16:35:52.43
[stdout]                Firmware revision = not set
[stdout]                Device address binding = []
[stdout]                APDU segment timeout = 1000
[stdout]                Application software version = 1.0.1
[stdout]                APDU timeout = 5000
[stdout]        Analog Input 0
[stdout]                Status flags = [false, false, false, true]
[stdout]                Units =
[stdout]                Object identifier = Analog Input 0
[stdout]                Object name = randObj
[stdout]                Present value = 58.0
[stdout]                Object type = Analog Input
[stdout]                Out of service = false
[stdout]                Event state = normal
[stdout]                Device type = oneNiceDevice
[stdout]                Description = degreesInCelcius
[stdout]        Binary Output 0
[stdout]                Status flags = [false, false, false, true]
```

Figure 4.6: Data Output to the User Interface

Finally, on the monitor there are several other operation feedback and a prompt to the user to repeat the discovery or the data request is displayed. (The interested reader can refer to the literature - (2))

4.6 Conclusions

In the current chapter, a prototype of the IBC is presented. In addition, for testing and demonstration reasons a virtual building automation field is also developed,

providing virtual devices. The skeleton of the software and its operation is described and screen shots of its operation are provided. Moreover, the network traffic is monitored, captured and analysed with the Wireshark tool in order to gain deeper theoretical and practical knowledge on the protocol used and have a better understanding of the IBC functionality.

Finally, a demonstration scenario is provided, implementing a network data exchange between the prototype and the virtual field and giving feedback to the user interface.

Chapter 5

5 Conclusions and Further Research

5.1 Introduction

This chapter presents the concluding remarks of this thesis and topics for future research. A brief description of the research done as well as its results are also presented. Furthermore, an overview on the benefits gained from the integration of Information and Communication Technologies to the Building Automation Technologies is provided in order to simplify this thesis scope for the reader.

5.2 Conclusions

The main goal of this thesis was to define novel technological solutions that could enable improvements on the building performance in both energy saving and management. Moreover, it was vital to identify the barriers of the existing building automation management systems and technologies that could limit their extensibility and improvement. A research on today's most commonly used building automation technologies, followed by their presentation and comparison, was vital in order to recognise their advantages and disadvantages, and move on the next step.

After defining the existing technologies, solutions on how they could be advanced with the use of ICT, that could lead to an improved energy consumption profile, were provided based on real world requirements extracted by CAMPUS 21 demonstration sites and theoretical use cases (e.g. unifying the sub-systems for better improved management and use external data like forecast to dynamically calibrate the BMS operation profile).

Following the definition of the requirements, a design of a solution to cover them was provided. This thesis solution defines an intelligent proxy that interconnects

the existing and future building automation sub-systems, overcoming compatibility issues by providing interoperability between devices of various vendors and generations. Definitions on the technologies used by proxy's operating system and it's software architecture were provided. Moreover, innovative functionalities, such as API to external data sources, were included into the proxy design.

Finally, the implementation of a prototype based on the state of the art presented in Chapter 3 was the last stage of this thesis. The skeleton of the IBC and some basic functionalities were implemented. Furthermore, a virtual building automation field was developed for testing and demonstrating the prototype. A simple scenario, implementing communication between the proxy and the virtual devices was presented and analysed, providing deeper technical understanding to the reader.

5.3 Further Research Areas

As for future work, research and development of intelligent algorithms that could process real-time data and their integration to a BMS in order to dynamically calibrate and optimise its operation performance would be an interesting topic. In addition, further extension and development of the model provided by this thesis is necessary to realize and validate energy efficiency gains in real world deployments.

References

[1] “Building management system”, Wikipedia, [Online]. http://en.wikipedia.org/wiki/Building_Management_System (Last accessed: February 2012).

[2] “Building automation”, Wikipedia, [Online]. http://en.wikipedia.org/wiki/Building_automation (Last accessed: February 2012).

[3] H. Merz, T. Hansemann, C. Hübner, Building Automation Communication systems with EIB/KNX, LON and BACnet (Signals and Communication Technology), Springer 2009.

[4] BACnet - Official Website of ASHRAE SSPC 135, [Online]. http://www.bacnet.org/ (Last accessed: February 2012).

[5] Konnex– Official website of KNX Association , [Online]. http://www.knx.org/ (Last accessed: February 2012).

[6] Echelon - LonWorks Platform for Control Networking, [Online]. http://www.echelon.com/products/lonworks_control_networking.htm (Last accessed: February 2012).

[7] LonMark International, [Online]. http://www.lonmark.org/ (Last accessed: February 2012).

[8] OPC Foundation, [Online]. http://www.opcfoundation.org/ (Last accessed: February 2012).

[9] oBIX –Open Building Information Exchange, [Online]. http://www.obix.org/ (Last accessed: February 2012).

[10] OASIS Open Building Information Exchange (oBIX) TC, [Online]. http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=obix (Last accessed: February 2012).

[11] BuildingSmart Organization – international home of Open BIM, [Online]. http://www.buildingsmart.com (Last accessed: February 2012).

[12] ISO/PAS 16739:2005 - Industry Foundation Classes, Release 2x, Platform Specification (IFC2x Platform), http://www.iso.org/ .

[13] ISO International Standard 10303-11:1994, Industrial automation systems and integration — Product data representation and exchange — Part 11: Description methods: The EXPRESS language reference manual, International Organization for Standardization, http://www.iso.org/ , Geneva, Switzerland (1994).

[14] Green Building XML (gbXML) Schema: a Building Information Model Solution for Our Green World, [Online]. http://www.gbxml.org/ (Last accessed: February 2012).

[15] RFC 4944, IETF 6LoWPAN, [Online]. http://tools.ietf.org/html/rfc4944 (Last accessed: February 2012).

[16] CAMPUS21 D4.1 (WP4) Deliverable ,FP7, Seventh Framework Programme with project reference number: 285729 , (Submitted on 31 December 2011)

[17] “OLE for process control”, Wikipedia [Online] http://en.wikipedia.org/wiki/OLE_for_process_control (Last accessed: February 2012).

[18] “OPC Training Institute” [Online]. http://www.opcti.com (Last accessed: February 2012).

[19] “Control and Automation Management of Buildings and Public Spaces” [Online]. http://zuse.ucc.ie/CAMPUS21/index.html (Last accessed: March 2012).

[20] “Community Research and Development Information Service” [Online]. http://cordis.europa.eu/projects/100248_en.html (Last accessed: March 2012).

Bibliography

(1) H. Merz, T. Hansemann, C. Hübner, “Building Automation Communication systems with EIB/KNX, LON and BACnet (Signals and Communication Technology)”, Springer 2009.

(2) Joel Bender, Mike Newman, “BACnet/IP” A detailed technical description of how BACnet devices may communicate using the Internet Protocols. [Online] http://www.bacnet.org/Tutorial/BACnetIP/default.html (Last accessed: February 2012).

(3) Scott Cosby, “BACnet Architecture”, [Online]. http://www.chipkin.com/articles/bacnet-architecture (Last accessed: February 2012).

(4) “OSGi”, Wikipedia [Online] http://en.wikipedia.org/wiki/Osgi (Last accessed: March 2012).

(5) “Knopflerfish”, Wikipedia [Online] http://en.wikipedia.org/wiki/Knopflerfish (Last accessed: March 2012).

(6) “Knopflerfish – Open Source OSGi” [Online] http://www.knopflerfish.org/ (Last accessed: March 2012).

(7) “Mango, open source M2M” [Online] http://mango.serotoninsoftware.com/ (Last accessed: March 2012).

(8) “BACnet4J – BACnet I/P for Java” [Online] http://sourceforge.net/projects/bacnet4j/ (Last accessed: March 2012).

(9) National Joint Apprenticeship & Training Committee for the Electrical Industry and American Technical Publishers, “Building Automation: Control Devices and Applications”, January 1st 2008.

(10) National Joint Apprenticeship & Training Committee for the Electrical Industry and American Technical Publishers, “Building Automation: Integration with Open Protocols”, January 1st 2009.

(11) “Wireshark – The world's foremost network protocol analyzer” [Online] http://www.wireshark.org/ (Last accessed: March 2012).