18th International Conference on the Application of Computer
Science and Mathematics in Architecture and Civil Engineering
K. Gürlebeck and C. Könke (eds.)
Weimar, Germany, 07–09 July 2009

A Service Oriented Architecture for Building Performance Monitoring

P. Stack *, F. Manzoor, K. Menzel and B. Cahill

* *Department of Civil and Environmental Engineering*
University College Cork (UCC)
College Road, Cork, Ireland
E-mail: pstack@student.ucc.ie

Keywords: Service Oriented Architecture, Building Performance Monitoring, Web Services, Integration

Information technology plays a key role in the everyday operation of buildings and campuses. Many proprietary technologies and methodologies can assist in effective Building Performance Monitoring (BPM) and efficient managing of building resources. The integration of related tools like energy simulator packages, facility, energy and building management systems, and enterprise resource planning systems is of benefit to BPM. However, the complexity to integrating such domain specific systems prevents their common usage. Service Oriented Architecture (SOA) has been deployed successfully in many large multinational companies to create integrated and flexible software systems, but so far this methodology has not been applied broadly to the field of BPM. This paper envisions that SOA provides an effective integration framework for BPM. Service oriented architecture for the ITOBO framework for sustainable and optimised building operation is proposed and an implementation for a building performance monitoring system is introduced.

1 INTRODUCTION

In order to have effective BPM the performance of buildings need to be monitored continuously and any actions that cause waste and inefficiencies needs to be recognised. Aspects that have been addressed are methods of collecting information from sources of building data and combining all the information for end-user applications or other applications that function based on the results of the building performance.

The combination of cheap, high-performance microcomputers together with the emergence of high-capacity communication lines, networks and the Internet has produced exponential growth in information technology (IT) and in BPM systems (BPMS) [1]. A facility manager is able to monitor most items of equipment and processes that occur in the operations of the building through IT integration [2].

Traditional BPMSs rely heavily on a collection of separate system that operate independently and were implemented with proprietary communication protocols that make expansion, modification, updating and integration with other building or plant information and control systems extremely difficult [1, 3]. With integral elements to gauge energy and performance levels of a building, a BPM must incorporate a myriad of software applications.

Currently, there is a huge choice on the market for automated Building Management Systems (BMS), which are generally vendor-specific platforms. These systems are installed into the building fabric on a separate network to existing IT infrastructure. To monitor the facility a system must collect data from a sensor network. For control and actuation of building plant like the heating, ventilating and air conditioning (HVAC) system a BMS must send parameterised messages to actuators to adjust operation levels. Over time a buildings infrastructure may require retrofit and extensions so a monitoring system must be easily extensible to interpret a new environment. With regard to occupancy levels, an optimised and calibrated energy simulation can be performed. Everyday operations in terms of facility management (FM) and enterprise resource planning (ERP) can be more easily determined by extracting relevant information from an improved monitoring platform.

This paper purposes the usage of Service Oriented Architecture (SOA), based on web service protocols like Simple Object Access Protocol (SOAP) and traditional Transmission Control Protocol (TCP)/Internet Protocol (IP) for communication between elements of the BPM system, allows for the creation of a more effective and standardised platform for BPM.

SOA is a software development paradigm that is tailored to increase the responsiveness and agility of a system. SOA has been adopted by many enterprises to help break down the complexity of IT solutions to help improve their business processes. The dismantling of the current business process into smaller, distributed services is a challenge. The increased flexibility makes SOA a natural improvement over the object-oriented (OO) and the component-based architectures for software development [4].

The basic concept of SOA is to split a business processes in small pieces of software called components. A component is self-contained and has a well-defined interface and can be developed, delivered, installed and run independently [5]. The component is built so that it can be easily composed of and combined with other components for enterprise needs [6]. The component has a software interface with infrastructure. The business logic within these components are based on the principles of OO development. These business processes are termed as services in the analogy of SOA [4].

Web services technology, a platform to implement SOA distributed computing, provides the core capabilities to those offered by Electronic Data Interchange (EDI) systems but are simpler and less expensive to implement. With these advances in technology and with computers and software applications becoming integral to organisations, there is a quest for organisations to perform their processes more efficiently [7]. A key to unlocking the interoperability issues of a multi-system organisation can be gained through technology for EDI. Early attempts have been made in the supply chain management [8] and this enables business to conduct secure and reliable electronic transactions. EDI systems improve efficiency and promote better accounting practices but can be difficult to maintain due to tightly coupled implementation methods and are expensive for companies to adopt [9]. Web services are built using open standards so fewer compatibility issues arise.

In order to define an effective SOA solution for BPM, an identification process must be performed to find the services of the core components. A service can be an individual logical unit of a business process. The business logic is encapsulated within the service and it interacts with service consumer through an interface. Services are designed to be flexible, so that extra functionality can be easily upgraded. They can be reusable so other processes in the platform can avail of their processing. Plus, services can be published by the service provider so a service consumer can bind through a service handle.

2 BUILDING PERFORMANCE MONITORING

A number of various interrelated factors influence the overall building performance and a complete BPM is a complex function depending on different parameters. Simulation and monitoring programs divide these parameters on the basis of four main groupings of variables: energy consumption of building components and zones, user thermal comfort provided, environmental conditions, and occupant behaviour. When providing information of BPM, one is faced with many challenges that are either conceptual or practical. The conceptual challenges can be the definition of BPM to different stakeholders of a building and their expectations from the output data. Practical challenges, to name a few can be actual vs. modelled performance, data quality, availability and consistency, isolating effects of individual buildings, and benchmarks for comparisons, etc.

The variety of BPM components requires it to be performed with a collection of separate systems. These systems need to be integrated in order to create a common platform and obtain the benefits from a cohesive system. Taking into account multiple stakeholders is an essential aspect of any business model [10]. Software must fulfil the objectives of prospective users and be tailored to their respective roles. Effective communication channels between individual components can also be automated to avoid manual interaction.

Our research aims to achieve a complete monitoring platform from sensing and actuation layers to data management systems, taking into account current building architecture and structure from Computer-Aided Design (CAD) models with relative occupancy patterns. This should facilitate the production of more accurate energy simulations as well as creating, and presenting context information to relevant stakeholders to assist in their daily operations. Figure 1 gives an overview of the framework design resulting from the Irish BuildWise project [11, 12]. This O Model focuses on the activity information flow in building monitoring. It is currently extended in the ITOBO project (cf. Section 6). The basic elements and extensions will be introduced next.

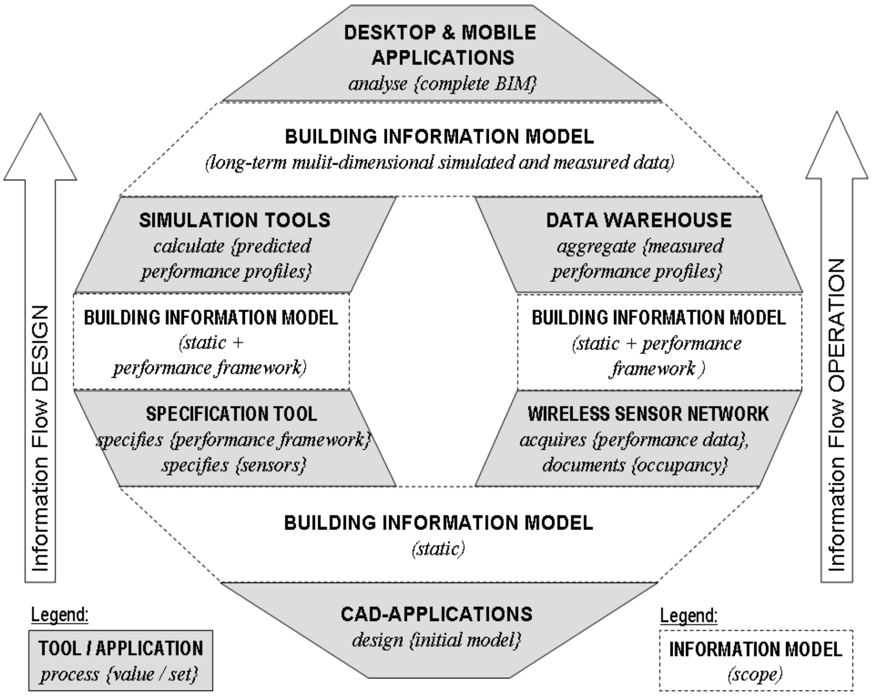

Figure 1: The “O” Model – Building Performance Framework

2.1 Sensor and Actuation Networks

Sensor and actuation networks provide the data for online building performance evaluation like the temperature, humidity, or CO_2 levels of rooms, or the energy consumption of equipment. As data sources, both wired and wireless networks are currently integrated in the ITOBO project. Building Automation Systems (BAS) usually consist of wired network technologies like LON, EIB/KNX, or BACnet to monitor and control a building. They are normally integrated in the Building Management System (BMS) via Gateways connected to an Ethernet backbone. Nevertheless, Wireless Sensor and Actuator Networks (WSAN) receive a high attention as extension or replacement of wired BAS. WSAN require no wiring and, thus, are easy to integrate in a building. Supported by energy-harvesting approaches they work without batteries, and allow maintenance free service. Especially in existing building structure they allow a quick and inexpensive equipment of sensors in a building.

2.2 Radio Frequency Identification (RFID)

RFID [13] is a technology that enables wireless and automatic identification of small electronic devices called RFID tags or RFID transponders. The identification is done by each tag having its own unique tag ID and these tags when applied on equipment or handed out to occupants of a building can identify and track them in certain building spaces and retrieve specific information. Data collected in this way can be forwarded for storage in a central database system where other information parameters are related to each tag ID.

This easy tagging and identification of equipment and persons supports a broad variety of application scenarios reaching from facility management, to localisation and tracking, as well as decentralised information management. The availability of active RFID devices allows also for small form-factor wireless sensors. The integration of RFID into a BPM framework allows further unique features like the tracking of energy usage by occupants or the automatic appliance of user profiles to the thermal comfort preferences in the intelligent control of a building.

2.3 Data Management Systems

A key element for the collaboration of tools in the ITOBO framework is a shared data source allowing information exchange and aggregation. In contrast to normal database, a Data Warehouse (DW) is used in ITOBO. Data warehouses provide not only the storage and the historic consistent electronic archiving of data, but also the aggregation of information for reporting and decision support. Therefore, the data is structured in pre-specified *materialised views* and *cubes*. DWs use structural information classified in so-called *dimensions* (e.g. tenants in a building or the time), to aggregate collected data like the energy consumption of all rooms rent by a tenant per month.

2.4 Building Product Models

Building Product Models like IFC or aecXML contain much information relevant for BPM like the structural geometry of the building including storeys, rooms, the location of sensors, or the usage and tenants of rooms. This information should be imported into the data warehouse for general availability to other tools and also to provide the structural information used by the data warehouse for aggregation.

2.5 Energy Simulation Packages

The ideal performance of a building based on the design information is very relevant to evaluate and optimise a building's performance in terms of operational procedures. Energy Simulation packages generate a building's energy requirements and consumptions estimation. They should be integrated in a way that allows them on the one hand to import information already specified in Building Product Models and export the ideal performance to the data warehouse. Besides building elements and components, the environmental impact and occupant behaviour also need to be taken into account [14]. Any client tool may access the ideal performance to compare it with real measurements from the sensor and actuator network, leading to complete monitoring of building performance.

2.6 Maintenance Tools

The various information collected in the data warehouse also benefits FM tasks. First, the availability of online performance measurements allows common Fault Detection & Diagnosis (FDD). Second, additional ideal performance data and data mining in the data warehouse enable enhanced FDD approaches. This also provides more relevant data for energy efficient space usage and maintenance scheduling. Finally, RFID simplifies maintenance actions with decentralised information management and easy equipment management.

2.7 Monitoring Tools

A first step in the ITOBO project is to present the collected building performance data available to stakeholders. Therefore, user friendly Graphical User Interfaces (GUI) are developed that take advantage of the unique features like the data warehouse. The integration aspects of these interfaces are introduced in this paper. The GUI also provides a mechanism for the collection of feedback about the thermal comfort of the user or occupant, which is stored in the data warehouse for further usage in the intelligent control.

2.8 Intelligent Building Control and Automation Systems

Intelligent control is the next step from monitoring a building by controlling a building from its main physical measurements like temperature, humidity or illumination, and towards optimising the control to provide simultaneous individual user comfort and reduction of a building's energy consumption. Therefore, both the user thermal comfort, based on user feedback, and the building's energy consumption are continuously monitored and used as a decision base for an artificial intelligence control approach.

3 SERVICE ORIENTED ARCHITECTURE

In a world of increasing globalisation, rapid business development, and technological advancements, there is a growing need for integration and compliance between different stakeholders. This integration is necessary to increase output efficiency and optimisation of organisational processes and to leverage capabilities across different domains [15]. This multi-organisational integration and inter-domain collaboration requires appropriate frameworks, paradigms, methodologies, models, and architectures. The industry, ranging from analyst firms, IT consortiums, vendors, governments and the building and construction sector are putting in efforts for its developments and adoption for automated environments [16].

According to OASIS [17], "*SOA is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. It provides a uniform means to offer, discover, interact with and use capabilities to produce desired effects consistent with measurable preconditions and expectations*". It is an approach to organise IT resources and data collectively. It was created in order to enable integration between different technologies and allow for standardised data interaction.

SOA can also be termed as a software architecture that starts with an interface definition and builds the entire application topology as a topology of interfaces, interface implementations, and interface calls [18]. Many businesses are moving towards SOA to effectively eliminate redundancy while simultaneously accelerating the delivery of projects through consolidation and the reuse of services.

SOA implementation depends upon Information and Communication Technologies being able to build interoperable, robust, reusable, and composable services that abstract the application functionality and data of each technology. The key to enabling SOA depends upon abstracting interfaces through different data access technologies. The fundamentals of SOA are based upon service, message, dynamic discovery and web service.

Services are distinct logical units. A combination of services forms a distributed enterprise solution. The business logic is encapsulated within the services. Services must contain a service description which defines the name, how the services can be accessed and the location of the service. Services have to comply with a communication standard so that information can be transferred through an enterprise in an understandable format. This is achieved through the use of messages from the interface designed for the services. Messages are exchanged using XML formats to ensure a platform-neutral system.

The core components of SOA are described in Figure 2. Services can be published to a *directory* service to allow for *discovery* by potential service consumers. The consumer seeks a set of business goals can search through a directory to find a service or services that fulfil these goals. If a service is found, a service consumer can bind to it and execute the processing logic.

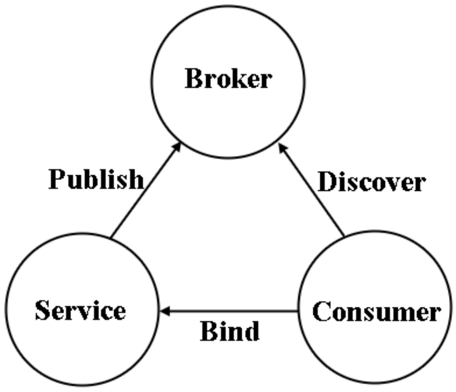

Figure 2: Three Core Components of SOA

Giving an example of the use of SOA, Badawi discusses an SOA project [19] about an Employee Carbon Footprint Dashboard to estimate the environmental impact of an organisational workforce. The implemented architecture is based on an open SOA framework and consists of three layers; the user interface layer, the business logic layer, and the database layer. The Carbon Footprint Dashboard an example of BPM considering one aspect of occupant energy carbon production.

Some of the key design principles of SOA given below are reusability, interoperability, discoverability, and composability.

3.1 Reusability

In a service-oriented paradigm, like SOA, many principles depend upon reusability of several software components. One application is used many a times for different applications to produce different results [20]. For any business logic to be used, it is divided into smaller logical units. The services then to each smaller logic unit can be re-used. In this way, utilisation of SOA-based solutions are enhanced, having a cascading affect on service delivery and execution.

3.2 Interoperability

Interoperability is a property referring to the ability of diverse systems and organisations to collaborate. The common usage of platform-neutral XML messages in SOA, and the support of structural, semantic service descriptions like Web Service Description Language (WSDL) simplify the creation of interoperable solutions that are loosely coupled and with less complexity [20]. At a granular level, services are being used to interact with vendor applications and systems. This provided a method of communication which moves away from proprietary integration solutions to open standards-based solutions.

3.3 Discoverability

Through the principle of discoverability, SOA implementation offers a means of consistently communicating information about different services and resources. This information exchange can be made to other services and users that are have the ability to *discover*. This discoverable information can be stored and maintained in a searchable format. A key aspect of discoverability is that it is not known whether a service exists or not before an attempt to discovery is made.

3.4 Composability

Taking into account the organisation may be using legacy applications, monolithic software units and partially networked solutions. Organisation assets need to be broken down to a number of business processes. The application will be made up of underlying business logic and a front end can be presented in an abstract from to the end users.

4 WEB SERVICES

Web services technology is a common integration solution for different software systems. Web services operate using open standards enabling components written in different programming languages and for different platforms to communicate. Thus, they can significantly reduce the costs of enterprise application integration (EAI) and business-to-business (B2B) communication and offering companies tangible returns on their investments. The use of this technology provides a mechanism for defining the services which different types of applications can create and interact with.

The introduction of Web services has revolutionised the domain of distributed computing. Common Object Request Broker Architecture (CORBA) and Distributed Component Object Model (DCOM) were the main technologies in this domain but migration to a different platform costs, businesses time, money and employee productivity [21]. Industry’s experience with interoperability problems led to the development of open standards for Web service technologies, in an effort to enable cross-platform communication [22].

The main idea of Web services is the combination of Web with component-based systems into a single framework where the user-to-component as well as component-to-component interactions are conducted using standard Web technologies. The eXtensible Markup Language (XML) is used for information exchange and messages are usually delivered using Hypertext Transfer Protocol (HTTP). This provides an interoperable paradigm for all types of platforms and hardware devices from mobile phones and Personal Digital Assistants (PDA) to laptops and back-end servers [6].

XML was designed by the W3 Consortium in 1998 [23]. It was designed for data exchange purposes. The advantages that XML provide are its structured semantics, portability, easily extensible and text format. A platform-independent technology for describing data and creating markup languages. XML’s data portability integrates well with portable applications and services.

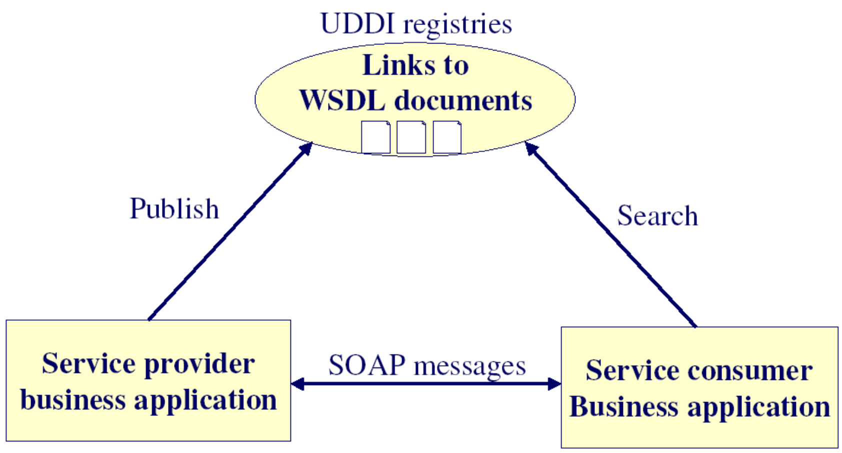

Figure 3: Web Services Architecture

Figure 3 describes the Web services architecture. SOAP, WSDL and Universal Description, Discovery and Integration (UDDI) are the core technologies used in web services. These technologies enable communication among applications in a manner that is independent of specific programming languages, operating systems and hardware platforms. SOAP provides a communication mechanism between web services and other applications across a network. WSDL offers a uniform method of describing web services to other programs. UDDI enables the creation of searchable Web services registries following the model of a phone book to organise Web services.

Web services provide application building blocks based on XML that connect to each other over the Internet. They support mechanisms to describe the services provided by business components and make the service available to other components over the Internet. Interface description are create to allow other components to invoke their services without regard for which platform they execute from or what type of device provides the service.

SOA implementation through web services also provides a reliable, human readable XML-protocol that facilitates interaction with heterogeneous data sources and stores and provides structured and unstructured information that the complete BPM system would be running on.

4.1 Web Services Implementation

Currently all major programming languages are suited for developing distributed systems with Web services. Java as one of them has rich support for XML and provides standard libraries for networking [24]. To keep in line with the emergence of Web services further Java Application Programmer Interfaces (API) have been developed for XML Messaging (JAXM) and XML-based Remote Procedure Calls (JAX-RPC), which was later renamed to Java API for XML Web Services (JAX-WS 2.0). These APIs enable developer to work at a high level of abstract when designing standards compliant Web services. Currently, there are many Web service implementations available to choose from including Apache Axis, Spring and XFire through the Java development environment [4]. Web services technology is widely available on other development environment like .NET from Microsoft to support interoperability across different platforms and operating systems.

5 DEMONSTRATION SCENARIO & IMPLEMENTATION

ITOBO project is an Irish research project focussing on advances in Information and Communication Technologies to develop a holistic, methodological framework for life-cycle oriented information management and decision support in the construction and energy-management sectors [25]. Therefore, it combines the aspects introduced in Section 2 into one BPMS. As an output of this paper, the general SOA based framework architecture is introduced and a single platform will be implemented using SOA in order to prove the benefits for BPM. This research is seen as the first step to reaching the domain-specific goal of developing an anticipating, smart building that operates on an energy-efficient and user-friendly basis while reducing its maintenance costs.

In Figure 4 shows a simplified version of the ITOBO platform. The core components were already introduced in Section 2 and will be incorporated into a single functioning system to optimise building operations. Using an SOA methodology the goal is to create a standard set of interfaces to alleviate traditional interoperability issues. The data interchange between components fulfils the business requirements of the platform to form easily extendable and reusable segments of business logic offering their services to other components within the system.

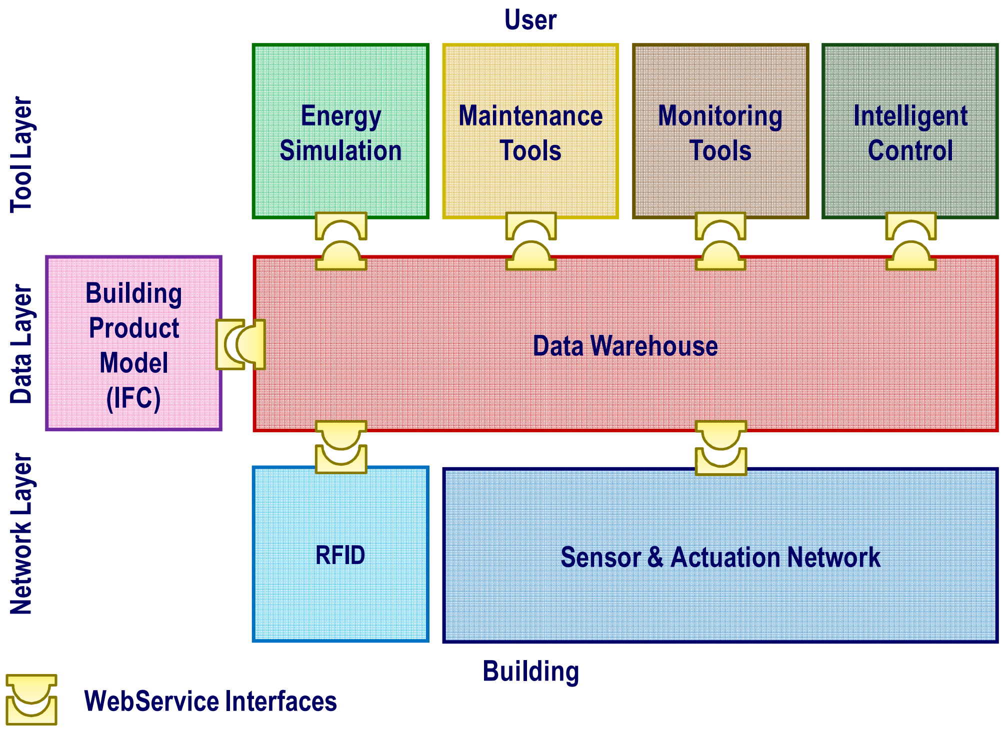

Figure 4: Simplified ITOBO package diagram highlighting the Web service interfaces of the components

The initial steps to creating the SOA design involve gathering data required to support business processes within BPM. For each component in Figure 4 the input data and output data is defined that is required in order to achieve their business goals. Then, Web service interface are defined for each component. The decision, which component offers a service or implements a client depends less on the data flow, but mainly on the control flow, as the Web services work in a server-client relationship, which means that a client calls a service offered by a server. The data flow can then be organised either in a push (a client pushes data to the server) or a poll (a client polls data from the server) way.

The components in the ITOBO SOA are connected using distinct Web service interfaces, resulting in several specific Web service interfaces for interaction with the data warehouse. This separation of business logic simplifies the updating of the Web services interface between two components without conflicting with other components.

The aspects of developing and integrating these Web service interfaces are demonstrated in this paper for a monitoring scenario that collects data from a wireless sensor network [26], stores the data and presents it to various stakeholders. A detailed overview of the stages of development of this scenario is presented in the following sections.

5.1 Scenario

Figure 5 shows the UML use-case diagram for the scenario of BPM. The BPM platform provides multiple user views into the building data. One view is from the perspective of the *owner* who favours information directly related to energy consumption of buildings and the related billing of tenants. This energy information could also be subdivided into more precise readings related to the performance and efficiency of individual elements within each building. This staging of building performance data will assist in diagnosis of operation faults by a *building operator*. Another aspect of providing communication with the building *users* is the collection of tenant requirements. For preparing the data for these different views, the data warehouse operator defines so called *dimensions* for aggregating data.

Once all the needs of the building are catered for, the integrated components will form the basis of a complete BPMS. SOA provides the backbone for intelligent communication of relevant building information to end-user applications by connecting various GUIs to a data warehouse system. A prototypical GUI, monitoring client, was designed to address the needs of the different stakeholders.

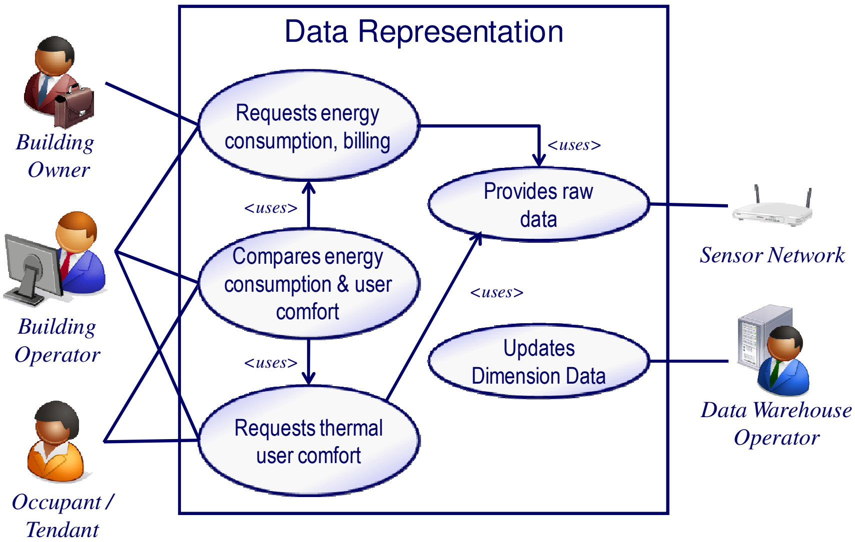

Figure 5: Use-Case diagram of Scenario 1 in the ITOBO platform

Wired and wireless sensor and actuator networks are used as data source. The demonstrator is currently fed from the sensors and actuators installed in the Environmental Research Institute (ERI) building in UCC [27]. The building contains a total of 180 wired and 11 wireless nodes, which shortly will be extended by an additional 90 wireless nodes. The wireless sensor network (WSN) operates using a ZigBee-based protocol and is connected to a Wireless Fidelity (WiFi) backbone via embedded PCs. The system monitors not only common values like temperature, humidity, CO_2 and the energy usage of HVAC equipment, but also many sustainable energy features such as solar panels, geothermal heat pumps and heat recovery systems. Furthermore, the ERI building is used by multiple research groups ranging from biology, chemistry, as well as engineering. The mixed usage with office and laboratory spaces and the modern sustainable energy features define a wide set of requirements for the building operator to optimise energy usage while maintaining steady occupant thermal comfort.

The monitoring client of the BPMS and the network layer exchange information via a data warehouse (DW). It allows the consistent collection of historical data and aggregation of data like the energy usage. This architecture requires two interfaces, one to facilitate data collection from the WSN to the DW and a second to provide relevant context data from the DW to a monitoring client. Figure 76 presents a component based view of the scenario. The wireless sensors collect data and wirelessly communicate the data to an embedded PC on the WiFi backbone. The DW stores and aggregates this data. A monitoring client retrieves the aggregated information to present it to the different stakeholders.

A more detailed description is given of the interfaces that enable interoperability between the components is covered in the following sections.

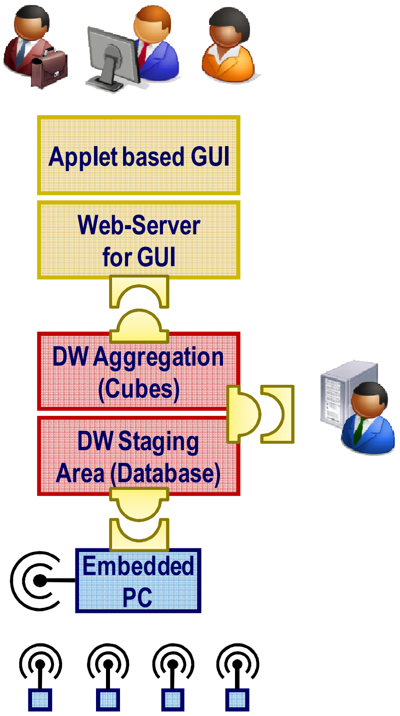

Figure 6: Demonstrator Interfaces from WSN to DW and DW to GUI

5.2 Data Warehouse

Data warehouses are designed to incorporate data from different sources like databases, files or streams. The first stage of a data warehouse is therefore the so called *staging area* where the various input data is collected via an ETL-process (Extract, Transform, and Load) [28] that also conforms the data. From this staging area the data is then aggregated. This staging area is a common relational database in the 11g Oracle Data Warehouse used in the ITOBO project and therefore allows quick data transactions to add, retrieve and alter data.

The structure of the staging area has to be as generic as possible to support the easy collection of data from different wired and wireless network sources as well as a consistent data access from the DW. For example, the demonstrator in the ERI building uses wireless, multisensory nodes that measure the temperature, humidity and light level, specific wireless nodes with single sensor values (like wireless energy meters), and several wired sensors connected via the BMS for all kinds of readings. Each device supports different numbers of sensors, measurement resolutions, message formats, etc. But, if a user wants to see the temperature in the monitoring GUI for his room, then he does not want to name the specific sensor with all these properties. Instead, the data warehouse has to identify the "temperature reading" on any wired or wireless sensor in his room and interpret it correctly.

Figure 7 shows the data structure of the part of the staging area related to the sensor information, which is relevant for the scenario examined in this paper. Each device is represented by a `Node` instance, defining individual node attributes like its IP-address. The `Node` is composed of a `NodeType`, which defines general attributes common to a family of nodes like the manufacturer. A `Node` can have several `DataPoints` for each reading that also are composed of a `DataPointType` that is shared for example by all "temperature readings". To identify the node in the room of a specific occupant, the `Occupant` and `Node` are association to the same `Location`.

These objects can be retrieved, added and changed in the data warehouse's staging area via Web services. However, not all Web services should offer the same information to the users. Like, the network needs only to add sensor readings to the data warehouse, and the stakeholder mainly wants to retrieve this data via the GUI or change a user location. Most of the structural data like the list of sensors may be altered by the data warehouse operator.

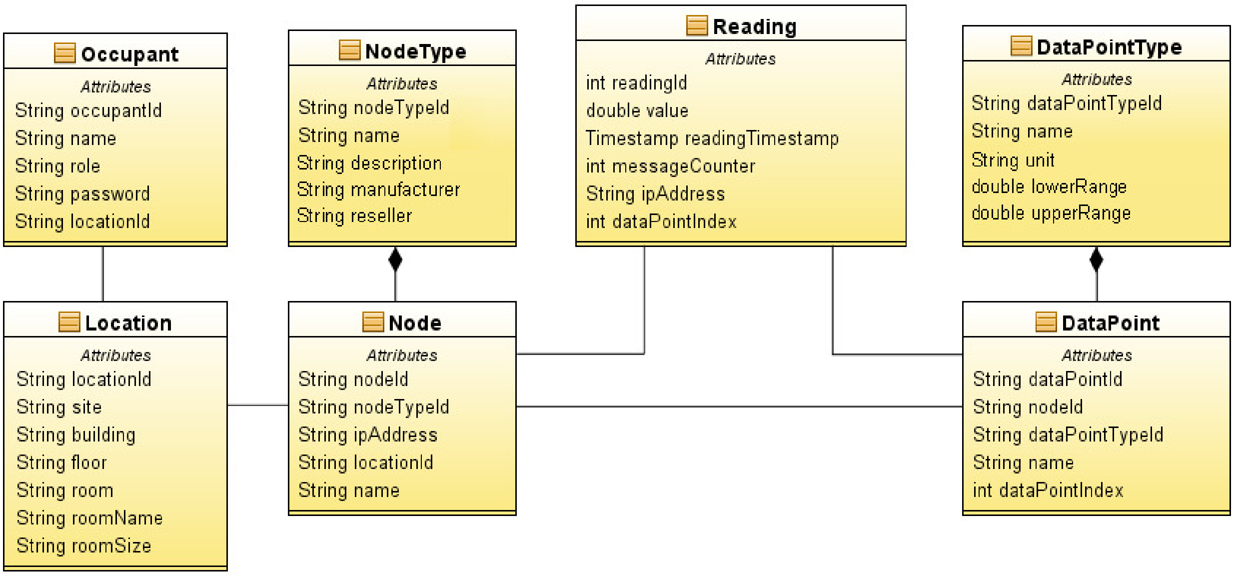

Figure 7: Simplified UML Class Diagram of Data Structures

5.3 Assigning Client and Server for Web Service Interfaces

The issues of optimal communication channels are key to developing an efficient networking solution. As this is the initial step in integrating a BPM platform issues such as possible network bottlenecks and overloading services must be taken into consideration. A DW can collect data from the WSN's embedded pc at certain intervals but this would require the embedded pc to temporarily store readings and sent bulk messages when the DW requires data. The optimal solution is for the DW to provide the service to store readings and the embedded pc to forward messages from the WSN as messages arrive.

From the GUI side, the choice of client and server an easier decision as the process is user driven. As the user interacts with certain features on the GUI and service requests are sent to the DW. The DW responds with the relevant data and the GUI displayed information is updated to inform the user.

If a web service were to go down the clients in this platform could choose an alternative route. This flexibility covers certain aspects of fault tolerance that are necessary in network based solutions. This is a key issue in a mobile scenario where a client device may be intermittently connected to the network.

5.4 WSN to DW Web Service Interface

The sensor nodes sample at a constant interval of 15 minutes for the wired sensors in the BMS and three minutes for the wireless sensors. Once a wireless sensor reaches its duty cycle the data is collected and broadcast as a network packet to the embedded PC. The elements of this packet are the IP address of the node, a message sequence number, the timestamp for the reading, and one or several sampled values like temperature, humidity, illumination or the battery voltage value. A generic method for collection of building readings was created to support such sensor nodes that contain multiple sensing abilities.

Figure 8 demonstrates the general design for the Web service implementation used. A shared interface (DB_WSN) specifies the functions offered by the Web service. Both the client and server side implement this interface. This allows both services to be directly connected without the Web service if needed, like if they run on the same machine and for performance reasons should be connected directly.

DB_WSN_Server implements the service process and connects to the Oracle Data Warehouse via a Java Database Connectivity (JDBC) interface. The server is published by the JAX-WS framework on the web and a WDSL describing the server interface is automatically created. The client service code can be generated from this WDSL.

Once a message is received on the embedded PC with a sensor device set of readings, the embedded PC extracts the message data and sends it to the DW service via the client interface. The handling of the XML messages is completely encapsulated by the JAX-WS and doesn't need to be implemented. A Boolean value is returned from the server to inform the embedded PC if the data submission was successful or some error occurred. The server relates the submitted data to the sensor node via the supplied IP address and adds the new reading for the staging area to the reading table. If a device has multiple data points it may transmit all readings in one message using a separate service.

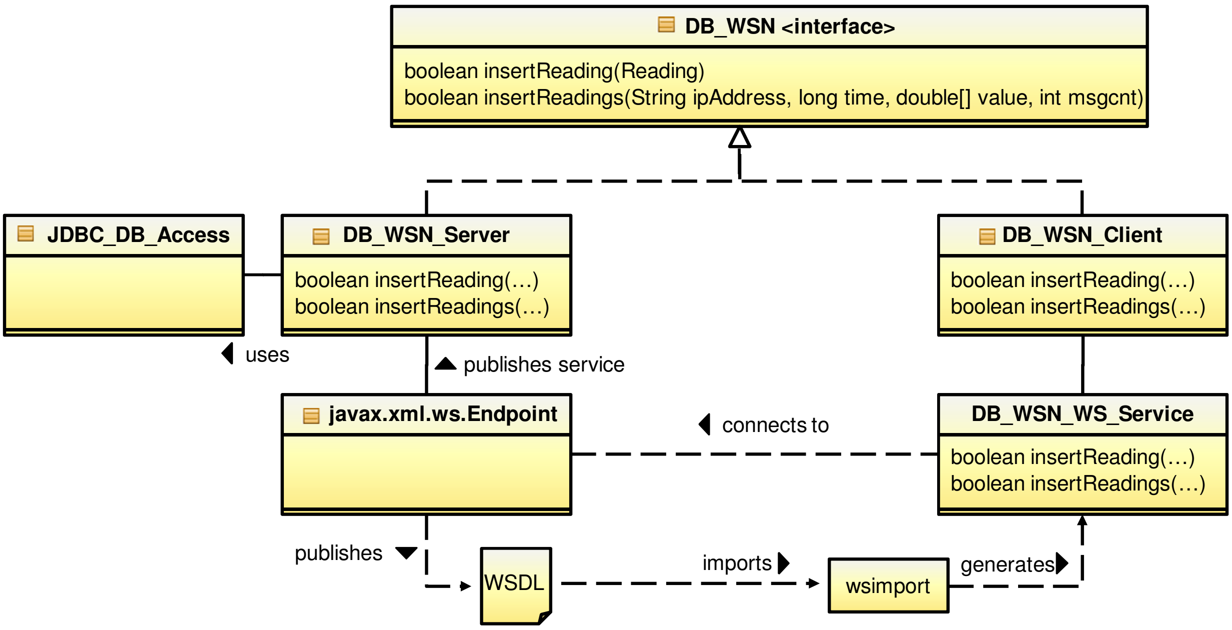

Figure 8: WSN to DW Web Service Interface

5.5 GUI to DW Web Service Interface

The monitoring client was implemented using a Java Applet to enable the user to access building information from a Web browser. The applet web page is run on an Apache web server.

Once data is collected and stored in the DW, now the task is to make this data available to the occupant. As with most software applications a degree of authorisation is required with the user accessing the system. Once the user is granted access they can view relevant data according to their role within the building.

A screenshot of the occupant user view is presented in Figure 9. In order to access this view a successful log in has occurred. The services required are the current state of the occupant room environment which is represented by the current sensed values, if available. Another requirement is the graph which presents the values recorded for the current days and for the selected sensing type, in the screenshot the temperature on the 14th May 2009 is plotted. An occupant thermal comfort level feedback is also available but this implementation is planned for the next stage of development.

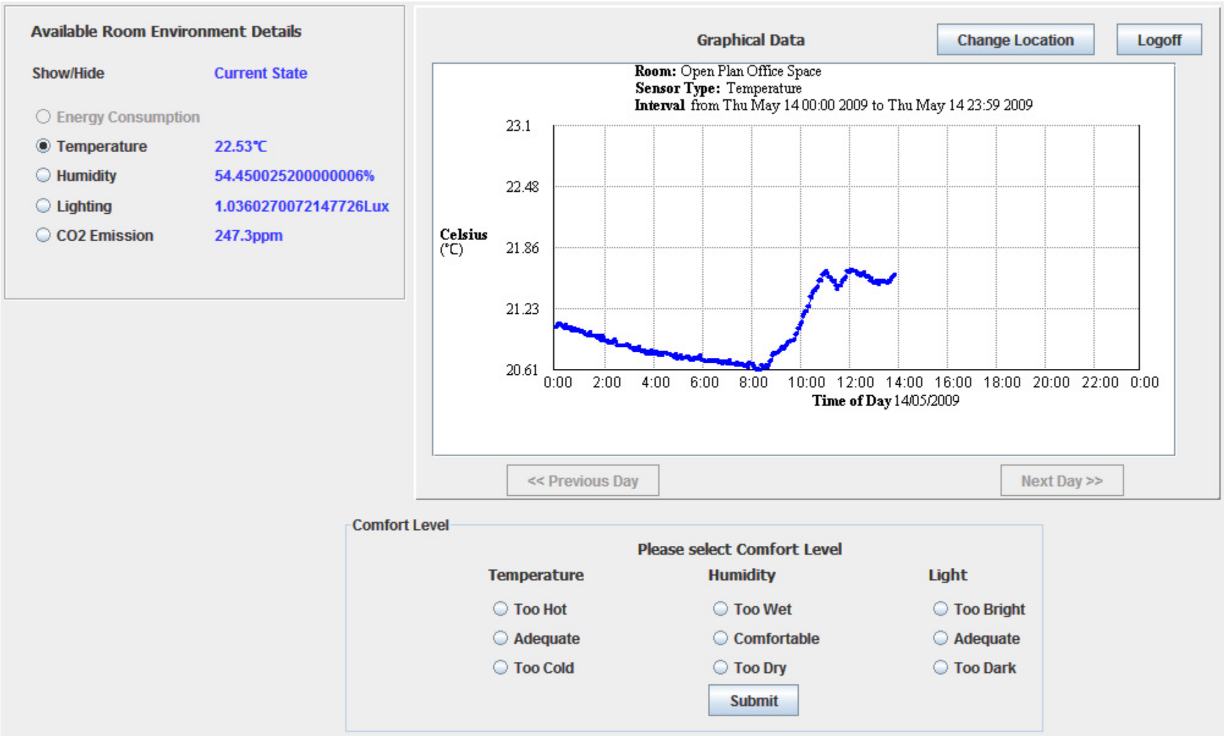

Figure 9: Demonstrator GUI for Occupant

To fulfil these requirements the Web service interface in Figure 10 was formulated. The first method is an authorisation process to check if the user has access to the application. Once access is granted relevant user information is retrieved through the getUser and getUserLocation functions. Finally, sensor data is requested relevant to the current user. Sensor readings for the current day are returned and displayed on the screen. As sensor data is collected every 3 minutes a thread process is implemented in the applet to enable cyclical updating of environmental conditions on the GUI by calling the getUserReadings method repeatedly.

```
public interface GuiDw {
    public abstract boolean validateUserPassword(String username, String plainpassword);
    public abstract User getUser(String username);
    public abstract SensorData getUserReadings (String userId);
    public abstract Location getUserLocation(String usedId);
}
```

Figure 10: GUI to DW Web Service Interface

6 FUTURE WORK

The scenario described is the first step towards creating a flexible software platform for BPM. Initial development has provided an excellent insight to the modular approach that a

SOA framework provides for research in the ITOBO project. The next stage of development is the integration of a mobile device that supports a maintenance view for a building operator. Figure 11 presents the ITOBO components involved to create a platform for sustainable and optimised building operations. This provides an insight into the role of SOA within the ITOBO framework.

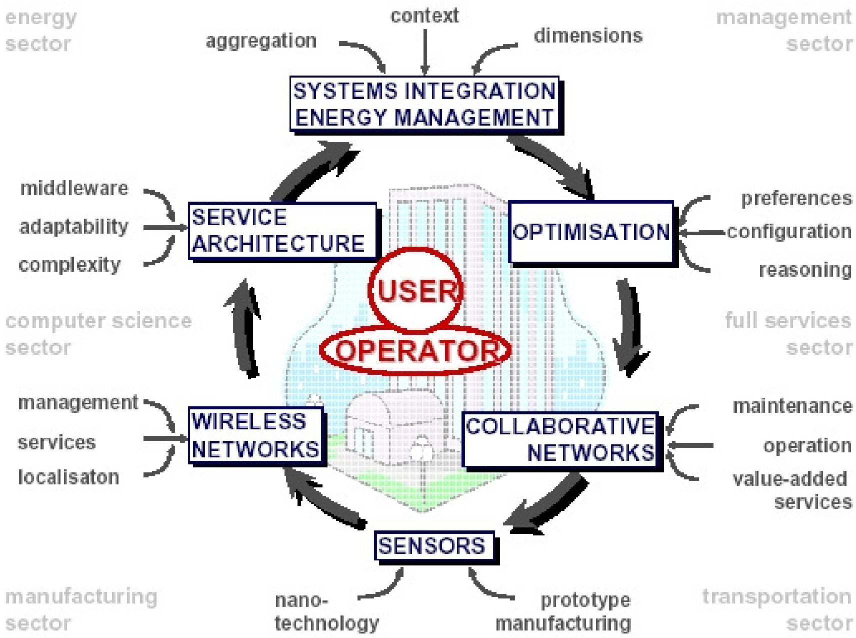

Figure 11: ITOBO Components

Further developments are planned to improve the efficiency, security and reliability aspect of the current implementation. For communication with the actuation elements in the platform the embedded pc will require a service to enable control commands to be interpreted within the wireless network. Similarly, user feedback needs to be collected by the DW and is a critical aspect in the project to enable optimisation of building HVAC systems.

A critical aspect of SOA is the standardisation of data within the problem domains. Ontologies are a methodology of creating a link between the different domain languages and form a common understanding between applications. By leveraging this concept we can organise and share enterprise information, as well as manage content and knowledge, which allows better interoperability and integration. An ontology offers the extension of a framework for maintaining and standardising the data passed between applications and the BPM system.

Furthermore, the concepts of business process modelling and collaborative networks offer huge potential to the development process. The solution can be tailored towards an organisation's goals and any change in operations can be easier to deal with through realigning the services. Existing methodologies such as Business Process Execution Language (BPEL) already offer abstract views of Web services and provide a method to enable collaboration and coordination between these services.

7 CONCLUSION

The ERI Building of UCC is our test bed for implementation of BPM. The application is available for building occupants to review their room conditions. As the ITOBO project involves researchers from multiple institutes, SOA has proved an effective methodology for collaborative work. With different areas of expertise the Web services technology has proved to be a flexible technique to concurrently develop a bridge between domain specific components based on standard information interchange requirements.

For BPM, further efforts will be invested in the integration of IFC and the DW. Our current prototype contains manually entered building information but the import of building structural information, including the placement of wireless sensors and building HVAC system, provides an adaptable approach to changes over the building lifecycle.

With effective monitoring, building performance can be diagnosed to consume excess energy or produce excess CO_2. These relate to extra expenditure and environmental issues which an effective BPM platform can provide awareness to system users. This development is the first step towards developing such a flexible platform to support stakeholder decision making and assist more effective and energy efficient building operations.

8 ACKNOWLEDGEMENT

The ITOBO researchers and the iniatives for the Interfaces Group, Joern Ploennigs, Matteo Burillo, Weiping Song and Rosta Spinar who assisted in the creation of this demonstrator and Emmanual Tumwesigye for his advice.

REFERENCES

[1] B. L. Capehart, W. Turner and W. J. Kennedy, Guide to Energy Management, The Fairmont Press, 2005.

[2] B. L. Capehart, Information Technology for Energy Managers, The Fairmont Press, 2003.

[3] R. J. Hitchcock, High-Performance Commercial Building Systems Program, Element 2 Project 2.1, Task 2.1.2, Standardized Building Performance Metrics, Final Report, Lawrence Berkeley National Laboratory, LBL-53072, 2003.

[4] Bilindas CA, M. Barai and V. Caselli, Service Oriented Architecture with Java, Packt Publishing, 2008.

[5] P.Hersum and O. Sims, Business Component Factory, John Wiley, 2000.

[6] A. Umar, Mobile Computing and Wireless Communications – Applications, Networks, Platforms Architectures, and Security, NGE Solutions, 2004.

[7] R. F. Bruner, M. R. Eaker, R. E. Freeman, R. E. Spekman, E. O. Teisberg and S. Venkataraman, The Portable MBA, John Wiley & Sons,1998.

[8] R. A. Lancioni, M. F. Smith and T. A. Oliva, The Role of the Internet in Supply Chain Management, Industrial Marketing Management, **29**, 45-56, January 2000.

[9] M. McGarr, Transforming Business Processes with EDI, Electronic Commerce World, 2002.

[10] L. Allan, K. Menzel and T. Iqbal, Virtual Enterprises for Integrated Energy Service Provision, 10th IFIP Working Conference on Virtual Enterprises, Thessaloniki, in press 2009.

[11] BuildWise, http://www.zuse.ucc.ie/buildwise, 2007.

[12] M. Keller, J. O'Donnell, K. Menzel, M. Keane and U. Gokce, Integrating the Specification, Acquisition and Processing of Building Performance Information, 12th International Conference on Computing in Civil and Building Engineering, Beijing, 2008.

[13] K. Finkenzeller, RFID Handbook, Fundamentals and Applications in Contactless Smart Cards and Identification, Wiley, 2003.

[14] F. Manzoor, Z. Cong, P. Stack, and K. Menzel, Tracking Occupants and Inventory Items in buildings using RFID Technology, International Conference on the applications of Computer Science and Mathematics in Architecture And Civil Engineering, Weimar, 2009.

[15] R. Raju, An Evaluation of SOA Frameworks, LEADINGedgeforum, Consulting Systems Integration Outsourcing, http://www.csc.com/aboutus/leadingedgeforum/knowledgelibrary/uploads/SOA%20Frameworks%20-%20RRaju.pdf.

[16] F. Jammes and Harm Smit, Service-Oriented Paradigms in Industrial Automation, IEEE Transactions on Industrial Informatics, **1**, February 2005.

[17] OASIS Reference Model for Service Oriented Architecture 1.0 Committee Specification 1, http://www.oasis-open.org/committees/download.php/19679/soa-rm-cs.pdf, August 2006.

[18] R. W. Schulte and Y. V. Natis, "Service Oriented" Architectures, Part 1, SSA Research Note SPA-401-068, Gartner, 1996.

[19] H. Badawi, How to measure employee carbon footprints: Part 1, Sample SOA implementation architecture, http://www.ibm.com/developerworks/websphere/techjournal/0904_badawi/0904_badawi.html, IBM developerWorks, 2009.

[20] T. Erl, SOA – Principles of Service Design, Prentice-Hall Inc., Boston, 2007.

[21] D. Grisofi, Web Services Architect, Part 3 – Is Web Services the Reincarnation of CORBA?, http://www.ibm.com/developerworks/webservices/library/ws-arc3/, IBM developerWorks, July 2001.

[22] H. M. Deitel, P. J. Deitel, J. P. Gadzik, K. Lomelí, S. E. Santry, S. Zhang, Java Web Services for Experienced Programmers, Deitel Developer Series, Prentice Hall, 2003.

[23] Extensible Markup Language (XML), W3C/World Wide Web Consortium.

[24] Java, Sun Developer Network, http://sun.java.com, 2008.

[25] ITOBO, http://www.zuse.ucc.ie/itobo, 2008.

[26] R. Spinar, P. Muthukumaran, R. de Paz, D. Pesch, W. Song, S. A. Chaudhry, C. J. Sreenan, E. Jafer, B. O'Flynn, J. O'Donnell, A. Costa, M. Keane, Demo Abstract: Efficient Building Management with IP-based Wireless Sensor Network, 6th European Conference on Wireless Sensor Networks, Cork, 2009.

[27] Environmental Research Institute (ERI), http://www.ucc.ie/en/ERI, 2002.

[28] A. Ahmed, K. Menzel, J. Ploennigs and B. Cahill, Aspects of multi-dimensional Building Performance Data Management, 16th Annual Workshop of the European Group for Intelligent Computing in Engineering, Berlin, in press 2009.