A Systematic Engineering Tool chain Approach for Self-organizing Building Automation Systems

Alan Mc Gibney(1), Susan Rea (1), Matthias Lehmann(2), Safietou Thior(3),Suzanne Lesecq(3), Martijn Hendriks(4), Claire Gardeux(3), Linh Tuan Mai(2), François Pacull(3), Joern Ploennigs(2), Twan Basten(4), Dirk Pesch(1)

(1) Cork Institute of Technology, (2) Technische Universitaet Dresden,
(3) Commissariat à l'Energie Atomique, (4) TNO-ESI

***Abstract*—There is a strong push towards smart buildings that aim to achieve comfort, safety and energy efficiency, through building automation systems (BAS) that incorporate multiple subsystems such as heating and air-conditioning, lighting, access control etc. The design, commissioning and operation of BAS is already challenging when handling an individual subsystem; however when introducing co-operation between systems the complexity increases dramatically. Balancing the contradictory requirements of comfort, safety and energy efficiency and coping with the dynamics of constantly changing environmental conditions, usage patterns, user needs etc. is a demanding task. This paper outlines an approach to the systematic engineering of cooperating, adaptive building automation systems, which aims to formalize the engineering approach in the form of an integrated tool chain that supports the building stakeholders to produce site-specific robust and reliable building automation.**

Keywords—Systematic engineering; tool chain; co-operation; building automation system

I. INTRODUCTION

People spend the majority of their time in buildings. Creating comfortable and safe living and working environments is therefore rewarding in many ways. Buildings moreover account for a large part of society's energy consumption, 42% in Europe [1], making investments in energy saving technology worthwhile. There is a strong trend towards smart buildings that aim to achieve comfort, safety and energy efficiency, through building automation systems (BAS) that cope with heating and air-conditioning, lighting, access control etc. Balancing the contradictory requirements of comfort, safety and energy efficiency and coping with the dynamics of constantly changing environmental conditions, usage patterns, user needs etc. is challenging. Solutions need to be found in co-operation between and adaptation of BAS in an attempt to achieve common goals.

II. CURRENT OUTLOOK

At present, the building automation systems market is quite fragmented with vendors providing a heterogeneous range of systems, sensors and actuators. System level management usually lacks co-operation as interfaces and semantics for interoperation are not available. This is a result of buildings that are not designed with an integrated BAS but rather utilize different systems for heating, ventilation and air conditioning (HVAC) control, lighting control, fire and security applications, etc. These systems are usually installed by different system integrators and most of the applications at the device and systems layers are vendor proprietary. This heterogeneity leaves little capacity for interoperability and co-operation among the different devices and systems [2].

This lack of co-operation between individual subsystems hampers the increasing demand to operate the whole system in an optimal manner. Sensor data gathered in one subsystem could improve overall system operation and robustness if it were easily accessible by other subsystems. In addition to this, subsystems are individually controlled using often contradictory goals such as operating energy-efficiently, with high service, and safety quality at a low price. This leads to inefficient operation and occupant discomfort. Addressing such complex system goals in an integral and robust manner requires moving from individual, centralized systems towards collaborative systems that cooperate inside and between the device and system level to serve overall system wide operation strategies in an efficient manner.

System co-operation is not a problem that can be simply overcome by the use of middleware approaches and self-organizing techniques. Middleware approaches simplify the transformation of data between technologies. Self-organization addresses a system's ability to adapt its structure autonomously to perform a function [3], which allows it usually to be self-adaptive and more robust to failures [4]. Common plug-and-play concepts follow this idea. But they predefine interoperable design patterns with fixed devices or device profile combinations [5]. This limits the ability of systems to adapt to changing goals of the building stakeholders (facilities managers, building owners, or tenants), and to evolve with time-variant dynamics or to adequately manage often unexpected or unpredictable behavior.

System co-operation has to be viewed as a critical consideration during system engineering. If during this phase components are selected that do not fulfill the requirements of later scenarios or are not interoperable, then the self-organizing capabilities of a system are limited to the poor design choices and it can never reach optimal performance. Nonetheless, selecting interoperable devices is not an easy task due to the high diversity of the many components available on the market. As a result, device selection and system interoperability is still a major, prevalent problem in engineering complex systems and must be addressed for reliable, robust systems.

The authors wish to acknowledge the support of the EU Commission under the FP7 SCUBA project (Grant Agreement No 288079) in part funding the work reported in this paper.

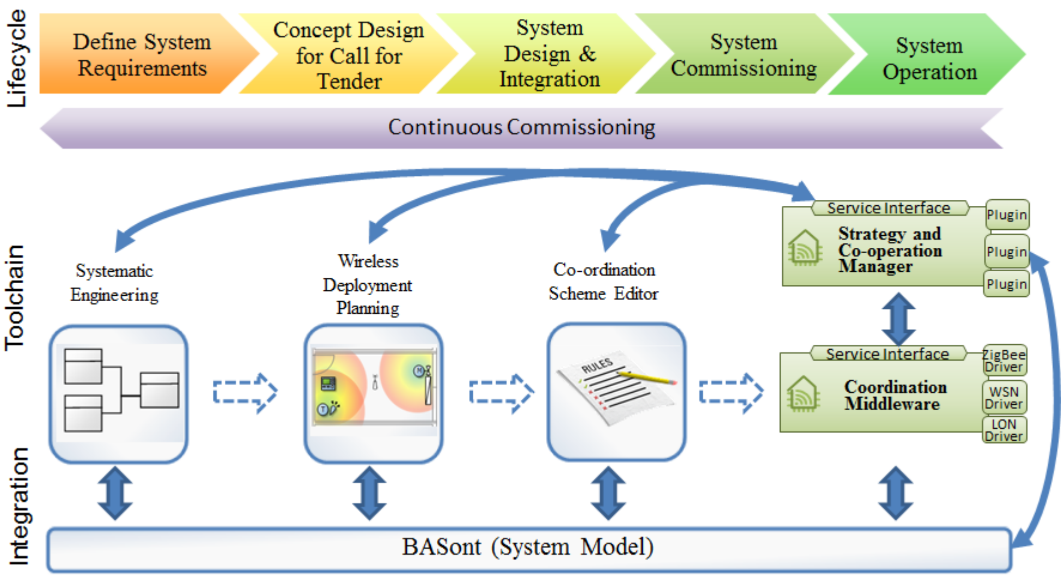

Fig. 1. SCUBA Tool chain and workflow

BAS traditionally rely on wired technology with wireless communication and networking technologies becoming increasingly more prevalent [6][7]. Wireless sensors are flexible and low-cost in comparison to wired solutions [8] especially in facilitating retrofit scenarios. However the installation of a reliable wireless network infrastructure in a building is a labor- and cost-intensive process due to the complex nature of indoor radio signal propagation. Propagation modeling and deployment support tools [9]-[11] can simplify the design process, but often the complexity of the problem leads to the use of simplistic propagation models, basic optimization criteria, and the use of optimization approaches that do not scale well to large scenarios [12][13]. Also beyond the design phase the adaptation of existing deployment optimization tools [14][15] is necessary to link off-line design with on-line network data to support continuous commissioning.

Within the context of the on-going FP7 SCUBA project [16] this paper introduces a systematic engineering approach for cooperating BAS that integrate wireless technology. The approach covers the design, commissioning, and operation phases. The proposed approach includes the development of a methodology to derive system specifications with interoperable components from system requirements, which will create interoperability that is consistent with requirements. Complementing this will be the development of device deployment optimization techniques for wireless sensor network installations in order to address robustness concerns and to support commissioning problems by linking this to the co-ordination and adaptive features of the system. Self-organization at device and system layer will be used to ease commissioning time and complexity of building monitoring and control systems.

The remainder of this paper is structured as follows: Section III discusses the SCUBA tool chain approach for cooperative BAS with Section IV providing details of the individual SCUBA tool chain components. Section V summarizes the approach and presents future directions for the presented research.

III. The SCUBA Systematic Engineering Approach

SCUBA aims to address the challenges associated with the engineering of BAS by creating a novel systematic engineering approach delivered via an integrated design tool chain and an online integration and control framework. The proposed tool chain encapsulates system specification and design, placement optimization, deployment support, a co-ordination middleware and co-operation management modules. The realization of the SCUBA tool chain requires expertise to be drawn from several domains including system design tools, wireless deployment planning, semantics, self-organizing systems, middleware, adaptivity, monitoring, and data storage, machine learning, agent technology and control. Fig. 1 presents the components of the SCUBA tool chain. The workflow can be described in terms of two distinct phases, a design phase and an operation phase. The design phase encapsulates the Systematic Engineering and Wireless Deployment Planning tool and the Co-ordination Scheme Editor. The operation phase constitutes the use of the Co-ordination Middleware and Strategy Manager.

The *systematic engineering tool* facilitates the specification of system requirements for different room usage types and automatically creates various designs for these use cases including selecting and binding the required components. These detailed designs specify which devices shall be installed in each room or building zone. Based on this information the *wireless deployment planning tool* supports the user in defining robust wireless communication links between the devices. The resulting system specification contains all the relevant information for installing the automation system including the requirements, the devices to install and their physical installation positions in each room.

This information is transformed by the *co-ordination scheme editor* into commissioning and self-organization rules that are deployed to the co-ordination middleware. The co-ordination scheme editor provides a simple user-friendly interface to edit these rules and implement supplementary rules that provide additional co-ordination and co-operation functionality on the middleware. The *co-ordination middleware* is responsible for the self-organizing operation of the BAS. For our purposes self-organization is viewed as consisting of two parts, firstly self-commissioning post design, and secondly self-adaptation to operational changes. The presented tool chain is a key enabler for self-commissioning; this is complemented with self-adaption during online operation by the use of a strategy manager. The *strategy manager* provides a monitoring platform in which plug-ins can be deployed to monitor the system in real-time and that have access to historical system data. This data can include measurements of energy consumption, sensor values (temperature, CO2, motion, etc.), external data like weather forecasts, and also statistics data extracted from the wireless network. The strategy manager uses this data to provide a consolidated control strategy to the individual subsystems via the co-ordination middleware. The holistic engineering approach is supported by a data exchange service known as BAS Ontology (BASont) [17]. It is used by the tools during engineering as well as by the middleware during operation. The following section describes the specific components and their relationship to the user workflow by defining the steps a user would take when using the tool chain.

IV. SCUBA Tool chain Components

A. *Systematic Engineering Tool*

The systematic engineering approach in our tool chain develops the system configuration along a three step top-down approach. It starts with the acquisition of the functional requirements. Based on these functional requirements, as a second step, an abstract design is created that already allows the evaluation of the functional concept of the system and simplifies the specification of non-functional requirements. Only in the third and last step a technology-dependent detailed design is created.

<u>Step 1 - Functional Requirement Specification:</u> System requirements can be of functional and non-functional nature. Functional requirements determine the envisioned functionality of the installed system, for example, whether a light control should be realized. Non-functional requirements cover all other aspects such as whether a wireless device should be used or whether the systems should fulfill specific cost, performance or safety requirements. A generic requirement engineering tool has been developed to provide support for the functional requirement specification [18]. It is based on a feature model and a Generative Programming approach [19]. The feature model describes the functions that may be realized by the BAS. These functions have a hierarchy. A lighting system, for example, can be separated into sensors, operators, control functions, and actuators. The provided requirements are the standardized functions defined in the German regulation VDI 3813. Functions can also have dependencies, e.g. that a constant light control function requires a luminance sensor and a dimmable actuator. To simplify the use of the tool, selecting such a function automatically selects the required features and disables the excluded features. With this mechanism the user can not choose requirements that are conflicted to each other.

<u>Step 2 - Abstract Design Creation:</u> An abstract design is generated from the functional requirements. The abstract design is a function-block based representation of the realized functions and the implementation-independent interaction. It conforms to the German regulation VDI 3813. The abstract design tool supports the specification of additional non-functional requirements such as that the specific sensor functionality should be realized wirelessly. Abstract designs are applied as site-specific design templates. A design template groups all requirements for subsystems with similar usage. This can be for example an office room template that is used multiple times along a corridor. To capture the flexibility of adaptive systems, it is possible to define different variants of such templates. The office room template could define not only the normal usage variant but also a fire emergency case that specifies what the system configuration should be in this emergency case. In order to avoid an exponential growth of the variant number, variants should only be created for reasonable, predictable cases.

<u>Step 3 - Detailed Design Optimization:</u> The next step is to create a detailed system design for the templates including selecting and composing devices to realize the functionality defined by the requirements and the abstract design. To improve the efficiency of system design and address the interoperability problem, the system integrator is supported by an optimization approach that selects interoperable devices and composes the final system by connecting the devices' data-points. It implements a systematic engineering process that develops the final design over different user-interactive abstraction levels [20]. The device selection and composition uses an Evolutionary Optimization approach [21]. The multi-objective optimization approach evolves several candidate solutions in parallel and evaluates a set of optimization criteria, which describe the cost, validity and the quality of the design. The output of the systematic engineering tool is a complete design set for a building with all contained devices and the bindings between them. These design templates are stored in the BASont so the other components of the tool chain can re-use them.

B. *Wireless Deployment Planning Tool*

The wireless planning tool focuses on the impact of the quantity, position and topology of a wireless infrastructure on the design while considering site specific constraints and requirements generated by the systematic engineering tool. The global objective of the design tool is to select the optimal configuration (number, position and topology) of sensor, repeater and gateway devices in order to maximize link quality across all nodes while minimizing the overall infrastructure cost. Understanding the site specific requirements is vital to encapsulate application objectives and to form the basis for defining the success of the design output. Due to the often conflicting nature of these objectives, e.g. low cost, long lived, low-delay, robustness, the design of the wireless sensor actuator network (WSAN) needs to be tailored in order to achieve site-specific requirements. The following are the required steps to successfully create a design.

Step 4 – Define Placement Requirements: The first step in wireless design, the fourth step overall, is the specification of an environment model which defines the layout of a building using the basic geometric features, such as walls, doors and windows, this is automated by the importing industry standard file formats such as AutoCad™ or IFC. Every element of the environment model is associated with a material type that can influence Radio Frequency (RF) propagation characteristics within the space. The user can map placement zones that are defined during Step 2 and Step 3 when using the engineering and system specification tool, to physical sensing locations. The placement zones are representative of an area or a point within the floor plan schematic of a building which is deemed the most suitable to maximize the sensing or actuating capabilities of the device. Furthermore, devices can be classified as either a sensor or an actuator.

Step 5 – Define Technology and Communications Properties: The deployment tool extracts the real devices and bindings that have been defined by the systematic engineering tool and stored in the system model. This will provide the critical information on the type of technology that should be used and includes the frequency the device operates at for propagation modeling (e.g. 2.4Ghz, 868Mhz), the power range used by the radio chip, a sensitivity threshold which indicates the bounds of a good link between devices and the battery capacity. These placement and device constraints form the basis of the automated design and are used to gauge the success of a design output.

Step 6 – Automated Design: The tool is a design framework which is underpinned by the use of various models; these include device, propagation, physical and network models. This model repository provides an abstraction from the complexity of design and implementation of wireless sensor and actuator applications, a shorter design time through model reuse, improved adaptability to requirement changes and overall a higher quality system design through the use of formal design approaches.

A key feature of the design tool is the use of an agent-based optimization methodology [15]. Agents are representative of a device trying to find its best position within the environment by maximizing its personal utility governed by the design problem (user-defined constraints). An agent represents an optimization entity and comprises of its internal mechanism dictating which actions should be taken. This is driven by the use of an ordinal utility function, which enables an agent to rank possible actions. Actions include moving a location of a repeater, the addition of another gateway to satisfy communication requirements or the removal of a redundant device from the current topology definition. The definition of the agent utility function is a key driver towards a successful design output and is a weighted function considering the link quality, topology, power consumption and design cost. The outputs of the design (number of devices and locations) are submitted to the system model for system update. By using the agent-based approach in conjunction with the model repository, the tool supports an online refinement of an existing deployment. This is achieved by capturing live statistics data from a network and tuning the prediction models based on the dynamics of the environment it is currently deployed in. The on-line optimization is achieved in co-operation with the strategy manager explained below.

C. Co-ordination Scheme Editor

This tool provides a Graphical User Interface (GUI) that enables the generation and validation of high level co-ordination rules for the co-ordination middleware. A co-ordination rule is a set of actions that are executed when a set of conditions is satisfied. Using the tool a user can:

Step 7 – Create Self-organization Scenarios: The tool can import the different variants from the detailed design templates that were generated in previous steps and define execution scenarios for them. In each scenario, the configuration and binding of sensors and actuators is defined. Scenarios can be triggered by specific conditions or sensor events and can be enabled or disabled during run-time which results in the reconfiguration of the system according to the scenario definition.

Step 8 – Define Soft Sensors: the tool provides different options to define additional control logic. One way is to process sensor information to create a soft sensor which in turn enables the creation of a new type of sensor that provides an abstract value. It can be an aggregation of sensors providing the same type of information (e.g. an average temperature soft sensor is an aggregation of different temperature sensors), or different types of information (e.g. a comfort soft sensor is a aggregation of temperature, humidity and luminosity sensors).

Step 9 - Define Co-ordination Logic: The co-ordination between different monitoring and control systems requires communication between devices from various technologies. This is achievable by using the tool to build complex scenarios and automatically generating co-ordination rules for the middleware. The generated co-ordination rules enable the verification of conditions and execution of actions within the deployed systems. For example if there is a wired presence sensor and a wireless lux level sensor that are not ordinarily connected in the space a scenario is created in the GUI to bind these devices and create a scenario that is invoked when certain conditions are met, e.g. "if presence is detected the lux level is below a threshold then turn on the lights".

Step 10 – Visualize the System State: the tool may retrieve device information (status, values read, location, type of information provided, etc.) and displays it to the user. Once the design phase is complete and the physical infrastructure is deployed the remaining components of the SCUBA tool chain manage and monitor the operation of the system.

D. Co-ordination Middleware

The co-ordination middleware combines three basic paradigms: associative memory, production rules and distributed transactions to provide, at the middleware layer, high level properties [22]. The associative memory provides, to all system components, transparent access to all relevant data. Production rules execute specified actions when given conditions are met. Distributed transactions ensure that the system operates in a robust manner. These properties allow it to play a number of roles within the SCUBA tool chain.

Firstly it can enact the co-ordination rules corresponding to the scenarios defined by the systematic engineering tool and the co-ordination scheme editor. These scenarios are either managed by the middleware when several heterogeneous technologies are used or managed by the subsystem when a single technology is considered. In the latter case, the co-ordination rules are responsible for the configuration of the bindings between devices. Secondly it provides the meta-level control which is responsible for enabling or disabling the defined scenarios according to the current context, ensuring the mutual exclusion of competing scenarios. The switching from one scenario to another one can either be triggered by an external tool (such as the strategy manager discussed below) or directly via a set of rules adapting the system to a given context. Thirdly, the middleware can be used to monitor the health of a system by using the production rules to capture and forward any abnormal phenomena that are observed within the system to the strategy manager, such as a sensor going outside predefined range or not reporting within the expected timeframe. With this information the strategy manager can adapt to current conditions to ensure robustness (e.g. via soft sensors). In the same way a rule that cannot perform the planned actions due to a faulty actuator informs the appropriate tool of the abnormal situation. Finally, the co-ordination middleware can be configured to transmit all relevant data to a historical database for further processing/analysis.

The biggest advantage, from a user's perspective, of using the co-ordination middleware within the SCUBA tool chain is that it minimizes the effort required to commission the deployment as rules and bindings defined in the design phase are automatically deployed by the middleware. It also reduces the burden of re-configuration during operation as the middleware can determine the current operational state and adapt to a dynamic environment within pre-designed bounds. The ability to easily create and apply new design templates during operation provides more flexibility over common plug-and-play approaches where the pairing is programmed to the individual devices.

E. *Strategy Manager*

The Strategy Manager collects and stores data from the building and its context in order to provide a historical view as a basis for strategy management. The approach of [23] is followed in that data warehousing techniques are used to store the building data according to a dimensional data model. Fig. 2 shows the core of the data collection framework. The data flow (concentrating on wireless network health) is as follows: First, a co-ordination middleware instance collects statistics packets from the wireless network it is responsible for. It then uploads these packets to the Strategy Manager via a REST-based protocol. The Strategy Manager buffers these data items in memory or on disk to be resilient to data bursts, which completes the Extraction phase. Uploaded data items are then pushed into the Transformation phase of the data-warehousing architecture. The Transformation phase adds dimensional information to the statistics measurements.

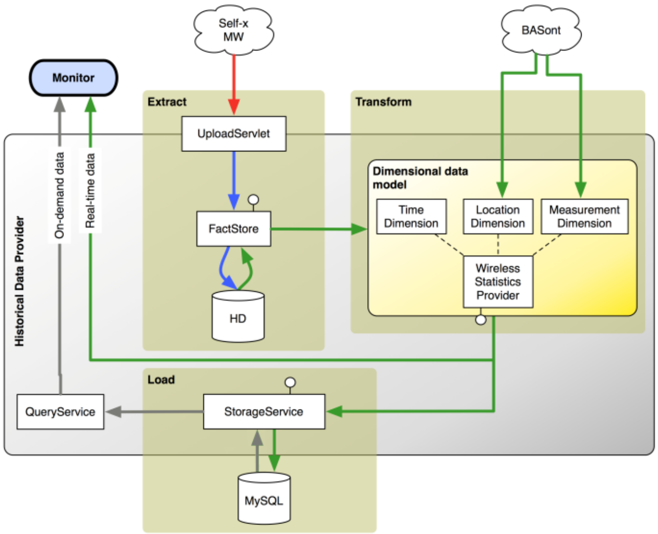

Fig. 2. The data collection core

For instance, the precise location is retrieved from the BASont, and added to the measurement. In general, various dimensions (time, location, measurement) are added to the data items. A main advantage of the deformalized dimensional data model is query efficiency. The results are then loaded to a relational database management system; MySQL in the current prototype. The implementation of the Strategy Manager is based on Java and OSGi and Spring technology. This results in a highly flexible implementation, which allows easy extension of functionality through the use of plug-ins.

The transformation part shown in Fig. 2, for instance, can be deployment specific if dimensional information is to be retrieved from custom back-end systems. An example is an organizational dimension that links measurements to certain parts in the organization. Therefore, the design of the transformation part is completely customizable. The Strategy Manager currently allows two mechanisms for plug-ins to tap into the data streams. First, historical data can be accessed via a query service component. This component currently provides time-series data. Second, an implementation of the Observer pattern allows plug-ins to register to semi real-time data streams. Fig. 2 shows how a custom Monitor plug-in used in both ways.

The strategy manager supports continuous commissioning by closing the loop between the system operation and the design engineering tools in order to adapt or evaluate the impact of a change in the system design or operation. This may result in an automated re-configuration of the system or decision support for the user (e.g. facilities manager) to ensure optimal operation of the BAS.

F. *Tool Integration*

To support tool integration, a common representation of the system model is used, known as *BASont*. It is designed to integrate the results of the various tools in a consistent structure and to provide the information as input for the tools. Therefore, the *BASont* is divided into several parts to allow specific model views and extensions, see Fig. 3.

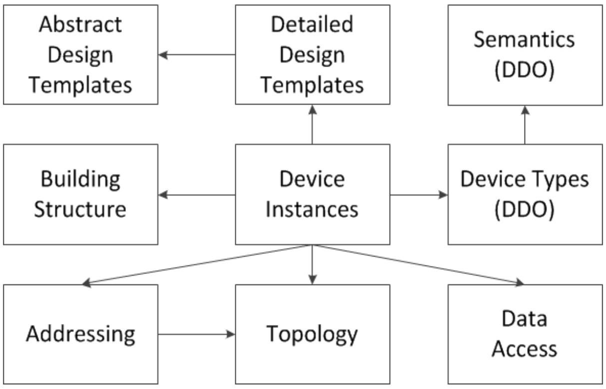

Fig. 3. BASont Structure

The central device instance model represents the currently active system configuration running in a building. It is extended by the abstract and detailed design templates, which are added by the system specification tool. The tool uses semantics and device types from the device description ontology (DDO) [24]. The wireless design tool extends the device instances with addressing, topology and building structure data. The strategy manager stores online data.

V. Conclusion And Future Work

Today the design, deployment, retrofit, commissioning and reconfiguration of building systems is problematic due to the lack of standardization, support tools, and due to a vertical market with many different stakeholders. As a result the majority of buildings are equipped not with one system that integrates all functionality, but with different systems that are provided by different vendors and use different technologies. To achieve co-operation among these systems is challenging. This paper presents an integrated tool chain to support the first phases of the building automation life-cycle. It is possible to describe a building with all integrated building automation devices, optimize their positions and configuration, and create co-ordination rules for the co-ordination middleware. The tool chain goes a step further by facilitating continuous commissioning of deployed systems by linking the physical system back to the design tools via a strategy manager. This enables a holistic approach to the design and management of robust cooperative BAS. It is envisaged that the burden and challenging task of producing an optimal design is reduced by the use of the integrated toolset. Currently the individual tools have been assessed independently and future work will involve the evaluation and validation of the complete systematic engineering approach by applying the tool chain to real design scenarios.

References

[1] L. Itard, F. Meijer, E. Vrins and H. Hoiting, “Building Renovation and Modernisation in Europe: state of the art review”, Delft University of Technology, Jan. 2008.

[2] D. Dietrich, D. Bruckner, G. Zucker and P. Palensky, “Communication and Computation in Buildings: A Short Introduction and Overview”, IEEE Trans. Ind. Electron., vol. 57, no. 11, pp. 3577–3584, Nov. 2010.

[3] C. Gershenson, “Design and Control of Self-organizing Systems”, PhD Dissertation, Vrije Universiteit Brussel, 2007.

[4] F. Heylighen, “The Science of Self-organization and Adaptivity.” In The Encyclopedia of Life Support Systems, L. D. Kiel, (Ed.). EOLSS Publishers, Oxford, 2001.

[5] W. Kastner, G. Neugschwandtner, S. Soucek and H. M. Newman, “Communication Systems for Building Automation and Control”, Proceedings of the IEEE, Vol. 93, Nr. 6, pp. 1178-1203, 2005.

[6] M. Kintner-Meyer, M. Brambley, T. Carlon and N. Bauman, “Wireless Sensors: technology and cost-savings for commercial buildings”, ACEEE Summer Study on Energy Efficiency in Buildings, vol. 7, 2002, pp. 7.121–7.134.

[7] L. Jagemar, D. Olsson and F. Schmidt, “The EPBD and Continuous Commissioning”, Project Report, Building EQ, EIE/06/038/SI2 .448300, Oct. 2007.

[8] EnOcean Alliance, “Wireless Lighting Controls: A Total Cost Analysis”, Whitepaper

[9] W. You-Chiun, H. Chun-Chi, and T. Yu-Chee, “Efficient Deployment Algorithms for Ensuring Coverage and Connectivity of Wireless Sensor Networks”, 1st Int. Conference. on Wireless Internet (WICON), 2005.

[10] J. Chang, P. Hsiu, and T. Kuo, “Search-Oriented Deployment Strategies for Wireless Sensor Networks”. IEEE Symposium. on Object/ Component/Service-Oriented Real-Time Distributed Computing (ISORC)., pp. 164 – 171, 2007.

[11] A. Pinto, M. D'Angelo, C. Fischione, E. Scholte and A. Sangiovanni-Vincentelli, “Synthesis of Embedded Networks for Building Automation and Control, American Control Conf., pp. 920-925, 2008.

[12] S.S. Dhillon and K. Chakrabarty, “Sensor Placement for Effective Coverage and Surveillance in Distributed Sensor Networks”, IEEE Wireless Communications and Networking (WCNC), 2003.

[13] K. Chakrabarty, S. S. Iyengar, H. Qi, and E. Cho, “Grid Coverage for Surveillance and Target Location in Distributed Sensor Networks”, IEEE Transactions on Computing, vol. 51, no. 12, pp. 1448-1453, 2002.

[14] A. Guinard, M. S Aslam, D. Pusceddu, S. Rea, A. McGibney and D. Pesch, “Design and Deployment Tool for in-building Wireless Sensor Networks: A performance discussion”, IEEE 36th Conference on Local Computer Networks (LCN), 2011.

[15] A. Mc Gibney, M. Klepal and D. Pesch, “Agent-Based Optimization for Large Scale WLAN Design”, IEEE Transactions on Evolutionary. Computation, vol. 15, no. 4, pp. 470–486, Aug. 2011.

[16] FP7 SCUBA Project (Grant Agreement No 288079) http://www.ict-scuba.eu, Last accessed 22-07-2013.

[17] J. Ploennigs, B. Hensel, H. Dibowski and K. Kabitzsch, “BASont – A Modular, Adaptive Building Automation System Ontology”, IEEE IECON, pp. 4827–4833, 2012.

[18] J. Ploennigs, H. Dibowski, U. Ryssel, and K. Kabitzsch, “Holistic Design of Wireless Building Automation Systems.” IEEE 16th Conference on Emerging Technologies & Factory Automation (ETFA) (pp. 1-9). IEEE. Sep. 2011

[19] K. Czarnecki and U.W. Eisenecker, “Generative Programming: Methods, Tools and Applications”, Addison-Wesley, 2000.

[20] H. Dibowski, J. Ploennigs and K. Kabitzsch, “Automated Design of Building Automation Systems”, IEEE Trans. Ind. Electron., vol. 57, no. 11, pp. 3606–3613, 2010.

[21] A. Oezluek, J. Ploennigs and K. Kabitzsch, “Designing Building Automation Systems using Evolutionary Algorithms with Semi-Directed Variations”, SMC - IEEE Int. Conf. on Systems Man and Cybernetics, pp. 2328–2335, 2010.

[22] L.-F. Ducreux, C. Guyon-Gardeux, S. Lesecq, F. Pacull, and S. R. Thior, “Resource-based Middleware in the Context of Heterogeneous Building Automation Systems”, IEEE IECON, pp. 4847–4852, 2012.

[23] A.A. Ahmed, J. Ploennigs, K. Menzel and B. Cahill, “Multi-dimensional Building Performance Data Management for Continuous Commissioning”, Advanced Engineering Informatics 24, Elsevier, 2010.

[24] H. Dibowski, and K. Kabitzsch, “Ontology Based Device Descriptions and Triple Store Based Device Repository for Automation Devices”, IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), 2010.