Informatics

Towards an Automatic Configuration Process for Building Automation Systems

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Software Engineering & Internet Computing

eingereicht von

Josef Wechselauer, BSc
Matrikelnummer 01326671

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Wolfgang Kastner

Wien, 4. März 2021

Josef Wechselauer	Wolfgang Kastner

Informatics

Towards an Automatic Configuration Process for Building Automation Systems

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Software Engineering & Internet Computing

by

Josef Wechselauer, BSc
Registration Number 01326671

to the Faculty of Informatics

at the TU Wien

Advisor: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Wolfgang Kastner

Vienna, 4th March, 2021

Josef Wechselauer   Wolfgang Kastner

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Josef Wechselauer, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 4. März 2021

Josef Wechselauer

Danksagung

Ich möchte mich an dieser Stelle bei allen jenen bedanken, die mir den Abschluss dieser Diplomarbeit und meines Studiums ermöglicht haben.
An erster Stelle gilt mein Dank natürlich Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Wolfgang Kastner, der mir nicht nur die Möglichkeit geboten hat, diese Arbeit unter seinen Fittichen zu schreiben, sondern mir auch stets mit seiner Expertise und Unterstützung zur Seite gestanden ist.
Ein großer Dank gilt Lukas Krammer und Daniel Lechner, die maßgeblich dazu beigetragen haben, dass diese Arbeit zustande und zu einem erfolgreichen Ende gekommen ist. Ihre Unterstützung und Motivation waren ausschlaggebend für den Fortschritt der Arbeit, wofür ich sehr dankbar bin.
Ich möchte an dieser Stelle meinen Eltern danken, dass sie mir die Möglichkeit geboten haben, dieses Studium zu belegen und abzuschließen. Meine ganze Familie hat mich stets unterstützt und motiviert, und ohne sie wäre das nicht möglich gewesen. Ich bin wirklich dankbar dafür.
Generell gilt mein Dank allen, die mich während meiner Studienzeit begleitet und unterstützt haben.

Acknowledgements

Hereby, I would like to thank everyone who helped me finish this thesis and my studies.
First and foremost, I would like to thank Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Wolfgang Kastner, who not only gave me the opportunity to write this thesis under his guidance, but also always provided his expertise and support.
A big thanks goes to Lukas Krammer and Daniel Lechner, who were essential for the success of this thesis. I am really grateful for their support and motivation, which was crucial for the progress of this work.
I would like to thank my parents, who provided me with the opportunity to attend and complete this study. My whole family always supported and motivated me, and graduating would not have been possible without their backing. I am very grateful for that.
In general, I would like to thank everyone who accompanied and supported me during my studies.

Kurzfassung

Effiziente und optimierte Gebäudeautomationssysteme (eng. building automation systems, BASs) zielen darauf ab, Bewohnern und Nutzern ein hohes Komfortlevel zu bieten, den Energie- und Ressourcenverbrauch niedrig zu halten und das Management, die Instandhaltung und den Betrieb von Gebäuden zu erleichtern. Durch die Anwendung von gezielten Optimierungsstrategien werden diese Systeme stetig effizienter.
Um jedoch auf die erforderlichen Informationen, Daten und aktuelle Messwerte zugreifen zu können, wird eine initiale Konfiguration des BAS Systems vorausgesetzt. In typischen BAS Projekten müssen sich Experten unter anderem durch vorhandene Gebäudepläne, System Design-Pläne, Gerätelisten und dokumentierte Anforderungen arbeiten, um die nötigen Informationen zur Konfiguration zu sammeln. Aufgrund von oft mäßiger Zusammenarbeit der verschiedenen Expertengruppen in solchen Projekten können die erforderlichen Informationen unter Umständen schwer verfügbar sein. Dadurch wird die Konfiguration von BASs zeitaufwendig, fehleranfällig und repetitiv.
In dieser Arbeit wird ein neuer, generell anwendbarer Prozess zur automatisierten Konfiguration von BAS, im Speziellen von Raumautomationssystemen (eng. room automation and control systems, RACs), vorgestellt. Der Prozess wurde aufbauend auf bekannten Vorgehensweisen zur Konfiguration von BASs entwickelt. Hierbei liegt das Hauptaugenmerk darauf, den Informationsverlust minimal zu halten, während die Anwendbarkeit des Prozesses in realen Projekten gewährleistet wird. Deshalb werden die zur Konfiguration notwendigen Informationen von verschiedenen Quellen, wie zum Beispiel System Design-Dokumenten oder Gerätelisten, gesammelt und in einer digitalen Repräsentationsontologie gespeichert.
Danach wird ein Arbeitsablauf vorgestellt, um diese Informationen mit Wissen über die Funktionalität von RAC Systemen anzureichern und in der digitalen Repräsentation darzustellen. Auf Basis dieser Informationen kann RAC System Software generiert werden, welche zur Konfiguration und dem Betrieb von RAC Systemen verwendet wird. Um die Anwendbarkeit des Prozesses zu zeigen, wird eine beispielhafte Durchführung der einzelnen Schritte zur Verfügung gestellt. Der neue Prozess erlaubt eine hohe Flexibilität in der Designphase von BAS Projekten, während die Wiederverwendung von verfügbaren Informationen und Konfigurationen im Fokus steht. Außerdem wird dadurch die Wahrscheinlichkeit von menschlichem Versagen sowie der technische Aufwand während der Konfiguration von RAC Systemen verringert.

Abstract

The main goals of efficient and optimized building automation systems (BASs) are to provide a high level of comfort for occupants, keeping energy and resource consumption low and easing management, maintenance and operation of buildings. Through the application of specific optimization strategies, such systems are becoming increasingly efficient.

Though, an initial configuration of BASs is required to provide the information, data and live measurements essential for optimization and controlling. In typical BAS projects, experts have to work through existing building plans, system design plans, device lists and documented requirements, among other information sources, in order to configure such systems. The required knowledge is often difficult to obtain, due to modest collaboration between the different fields of expertise in such projects. This makes the configuration of BASs a time-consuming, error-prone and repetitive task.

In this thesis, a new, generally applicable process for automated configuration of BASs, specifically room automation and control (RAC) systems, is presented. This process is based on well-known approaches for configuring BASs, but improves upon them by minimizing information waste while focusing on the applicability in real world projects. Therefore, the information required for the configuration of (RAC) systems is gathered from various sources, such as system design documents or device lists, and stored in a digital representation ontology.

Next, a workflow is introduced, which enriches this information to describe the functionality of RAC systems in the digital representation. Utilizing the gathered knowledge, RAC system software used for configuring and operating RAC systems can be generated. An example application of the introduced process is provided in this thesis, showing the applicability of the approach. The new process allows high flexibility during the design phase of BAS projects, while emphasizing the reuse of available information and configurations, reducing human errors and decreasing the engineering effort for configuring RAC systems.

Contents

CHAPTER 1

Introduction

1.1 Motivation

Installing and operating building automation systems (BASs) provides various benefits for building owners, stakeholders and residents. Such systems help to improve the comfort level of occupants while minimizing the energy consumption by means of automation and optimization. [1, 2] The design, installation, configuration and operation of BASs is typically orchestrated by experts, although sometimes interested building owners take this process in their own hands. [3, 4].

Different approaches and processes for establishing such BASs, especially for the subset of room automation control (RAC) systems, have been developed. They provide guidance in handling multidisciplinary information exchange, e.g. between architects and BAS planners. Such methods also often contain workflows for creating such systems. [5, 6, 7] Unfortunately, a structured approach is not always feasible in practice due to project-specific circumstances such as time or budget constraints. Subsequent deviations from processes can lead to extra effort and expense for project participants. [8, 9]

A generalized process, which takes real world problems and constraints of BAS projects under consideration while minimizing its information waste could be of great benefit to lighten the burden on BAS system designers and integrators. In addition, an automated utilization of already available documents and knowledge would be beneficial for project owners, stakeholders and other participants. Such a generalized process could help to reduce the necessary time and effort for establishing, configuring and operating BASs.

1.2 Problem statement

Modern smart buildings and their building automation systems (BASs) rely on a network of many sensors, actuators and controllers placed on selected locations all over the building. [1] Such intelligent systems are intended to increase the comfort level of occupants while saving money by keeping the energy consumption low. [2]

At some point, this network of interconnected devices in BASs has to be initially configured. Furthermore, within the life cycle of a building, some reconfigurations of such BASs may be necessary, for example when components are exchanged or replaced with newer ones, they are moved to new positions, the physical context of the building changes or components are deactivated. [2]

In order to configure such a system, typically a mechanical, electrical and plumbing (MEP) design engineer provides specifications including equipment, process and location information of the devices controlled by the BAS to the control system integrator. Also a narrative "Sequence of Operations" is delivered by the MEP design engineer, which is an overall specification for the heating, ventilation and air conditioning (HVAC) system strategy for the building. As the level of detail in this specification is varying, the control system integrator often has to determine the control strategy manually. Based on the defined strategy, a configuration database is manually created with the aid of engineering tools. [3]

In this configuration database, additional contextual information of devices and relations between them, for example *Light A is controlled by Light-Switch B*, has to be provided by the programmer. There exist tools which help to ease the configuration steps through a graphical user interface, but the process remains exhausting, error-prone and time consuming. [10, 7]

In order to automate the configuration process, information such as floor plans, unique identifiers of devices or locations of BAS components may be needed in a machine-readable representation. [11] This knowledge should be provided by MEP specifications. [3] Unfortunately, such MEP specifications might be incomplete, inconsistent with the as-built state of the building or not digitalized at all. [12] Therefore, additional information sources may be needed to configure such a system. An obvious source of information is the physical, as-built context of the installed BAS and its field components within the building.

In modern building projects, as-built information is partly available in a common, machine-readable knowledge base provided by building information modeling (BIM). [13] Unfortunately, information necessary for BAS configuration processes, such as knowledge about IT components, associated protocols and logical relationships between components may not be available or cannot be defined in such models. [12, 14] Therefore, BIM models might provide some additional information to MEP specifications for automating the configuration of BASs, but are also not sufficient as a single source of knowledge.

1.3 Aim of the work

The goal of this work is to create a generalized process that leads towards automated configuration of RAC systems. This should be done by identifying information necessary for designing and configuring RAC systems. The gathered knowledge should be subsequently used in the engineering and configuration phase. Therefore, existing processes for designing, implementing and configuring RAC systems are studied and relevant information and programming steps included in these processes are extracted and documented.

Based on this knowledge, sources which provide the necessary information for configuring RAC systems should be identified. Typical information providers in existing system design and configuration processes are studied and suitable sources of information for automatic processing are gathered. Therefore, already available information sources, such as BIM models, should be examined to cover all the required information for configuring RAC systems.

The knowledge collected by analyzing existing design and configuration processes is then used in a new process for configuring RAC systems. This process should be technology and context independent. The application of the proposed process and contained concepts is shown by means of examples. Thereby, previously identified information sources for configuration processes are utilized.

The proof-of-concept should provide insights, how the newly developed process and its included steps are applicable using specific technologies as well as in a general, technology independent context. Furthermore, an evaluation in comparison with existing processes is made and possible decrease of manual configuration effort, additional benefits and future implications caused by the proposed automatic configuration process are highlighted. Shortcomings and possible additional effort necessary to prepare information for automatic processing are also discussed.

1.4 Methodology

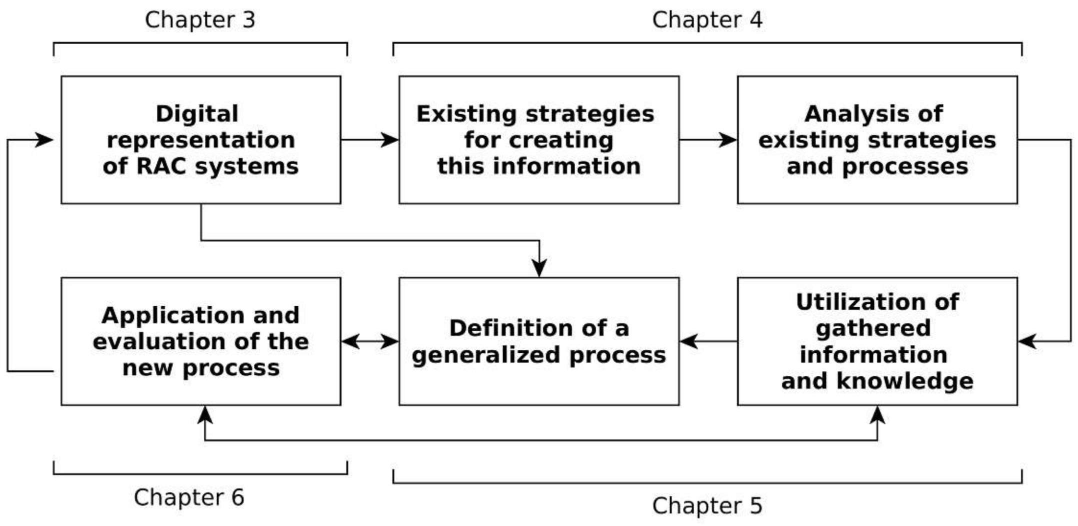
This section comprises all steps of the methodological approach for the thesis. Figure 1.1 provides an overview of the methodology.

At first, the information needed for creating and configuring RAC system software has to be identified. This is done by critically analyzing the recommended forms and structures used in standards and guidelines for RAC system design and software implementation, including the functionality of such systems. A generally applicable, digital representation of RAC systems is developed based on these findings. This step is documented in Chapter 3.

In the second part, common strategies for creating the information identified in step one are discussed. On basis of these findings, problems and deficiencies of these strategies are identified and analyzed as third step. This includes an argumentation for the selection of a common system level design approach, which is built upon in this thesis. In part four of the methodological approach, possible problem solutions for the identified deficiencies of existing RAC system design and development processes are discussed. Furthermore, the information gathered in previous steps is utilized. This includes the search for possible alternative information sources and ways to process and combine the available knowledge to satisfy the information requirements defined in step one of the methodological approach. Steps two and three are covered in Chapter 4, part four of the methodology can be found in Chapter 5.

Based on the solutions gathered in step four, an underlying generalized process for creating and configuring RAC system software is defined in part five. This process focuses on creating digital representations of RAC systems as defined in step one. All required information for creating RAC system software has to be available when following the sequence of tasks included in the new process. This step of the methodological approach can be found in Chapter 5 of this thesis.

The sixth part of the methodology covers the evaluation of the previously defined process. Thereby, the applicability of the generalized process and problem solutions defined in part four are evaluated. Deficiencies and problems regarding these solutions and the target representation created in step one are discussed. A proof-of-concept of the generalized process in order to demonstrate its applicability is provided based on examples. The new process is revised based on these findings and possible future variations and modifications are outlined.



Chapter 3
Chapter 4
Digital representation of RAC systems
Existing strategies for creating this information
Analysis of existing strategies and processes
Application and evaluation of the new process
Definition of a generalized process
Utilization of gathered information and knowledge
Chapter 6
Chapter 5

Figure 1.1: Visualization of the methodological approach

CHAPTER 2

State of the art

This chapter contains the the fundamentals and state of the art concepts this thesis relies. Relevant perspectives of information management and types of information representation are discussed. Furthermore, an introduction on the concept and composition of building automation systems (BAS)s, including standards and guidelines, is provided. The importance of semantic web and ontologies in this field of research is also discussed. In addition, conducted research projects concerned with automated design and configuration of BASs, which are relevant for this thesis, are presented. At last, methodologies for analyzing and optimizing existing processes, especially for BAS projects, are outlined.

2.1 Information management

Effective and efficient information access, processing and utilization are key components for the competitiveness and strategic operation of organizations. In this context, information is viewed as data that has been converted to be meaningful and useful for further processing and utilization. Information management helps people and organizations in the creation, organization, acquirement, distribution and storage of information and profit from arising, subsequent benefits. [15, 16]

2.1.1 Information management perspectives

There exist multiple perspectives regarding information management. The three most important perspectives are summarized in Figure 2.1 and discussed in the following. [15, 16]

Organizational perspective

Information management from an organizational perspective covers the full lifecycle of information processes. Starting from the creation of information until its utilization,

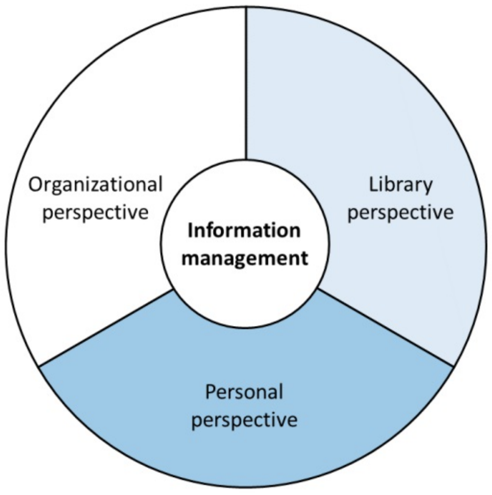


Figure 2.1: Information management perspectives
[15, 16]

it is intended to improve the performance of organizations. It provides a strategic advantage, which leads to benefits such as cost reduction, a limitation of risks and uncertainties, improvement of services and values and benefits through the establishment of new information-based products and services. [15]

Thereby, viewing and managing information as a strategic resource, in a similar fashion as equipment or staff, can help organizations to create a competitive advantage in their markets. [17] Though, information management is not only restricted to managing raw data such as customer names or unit prices. It helps to handle a varying set of information which is enriched with additional, useful context knowledge. The goal is to find patterns and trends in operational data which justify organizational and business decisions. [15]

From an organizational perspective, proper information management is also intended to increase the information processing capacity of organizations. In addition, it should reduce the occurrence of ambiguous information and the general need of information processing within an organization. [15]

Library perspective

In contrast to the organizational perspective, the view respective does not concern itself with the creation or usage of information. The purpose of libraries is to manage information collections, e.g. books and journals. From this perspective, information management controls processes such as information acquisition and organization. Storage, retrieval, access and dissemination of information are also managed. Through means of digitization, the required resources and skills for managing digital libraries and collections have changed over time. [15]

Personal Perspective

Similar to the organizational perspective, the personal perspective deals with the management of information processes through the whole lifecycle of information entities, including its creation and utilization. In contrast to the organizational perspective, the management of information which is relevant to the individual instead of organizations is of primary concern. This includes the management of various types of information, such as personal notes, journals, emails, calendar dates, etc. Personal information management technologies and tools help to simplify the creation, organization and access to such information. This perspective is becoming more and more important due to the increase of personal information that is created, generated and used through the help of personal computing devices. [15]

2.1.2 Types of data

Three forms of data representation, which all serve a different purpose and have advantages and disadvantages, are discussed in the following paragraphs.

Unstructured data

As the name implies, unstructured data represents data that is not bound to an identifiable structure. [18] Though, it is a common data type and its usage widespread, e.g. in form of videos and images. The main characteristic of unstructured data is, that it cannot be stored in a relational database with rows and columns. Therefore, controlled navigation within this kind of data using queries is not possible. Instead, full-text search is used to search within representations of such data itself. This is a flexible way of searching textual files, as it is decoupled from the data. Though, images and videos cannot be searched using this method. [19]

Information within organizations, companies and enterprises is often stored in an unstructured form. Proper management of such information, e.g. documents, PowerPoint presentation content, reports, industry trends etc. can help to gain an advantage and stay ahead in competitive situations. [15]

Fully structured data

In contrast to unstructured data, fully structured data is based upon a predefined schema, which defines the structure and type of data and relations. Schemas are often visualized using entity relationship diagrams and have to be defined before a database (e.g. a relational database) is filled with content. The structured design helps to efficiently process, store and navigate through the data but reduces flexibility and scalability in comparison with unstructured data. Extending already established schemas, which are filled with content, can be difficult and inefficient. [19]

Semi-structured data

In the case of semi-structured data, the definition of a schema is optional. Data instances can exist without a schema, as semi-structured data is often referred to as self-describing because its structure is known implicitly. A schema can be added to already existing instances or may only apply to a part of the data. In addition, data instances can have multiple types defined in a schema. An advantage of semi-structured data is, that it can be created according to a specification, but can also variate from the defined structure. For example, duplicated fields or missing data are allowed. This means, that semi-structured data is irregular, as some instances may be incomplete and others can contain extra information, e.g. by using annotations. In general, schemas of semi-structured data are expected to change often and evolve rapidly. [19, 20]

In semantic web technologies, which are of importance for this thesis, information can be represented in form of resource description framework (RDF) triples. They are created according to RDF schema (RDFS) definitions and are a subtype of semi-structured data. [19]. The semantic web is discussed in more detail in Section 2.3.

In Table 2.2, an overview and comparison of the three different data types discussed in this section is presented. The table is taken from [19], and includes the most important characteristics of each data type. As can be observed from this comparison, each one of the three data-types has its benefits, dependent on the context and requirements. In this thesis, a digital representation ontology is developed in Chapter 3. Therefore, the focus is shifted towards semantic web technologies which utilize semi-structured data such as XML and RDF. [21]

Aspect	**Unstructured data**	**Fully structured data**	**Semi-structured data**
Technology	Character and binary data	Relational database tables	XML/RDF
Transaction management	No transaction management, no concurrency	Matured transaction management, various concurrency techniques	Transaction management adapted from relational database management systems, not matured
Version management	Versioned as a whole	Versioning over tuples, rows, tables, etc.	Not very common, versioning over triples is possible
Flexibility	Very flexible, absence of schema	Schema-dependent, rigorous schema	Flexible, tolerant schema
Scalability	Very scalable	Scaling database schema is difficult	Schema scaling is simple
Robustness	-	Very robust, enhancements over a long period of time (30 years in 2009)	New technology, not widely spread
Query performance	Only textual queries possible	Structured query allows complex joins	Queries over anonymous nodes possible

Table 2.2: Comparison of the different data types. Table is taken from [19].

2.2 Building automation systems

Building automation and control system (BACS) is used as a term in the international organization for standardization (ISO) 16484 standard [22] to comprise a variety of systems. These systems are, among other things, concerned with energy management, optimization, automatic control and monitoring of buildings. This includes, for example, energy management systems (EMS)s, central control management systems (CCMS)s, building management systems (BMS) and building energy management systems (BEMS)s, which are more concerned with the energy management aspect than BMS. [23, 24] The term building automation system (BAS) is commonly used to refer to systems of this field of research. [25] Therefore, BAS will be used in the remainder of this thesis.

2.2.1 Incentive

Various goals can be pursued with the implementation of BASs. According to the 3814 series of guidelines of the *Verein Deutscher Ingeneure* (VDI, ger.: society of German engineers), the most important concerns are operational safety and reliability of supply, comfort level, energy savings and efficiency, monitoring and accessibility. [26] Relevant VDI guidelines are discussed in Section 2.2.3.1.
Reducing energy consumption while keeping a high level of comfort for occupants is of special interest when planning, designing and operating BASs. Without energy management, an increase in thermal, visual and indoor air-quality comfort usually brings along a higher consumption of energy. This can be countered by having automated control strategies for heating, ventilating and air conditioning (HVAC) and lighting systems, among other aspects. [27] In order to monitor, operate and maintain such systems, sensor measurements of the physical environment, e.g. temperature, occupancy levels or CO2 concentration, are required. [28]

2.2.2 Hierarchical levels

BASs can be divided in three hierarchical levels: the field level, the automation level and the management level. [25, 29, 30] They are organized in the form of a pyramid as illustrated in Figure 2.2. Following this hierarchy, lower level entities and elements are controlled and orchestrated by upper level applications and devices. In turn, upper level components rely on data and information which is provided and processed in lower levels. [30]

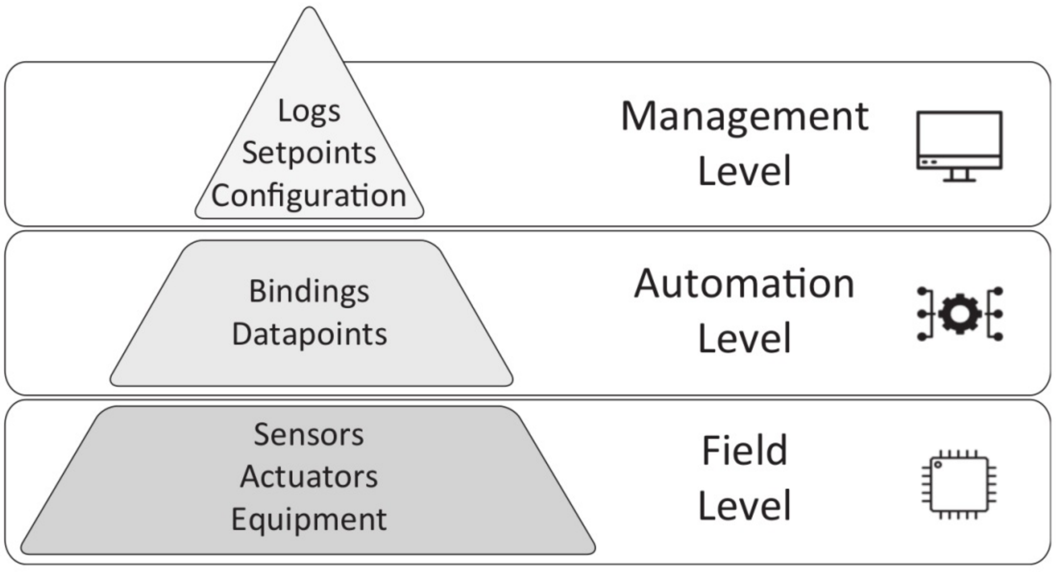


Figure 2.2: Hierarchical levels of BASs; Figure taken from [30]

Field level

The field level represents the lowest level of the BAS hierarchy. Devices at this level are used to interact with the physical environment. This is done by collecting data through measuring, counting or metering and controlling parameters using switching, setting or positioning commands. [25]

At the field level, information about the wiring concept including the connection and connection type of devices is required. Equipment specifications, which contain specific device information characterizing devices, have to be available. Spatial elements including devices and their position inside an element are defined to localize and place equipment in buildings. The influence zone concept is described at the field level to identify, which area is influenced by which control device. Also the type of influence, e.g. monitoring or actuating, is documented. [30]

In accordance with the hierarchical levels of BASs, a distinction between equipment, device and control device is made. Equipment describes electrical and mechanical entities used to interact with a system. Devices are a form of equipment, which are used to interact with the environment or other equipment. Control devices are connected to networks and interchange control commands through addressable, logical units called datapoints. [30]

Automation level

At the automation level, a list of datapoints and bindings is created. Datapoints contain a value and attributes which describe them, including their network address and measuring unit. A binding describes, how the communication between datapoints is established, containing the input/output direction. Furthermore, control devices and datapoints are mapped together at the automation level to describe their relationship. [30] The focus is on automatic control of devices at the field level through the implementation of control loops. Also processing the data gathered and prepared at the lower level of the hierarchy is emphasized. Through horizontal communication, data between elements at the automation level is exchanged for processing. [25]

Management level

At the top of the BAS hierarchy is the management level. Here, data and values prepared at the automation level are processed through vertical access. [25] At the management level, a list of setpoints, their unit of measure and relation with datapoints, is defined. Lists of scenarios with presets and trigger events as well as schedules, which affect datapoints, are also part of the management level. Furthermore, datapoints can be assigned to logical groups and zones, for example if they have a common interface. Lists of events and alarms, their relation with datapoints and their triggers are also part of the management level. [30]

The focus at the management level is on providing user interaction, management and configuration capabilities. Therefore, user interfaces, state monitoring and notification systems including historical data, manual management and configuration applications as well as event logging systems are provided. [30]

2.2.3 Standards and guidelines

In this section, popular standards and guidelines for BAS development, configuration and installation are presented. These documents aim to provide a common baseline for BAS development and can be used as reference works.

2.2.3.1 VDI guidelines

The guidelines provided by the (VDI) are indicatory, convenient documents that provide evaluation criteria in order to make better decisions in technical processes. Common definitions and terminology improve the collaboration between building owners, investors, operators, users, installers, planners, executing companies and product manufacturers which are involved in planning, building and operating buildings. The recommendations included in VDI guidelines are not mandatory and not required by law. [31]

VDI 3813 guideline - BACS and room control
Part 1 of the VDI 3813 guideline contains definitions for functions, terms and fundamentals in room automation systems. A list of room automation functions can be found in part 2 of the VDI 3813 guideline, whereas application examples for room types and function macros of room automation and control are presented in part 3. All definitions provided by the guideline are technology independent, i.e. no specific hardware, software and IT-system implementation is specified. [5, 32, 33]

VDI 3814 guideline - BACS
In 2004, the VDI started with a series of standards on room automation in addition to the already existing VDI 3814 guideline, which by then only included technical standards for BAS. This addendum was initially published as a separate standard with the number VDI 3813. The previously existing parts of the VDI 3814 have been withdrawn and a new, revised version has been published in order to include standards and terminology developed by different expert groups across the world. [26]
The VDI 3814 guideline applies to fields in which BASs control at least a part of the function range of room control and system automation functions, e.g. sunshade and lighting systems or automated facade controls. [26] This guideline also applies to all phases in the life cycle of a building, including the conception, planning, construction and operation and use phases. [26]
Currently, the 3814 standard is again revised with the goal to combine separate planning approaches for system automation in VDI 3814 and room automation in VDI 3813. [31, 26]
Part 1 of the VDI 3814 (VDI 3814-1) standard provides fundamentals of BAS and an introduction to the VDI 3814 series of guidelines. [26]
Part 2.1, 2.2 and 2.3 aim to provide assistance in planning BASs. In detail, part 2.1 deals with requirements planning including an BAS operator concept and user requirements specifications. [34] Part 2.2 guides through the different planning phases in BASs while also outlining system integration steps and interfaces used in BAS planning. [35] VDI

3813 2.3 provides assistance in planning an operating concept and user interfaces for BASs. [36]
Of special interest in the scope of this thesis is part 3.1 of the VDI 3814 guideline, which provides BAS automation function basics and specific BAS descriptions. [37] The VDI is planning to introduce a part 3.2 for the VDI 3814 series in October 2021. This text is planned to include a collection of examples for macro function used in BASs. Manufacturers should be able to provide enhancements in form of a modular system. The suitability for IT integration is specifically considered with the aim to make the function descriptions *"BIM-capable"*. [38]
Part 4.1 and 4.2 of the VDI 3814 series outline methods and tools for planning and building acceptance tests. Specifically, part 4.1 provides a marking and addressing system for BAS and examples for system, consumer, device and quantity survey lists. [39] Part 4.2 outlines checklists for requirements planning, an operator concept and specification sheets. Furthermore, a checklist for the planning of building automation and a system integration table to be used in a methodical approach is included. [40]
The reference text VDI 3814-6 deals with the graphical description of logic control tasks. It assists in creating descriptions of logic control tasks by means of a state graph while also providing specific examples for room control and air-conditioning systems. [41]

2.2.3.2 ISO 16484 standard - BACS

The ISO 16484 standard comprises 6 parts. It provides guidance in the application of BASs in new or already existing buildings. This standard should help to create an environmentally sound design of buildings equipped with BASs by focusing on energy efficiency and conservation while providing an *"acceptable indoor environment"*. It is intended to be used through the whole lifecycle of a BAS, beginning at the design phase. An unambiguous terminology including definitions for building automation and control is defined in all parts of the ISO 16484 standard. The standard provides guidance in a generalized, vendor, hardware and software independent way. [6]
Part 1 of the ISO 16484 standard is concerned with the specification and implementation of BAS projects. A process is defined, which contains a description of all phases of a BAS project. This process is discussed in more detail in Section 4.1.1. In addition, recommendations for documentation and providing training material for BASs are presented. [6]
The second part of the standard defines characteristics and properties of BAS hardware, limited to physical devices and entities. This includes interfaces for operation and human-system interaction, management function devices, general and application specific controllers, automation stations, field devices including their interfaces, connecting and wiring of devices and tools for engineering and commissioning. In addition, a general system model which is applicable for all types of BASs and their connection networks is provided. [22]
In part 3 of the ISO 16484 standard, design and documentation of functional requirements of BASs are discussed. General function block examples are provided to describe the approach taken in this standard. It includes requirements for application software,

functions concerned with project and plant specific applications and engineering functions used for automated operation and control of buildings. Furthermore, functions for communication and integration of other systems are described. Furthermore, a process for defining the functional requirements to operate BASs and creating a complete functional specification is provided. [42]
ISO 16484-5 includes a definition of data communication protocols, services and objects for controlling and monitoring BAS elements such as HVAC systems. It contains lists of objects which are used to exchange binary, analogous and alphanumeric data. Furthermore, an information representation in form of an object-oriented structure is provided to allow communication between equipment in an abstract way. [43]
In part 6, methods for testing and verifying implementations in compliance with the building automation and control networking (BACnet) protocol implementation conformance statement (PICS) are presented. This includes support of BACnet services, object-types, the network layer protocol, data link options and required special functions. [44]

2.2.3.3 Open BAS technology standards

Based on the hierarchical levels introduced in Section 2.2.2, some open standards, which reach across multiple levels of this hierarchy, are available. A subset of these technologies is visualized in Figure 2.3. [29] The most important standards for the purpose of the thesis are outlined in this section. The focus is in how the functionality of BASs is established using these technologies.

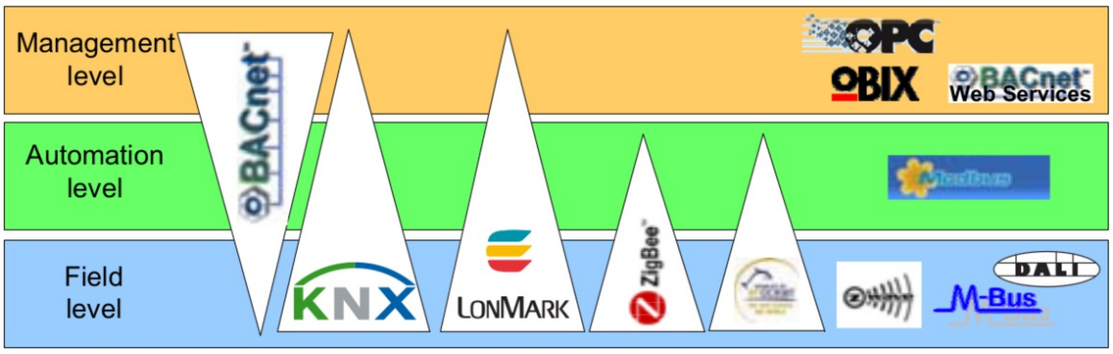


Figure 2.3: BAS technology hierarchy; Figure taken from [29]

Wireless solutions

In recent years, automatically configuring BASs equipped with wireless sensor networks have already been researched extensively. [45, 46, 11, 47] Unfortunately, these approaches are often not applicable in real world scenarios, as wired networks are still more common in BASs than wireless technologies. [48, 49]

BACnet

The purpose of the building automation and control networking (BACnet) protocol is to reach high interoperability between BAS applications and devices, which are often produced by different vendors. Data elements, also called properties, are grouped together to BACnet objects, which are used to describe the functions of BASs and represent datapoints. Properties contain information such as values, minimum and maximum boundaries, resolutions or units. BACnet objects can have different types, for example analog or binary input and output or complex types, such as scheduling or alarming. The *"conformance code"* of an object type describes, if it is read-only, writable, optional or required. A BACnet device has to have the type *"Device"* assigned. Other than that, each device can have zero or more objects of any type associated with it. [25]
Services are following a client/server model and are used for the communication between BACnet devices and BASs. They are grouped by file access, alarm and event, object access, virtual terminal, and remote device management services. [25]
BACnet interoperability building blocks (BIBB)s have been introduced to describe the functions a BACnet device supports. These are divided in data sharing, device and network management, alarm and event management, scheduling and trending blocks. BIBBs are available in client/server pairs to describe, which device is the requester and which one the recipient of a service request. The protocol information and performance statement (PICS) of a BACnet device provides information, which elements of the BACnet protocol it supports. A PICS includes for example the supported objects and their operations, properties and services. [25]

KNX

In the KNX standard [50], function blocks, which can consist of multiple datapoints, are used to describe the functionality of a system. Each datapoint has a datatype and represents a parameter, an input or an output. The datatype includes information about the format, encoding, range and unit of a datapoint. [29]
To program individual devices and parameterize them, individual addresses are assigned to them. These addresses are also used to identify network participants. For actual communication between devices, group addresses are assigned to them during the programming step. [51]

LonWorks

To describe the functionality of LonWorks devices, standard functional profile templates (SFPTs) have been introduced. These SFPTs contain network variables and configuration properties to describe the behavior of devices. Configuration properties are used by applications to access configuration functions of devices. To get information about the process data, e.g. the unit or encoding, applications access shared network variables. Standard configuration property types and standard network variable types, which describe aspects such as format, scaling or encoding, are used to ensure interoperability between LonWorks devices and applications. [29]

ZigBee
In ZigBee, application objects are used to describe the functionality of applications and are spread among devices. Application profiles are used to guarantee interoperability between devices in ZigBee and contain device specifications to describe the functionalities of profiles. These specifications contain mandatory and optional ZigBee clusters. A cluster provides information about attributes and commands to control attributes and devices. The ZigBee Cluster Library Specifiction provides specifications for ZigBee clusters. Application objects then implement these clusters within physical ZigBee devices. [29]

2.3 Semantic web

Semantic web describes the idea of making data in the world wide web accessible, processable and meaningful, not only for humans but also machines. This is done by structuring content, adding information and key terms for especially targeting software agents and interlinking definitions and rules for reasoning about them. A goal of semantic web is to *assist the evolution of human knowledge as a whole.* [21]
In this section, the application of semantic web concepts in the form of ontologies in BASs is discussed.

2.3.1 Ontologies

An ontology is a knowledge base, in which information is collected to define relations between terms. If multiple databases have various identifiers for the same thing, an ontology can be used to discover such overlaps and define a common term. [21] A big reason for using ontologies is the ability to capture and describe functional knowledge of an underlying system in a consistent way. [52]
In an ontology, classes, subclasses, relations between the objects of classes and class properties are defined. This so-called taxonomy of an ontology is used to create a web of information and to describe various relations between different kinds of entities. Entities are identified on basis of universal resource identifiers (URI)s. [21]
In addition, inference rules can be defined to allow automated reasoning of available information to provide additional or more useful information. [21] Different languages for describing ontologies have been developed. The most important are the resource description framework (RDF) and schema (RDFS), and the web ontology language (OWL). [53]
In RDFS, information is represented in the form of triples, which consist of a subject, predicate and object. [19] An example of such a triplet, taken from the digital representation ontology developed as part of this thesis in Chapter 3, is stated in Listing

2.1. This example is stated in turtle, a terse RDF triple language (TTL) syntax, which allows a compact representation of RDF triples. [54] The first and second line provide information about the complete URI of the described resource in form of a comment and is automatically created by the ontology tool which is used for ontology development in this thesis, Protégé. [55] In the next line, *:influencesLocation* represents the subject, *rdf:type* the predicate and *owl:ObjectProperty* the object of the triple. Using a semicolon at the end of the triple allows creating object lists, which are associated with the previous subject, in this case *influencesLocation*. Therefore, the predicates *owl:inverseOf* and *rdfs:range* and objects *:isInfluencedLocationOf* and *:Location* complete additional triples. The statement is terminated with a period ('.'). [54]

```
###  http://www.semanticweb.org/automation-configuration-
   ontology#influencesLocation
:influencesLocation rdf:type owl:ObjectProperty ;
                    owl:inverseOf :isInfluencedLocationOf ;
                    rdfs:range :Location .
```

Listing 2.1: RDF triple example in TTL syntax

2.3.2 Ontology development

When developing an ontology, it is recommended to follow a methodology consisting of the following steps [56]:

- Ontology capture
  In this step, key concepts and relationships in the domain an ontology refers to, have to be identified. These concepts and relations are then described by using distinct text definitions. Terms are identified to reference such definitions of concepts and relations. [56] During ontology capture, additional terms can be defined to refer to the same concepts. [57].
- Ontology coding
  The concepts and relationships captured in the previous step are represented using a formal language by creating code. [56, 57] It is possible to combine the capture and coding stages into one step, for example by using ontology editors. [56]
- Integrating existing ontologies
  Identifying and integrating ontologies which already provide concepts that can be reused by a newly developed ontology should be considered during both ontology capture and coding. [56, 57]

The development of an ontology according to this methodology is not trivial and can be time-consuming. [57] In Section 3.2, the approach taken for developing an ontology to represent a RAC system is described in detail.
Ontologies, that have been developed and published specifically for the use in internet of things (IoT) and BAS research fields, are presented and discussed in Table 2.4. [24]

Ontology	**Purpose and Concept**
	BAS ontologies
SAREF [58]	Fundamental concepts of smart devices; Three main sections: • Devices and functions (e.g. turn on/off) which are related with each other; each function is related with a command (e.g. receive instrument readings) • Energy production/consumption • Building
SAREF4EE [59]	Expands SAREF for compatibility with EEBus and Energy@Home
BRICK [60]	Ontology based on SAREF with a simplistic subject-area model, which is *sufficient for most applications in office and university buildings"*. It covers: • HVAC • Lighting • Spatial and energy infrastructure • Relations between objects
BOnSAI [61]	Ontology focused on services for smart home and web
DogOnt [62]	Ontology geared towards smart home systems and energy consumption device modeling; distinguishes between controllable (with states and functionality) and non-controllable devices
ThinkHome [63]	Ontology focused on smart homes and energy supply and consumption concepts for resources, building structure, actors, energy, comfort and exterior influences [64]
	Ontologies for controlling and monitoring
SSN [65]	Ontology for modeling sensor- and observation-based data
M3 [66]	Extends SSN with the capability of modeling sensor, observation and measurement unit descriptions
OntoSensor [67]	Extends the SensorML standard with categories of sensors (including metadata specifications, performance and reliability), behaviors, relations and functions
	IoT ontologies
IoT-Lite [68]	A lightweight SSN implementation for modeling sensor placement and configuration data
Open-IoT [69]	An ontology based on SSN which extends it with "IoT-crucial" concepts

IoT-O [69]	Reuses and combines concepts from SSN, SAN, DUL, oneM2M and other ontologies
FIESTA-IoT [70]	Reuses and combines concepts from SSN, IoT-lite, m3-lite, Time, DUL and WGS84 and other existing IoT ontologies.

Table 2.4: Overview of existing, relevant ontologies in BAS and IoT [24]

2.4 Automated design and configuration approaches for BASs

In this section, relevant approaches and projects with the goal of automating the design and configuration of BASs are discussed.
Most research projects in the field of automating engineering tasks have been focused on the design phase of products. Especially non-creative, repetitive tasks are targeted in order to save time and cost. [71] In the following, important configuration and design automation (DA) approaches in BAS projects are outlined. In Sections 4.1.2 and 4.2.2, specific methods and potential disadvantages are discussed in more detail.

AUTEG, AUDRAGA and AUTERAS
The project AUTEG was conducted between the Technical University of Dresden and the Helmut-Schmidt-University of Hamburg. The goal of this project was to create an automated approach for designing, installing and operating BASs to reduce the cost and energy consumption of such systems through optimized functional behavior. The approach includes guidance in requirements engineering, abstract design and a platform-specific design of a BAS. It includes a semi-automated selection of devices and their configuration and operation based on the gathered requirements. [7, 9]
AUDRAGA was a project which was conducted in succession of AUTEG. The goal of AUDRAGA was to define new methods for the automated integration and installation sensor systems in BASs. The same holistic approach as in AUTEG was taken to tackle challenges, such as interoperability problems between devices of different vendors or signal distribution in wireless systems. [8, 9]
AUTERAS [72] is a software tool which was developed on basis of the AUTEG and AUDRAGA projects. By selecting desired room automation functions, the tool provides devices which support those functions by accessing vendor specific database systems. To take advantages of this tool, the user should have professional expertise in BAS engineering. [9].

SCUBA and TOPAs
SCUBA concerns itself with the development of semantic models and new engineering methods for self-organizing, cooperative and robust building automation (SCUBA). Interoperability problems and distributed, self-configuring and complex systems which remove identified failures themselves, have been the main points of this project. A workflow

with an integrated tool chain which is applicable in the requirements engineering, design, commissioning and operational phase of BASs was developed during the course of the SCUBA project. [73, 9]
Tools for continuous building performance auditing (TOPAs) is a project conducted in succession of SCUBA. Its goal is to develop a method for continuous monitoring of the energy consumption and distribution in buildings in a holistic approach. It should help to improve the gap between anticipated and actual energy consumption of buildings. [9]

2.5 Methodologies for BAS process analysis

Through the introduction of DA and knowledge based engineering (KBE), important steps towards the automation of engineering related tasks in BAS projects have been made. Such approaches are essential for reducing the design effort, though failures as a consequence of automation are rarely documented. [71]
Methodology and tools oriented to knowledge-based applications (MOKA) have been developed in order to help during the development lifecycle of KBE applications, including DA approaches for BASs. MOKA includes a description of a KBE application lifecycle, which is illustrated in Figure 2.4. [71]
When applying this methodology, technical risks and the feasibility of the envisioned KBE system are studied. This is done based on factors such as available expert knowledge or product knowledge. Furthermore, the industrial need for the planned application is assessed. [74, 71]
A part of this methodology is the estimation of cost and time of the project before its inception. Possible project and financial risks on top of technical risks are assessed and criteria for verifying the success of the envisioned project are defined. [74, 71]
An adaptable methodology for automation application development (AMAAD) is available, which also provides criteria for the identification and justification of development processes. Thereby, automation techniques of similar tasks used in other processes are examined to assess the feasibility of automation in a process. This is based on resource, cost and time estimations, technical feasibility and the assessment of possible risks. [75]

Based on these existing methodologies, an integrated method for process value evaluation (IMPROVE) was developed. The core of this method is to identify, if a process is suitable for automation and estimating the potential of automating a task or a complete process. This method divides the analysis of a process into two levels: the macro-level analysis and the micro-level analysis. An overview of the IMPROVE method is visualized in Figure 2.5. [71]
During the macro-level analysis phase, key aspects and the most important functions and tasks of the process are identified. Furthermore, the distribution of the workload between process functions and the information flow are analyzed. Tasks, which take a long time or show signs of information waste, are marked to be studied in the micro-level analysis phase. Such tasks have a higher improvement potential and should therefore be optimized first. [71]

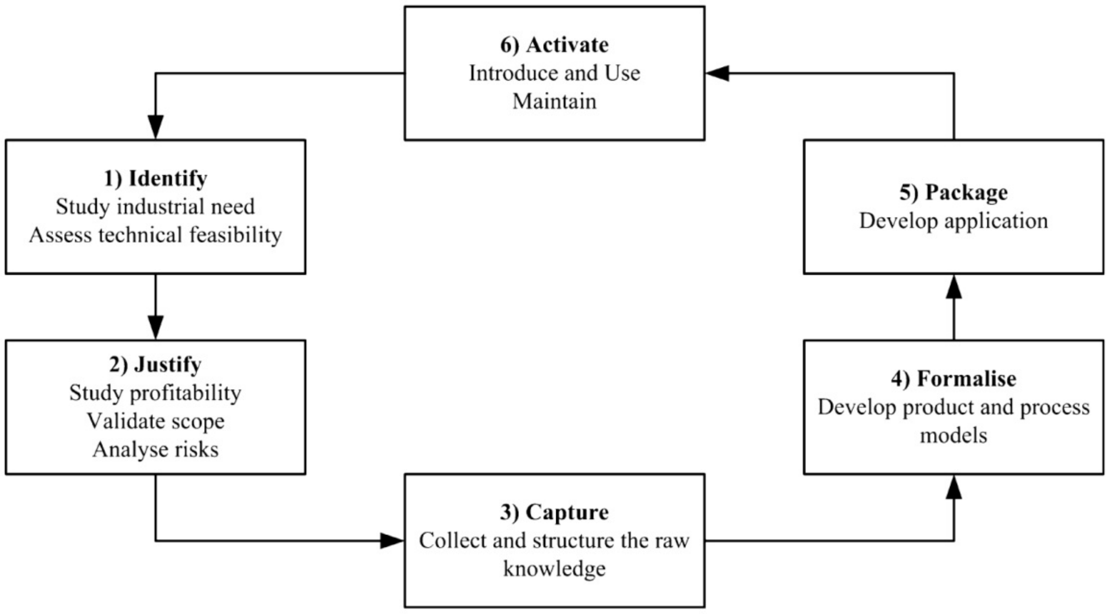


Figure 2.4: KBE-lifecycle; Figure taken from [71]

The micro-level analysis phase is used to outline the detailed steps of tasks that have been identified in the macro-level phase. This includes a detailed specification of the input and output information of each step. During the analysis, the information waste of tasks within processes is quantified between the 'as-is' and 'to-be' state, e.g. through time measurements. Based on *"performance objectives"*, a business case can be developed which takes potential improvements of processes, e.g. through the automation of tasks, and the cost of changes and implementations under consideration. [71]

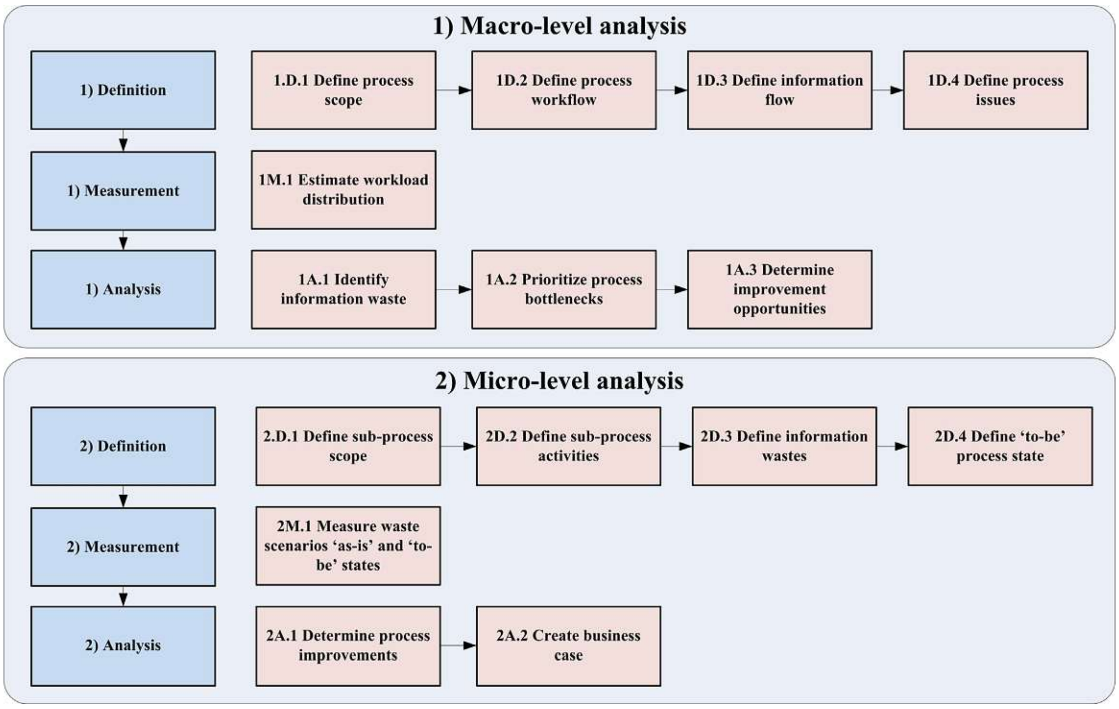

1) Macro-level analysis
1) Definition
1.D.1 Define process scope
1D.2 Define process workflow
1D.3 Define information flow
1D.4 Define process issues
1) Measurement
1M.1 Estimate workload distribution
1) Analysis
1A.1 Identify information waste
1A.2 Prioritize process bottlenecks
1A.3 Determine improvement opportunities
2) Micro-level analysis
2) Definition
2.D.1 Define sub-process scope
2D.2 Define sub-process activities
2D.3 Define information wastes
2D.4 Define 'to-be' process state
2) Measurement
2M.1 Measure waste scenarios 'as-is' and 'to-be' states
2) Analysis
2A.1 Determine process improvements
2A.2 Create business case

Figure 2.5: IMPROVE-method; Figure taken from [71]

CHAPTER 3

Ingredients for the digital representation of RAC systems

In this chapter, information required for creating RAC software and findings regarding this subject are presented in Section 3.1. Furthermore, the selected form and structure of digital representation are discussed in Section 3.2. Section 3.3 presents the developed knowledge base, which is used as target representation for creating and configuring BAS software in subsequent steps. This chapter comprises the first part of the methodological approach as visualized in Figure 3.1

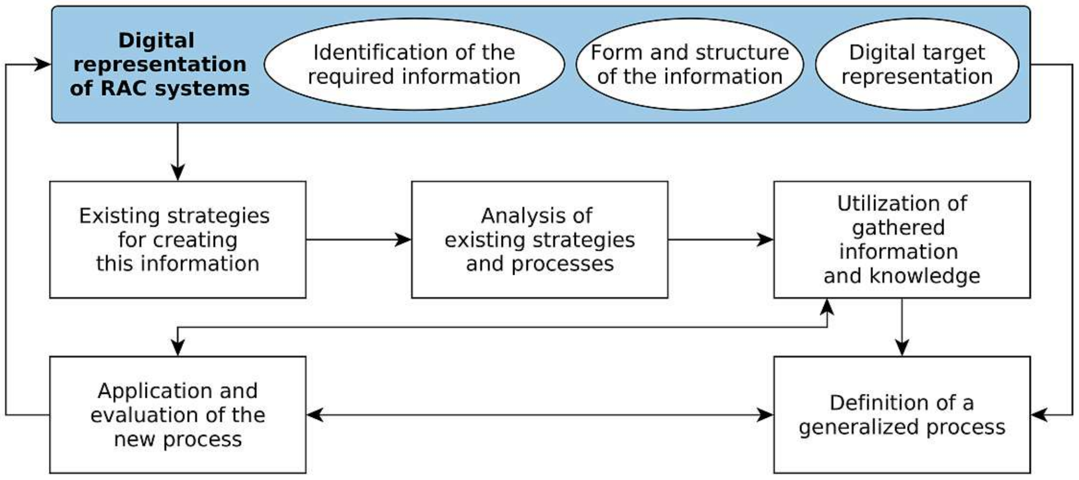

Digital representation of RAC systems
Identification of the required information
Form and structure of the information
Digital target representation
Existing strategies for creating this information
Analysis of existing strategies and processes
Utilization of gathered information and knowledge
Application and evaluation of the new process
Definition of a generalized process

Figure 3.1: Part 1 of the methodological approach

3.1 Information required for creating BAS software

According to part 1 of the revised VDI 3814 guideline, published in January 2019, BASs are composed of *System Automation and Controls (SAC)*, *Room Automation and Controls (RAC)* and *Management Building Automation and Control Systems (M-BACS)*. Figure 3.2 provides an overview of this concept.

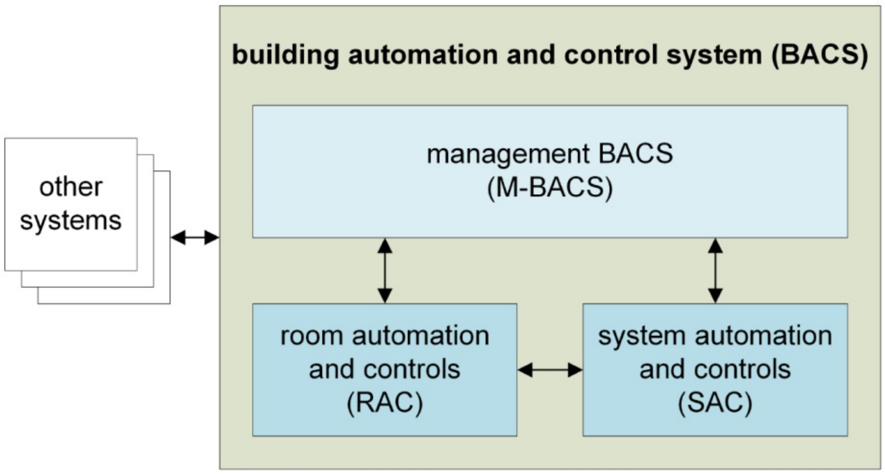

building automation and control system (BACS)
management BACS (M-BACS)
other systems
room automation and controls (RAC)
system automation and controls (SAC)

Figure 3.2: Building blocks of a BAS; Figure taken from part 1 of the VDI 3814 guideline [26]

The SAC component is, among other things, responsible for air conditioning, central ventilation, heat generation, cooling generation and safety systems. It describes the automation of systems in a building and represents the functionality and components of mechanical equipment rooms in buildings. The RAC building block contains all automation and control systems of rooms in a building, including local control and display components. The M-BACS part represents technical, commercial and infrastructural building management and is especially important in the operation phase of a building. [26]
To provide a complete set of BAS functions, these building blocks have to communicate and adjust to the demands of each other. For example, the SAC part has to provide exactly the amount of energy that is required by the RAC system controlled rooms it is responsible for. [26] These three components have to be covered in order to describe overall BASs.

3.1.1 Structural function description - guidelines and standards

The VDI and the ISO have published various guidelines and standards which provide assistance in designing, engineering and operating BASs. An overview of the content and current state of these documents can be found in Section 2.2.3. These standards and guidelines use function blocks to describe the functionality of individual parts of BASs.

Function blocks can be interlinked with each other and connected with data points to create function schematics. [37, 32, 6]
When designing RAC systems, these schematics can be used in combination with function lists and optional supplementary function descriptions, control flow charts and state graphs to describe the functionality of a RAC system in detail. These requirements are illustrated in Figure 3.3. The according function blocks, which are technology and vendor independent, and the approach for creating such schematics and lists for RAC systems are outlined in part 2 of the VDI 3813 guideline, which was published in March 2011. [32]

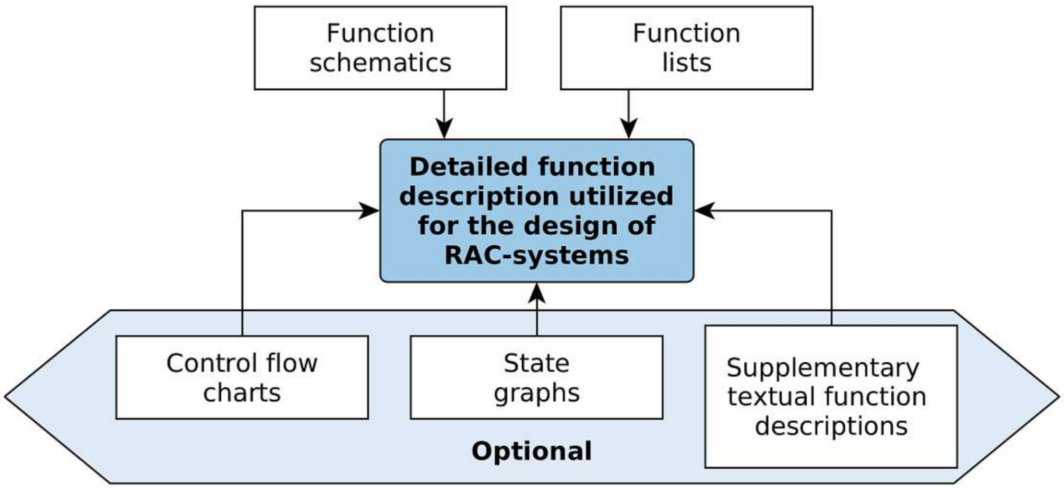

Function schematics
Function lists
Detailed function description utilized for the design of RAC-systems
Control flow charts
State graphs
Supplementary textual function descriptions
Optional

Figure 3.3: Function description overview of a RAC system [32]

The VDI intends to provide guidelines for schematics and function lists, which are not only limited to RAC systems but help to describe the functionality of complete BASs, comprising RAC, SAC and the M-BACSs. Part 4.3 of the VDI 3814 guideline, which targets the structure of BASs and includes function lists, has been published in November 2020, which is unfortunately not in time to incorporate it in this thesis. [76]
Part 3 of the ISO 16484 standard, published in December 2005, also provides function blocks for describing the functionality of BASs. It is based on prior published VDI guidelines and therefore provides a similar approach and strategy for creating automation schematics and function lists. Although some parts of the functionality of RAC systems are covered by this standard, it primarily concerns itself with SAC systems. Therefore, the provided examples for creating function schematics and lists in this standard are also focused on describing the functionality of SAC systems. [42]
The reason that part 4.3 of the VDI 3814 guideline is planned to be published is, that portions of the ISO 16484 standard are outdated and not applicable for all regional requirements of BASs. It is intended to provide a new reference, including state of the art technical knowledge for future revisions of the ISO 16484 standard. [26, 76]

In contrast to the ISO 16484 standard, part 2 of the VDI 3813 guideline aims at providing function descriptions in form of function lists and schematics for RAC systems. It also contains extensive examples for such use cases. The VDI recommends integrating RAC systems, designed with the aid of the VDI 3813 guideline, in overall BAS solutions as published in reference works ISO 16484 and VDI 3814. [32]
In this thesis, the concepts and recommendations of the ISO 16484 standard, the VDI 3813 guideline and the already available parts of the VDI 3814 guideline are taken under consideration. Thus, the focus is on providing a generalized process for creating RAC system software. Therefore, the VDI 3813 guideline is primarily used as reference work. An overview of these documents can be found in Figure 3.4.

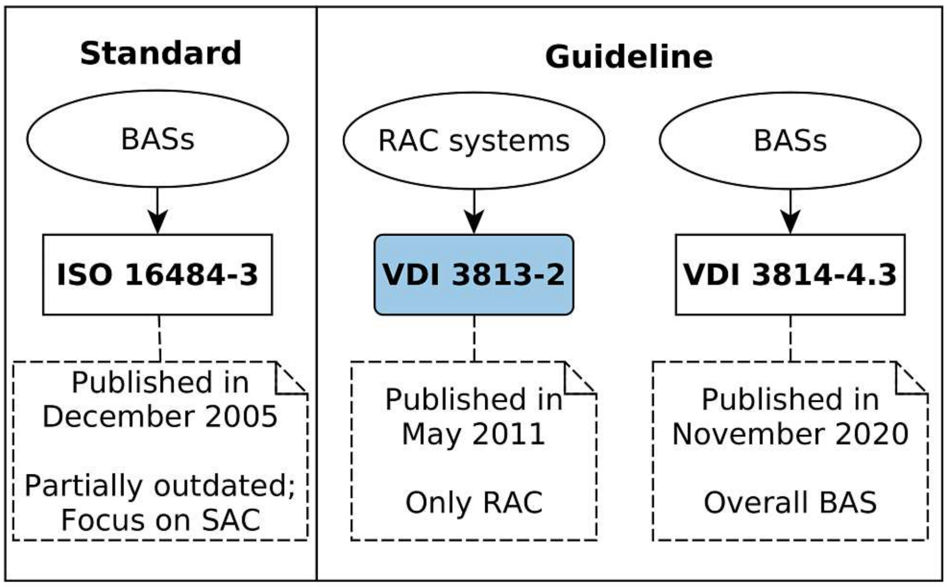


Figure 3.4: Overview of relevant guidelines and standards [42, 32, 76]

3.1.2 Function schematics and lists

Function schematics in combination with function lists are intended to provide detailed knowledge about the functionality of a system to subsequently design it on basis of this information. Most importantly, the relation between data points, function blocks and general building locations can be derived by utilizing these documents. However, they are only used as templates to describe the functionality of common rooms and buildings such as *"standard offices"* in order to ease the design of a system and to estimate costs. Concrete and identifiable data points, function blocks and building locations such as e.g. *room1-1* are not discussed in these documents. [42, 32]
Descriptions of function blocks, schematics and lists as published in part 2 of the VDI 3813 guideline are used to identify relevant information for designing RAC systems. Figure 3.5

provides an example of a function schematic, while Figure 3.6 illustrates a filled function list, which represents the counterpart to the schematic. All relevant information which is extracted from these documents to create the digital target representation defined in this thesis are modeled and documented in Section 3.3.

3.1.3 Additional information required

Unfortunately, the information provided by function schematics and lists is not sufficient to create and configure RAC system software. In this section, additional relevant aspects are outlined.

- **Device information**
  Devices, which are intended to be installed, have to be digitally represented in order to identify, establish communication and operate them. The device type, manufacturer as well as the model number have to be known. [8]

- **Address information**
  In order to identify and interact with a specific device in a communication medium such as an industrial network or LAN/WLAN, the address of the device according to the used protocol has to be known. For example, KNX uses individual addresses to identify network participants, program and parameterize devices. Individual addresses are seldom used in the operation phase. Instead, group addresses are assigned to devices, which mimic a connecting wire between two participants, e.g. a sensor and an actuator. These addresses are assigned in the engineering step of components. [51] Therefore, such protocol and medium dependent addresses have to be captured and linked with devices.

- **Association between data points and devices**
  A relation between data points specified in function schematics and lists and devices intended to be installed has to be established. [8]

- **Identification of locations**
  Data points, devices and function blocks in function lists and schematics are associated with general locations such as building, area, room or segment, as these documents are used as templates for subsequent system design and are not intended as one-to-one representation of real world systems. In order to tell which data point, device and function block belongs to which exact segment, each location has to be modeled individually and must be uniquely identifiable.

- **Association between data points and locations**
  Relations between the specific, identifiable locations discussed in the previous paragraph and data points have to be created in order to know their location within a building. Furthermore, not only the location of a data point is of interest, but also the location which are influenced by data points. There should exist the possibility to model such associations.

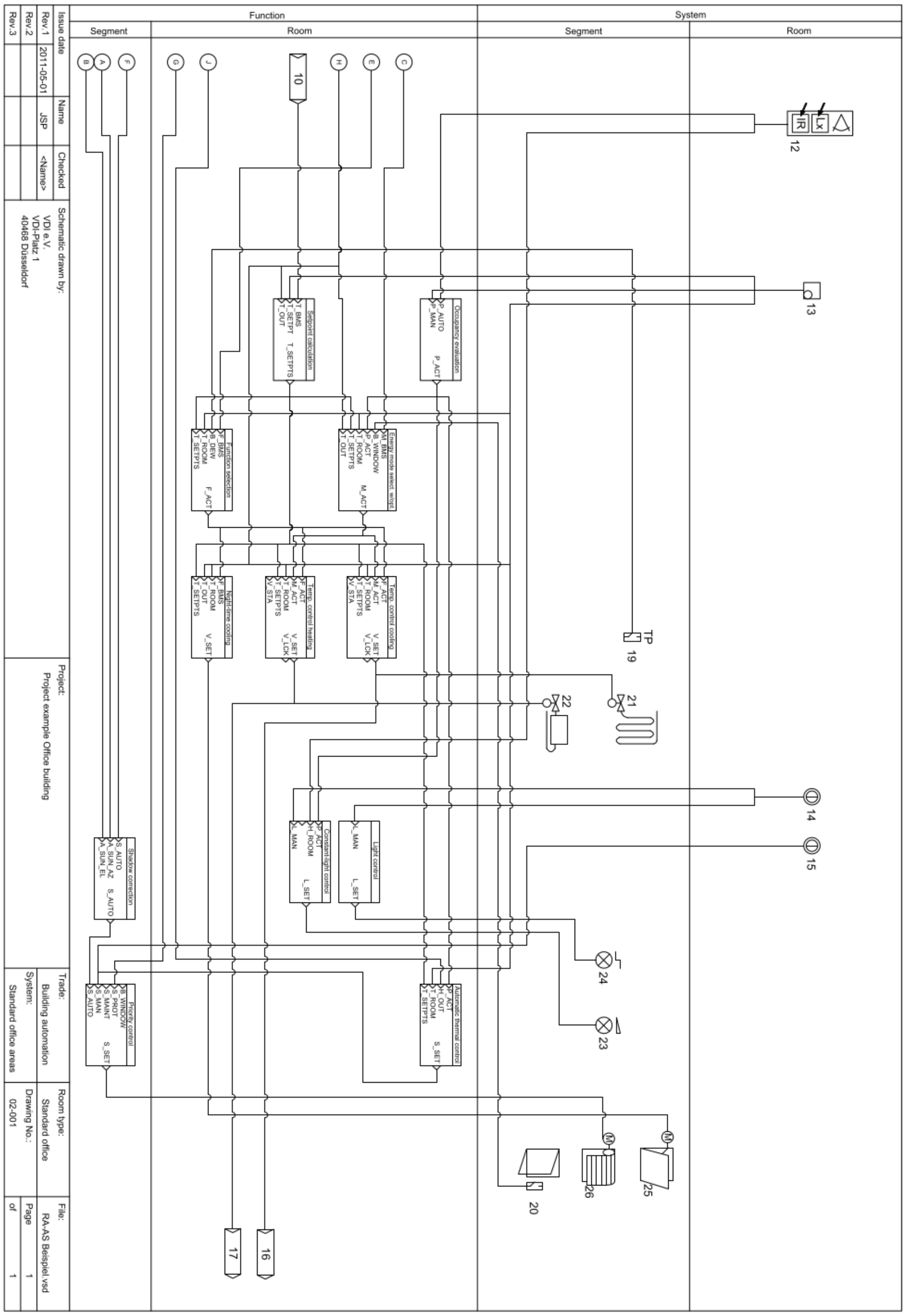
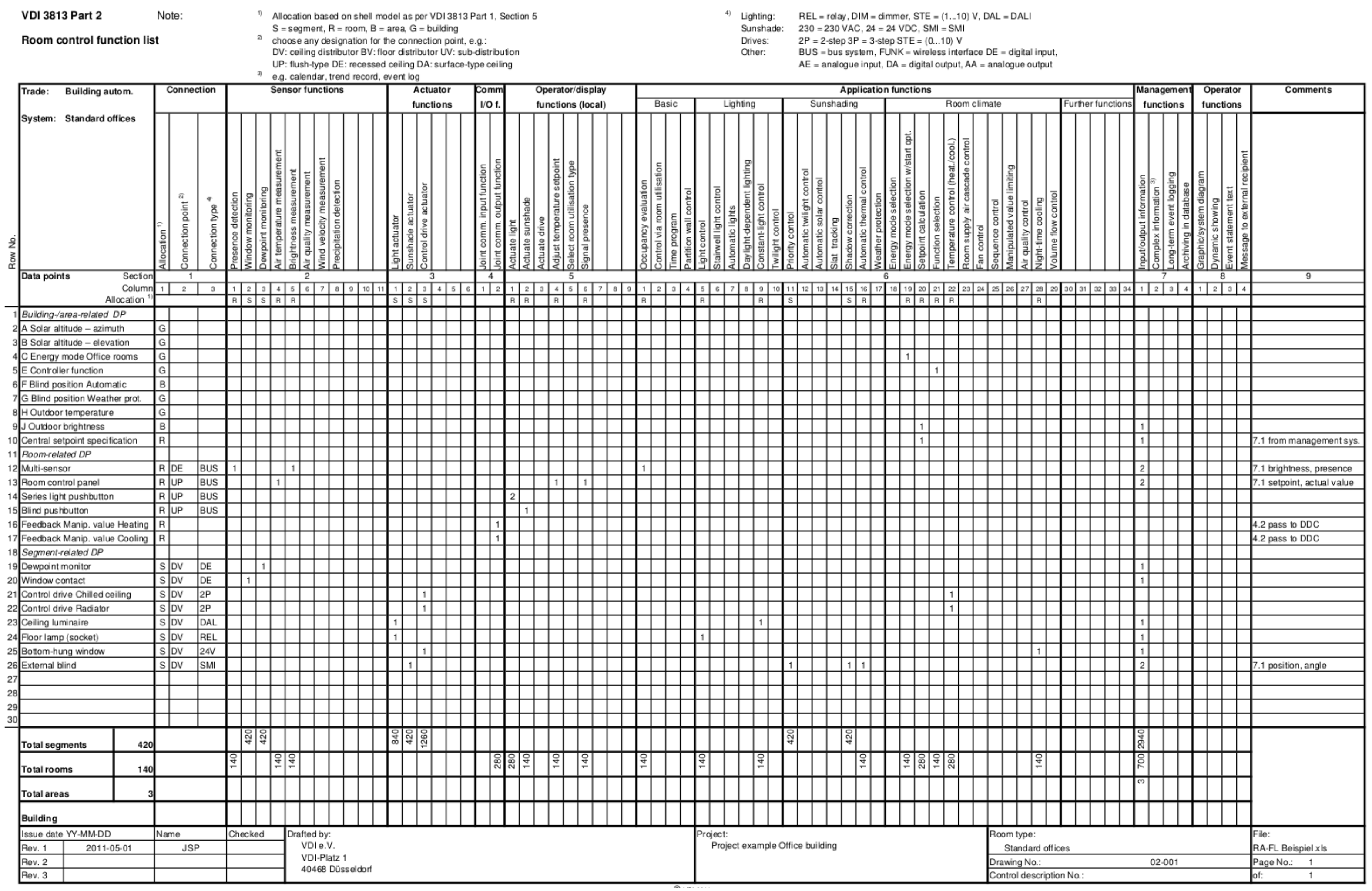


Figure 3.5: Function schematic example; Figure taken from part 2 of the of the VDI 3813 guideline [32]



VDI 3813 Part 2

Room control function list

Note:

[1] Allocation based on shell model as per VDI 3813 Part 1, Section 5
S = segment, R = room, B = area, G = building
[2] choose any designation for the connection point, e.g.:
DV: ceiling distributor BV: floor distributor UV: sub-distribution
UP: flush-type DE: recessed ceiling DA: surface-type ceiling
[3] e.g. calendar, trend record, event log

[4] Lighting: REL = relay, DIM = dimmer, STE = (1...10) V, DAL = DALI
Sunshade: 230 = 230 VAC, 24 = 24 VDC, SMI = SMI
Drives: 2P = 2-step 3P = 3-step STE = (0...10) V
Other: BUS = bus system, FUNK = wireless interface DE = digital input,
AE = analogue input, DA = digital output, AA = analogue output

Row No.	Trade: Building autom. System: Standard offices	Connection			Sensor functions											Actuator functions						Comm I/O f.	
		Allocation [1]	Connection point [2]	Connection type [4]	Presence detection	Window monitoring	Dewpoint monitoring	Air temperature measurement	Brightness measurement	Air quality measurement	Wind velocity measurement	Precipitation detection				Light actuator	Sunshade actuator	Control drive actuator				Joint comm. input function	Joint comm. output function
	Data points Section	1			2											3						4	
	Column	1	2	3	1	2	3	4	5	6	7	8	9	10	11	1	2	3	4	5	6	1	2
	Allocation [1]				R	S	S	R	R							S	S	S					
1	*Building-/area-related DP*																						
2	A Solar altitude – azimuth	G																					
3	B Solar altitude – elevation	G																					
4	C Energy mode Office rooms	G																					
5	E Controller function	G																					
6	F Blind position Automatic	B																					
7	G Blind position Weather prot.	G																					
8	H Outdoor temperature	G																					
9	J Outdoor brightness	B																					
10	Central setpoint specification	R																					
11	*Room-related DP*																						
12	Multi-sensor	R	DE	BUS	1				1														
13	Room control panel	R	UP	BUS				1															
14	Series light pushbutton	R	UP	BUS																			
15	Blind pushbutton	R	UP	BUS																			
16	Feedback Manip. value Heating	R																					1
17	Feedback Manip. value Cooling	R																					1
18	*Segment-related DP*																						
19	Dewpoint monitor	S	DV	DE			1																
20	Window contact	S	DV	DE		1																	
21	Control drive Chilled ceiling	S	DV	2P														1					
22	Control drive Radiator	S	DV	2P														1					
23	Ceiling luminaire	S	DV	DAL												1							
24	Floor lamp (socket)	S	DV	REL												1							
25	Bottom-hung window	S	DV	24V														1					
26	External blind	S	DV	SMI													1						
27																							
28																							
29																							
30																							
	Total segments 420					420	420									840	420	1260					
	Total rooms 140				140			140	140														280
	Total areas 3																						
	Building																						

Row No.	Data points	Operator/display functions (local)									Application functions: Basic				Lighting						Sunshading						
		Actuate light	Actuate sunshade	Actuate drive	Adjust temperature setpoint	Select room utilisation type	Signal presence				Occupancy evaluation	Control via room utilisation	Time program	Partition wall control	Light control	Stairwell light control	Automatic lights	Daylight-dependent lighting	Constant-light control	Twilight control	Priority control	Automatic twilight control	Automatic solar control	Slat tracking	Shadow correction	Automatic thermal control	Weather protection
	Section	5									6																
	Column	1	2	3	4	5	6	7	8	9	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
	Allocation [1]	R	R		R		R				R				R				R		S				S	R	
1	*Building-/area-related DP*																										
2	A Solar altitude – azimuth																										
3	B Solar altitude – elevation																										
4	C Energy mode Office rooms																										
5	E Controller function																										
6	F Blind position Automatic																										
7	G Blind position Weather prot.																										
8	H Outdoor temperature																										
9	J Outdoor brightness																										
10	Central setpoint specification																										
11	*Room-related DP*																										
12	Multi-sensor										1																
13	Room control panel				1		1																				
14	Series light pushbutton	2																									
15	Blind pushbutton		1																								
16	Feedback Manip. value Heating																										
17	Feedback Manip. value Cooling																										
18	*Segment-related DP*																										
19	Dewpoint monitor																										
20	Window contact																										
21	Control drive Chilled ceiling																										
22	Control drive Radiator																										
23	Ceiling luminaire																		1								
24	Floor lamp (socket)														1												
25	Bottom-hung window																										
26	External blind																				1				1	1	
27																											
28																											
29																											
30																											
	Total segments 420																				420				420		
	Total rooms 140	280	140		140		140				140				140				140							140	
	Total areas 3																										
	Building																										

Row No.	Data points	Application functions: Room climate												Further functions				
		Energy mode selection	Energy mode selection w/start opt.	Setpoint calculation	Function selection	Temperature control (heat./cool.)	Room supply air cascade control	Fan control	Sequence control	Manipulated value limiting	Air quality control	Night-time cooling	Volume flow control					
	Section	6																
	Column	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34
	Allocation [1]		R	R	R	R						R						
1	*Building-/area-related DP*																	
2	A Solar altitude – azimuth																	
3	B Solar altitude – elevation																	
4	C Energy mode Office rooms		1															
5	E Controller function				1													
6	F Blind position Automatic																	
7	G Blind position Weather prot.																	
8	H Outdoor temperature																	
9	J Outdoor brightness			1														
10	Central setpoint specification			1														
11	*Room-related DP*																	
12	Multi-sensor																	
13	Room control panel																	
14	Series light pushbutton																	
15	Blind pushbutton																	
16	Feedback Manip. value Heating																	
17	Feedback Manip. value Cooling																	
18	*Segment-related DP*																	
19	Dewpoint monitor																	
20	Window contact																	
21	Control drive Chilled ceiling					1												
22	Control drive Radiator					1												
23	Ceiling luminaire																	
24	Floor lamp (socket)																	
25	Bottom-hung window											1						
26	External blind																	
27																		
28																		
29																		
30																		
	Total segments 420																	
	Total rooms 140		140	280	140	280						140						
	Total areas 3																	
	Building																	

Row No.	Data points	Management functions				Operator functions				Comments
		Input/output information	Complex information [3]	Long-term event logging	Archiving in database	Graphic/system diagram	Dynamic showing	Event statement text	Message to external recipient	
	Section	7				8				9
	Column	1	2	3	4	1	2	3	4	
	Allocation [1]									
1	*Building-/area-related DP*									
2	A Solar altitude – azimuth									
3	B Solar altitude – elevation									
4	C Energy mode Office rooms									
5	E Controller function									
6	F Blind position Automatic									
7	G Blind position Weather prot.									
8	H Outdoor temperature									
9	J Outdoor brightness	1								
10	Central setpoint specification	1								7.1 from management sys.
11	*Room-related DP*									
12	Multi-sensor	2								7.1 brightness, presence
13	Room control panel	2								7.1 setpoint, actual value
14	Series light pushbutton									
15	Blind pushbutton									
16	Feedback Manip. value Heating									4.2 pass to DDC
17	Feedback Manip. value Cooling									4.2 pass to DDC
18	*Segment-related DP*									
19	Dewpoint monitor	1								
20	Window contact	1								
21	Control drive Chilled ceiling									
22	Control drive Radiator									
23	Ceiling luminaire	1								
24	Floor lamp (socket)	1								
25	Bottom-hung window	1								
26	External blind	2								7.1 position, angle
27										
28										
29										
30										
	Total segments 420	2940								
	Total rooms 140	700								
	Total areas 3	3								
	Building									

Issue date YY-MM-DD		Name	Checked	Drafted by:	Project:	Room type:		File:
Rev. 1	2011-05-01	JSP		VDI e.V.	Project example Office building	Standard offices		RA-FL Beispiel.xls
Rev. 2				VDI-Platz 1		Drawing No.:	02-001	Page No.: 1
Rev. 3				40468 Düsseldorf		Control description No.:		of: 1

Figure 3.6: Function list example; Figure taken from part 2 of the of the VDI 3813 guideline [32]

- **Association between locations**
  Locations in function schematics and lists are based on the multilayer-model published in part 1 of the VDI 3814 guideline. [26] However, locations are modeled individually as stated, they have to be interlinked with each other. For example, if a room r1 has two segments s1 and s2, these segments have to be associated with room r1 by using relations such as *s1 isPartOf r1* and *s2 isPartOf r1*.

3.2 Form and structure of the digital representation

Now that the required information for creating of RAC system software is defined, it has to be represented in a digital knowledge base. The reasoning for creating a digital twin using an ontology is discussed in Sections 3.2.1, 3.2.2 and 3.2.3.

3.2.1 Utilization of digital twin representations in building projects

Digital representations of construction plans and as-built physical contexts are becoming increasingly important in the lifecycle of buildings. IoT and BIM, used in digital construction processes, are two well known concepts for digitally representing design, construction, engineering, management and maintenance information of buildings. Digital twins share characteristics with these concepts and are utilized to digitally represent real world buildings, including measurements which provide information about the real-time state of physical objects. [24]
A digital twin is a model within a cyber-physical system (CPS). CPSs are a form of embedded systems that interact with physical entities through a communication network. The models in these systems, the digital twins, serve as digital representations of real world components. [77] They can be viewed as digital shadows which contain all information relevant to describe the state of a physical object. [78] Furthermore, the digital twins should be extended and kept up to date through the whole lifecycle of a building. This is important so they are of benefit for disposal, refinements, testing, operation optimization and to predict the future condition of a building. [24]
In this thesis, a process for creating, parameterizing and operating RAC system software is created. Detailed information about the current state of components within a building and the system design is important for this purpose. Therefore, a digital twin system approach is chosen as form of digital representation.

3.2.2 Architectural decisions regarding digital twin systems

In [79], the main architectural decisions for creating digital twin systems are introduced. These aspects are visualized in Figure 3.7 and discussed in this section. [79]
Most importantly, the internal structure as well as the content of a digital twin system has to be defined. Among other criteria, this includes creating a meta-model that describes internal models and relations, adapting tools which enable it to receive and provide information and modularizing the content of digital twin systems to be flexible in case of newly available information, e.g. via micro-services and Docker. Furthermore,

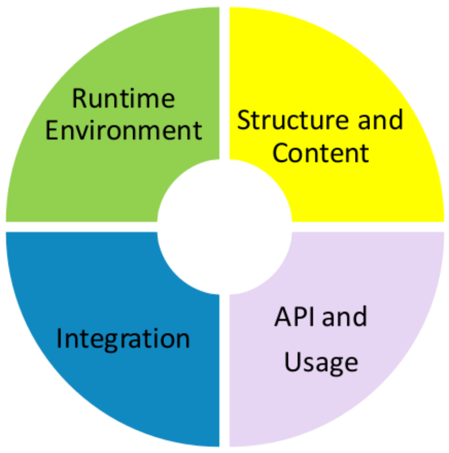


Figure 3.7: Architectural decisions regarding digital twin systems; Figure taken from [79]

standards have to be adopted to allow cross-organizational information exchange while implementing mechanisms that adjust existing information according to these standards. [79]
To allow digital twins to exchange and use their information as well as deriving more knowledge from them, application programming interfaces (API)s have to be developed. Different mechanism have to be considered when developing APIs, including cloud-to-device and cloud-to-cloud interactions, the information exchange frequency as well as access and authentication rights. [79]
A digital twin has to be associated and connected with a real world component. Therefore, it has to have a unique identifier, the ability to automatically detect a physical object on the network and the information stored in a digital twin has to be updated at an appropriate rate. Digital twins also have to be connectable with each other in order to create a composite digital twin. [79]

3.2.3 Digital twin ontology considerations

Ontologies are well suited as data- and knowledge bases for digital twin systems. They enable data sharing data between digital twins, their users and external systems while providing the ability to separate operational knowledge from domain knowledge. Furthermore, ontology representations simplify the process of data integrity and consistency monitoring. [24, 80]
The main characteristics of ontologies further emphasize their suitability for building and automation projects. They enable the reuse of domain knowledge and help to define and spread a common structure of information between software agents and people. Using ontologies, explicit domain assumptions and specifications can be created without the need of hard-coding them in programming code. This makes it easier to update this information in case of domain knowledge changes and also helps non-programming

experts to understand and learn domain specific terms and assumptions. The declaration of terms in ontologies furthermore enables a formal analysis of domain knowledge. [80]
In [80], a guideline for creating ontologies including design decisions is provided. The following three aspects are of special importance:

- There exist no single *"correct"* way for developing ontologies of a domain. The design decisions in ontology development are strongly dependent on how it will be used in the future and which extensions are expected. [80]
- The development of ontologies is an iterative process. After creating a simple draft of an ontology, it is consistently tested, verified, extended, corrected and redesigned while implementing the domain knowledge. [80]
- The terms and content of an ontology should be closely related to the domain objects it represents, by using nouns and verbs of the implemented domain. [80]

While keeping the above described aspects in mind, it is advisable to additionally implement the 4W1H method [24]. In this method, the four "W" questions What/When/Where/Who and How as "H" question are asked in each stage of a mobile crowd sensing lifecycle. [81] These questions can also be helpful when creating entities and object relationships for ontologies. [24]
Furthermore, the method created by [56] is also utilized in this work. Following this guideline, four initial questions before starting the development of an ontology are answered [56]:

- What should be built?
  - An ontology which represents the information and current state of the design of a building to create and operate RAC system software.
- How would the ontology be used and by whom?
  - A generalized process will utilize this ontology as target representation of RAC systems to create and configure RAC system software.
- Which language is appropriate for the ontology?
  - OWL [82] and Turtle [54], a specific syntax of RDF, are used as ontology language, as they are commonly used and supported by the ontology development tool used in this work, Protégé [55].
- Which method is appropriate for ontology capture?
  - In this work, no particular ontology capturing method is used. Key concepts and relationships are identified by analyzing existing ontologies and using common terms and definitions used in these knowledge bases, similar to the approach taken in [24].

In addition to the methods discussed above, the following set of requirements should be met by ontologies which are geared towards modeling digital twins for BASs according to [24]:

- Basic concepts and relations of the domain have to be included
- Digital twin components have to be able to interact and operate with each other
- It has to be modular
- It has to be extendable in order to create and maintain digital twin information
- It should be geared towards state of the art technology and methods, multi-agent systems and building automation hardware and software
- IoT, cloud services and data storage systems have to be supported

This set of requirements for BASs is kept in mind when developing the ontology, but it is not strictly followed, as the focus is on RAC systems.

3.2.4 Existing digital twin ontologies

In Table 3.2, the deficiencies of relevant, digital twin related ontologies based on [24] are outlined. A description of these ontologies, which have been considered to be the most important for this work, can be found in Section 2.3.1.

Ontology	**Disadvantages**
BAS ontologies	
SAREF [58]	Has a limited set of devices, functions and commands. [24]
SAREF4EE [59]	Analogous to SAREF. [24]
BRICK [60]	Too simplistic in specific cases such as function schematic and list modeling [24]
BOnSAI [61]	Some concepts are very abstract or too specific to web services and therefore of limited use for BAS modeling. [24]
DogOnt [62]	The location concept is not applicable for high rise buildings; HVAC systems are not included. [24]
ThinkHome [63]	Might be hard to grasp as class categorization and structure is very different in comparison to other ontologies. [24]
Ontologies for control and monitoring	
SSN [65]	Only includes basic concepts and has to be extended with domain specific terms. [24]

M3 [66]	Analogous to SSN. [24]
OntoSensor [67]	Is complex to use and does not provide the ability to describe sensor observations. [83]
IoT ontologies	
IoT-Lite [68]	Is specific to constrained IoT environments. [83]
Open-IoT [69]	Has similar problems as with IoT-Lite [83]
IoT-O [84]	*"IoT-O lacks a complete/holistic view of the context."* [83]
FIESTA-IoT [70]	Provides no support for context-aware services; limited notion of activity, actuators and virtual entities; context information such as place of interest cannot be annotated. [70]

Table 3.2: Overview of existing, relevant ontologies [24]

Considering the requirements for digital twin BAS ontologies listed in Section 3.2.3, none of the ontologies observed in Table 3.2 is suitable for the purpose of this work without adjustment. Therefore, an iterative approach according to the ontology development methods discussed in Section 3.2.3 is taken to create a new, suitable ontology.

3.3 Digital target representation - digital twin ontology

"A goal should be, not to reinvent the vocabulary repeatedly." [64]

The created ontology is focused on providing all information required for creating RAC system software. A common recommendation is, to keep ontologies as small and simplistic as possible, although in some cases such ontologies can also be hard to reason about. [85] Keeping that in mind, the ontology is only extended as need occurs, but is kept as small as possible. Required concepts, terms and vocabulary are taken from already existing ontologies listed in Tables 2.4 and 3.2 by reviewing and adjusting them accordingly. This is in line with the broad consensus of reusing parts of existing ontologies, which cover the need of a developer without reinventing them. [24, 64]

3.3.1 Function list and schematic modeling

At first, relevant information provided by function lists and schematics based on Figure 3.6 are modeled. This provides a core structure which is built upon in Section 3.3.2. The

knowledge extracted from function lists modeled as entities in the ontology is visualized in the following figures.
Based on the concept used in Brick, a *definition* annotation property published in the simple knowledge organization system namespace document (SKOS) is used to create in depth definitions and clarifications for ontology components. [86] These annotations are not visible in the illustrated figures to keep them simple and understandable.
The location implementation follows the examples of BOnSAI and Brick, which use classes such as *Building* and *Room*. [61, 60] However, it is adjusted to represent the multilayer-model outlined in part 1 of the VDI 3814 guideline, visualized in Figure 3.10. [26] As a consequence, *Room* is a subclass of *Area*, *Area* is a subclass of *Building*, etc. As an umbrella class, *Location* is introduced, which is the superclass of *RealEstate Portfolio* of the multilayer-model. The location class hierarchy is illustrated in Figure 3.11. Locations are related with each other using a *hasPart/isPartOf* relation following this hierarchy as discussed in Section 3.1.3.
Subclasses are in different colors and shapes to visualize the order of inheritance for classes that extend over multiple figures. For example, *ApplicationFunction* in Figure 3.15 is a subclass of *Function*, and *BasicApplicationFunction* in Figure 3.17 is again a subclass of the same *ApplicationFunction* illustrated in the previous figure. Object properties are visualized as arrows with filled black arrowheads and data properties are represented as arrows with dashed lines. *Device* is modeled as a subclass of *DataPoint*, as each device implicitly represents a data point according to the representation in function lists (see Figure 3.6). This is illustrated in Figure 3.9.

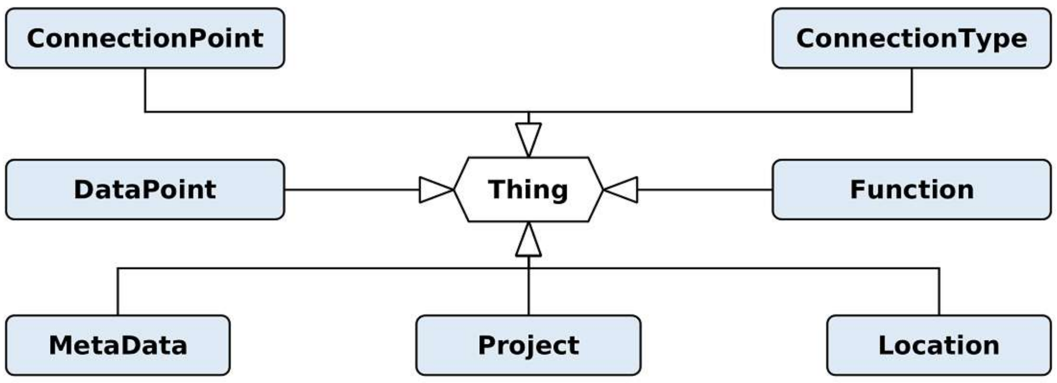


Figure 3.8: Classes created from the information provided in function lists compliant with part 2 of the VDI 3813 guideline [32]

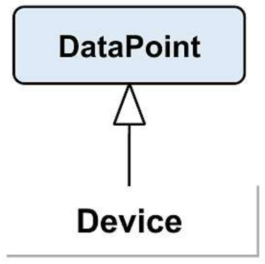


Figure 3.9: Devices modeled as subclass of data points, compliant with function lists from part 2 of the VDI 3813 standard [32]

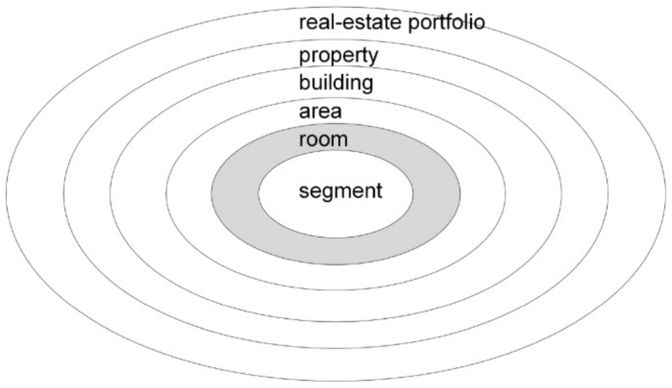


Figure 3.10: Multilayer-model; Figure taken from part 1 of the VDI 3814 guideline [26]

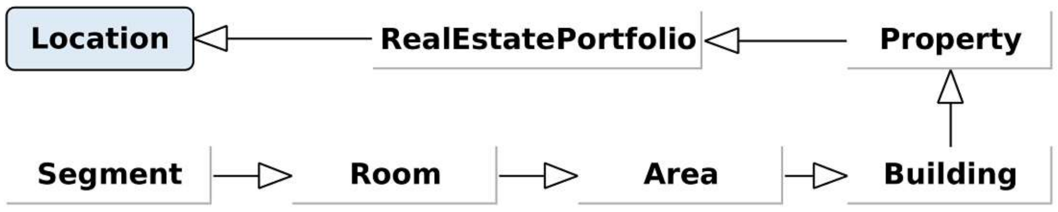


Figure 3.11: Location class hierarchy according to the multilayer-model presented in part 1 of the VDI 3814 guideline [26]

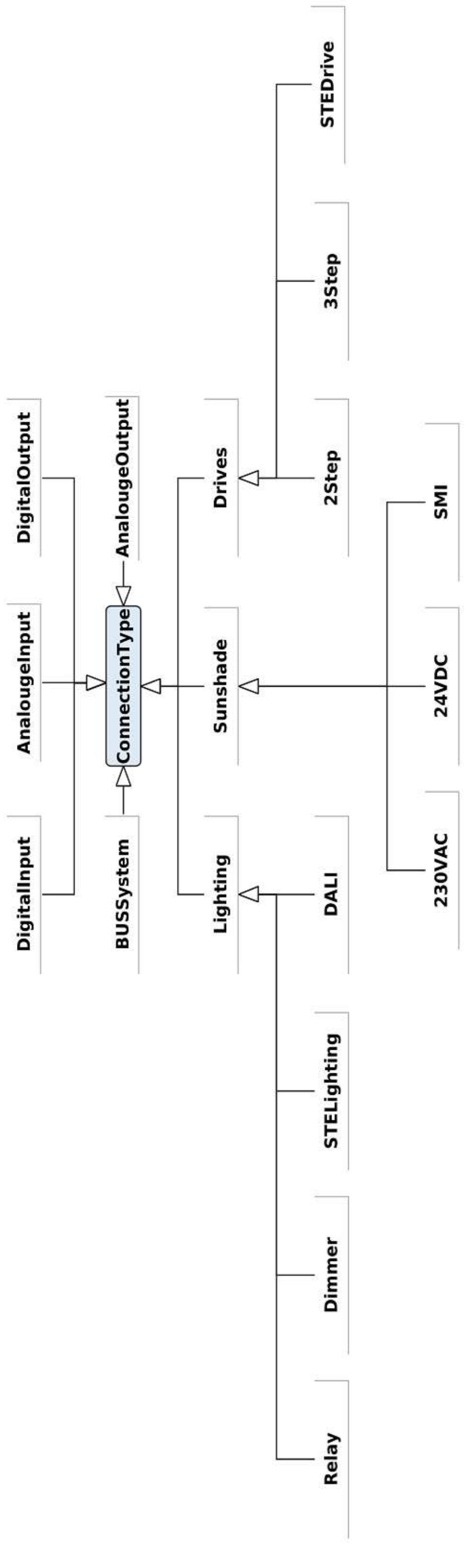


Figure 3.12: Connection type classes extracted from the information provided by function lists given in part 2 of the VDI 3813 guideline [32]

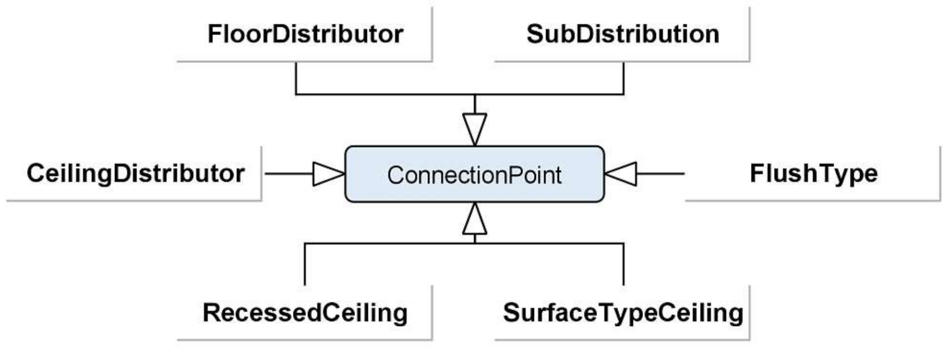


Figure 3.13: Connection point classes extracted from the information provided by function lists given in part 2 of the VDI 3813 guideline [32]

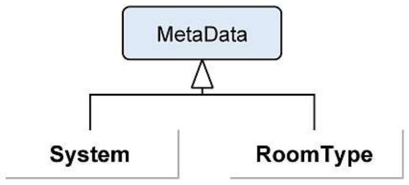


Figure 3.14: Metadata classes extracted from the information provided by function lists given in part 2 of the VDI 3813 guideline [32]

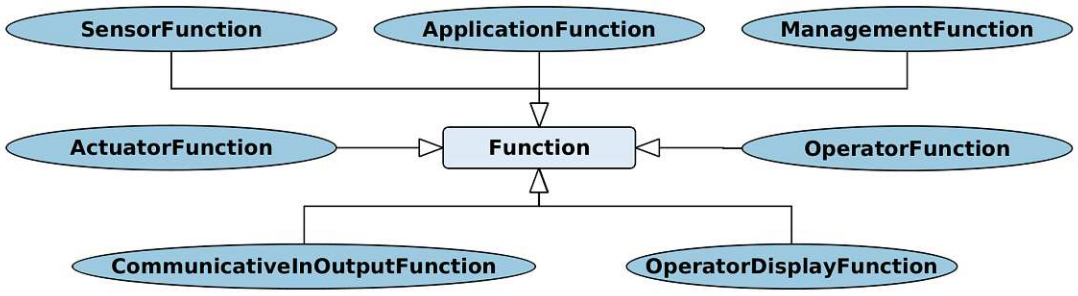


Figure 3.15: Basic umbrella functions modeled as classes and extracted from the information provided by function lists given in part 2 of the VDI 3813 guideline [32]

The function schematic does only provide two more object properties, but these are crucial for describing the functionality of a system. The *feeds* and *isFedBy* object properties, similar to the ones used in Brick, describe the relations between data points and functions (see Figure 3.5). [60] In contrast to Brick, they are modeled as transitive object properties. On top of that, three data properties are introduced in order to model values of different formats. The created entities are visualized in Figure 3.18.

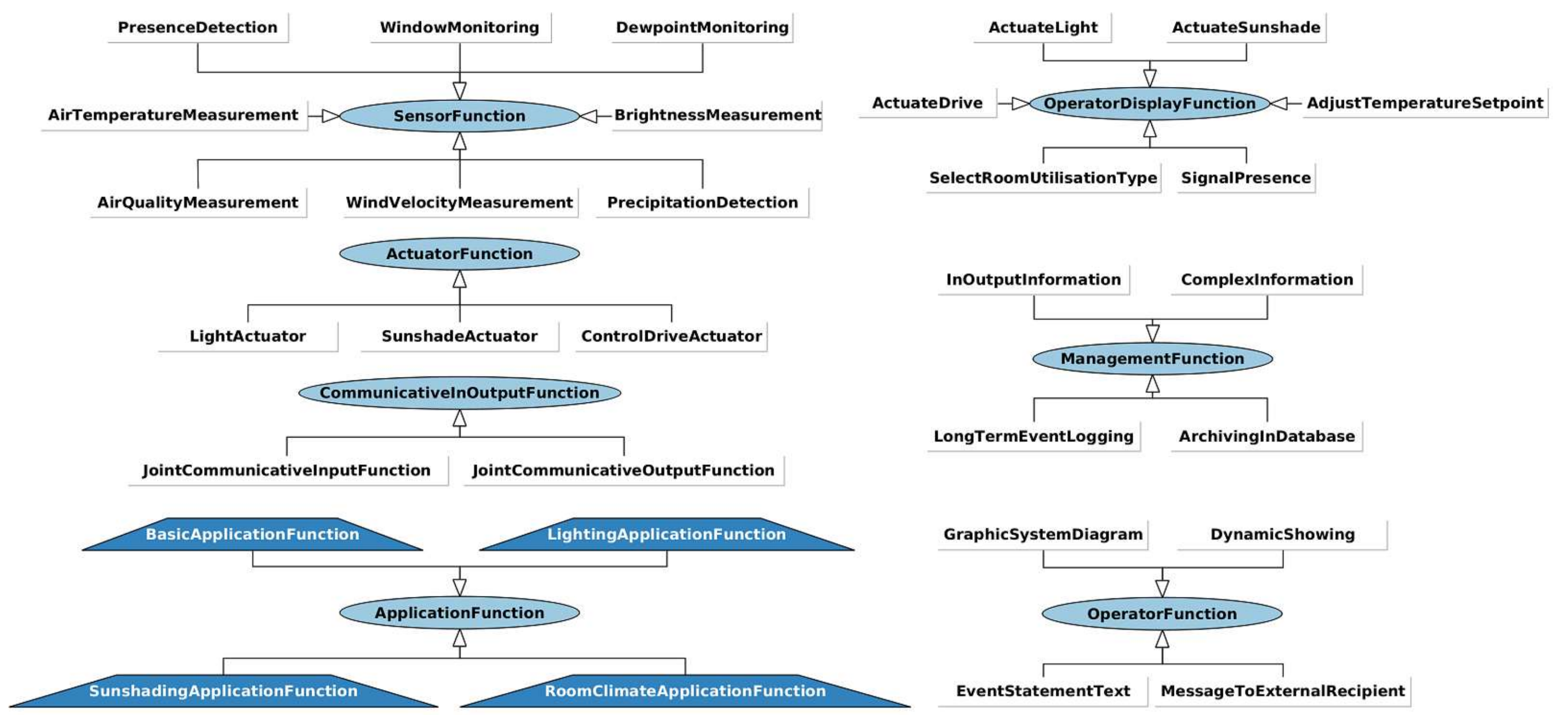


Figure 3.16: Overview of function blocks modeled as classes and extracted from the information provided by function lists given in part 2 of the VDI 3813 guideline [32]. The specific application functions are illustrated in Figure 3.17

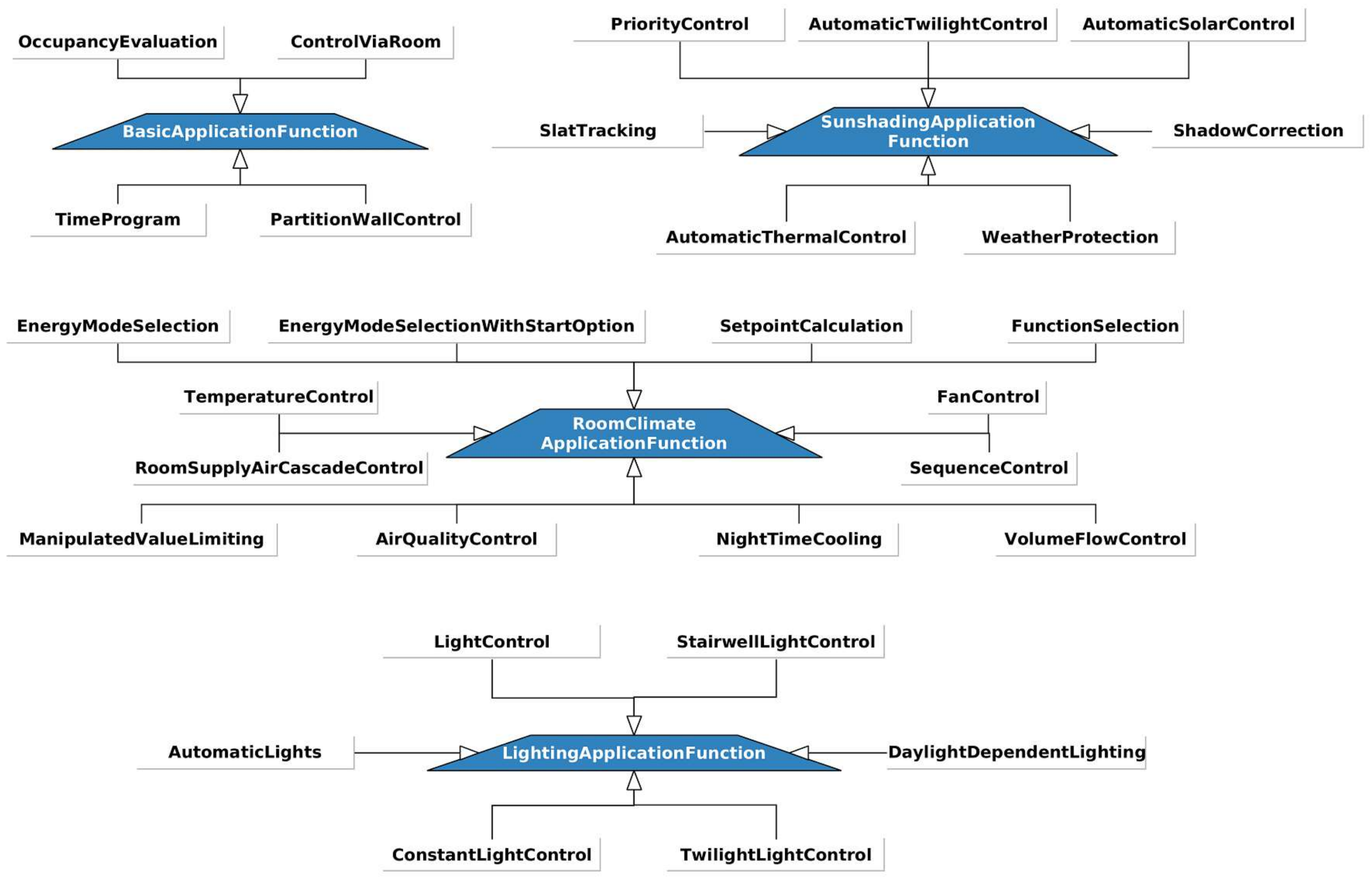


Figure 3.17: Application function blocks modeled as classes and extracted from the information provided by function lists given in part 2 of the VDI 3813 guideline [32]

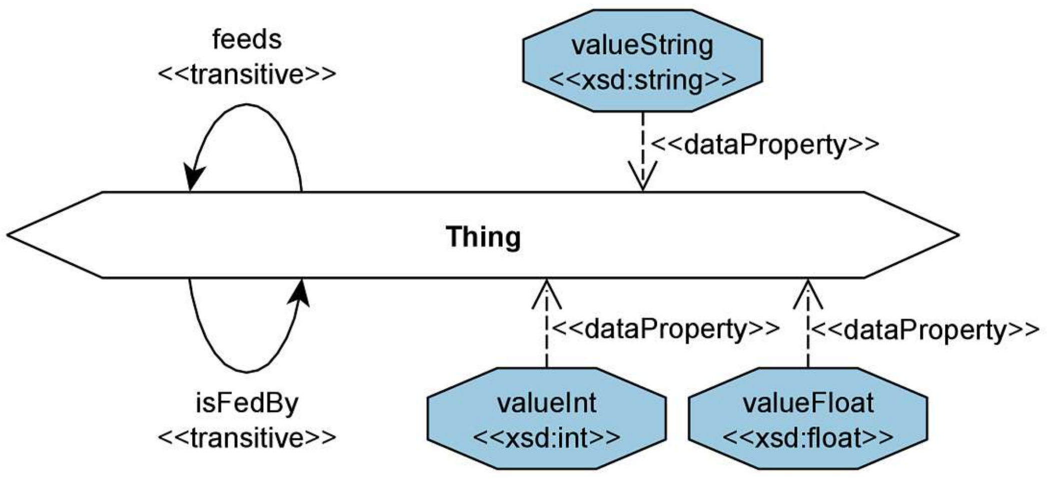


Figure 3.18: Entities modeled on basis of the extracted information provided by function schematics [32]

The following aspects, which can be derived from function lists and schematics, have not been modeled as it is unclear at this stage of development, if this information is of benefit for the purpose of this work:

- The total sum of functions relevant to a subsystem (e.g. segment, room, area). If needed, this can be calculated by using the information of individuals modeled in this ontology.
- Metadata of the function schematic
  - Issue date
  - Revision date and number
  - Checked by [person, organization, department, etc.]
  - Checked Status
  - Drafted by [person, organization, department, etc.]
  - Filename
  - Drawing number
  - Control description number
  - Page number and total amount of pages

3.3.2 Additional information modeling

On basis of the structure modeled in Section 3.3.1, the ontology is extended with additional required information as stated in Section 3.1.3. The introduced entities are visualized in Figure 3.19.

Data points are associated with functions, which covers all information that is needed at this stage of development of the ontology. This enables the possibility to derive the device type from the functionality it is associated with. It might be necessary to introduce subclasses such as *Lighting* or *HVAC* at a later stage of development. Furthermore, the data property *model* and *manufacturer* are introduced, to attach a model number and the manufacturer to a device.
Subclasses of *Device* represent specific network components or protocols, similar to the approach taken by the DogOnt ontology developers. [62] For now, only KNX devices with their address types are introduced. If the need arises, further classes such as e.g. *BACnetDevice* can be added.

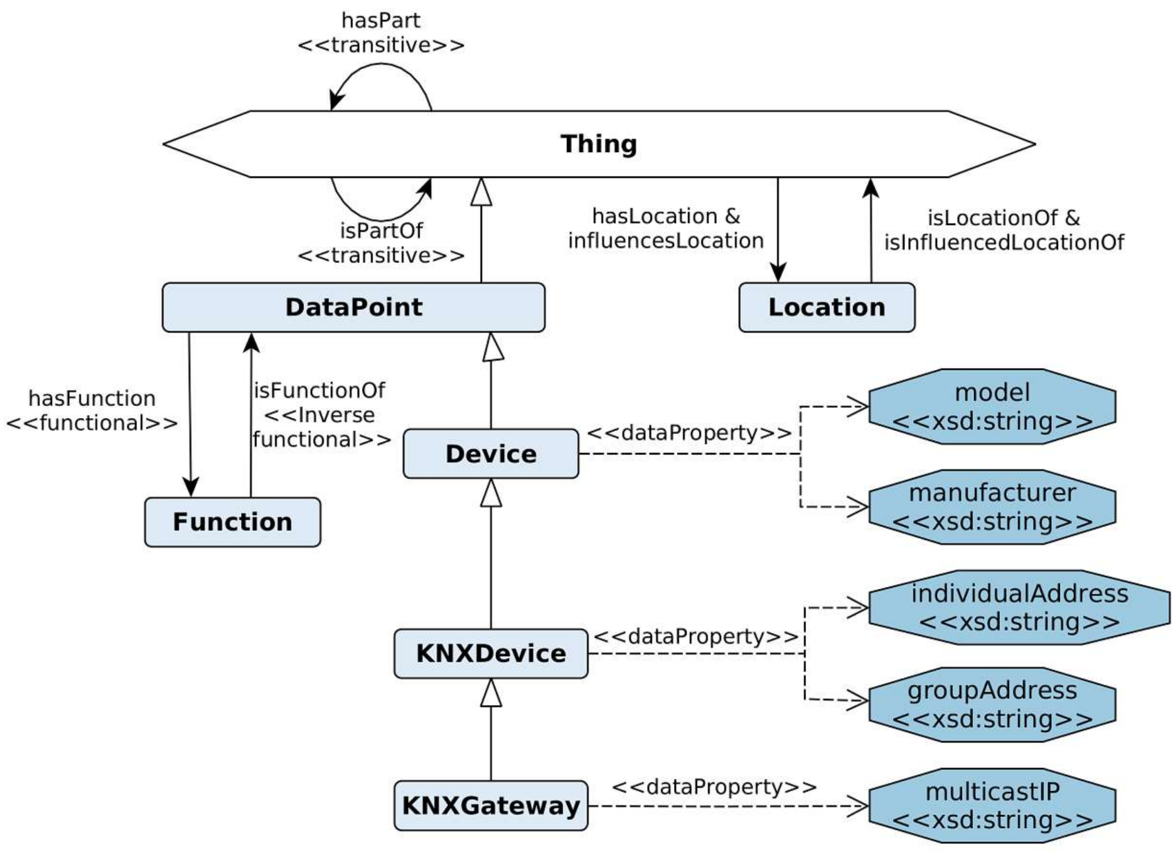


Figure 3.19: Entities modeled on basis of identified, required information

CHAPTER 4

Design processes for RAC systems

This chapter comprises part 2 and 3 of the methodology. Section 4.1 covers existing strategies and approaches for creating RAC system software. Problems and deficiencies in these strategies are analyzed in Section 4.2. Possible solutions to resolve the identified problems of a selected software creation approach are presented in Chapter 5.

4.1 Strategies for creating and sharing information in RAC projects

Conducting building automation projects is a complex task. Experts of different disciplines with various economic interest have to work together and share their knowledge in order to successfully create, operate and maintain a building with an effective automation system. [8] In building projects, typically different tools with various target formats are used in each development phase, leading to a broken information flow between project participants and additional time and effort that has to be dedicated to acquire and convert relevant information into suitable formats. [79]
There exist different strategies for designing, configuring, installing and operating RAC systems. The most relevant strategies for the research goal are discussed in this section, according to part 2 of the methodological approach (see Figure 4.1).

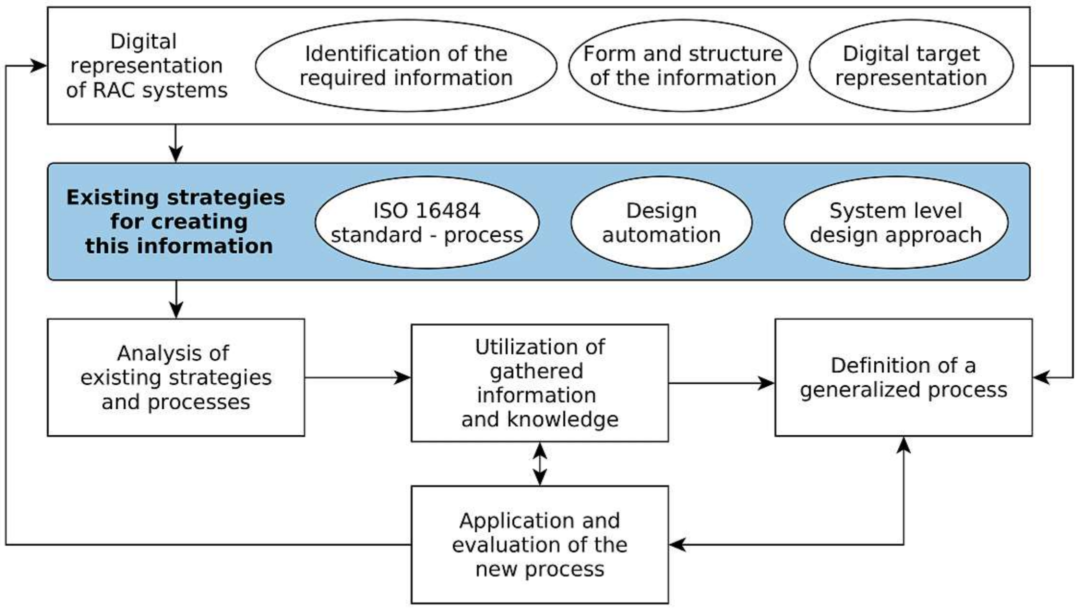

Digital representation of RAC systems
Identification of the required information
Form and structure of the information
Digital target representation
Existing strategies for creating this information
ISO 16484 standard - process
Design automation
System level design approach
Analysis of existing strategies and processes
Utilization of gathered information and knowledge
Definition of a generalized process
Application and evaluation of the new process

Figure 4.1: Part 2 of the methodological approach

4.1.1 ISO 16484 standard - process

In part 1 of the ISO 16484 standard, a generalized process for the conduction of building automation (BA) projects is defined. It guides stakeholders and participants through all lifecycles of such projects. [6] An overview of the phases within a building project according to this process is visualized in Figure 4.2. The process consists of the four phases *Design*, *Engineering*, *Installation* and *Completion*. The *Design phase* and the *Engineering phase* are highlighted, as they are of special interest for this work. During all phases, building performance should be continuously reviewed and improved. Furthermore, documentation and training should be considered and implemented during the phases *Engineering*, *Installation* and *Completion*. [6]

The *Design phase* includes the tasks *Determination of project requirements*, *Project planning and organization* and *Design documents and technical specifications*. These steps are executed in the given order. If a contract is declined, they may be revisited starting at the first task. Otherwise, the steps *Project planning and co-ordination* and *Detailed function and hardware design* are executed in this order. If the design is approved, the tasks *Hardware configuration*, *Control strategy configuration* and *Management and operator functions configuration* are conducted. It is advised to complete them in the given order, but project dependencies may change the sequence of execution. If the design is not approved, the task *Detailed function and hardware design* is revisited. If a system test is required, the *Engineering phase* is completed with the task *System test* and the *Installation phase* is entered subsequently. Otherwise, the *System test* task is

skipped. The tasks within the *Design* and *Engineering phase* of the ISO 16484 process are visualized in Figure 4.3. [6]
This sequence of actions and decisions is a suggestion made by the ISO and advised to be adhered, but not prescriptive. Project requirements may change the sequence of execution and move tasks to earlier or later stages in the lifecycle of a BA project. [6]

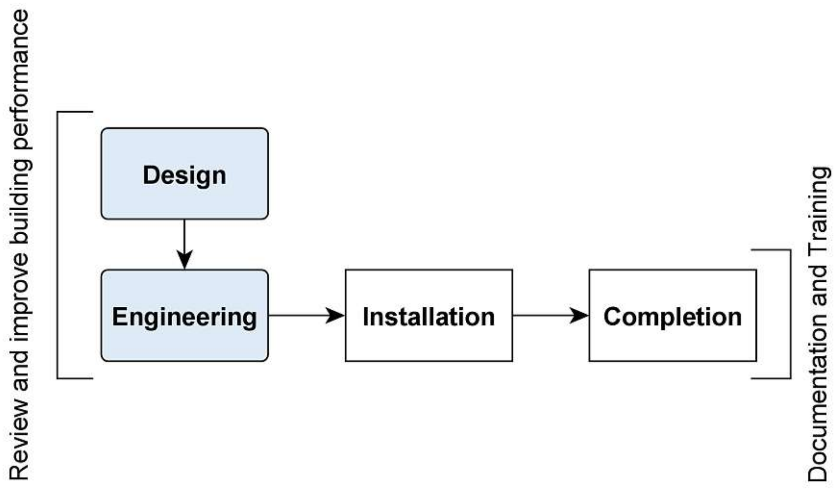

Review and improve building performance
Design
Engineering
Installation
Completion
Documentation and Training

Figure 4.2: Simplified overview of of the process presented in part 1 of the ISO 16484 standard [6]

Most information required for representing the functionality of BASs is included in function schematics and lists. [6] It is assumed that if one of them is given, the other one is also available, as both of them are not expressive and useful as stand-alone documents.

The function lists and schematics are created in the *Design documents and technical specification* task of the *Design phase.* They are produced on basis of the information gathered in the *Determination of requirements* step and are created on basis of given requirements, the know-how and experience of BAS planners. Function schematics and lists are part of the technical specification documents which are handed to clients and provide a baseline for establishing contracts. In the *Detailed hardware and function design* task outlined in the ISO 16484 process, specifically the *Preparation of submittals* part, these documents are revised and engineered in detail. [6] Unfortunately, there exists no commercial tool support and standardized file format for creating function lists and schematics. [8]
According to the part 1 of the ISO 16484 standard, function schematics and lists are recommended being used as foundation for the design of BASs, including RAC systems. However, they are stated to be an optional component of the technical specification. Therefore, it is possible that these documents are not created within the *Design phase* of the ISO 16484 process and therefore not available to system integrators. [6]

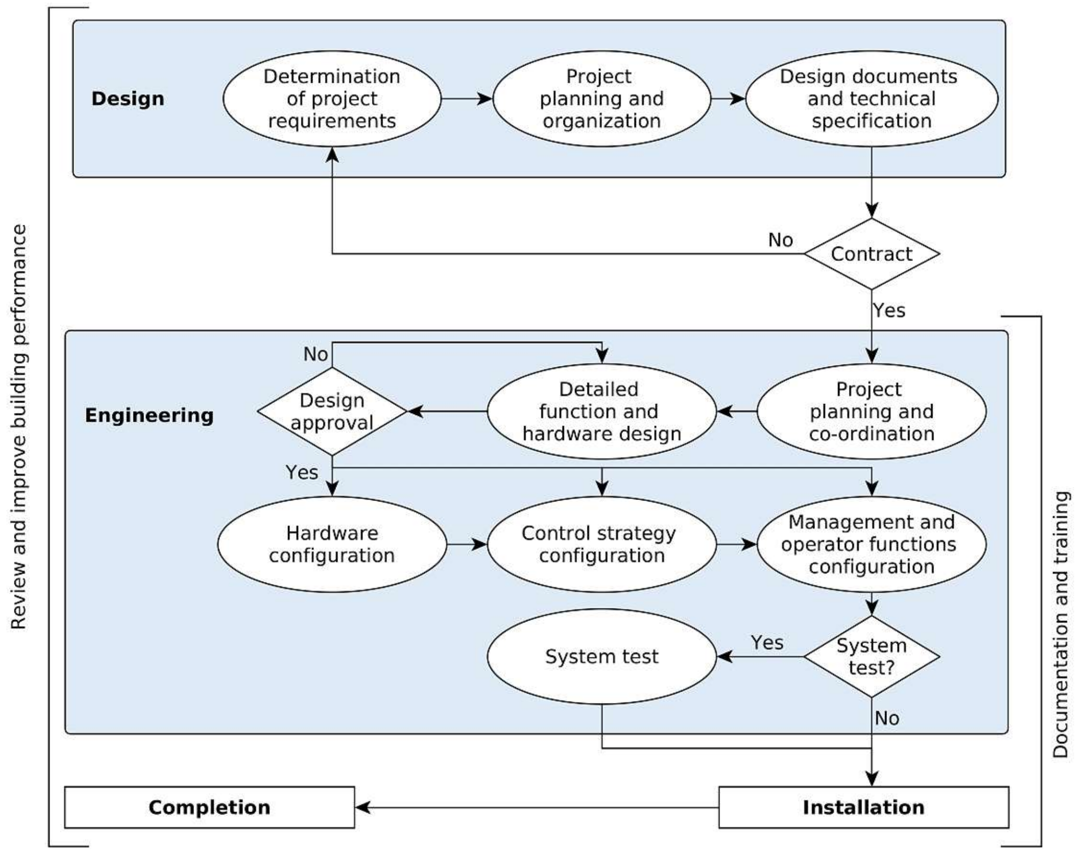

Review and improve building performance
Design
Determination of project requirements
Project planning and organization
Design documents and technical specification
No
Contract
Yes
Engineering
No
Design approval
Detailed function and hardware design
Project planning and co-ordination
Yes
Hardware configuration
Control strategy configuration
Management and operator functions configuration
System test
Yes
System test?
No
Completion
Installation
Documentation and training

Figure 4.3: Tasks within the *Design* and *Engineering phase* of the process presented in part 1 of the ISO 16484 standard [6]

4.1.2 Design automation

Various approaches for utilizing tools, which guide through the *Requirements engineering phase* of BASs, have been developed. They help to simplify or even automate the design of a such systems. When gathering the requirements of an envisioned system, the desired functional and non-functional requirements are defined and formalized in a structured, machine-readable representation. The following list includes information, which is gathered and digitized during the *Requirements engineering phase* in design automation (DA) approaches [8, 87, 88, 9]:

- Detailed functional requirements
  - Functionality of the system, including desired functions, e.g. constant light control based on a luminance sensor and a dimmable actuator
  - Usage flexibility
  - Energy efficiency classes based on the EN 15232 standard [89]

- Non-functional requirements, e.g. ingress protection or operating voltage
- Building plans including structure and identification of locations, i.e. based on the multilayer-model proposed in part 1 of the VDI 3814 guideline. [26]
- Room/Building templates used in subsequent DA processes

In order to gather this knowledge in a machine-readable format, the proposed tools have to be used through the whole lifecycle of a project, starting with the *Requirements engineering phase*. Thereby, functional and non-functional requirements are gathered by interacting with stakeholders and captured in a structured, digital database using tools dedicated for this purpose. Utilizing this knowledge, the DA software creates a system design based on templates which cover the functional and non-functional requirements for rooms and other building structures. These templates are predefined based on common requirements and use cases, e.g. office or seminar rooms, and stored in a template database. Furthermore, a template can contain multiple sub-templates in order to reuse functionality and information in numerous, reoccurring designs. Ultimately, the automation software is created on basis of the system design which was automatically designed beforehand. [8, 7, 87, 88] The approach and sequence of executed tasks is similar to the process defined in part 1 of the ISO 16484 guideline, except that the system is designed by software and not by a system planner. The BAS software is created automatically in contrast to the ISO 16484 process, in which an expert implements and configures the BAS software. [6, 8]

4.1.3 System level design approach

It is possible, that no formalized and structured information about functional and non-functional requirements, such as functions schematics and lists, are available at the time of BAS software creation. In other words, the system has been designed, including device selection, wiring plans etc. but there exist no dedicated documents which describe the functionality of the BAS. In this case, the system planner generally has to take the opposite of so-called top-down approaches, in which the functionality is available before the system is designed. In this work, such strategies are referred to as *system level design* approaches. In contrast to the top-down strategy as proposed in the ISO 16484 standard, which propagates the utilization of function schematics and lists as basis for the system design, the planner has to *"create a functional design in his mind"* when selecting the devices which should be integrated in the system. Furthermore, the expert which creates the BAS software has to recreate and derive this functionality from the given design documents. [8]
An overview and comparison of the strategies discussed in Sections 4.1.1, 4.1.2 and 4.1.3 is illustrated in Figure 4.4.

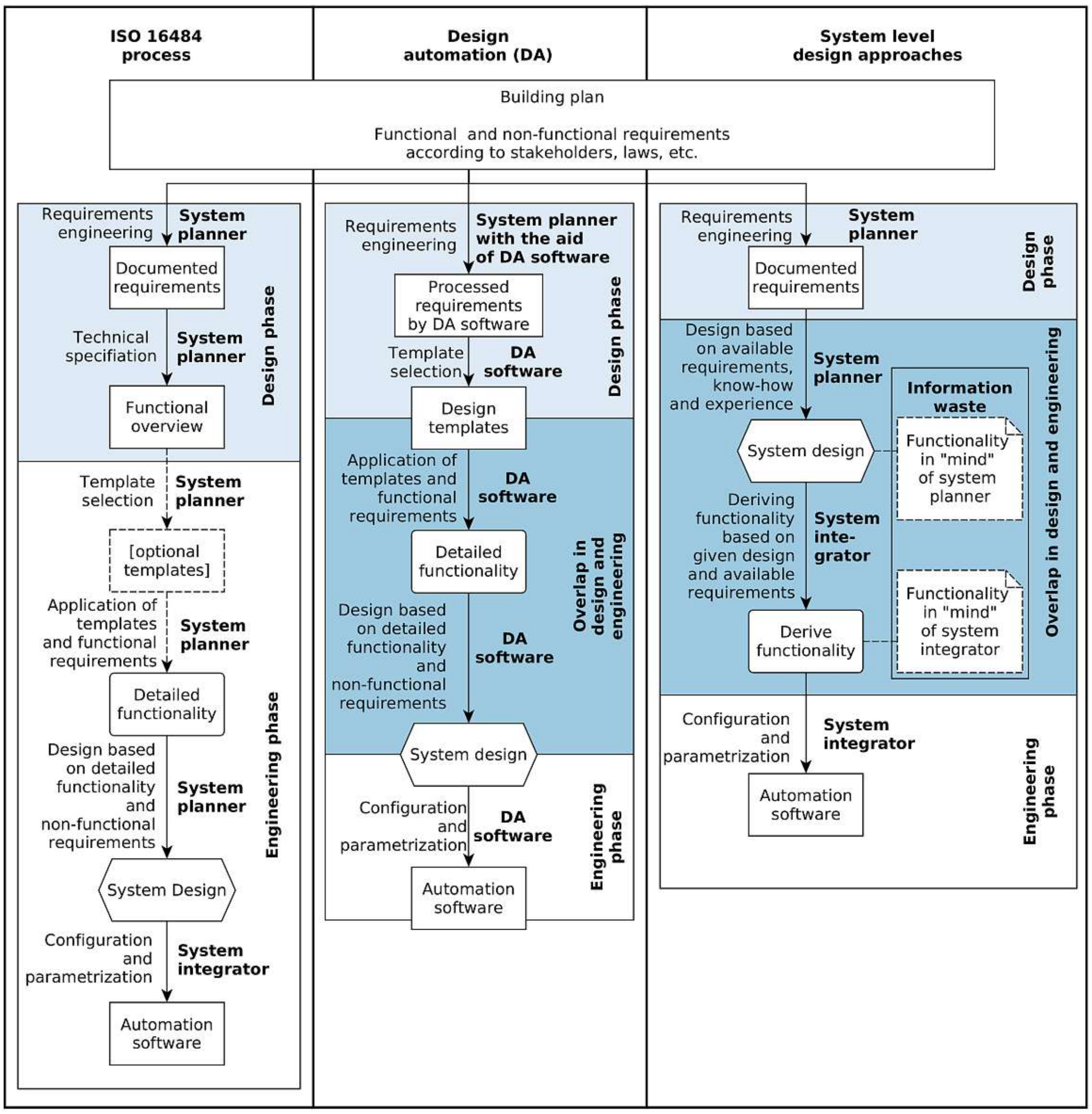

ISO 16484 process
Design automation (DA)
System level design approaches
Building plan
Functional and non-functional requirements according to stakeholders, laws, etc.
Requirements engineering
System planner
Documented requirements
Technical specification
System planner
Functional overview
Design phase
Template selection
System planner
[optional templates]
Application of templates and functional requirements
System planner
Detailed functionality
Design based on detailed functionality and non-functional requirements
System planner
System Design
Configuration and parametrization
System integrator
Automation software
Engineering phase
Requirements engineering
System planner with the aid of DA software
Processed requirements by DA software
Template selection
DA software
Design templates
Design phase
Application of templates and functional requirements
DA software
Detailed functionality
Design based on detailed functionality and non-functional requirements
DA software
Overlap in design and engineering
System design
Configuration and parametrization
DA software
Automation software
Engineering phase
Requirements engineering
System planner
Documented requirements
Design phase
Design based on available requirements, know-how and experience
System planner
Information waste
System design
Functionality in "mind" of system planner
Deriving functionality based on given design and available requirements
System integrator
Derive functionality
Functionality in "mind" of system integrator
Overlap in design and engineering
Configuration and parametrization
System integrator
Automation software
Engineering phase

Figure 4.4: Overview and comparison of strategies for creating BAS software

4.2 Analysis of RAC system software creation approaches

The strategies discussed in Section 4.1 have different advantages and deficiencies. The impact of these approaches is also dependent on the project context of a building and its intended purpose. In this section, these strategies are analyzed, covering part three of the methodological approach, as visualized in Figure 4.5. Differences in the approaches are summarized in Table 4.2 based on their main characteristics. The aspects are of varying importance and relevance for this work. To illustrate this fact, an extra column which represents the weight of importance for the research goal is added.

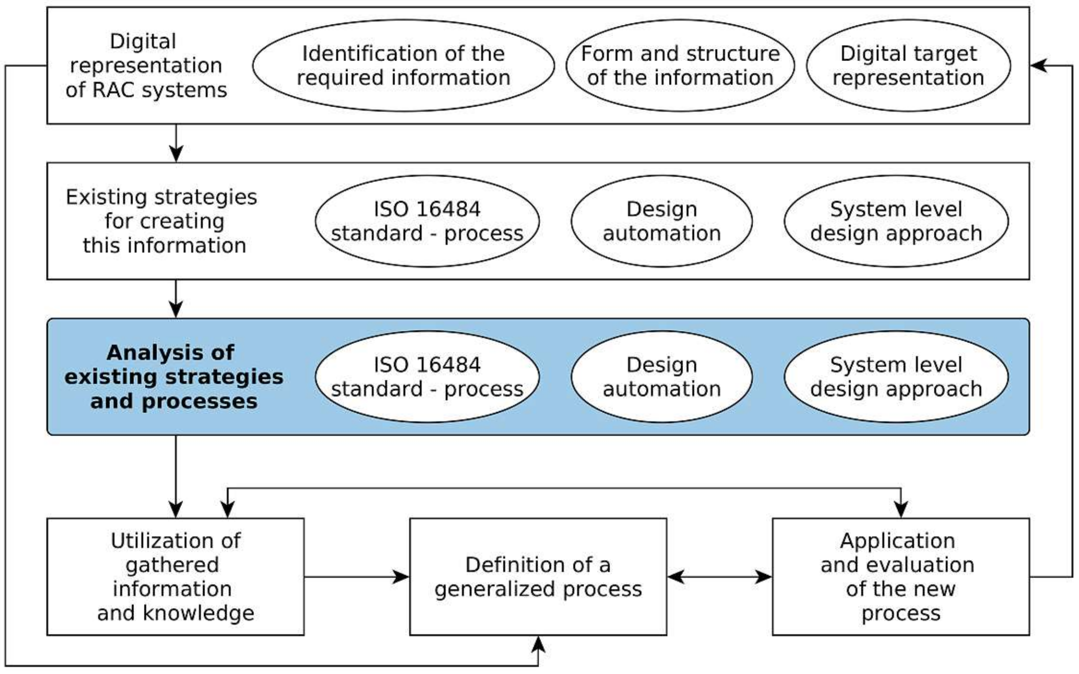

Digital representation of RAC systems
Identification of the required information
Form and structure of the information
Digital target representation
Existing strategies for creating this information
ISO 16484 standard - process
Design automation
System level design approach
Analysis of existing strategies and processes
ISO 16484 standard - process
Design automation
System level design approach
Utilization of gathered information and knowledge
Definition of a generalized process
Application and evaluation of the new process

Figure 4.5: Part 3 of the methodological approach

4.2.1 ISO 16484 process analysis

The main advantage of the process defined in part 1 of the ISO 16484 standard is, that it presents a structured, step-by-step approach while providing flexibility for variations in building projects. In Austria and Germany, the standard is not required by law but should help planners to design, install, operate and maintain BASs. Tasks are described in textual form and guide the user through the phases of the process. Furthermore, the required information output and format of each task are discussed.
For example, at the end of the *Design documents and technical specification* task within the *Design phase*, a technical documentation which contains *"all relevant documents to*

fully detail the requirements" of a BAS project should be provided. The content of these documents is outlined in part 1 of the ISO 16484 standard. [6, 90]
It is advised to utilize templates to reuse designs of similar rooms and structures. As this is not mandatory, the process can still be used in building projects including numerous individual and different rooms, e.g. for residential building projects. [6]
The ISO 16484 process furthermore advertises the usage of function lists in combination with function schematics. These documents are created in the *Design phase* and can be used for project tenders, cost calculation and billing. In the subsequent *Engineering phase*, these documents are refined to describe the functionality of the system in more detail. Still, these documents are not intended to be a direct copy of an installed system. Instead, they are used as reference for system design in subsequent steps. [6]
Unfortunately, the process defined in part 1 of the ISO 16484 standard is not applicable for all building projects. Time constraints, budget considerations and other project dependencies may lead to the necessity of shortening the *Design phase* and starting the *Engineering phase* earlier than advised in the proposed process, similar to the system level design approach (see Sections 4.1.3 and 4.2.3). Tasks within the process are intended to be executed by experienced, established system planners and integrators. As an advantage, the detailed design of BASs in the *Engineering phase* is eased, as functional and non-functional requirements should be available after the *Design phase* in a structured way, e.g. in form of function schematics and lists. [9, 8]
According to [8], no tool support for the creation of function blocks, as provided in the ISO 16484 standard or the VDI 3813 guideline, is available. Therefore, function schematics and lists including these blocks have to be created manually without the aid of specified applications. [8, 32, 6]

4.2.2 Design automation analysis

Similar to the ISO 16484 process, DA approaches are based on a structured step-by-step process. The functional and non-functional requirements are gathered with the aid of tools, which provides an advantage by having requirements available in a machine-readable format. Depending on the underlying architecture, these requirements are stored in a way so they can be linked with actual functions of devices. Often, ontologies are used as form of digital representation. [7, 8, 91, 87]
The main goal is to design and configure a BAS automatically by utilizing the information gathered in the *Requirements engineering phase*. In fully automated approaches, a template database is used in order to create system designs based on the gathered functional and non-functional requirements. These modular templates represent the design of common use cases and locations such as areas, rooms, segments, etc. (e.g. standard-office room) and are created in advance of the *Design phase*. Suitable templates are selected based on the requirements and mapped together to create an abstract system design, which is subsequently used for designing a concrete, detailed system, including selected devices. During installation and the creation of BAS software, the previously created design is utilized. These steps can be fully automated and do ideally not require interference of a system planner or integrator. Therefore, the main advantages of this

approach are possible time and cost savings, as the manual effort in system design and configuration is reduced and already existing designs can automatically be reused in the form of templates. [7, 8, 9]
Unfortunately, the reliance on such functional templates also includes a big disadvantage: inflexibility. This approach is not suitable for building projects with many differing rooms. For example, typical residential building projects would provide a problem, as rooms differ in functional requirements and size, and therefore have to be planned individually. Using a DA approach, this would lead to extra effort that has to be put in the creation of templates, decreasing their benefits. If no template is available, it has to be created in order to use it and design the system automatically. This deficiency is also present in semi-automated approaches, in which the requirements engineering and the system design are done manually with the aid of tools while the BAS software creation process is automated. Using such methods, rooms with no templates are still planned individually and additionally stored in separate template databases, which can lead to possible extra effort in the *Design phase*. [92, 8, 7]
Furthermore, to automatically select and interlink individual devices, an expert has to compile and maintain a knowledge base during the requirements engineering process. In case of missing templates, this has to be done additionally ind the system *design phase*. Such expert knowledge in the field of KBE cannot be expected from sectoral planners of BASs. [92]

4.2.3 System level design approach analysis

In contrast to the process defined in the ISO 16484 standard and DA strategies, in system level design approaches the functionality of a system is not planned before the system is designed, as can be observed Figure 4.4.
Design decisions often have to be made too early in the *Design phase* and without the aid of structured function descriptions, which have been planned beforehand. This overlap of the *Design phase* and *Engineering phase* leads to interoperability and device selection issues for system planners. [8] Furthermore, the person who programs and configures the system software also has no access to a structured, machine-readable function description. The system integrator has to implement the software on basis of the previously created system design by deriving the functionality from available design documents. Sometimes, documented requirements may be available to ease this process. [8] Therefore, a waste of information is identified in this approach, as the functionality of the system is first planned but not documented by a system planner and afterwards derived by a system integrator from the system design (see Figure 4.4).
In system level design approaches, the transition between *Design phase* and *Engineering phase* happens earlier in the project lifecycle in comparison with the ISO 16484 process and DA approaches. [9, 6] This may help to reduce the cost early in BAS projects, e.g. in case of a denial of propositions or project terminations.
There are also other advantages in utilizing such *"chaotic"* strategies like the system level design approach, which do not follow norms or processes. Loosening the procedure, a system planner is provided with more flexibility in design decisions and can make these

decisions earlier in the lifecycle of BAS projects. [90] The system level design approach is applicable for buildings with various rooms of different size and functionality, as in such processes each room is planned individually. [90, 9]

4.3 Process selection

Based on the analysis of the approaches discussed in Section 4.2, the system level design approach is selected and used as basis for further refinement in this thesis. It is applicable for creating RAC system software for all types of buildings, including residential buildings with many individual rooms in contrast to the DA method. Furthermore, the sequence of execution is more flexible in comparison with the other strategies. The process defined in part 1 of the ISO 16484 standard is not always applicable in its advised form, due to time and cost constraints in BAS projects. As the tools used in DA approaches are not standardized and templates are required for automated system design, this approach is also not suitable for all RAC system projects, e.g. for residential building projects.
Project owners frequently drift towards a more open method like the system level design approach. Furthermore, buildings often contain separate automation systems for different assembly sections such as ventilation engineering, heating or lighting engineering. As discussed in Section 4.2, the flexibility in the sequence of task execution also brings along disadvantages, but the benefits are valued by project owners and therefore, such methods as the system level design approach are frequently utilized. [90, 9]
At the time of writing this thesis, no problem solution strategies for the information waste in system level design approaches are known to the author. As such flexible strategies are common, the system level design approach is used as basis for a new, generally applicable process for creating RAC system software in this thesis. Furthermore, concepts, methods and information used in other approaches such as DA strategies or the ISO 16484 process are used to identify problem solutions possibilities and alternative information sources (see Chapter 5).

Characteristic	**ISO 16484 process**	**Design automation**	**System level design approach**	**Importance & relevance for this thesis**
Applicability for buildings with rooms of various size and functionality	✓	X	✓	+++
Flexible sequence of tasks and their information output in the design phase	~	X	✓	+++
Semi-automated RAC system software creation	X	✓	X	++
Functionality not required before system design	X	X	✓	+
Applicable without tool-driven requirements engineering	✓	X	✓	+
Utilization of templates possible	✓	✓	X	+
No specific software required for system functionality planning	✓	X	✓	+

Table 4.2: Comparison of RAC system design and software creation approaches

CHAPTER 5

Generalized process for creating RAC system configuration

In this chapter, a generalized process for creating RAC system software is introduced. This process is based on the findings highlighted in Chapter 4. The utilization of gathered information is in accordance with step 4 of the methodological approach as illustrated in Figure 5.1. Step 5 of the methodology is also included in this chapter and is elaborated in Section 5.5.

5.1 Towards a generalized approach

The incentive of this thesis is to automatically create RAC system software. As discussed in Section 4.3, a typical system level design approach is taken as template-process in order to satisfy this goal. From a process analysis perspective, already created information about the functionality of a system is wasted in the system level design approach. One expert designs the system to support the required functionality, while another expert has to derive the same functionality again by relying on provided technical documents, know-how and experience.
Therefore, the system level design approach, as shown in Figure 4.4, is adapted in order to get rid of this information waste and execute the necessary steps for a successful completion of the tasks *Derived system functionality* and *Automation software*. A new task *RAC system description* is introduced, which is performed instead of the *Derived system functionality* step. Instead of manual functionality derivation, which is executed by system integrators in common system level design approaches, the functionality should be derived from knowledge and documents which are available after the RAC system has already been designed. An overview of the adapted approach is illustrated in Figure 5.2. As can be seen in the figure, the *RAC system description* task processes all information

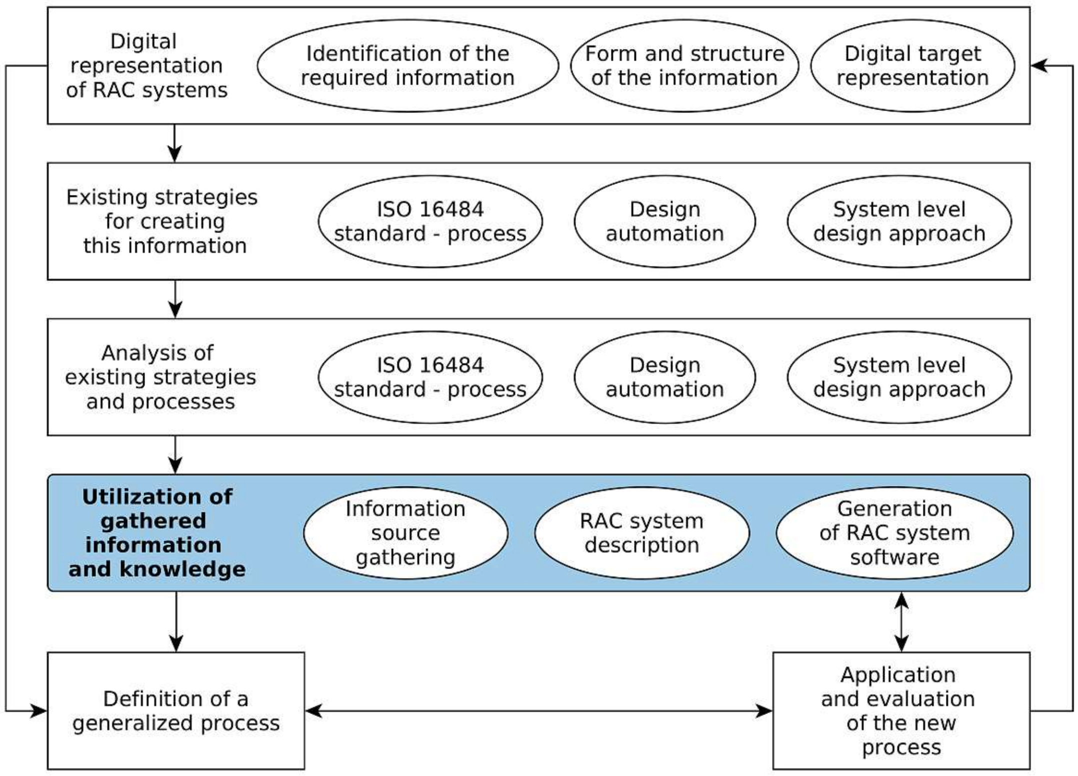

Digital representation of RAC systems
Identification of the required information
Form and structure of the information
Digital target representation
Existing strategies for creating this information
ISO 16484 standard - process
Design automation
System level design approach
Analysis of existing strategies and processes
ISO 16484 standard - process
Design automation
System level design approach
Utilization of gathered information and knowledge
Information source gathering
RAC system description
Generation of RAC system software
Definition of a generalized process
Application and evaluation of the new process

Figure 5.1: Part 4 of the methodological approach

that is available at this point in order to allow a derivation of the intended functionality of RAC systems. A detailed description of this task is outlined in Section 5.3.

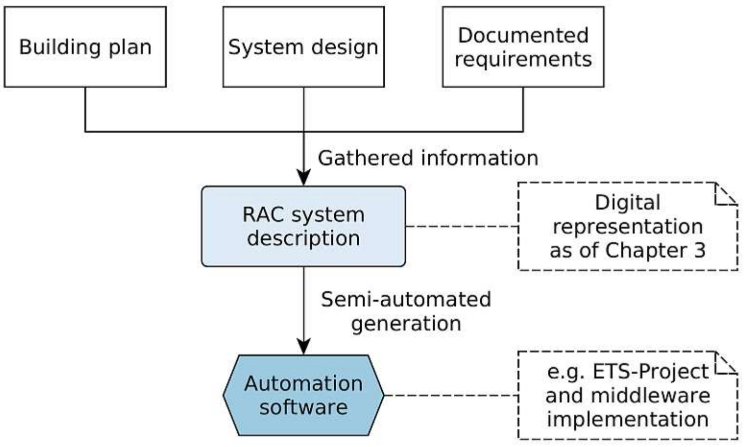


Figure 5.2: Overview of the adapted system level design approach

5.2 Information source gathering

In this section, possible information sources which are used to cover the required aspects discussed in Section 5.1 are examined. Adaptions and changes in the structure and form of the information are outlined.

5.2.1 Building plan information

All three approaches discussed in section 4.1 rely on building structure information. The process defined in part 1 of the ISO 16484 standard and system level design approaches do not require a specific form or representation of building plans.
In the *Determination of project requirements* task of the ISO 16484 process, it is advised to gather the following information about BAS project buildings [6]:

- Building structure, e.g. size, metal frame, concrete, etc.
- Type of building(s), e.g. high rise, open campus, tunnel etc.
- Building usage type and profile, e.g. hospital, commercial, residential, etc.
- Space usage profile, e.g. occupation schemes, diverse usage of space, etc.

This knowledge influences the decisions of system planners when designing BASs, including the functionality, data points and devices. [6]

5.2.1.1 Locations

In DA approaches, infrastructural data such as building layouts are required. Layout information up to the granularity level of rooms, according to the multilayer-model defined in the VDI 3814 guideline (see Figure 3.10), should be available. Identifiable buildings, areas and rooms are required, which are usually available through floor plans. The planning of segments is not necessarily required but could help to structure rooms. [92]
Computer aided (architectural) design (CAD or CAAD) applications are the most common tools for creating digital building plans. Programs which only export older AutoCAD file formats such as *.dwg* and *.dxf* are not suitable for automatically deriving the layout of buildings and have to rely on extra conversion or manual structuring effort. [9]
On the other hand, building plans available in the industry foundation classes (IFC) file format *.ifc* or the extensible markup language (XML)-based version *.ifcXML* are sufficient to derive building layouts at the granularity level of rooms as defined in the multilayer-model. A mapping between classes of IFC format and layers in the multilayer-model as well as an approach to automatically derive the layers has already been established. The mapping between IFC classes and building structure layers is illustrated in Figure 5.3. Models extracted from IFC files are manually extended with segments in order to get a more precise allocation of elements. [9] Real-estate portfolio locations can also be manually added, but are not required for the purpose of this work.

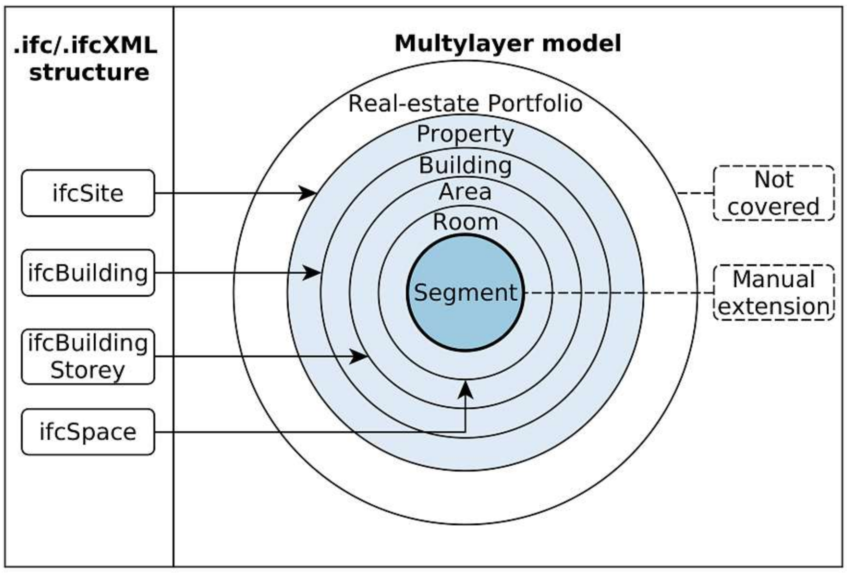


Figure 5.3: Creating a building structure according to the VDI 3814 multilayer-model using IFC files. [9]

5.2.1.2 Device locations in building plans

The location of devices is determined during the design of BASs. In the process defined in part 1 of the ISO 16484 standard and in DA approaches, the functionality of systems is planned beforehand. At the end of the *Design phase*, the physical location of system items should be available. In both the ISO 16484 process and DA approaches, devices are selected and placed based on basis of the structure of buildings, functional and non-functional requirements. [6, 7, 92] The device selection process is described in Section 5.2.3.1. Both approaches rely on building and floor plans for placing devices during the design of BASs. [6, 7, 92]
IFC supports modeling field aspects, especially field level features, of BASs. This includes wiring, equipment details and location. By relating devices to spatial location elements, they can be located within IFC models. Furthermore, a set of properties can be defined over device objects to create equipment details. In addition, *ifcPort* elements which are related to devices can be used in order to model wiring between device objects. [30]
Progress has been made in integrating BASs elements in BIM/IFC models, but there is still a lack of support. By enriching BIM models with BAS information, facility managers can contract system alterations, hire specialists and use third-party tools and technologies more easily, as they are not bound to the hardware vendors which design and install an entire BAS. Furthermore, information in BIM models is easier to grasp for building managers than a typical documentation of BASs, which contains complex and context specific information such as network typologies, device configurations or control loops created by BAS development tools. [30]

Preferably, the whole lifecycle of a building should be supported by BIM tools, including the architecture, engineering, construction and operation phases. Unfortunately, BIM still mainly focuses on the design and construction phase, disregarding the operational phase including modeling and sharing BA element information. This is due to the fact, that CAD software vendors focus on the design phase of buildings, influencing and restricting the development possibilities of BAS elements in BIM digital models. Furthermore, BAS information is more volatile and often subjected to change than constructive elements of buildings. As mentioned above, IFC provides the possibility to model device locations, equipment information and wiring relations, but logical and operational information is not supported in the current state. [30]
In summary, progress towards modeling BAS devices in BIM models has been made, but there is still a lot of work to be done in order to integrate BAS technology in BIM models. IFC provides the possibility to model devices and relate them to locations within a building. However, BAS devices and equipment are rarely modeled in IFC due to design application constraints and the focus on the building construction phase in BIM modeling.

5.2.2 Documented requirements

Information about the functional and non-functional requirements of BASs is needed in order to plan and design them. These requirements provide knowledge about specific functional requests as well as non-functional aspects, such as the operating voltage or mounting form of devices. [8]
DA approaches rely on knowledge-based requirements engineering. With the help of software tools, system planners, integrators and building owners can inquire requirements in a context-sensitive, knowledge-based process. In most approaches, the knowledge is gathered in a domain-specific ontology. [7, 8, 93, 87] This requires the intervention of experts, who can compile and maintain knowledge bases during the requirements engineering step, which is not expected to be within the skill set of all participants in BAS projects. [92]
Requirements are grouped in form of templates or patterns to prevent duplicate definitions of identical rooms. These templates are later on used to create abstract design templates, which are similar to function schematics as stated in part 2 of the VDI 3813 guideline and describe the functionality of the system. Generative programming techniques are used to create these designs. Thereby, a generator executes a generic construction plan to create such abstract design templates on basis of functional requirements specified in a knowledge base and abstract function blocks. This template creation process is illustrated in Figure 5.4. Available non-functional requirements are later on used when creating a detailed system design with included devices. They influence the device selection process by restricting the range of suitable devices through properties such as mounting form, platform or supported transmission medium. [7, 8, 87]

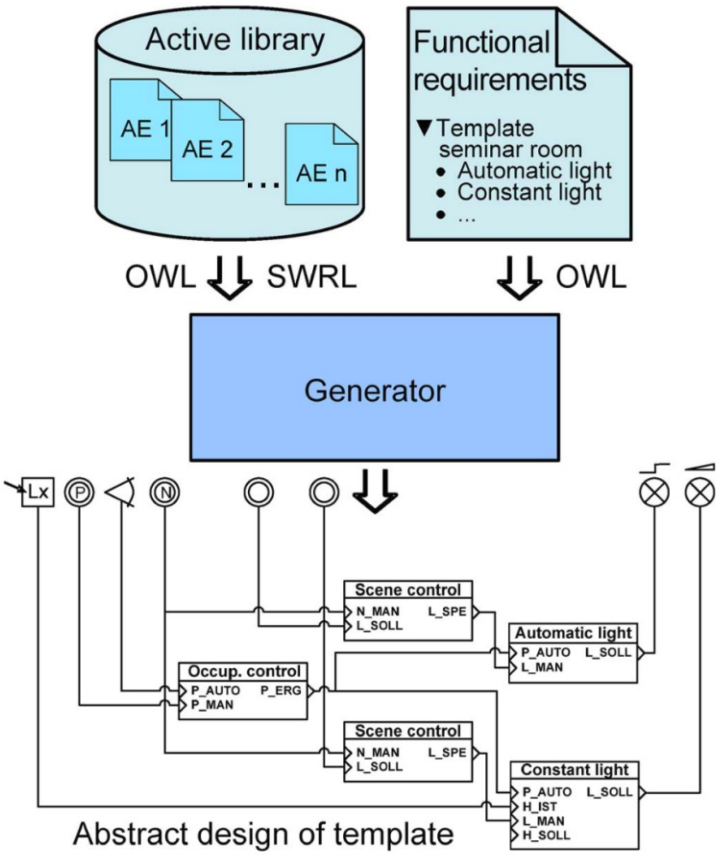


Figure 5.4: Generation of abstract design templates; Figure taken from [7]

Following the process defined in part 1 of the ISO 16484 standard, functional descriptions including the sequence of operation as text or diagram should be delivered at the end of the *Design phase*. Furthermore, function lists should be provided in order to prevent duplication of engineering in later, more detailed designing works. [6]
To create these lists, the following requirement categories are considered according to the process outlined in part 1 of the ISO 16484 standard [6]:

- General considerations, e.g. building structure, space usage, energy supply requirements, etc.
- Integration requirements, e.g. integration, operation, function, infrastructure, etc.
- Physical requirements
- Occupational requirements
- System requirements
- Site and client specific requirements

This list includes functional and non-functional requirements which are essential for the resulting design of a BAS. No recommended form of representation for these requirements is stated. [6] It is unclear, in which level of detail and structure the requirements are delivered to system planners and integrators. Therefore, the requirements created following the ISO 16484 process cannot be relied on as a single source of information for deriving the functionality of designed BASs.

5.2.3 System design information

This block of information includes all documents and plans, which have been created in advance to the RAC system configuration and installation steps during the *design phase* of projects.
In system level design approaches, this information is typically created without the use of structured and documented function descriptions such as function lists and schematics. Instead, the envisioned functionality is implicitly influencing the design of the system and later derived from design documents by system integrators. Among other sources, device lists and information about the functionality and location of devices as well as other documents such as wiring plans can be included in these design documents. [8]
According to part C of the DIN EN 18386 standard, a client has to deliver a number of documents to the contractor, including function lists and schematics compliant with part 2 of the VDI 3813 standard. In addition, function flow diagrams and descriptions, a compilation of setpoints, thresholds and operating times, a concept for addressing and other information have to be provided. Contractors have to use this information in order to create automation function schematics, including the essential functions of BASs in accordance with the designed system. Furthermore, circuit diagrams compliant with the DIN EN 61082-1 standard have to be delivered. Similar to the ISO 16484 standard, the DIN EN 18386 standard for German construction contract procedures is not legally binding. [94] It can be used as reference and helps to create contracts, which are suitable for projects in the BAS domain. Therefore, it cannot be expected that all documents and information stated as deliverables in the DIN EN 18386 standard are available in all BAS projects.

5.2.3.1 Device list

In DA approaches, manufacturer-specific devices are selected on basis of evolutionary algorithms. The device selection takes place after an abstract design of the desired system has been created (see Section 5.2.2). In order to automatically select interoperable devices and compose them for multi vendor solutions, knowledge about their functionality and properties is required. This information is provided by so-called *semantic device descriptions*, which are available through a knowledge-based component repository. In Section 5.2.3.2, the functional representation of devices and the application of this knowledge is discussed in more detail. Ideally, lists of devices, including semantic device descriptions, are provided by device manufacturers. The device selection process in DA approaches is not trivial, as a large number of device combinations can be valid.

Therefore, machine-readable functional and non-functional information about devices is critical for limiting the amount of possible design solutions. [7, 8]
In DA approaches, a list of all devices of a system can be found in the knowledge-base representing the RAC system after the detailed system has been designed. [7]
In course of the process defined in part 1 of the ISO 16484 standard, an equipment list is created in the *"Hardware configuration"* step within the *"Detailed hardware and function design"* task. However, the standard does not define a format and the level of detail, in which the selected BAS equipment has to be represented. [6]

5.2.3.2 Functionality of devices

Knowledge about the functionality and profiles of devices is necessary in order to map to abstract designs consisting of function blocks as utilized in DA approaches. This required semantic knowledge about devices includes the profiles and their specific functions, e.g. automatic light or constant light. Furthermore, knowledge about their parametrization, connectivity, their intended usage and the purpose of their input and output data points is needed. Device repositories and knowledge bases have been developed in order to represent this information in a machine-readable format. [7, 95]
This was necessary, as existing manual device descriptions, used by system integrators, are in natural language and cannot be used for automatic selection. [7] Such electronic device descriptions are often available in machine-readable form, e.g. EDDL, IODD, GSDML, CANopen, FDCML, XIF or FDI [96]. They include information about the interface of devices but do not provide knowledge about their specific functionalities. [7]
Semantic device descriptions include functional properties which describe semantics the of profiles. Such functions are represented in the same form as in abstract designs created within DA approaches. Thus, they can be mapped with each other on basis of the supported functions of devices and the function blocks used in the abstract design. Furthermore, knowledge about non-functional properties, such as mounting form, operating voltage or the supported transmission medium, is provided by semantic device descriptions. [7]
In addition to semantic device descriptions, semantic types are assigned to data points. They *"give a precise meaning to input and output data points, far beyond standardized variable types"*. Semantic types are used to analyze the interoperability of profiles by creating semantically correct connections between them.[7]

5.2.3.3 Device locations in system design documents

Device installation instructions are created in the *Engineering phase* as stated in part 1 of the ISO 16484 process. The placement and wiring of equipment is in accordance with submittal documents created earlier in the process. Furthermore, BAS equipment and cables should be tagged in compliance with the specification. There is no advice given, in which form or structure the tagging and localization of the equipment has to be documented. [6]

The IEC 81346 standard provides structuring principles and reference designations for objects which are used in BASs. In this standard, three topologies are defined, containing structural allocation of objects based on locations, functional assignments and product based distinctions. [94] Dependent on the utilization of reference designations in design documents of BAS projects and the level of detail, equipment and devices can be located within buildings, allowing to create a *hasLocation* object relation in accordance with the digital representation introduced in Chapter 3. Though, knowledge about a possible *influencesLocation* relation cannot be derived from reference designations.
In some BA projects, common CAD tools are used to create system design documents to illustrate the placement of devices within rooms and buildings. [8] This approach and the challenges it involves is discussed in more detail in Section 5.2.1.2.

5.2.3.4 Wiring and P&ID plans

In the process defined in part 1 of the ISO 16484 standard, submittals containing system plans are prepared during the *"detailed hardware and function design"* task. These documents include system architecture plans and system descriptions, optional data point and BAS function lists, the sequence of operation as diagram or text representing updated function descriptions, and other additional, required documents. [6]
A part of this *"detailed hardware and function design"* task is the *"hardware configuration"* step, in which field wiring plans and other schematics are created. Device, system and plant connectivity planning, cable type selection and terminal identification are also tasks which are executed and documented in this step. An input and output allocation, a network plan including addressing, setup, consideration of existing networks and selection of network devices as well as an equipment list are also created. The ISO 16484 standard does not propose a definitive format in which the information should be provided. [6] Therefore, the availability, form and structure of the documents and information created in this task are uncertain and depend on the context of the project, tools and the software that has been used.
Among other aspects, DA approaches have been developed because information such as installation plans often get lost or needs to be created multiple times due to badly organized processes. In such approaches, plans, device lists, function descriptions and other important information are exchanged and shared between different project phases using the same knowledge base and digital representation to minimize information loss. In contrast, system level design style approaches rely on common CAD tools to plan the physical placement of wires and devices. [8]

5.3 RAC system description

Similar to DA approaches, information about the structure of buildings and documented requirements are taken under consideration for creating and deriving the functionality of RAC systems in this thesis. [7] In addition, system design documents are also used as source of information, as it is assumed that the system has already been designed

when applying the new process defined in Section 5.5. In other words, the task *RAC system description* utilizes the knowledge provided by the information sources *Building plan*, *Documented requirements* and *System design* in order to derive the functionality of RAC systems automatically. The derived functionality is stored in form of individuals and relations in a new, project specific ontology, which imports the digital representation as presented in Chapter 3.
Therefore, a new workflow has been developed which utilizes the information in a structured, generally applicable way. This workflow is illustrated in Figure 5.5 and discussed in the following. A step-by step example execution of this workflow is outlined in Section 6.1.

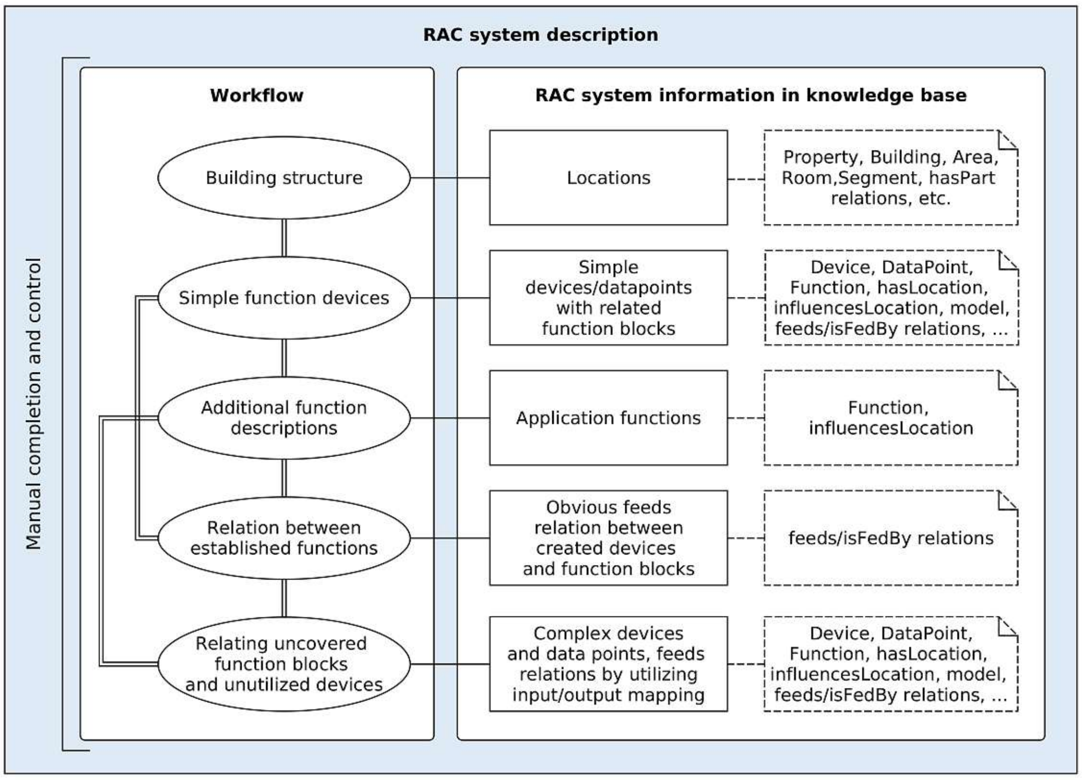

RAC system description
Manual completion and control
Workflow
RAC system information in knowledge base
Building structure
Locations
Property, Building, Area, Room,Segment, hasPart relations, etc.
Simple function devices
Simple devices/datapoints with related function blocks
Device, DataPoint, Function, hasLocation, influencesLocation, model, feeds/isFedBy relations, ...
Additional function descriptions
Application functions
Function, influencesLocation
Relation between established functions
Obvious feeds relation between created devices and function blocks
feeds/isFedBy relations
Relating uncovered function blocks and unutilized devices
Complex devices and data points, feeds relations by utilizing input/output mapping
Device, DataPoint, Function, hasLocation, influencesLocation, model, feeds/isFedBy relations, ...

Figure 5.5: Workflow for creating RAC system descriptions

5.3.1 Building structure

At first, the structure of buildings equipped with RAC systems is added to the digital representation. This is done by utilizing the information which is provided by building plan information sources as described in Section 5.2.1. After executing this step, a building structure compliant with the multilayer-model defined in the VDI 3814 standard should be available in the digital representation ontology in form of locations. An example for creating such a building structure is outlined in Section 6.1.3.

5.3.2 Simple function devices

As next step in the workflow, devices, which have a clear purpose and therefore only support a single, trivial function, are examined. In the following, these devices are referred to as *simple function devices*. Only actuator, sensor and operator/display functions are of relevance for this purpose. Other functions, such as application and management functions, are ignored at this step. The devices are identified by scanning device listss which have been created during the design of the system by a system planner. In addition, information about the model of listed devices has to be available. This knowledge is again used to identify the functions a device supports. This is done by looking up the model in a knowledge base which contains semantic device descriptions, as described in Section 5.2.3.1.
The only function which is supported by simple function devices is added as individual to the ontology and automatically linked with the associated device using a *feeds* or *isFedBy* relation, either if the device acts as a sensor or an actor. Associated connection points and types are either taken from available documents such as wiring, PI&D or building plans, or may be apparent from device lists and electronic or semantic descriptions of devices and models.
Next, devices are optionally associated with locations using the *hasLocation* property. This information may be taken from building structure plans as described in Section 5.2.1, from system design plans including reference designations as outlined in Section 5.2.3.3 or it has to be manually added to the ontology. Furthermore, an *influencesLocation* relation is established with the device. If this information is not available from documented requirements in form of textual, interpretable descriptions or structured, machine-readable requirements, it has to be manually added to the digital representation. This approach is illustrated in Figure 5.6.

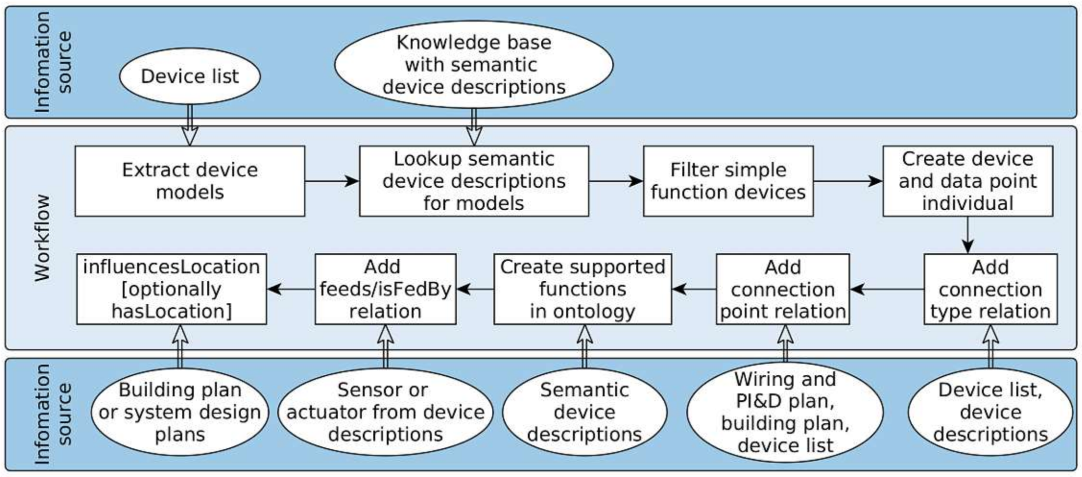

Infomation source
Device list
Knowledge base with semantic device descriptions
Workflow
Extract device models
Lookup semantic device descriptions for models
Filter simple function devices
Create device and data point individual
Add connection type relation
Add connection point relation
Create supported functions in ontology
Add feeds/isFedBy relation
influencesLocation [optionally hasLocation]
Infomation source
Building plan or system design plans
Sensor or actuator from device descriptions
Semantic device descriptions
Wiring and PI&D plan, building plan, device list
Device list, device descriptions

Figure 5.6: Workflow for adding simple function devices to the digital representation

5.3.3 Additional function descriptions

After identifying all simple function devices and adding them to the knowledge base, more complex application functions, which describe the functionality of RAC systems in more detail, are covered. These functions are used to satisfy functional as well as non-functional requirements of such systems. Documented requirements, e.g. in form of textual function descriptions available through requirements specification books, can be a source of information about the desired application functions of the system. They may be used to derive VDI compliant function blocks, which represent the desired functionality with the aid of text interpreters. After adding the new function blocks, these blocks are associated with locations using the *influencesLocation* object property. At this point, the functions are not connected to each other or with devices using *feeds* relations. An example for creating and adding such application functions is outlined in Section 6.1.5.

5.3.4 Relation between established functions

In order to describe the functionality of a system, the interaction and relation between separate function blocks has to be established. Therefore, the input requirements of all functions identified in the previous steps are examined. At first, function blocks associated with simple function devices are analyzed to determine possible relations with application functions. This is done by checking the compatibility of input and output variables.
Therefore, already established knowledge bases which implement function blocks defined in the VDI 3813 guideline can be imported by the new digital representation ontology. They provide information about the input and output variables as well as parametrization properties of functions, which are not modeled in the digital representation introduced in Chapter 3. Thus, already available information can be reused and a remodeling of

the same concepts and information is avoided. To utilize this method, the entities of an imported ontology have to match the representation ontology which is created in this thesis.
AutomationML is an example for such a knowledge base, which includes a comprehensive function library compliant with the function block definitions stated in the VDI 3813 guideline. [97]
Furthermore, the locations which are influenced by the function blocks are taken under consideration when creating *feeds* relations. Relations between function blocks within the same location or the next higher level are preferred. If the same information is provided by different function blocks within the same location, the user manually selects the preferred block which should be used to establish the *feeds* relation.
After the process has been executed for all available function blocks, the same approach is undertaken to check the input and output variables for established application functions in order to identify obvious relations between them. An example for creating such relations according to this step in the workflow is stated in Section 6.1.6.

5.3.4.1 Relating uncovered function blocks and unutilized devices

After executing the previous steps of the proposed process, there still remain function blocks with open input- and output variables. Therefore, unutilized devices are examined to relate them with still uncovered variables of established function blocks.
First, all function blocks which have unconnected input- and output variables are observed. Variables that are not covered are examined to find relations with devices, which have unused functional capabilities. This method is similar to the workflow for relating functions as described in Section 5.3.4. Devices which influence the same location or the next higher level according to the multilayer-model are observed first. If a device provides the functionality required by an uncovered input or output of a function block, a *feeds* relation between the two elements is established. If a relation between a function block and device cannot be created because there is no distinct solution, the relation has to be created manually by selecting from a set of suggested choices. This procedure is described in more detail using an example in Section 6.1.7.

5.3.4.2 Manual completion and control

It may be possible, that some input- and output variables as well as parameters of function blocks are still not covered after completing the previous steps of the process, because relations could not be automatically established or manual editions have been skipped. Furthermore, the overall functionality of the system is controlled on basis of agreed functional and non-functional requirements. Possible missing functionality or deviations from the intended purpose of the RAC system are manually added and fixed in this step. An example for solving such problems is described in Section 6.1.8. The manual completion and control procedure can be conducted during the whole workflow to identify possible failures as early as possible.

5.4 Generation of RAC system software

Based on the digital representation, which is created following the workflow outlined in Section 5.3, a RAC system software is generated on basis of established rules and concepts, as visualized in Figure 5.7.

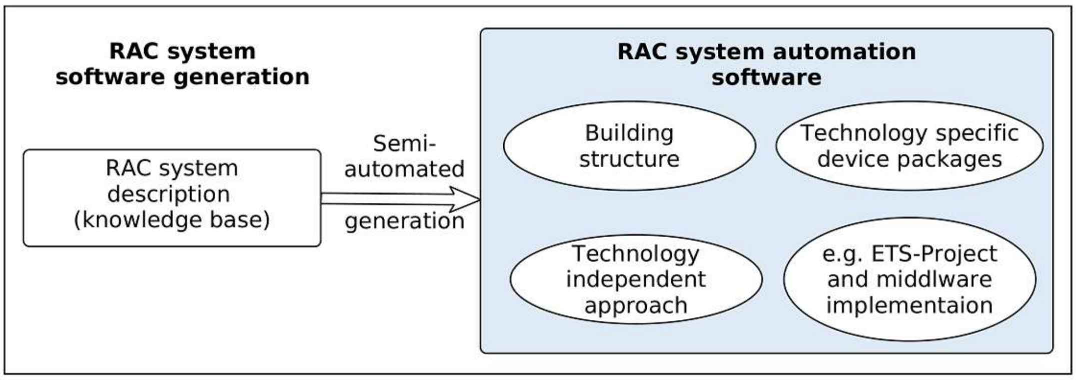
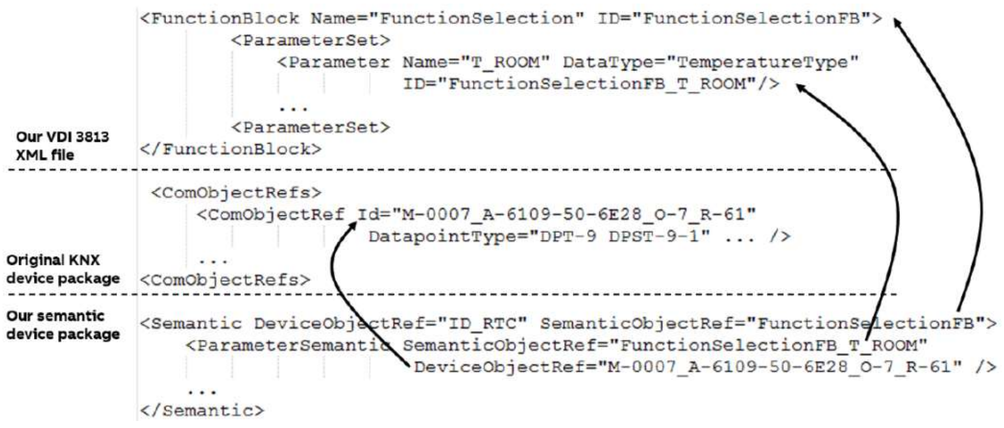

RAC system software generation
RAC system description (knowledge base)
Semi-automated generation
RAC system automation software
Building structure
Technology specific device packages
Technology independent approach
e.g. ETS-Project and middlware implementaion

Figure 5.7: Overview - RAC system software generation

Information contained within the knowledge base has to be mapped and processed in order to create RAC system software. There exist generators which utilize VDI 3813 compliant function schematics based on XML-files in order to create KNX- and ZigBee-specific configurations. [98] As the information provided in the digital representation ontology established Chapter 3 of this thesis is also compliant with VDI 3813 function schematics, this procedure can be applied to create RAC system software.
As long as a technology-specific generator for creating RAC system software is available, this approach is generally applicable for all RAC system technologies. Using *semantic device packages*, a mapping between VDI 3813 function blocks and technology-specific devices can be created. It is assumed, that such packages are provided by device manufacturers in future. These packages contain information about the input and output ports of devices, their internal modules, device communications, etc. An example for such a semantic device package, based on KNX, is illustrated in Figure 5.8. This example utilizes a VDI 3813 XML-file, a semantic device package XML file and a KNX prod-file in XML format, which represents original KNX device packages. In this example, an input variable *T_Room*, which is part of the VDI function block *Function Selection*, is mapped to a KNX specific communication object with the reference identifier *M-0007 A-6109-50-6E28 O-7 R-61*. [98]



```
<FunctionBlock Name="FunctionSelection" ID="FunctionSelectionFB">
        <ParameterSet>
            <Parameter Name="T_ROOM" DataType="TemperatureType"
                       ID="FunctionSelectionFB_T_ROOM"/>
            ...
        <ParameterSet>
</FunctionBlock>
```

```
 <ComObjectRefs>
     <ComObjectRef Id="M-0007_A-6109-50-6E28_O-7_R-61"
                 DatapointType="DPT-9 DPST-9-1" ... />
     ...
<ComObjectRefs>
```

```
<Semantic DeviceObjectRef="ID_RTC" SemanticObjectRef="FunctionSelectionFB">
    <ParameterSemantic SemanticObjectRef="FunctionSelectionFB_T_ROOM"
                       DeviceObjectRef="M-0007_A-6109-50-6E28_O-7_R-61" />
    ...
</Semantic>
```

Figure 5.8: KNX semantic device package example [98]

An example architecture for semantic-based engineering to create ETS/KNX projects and software is visualized in Figure 5.9. [98] Using the process described in this thesis, the information generated by the *Configuration Generator* illustrated in this figure is provided by the digital representation ontology containing the necessary information for describing RAC systems. A detailed description for creating RAC system software based on ETS/KNX projects based on this approach is stated in Section 6.1.9.

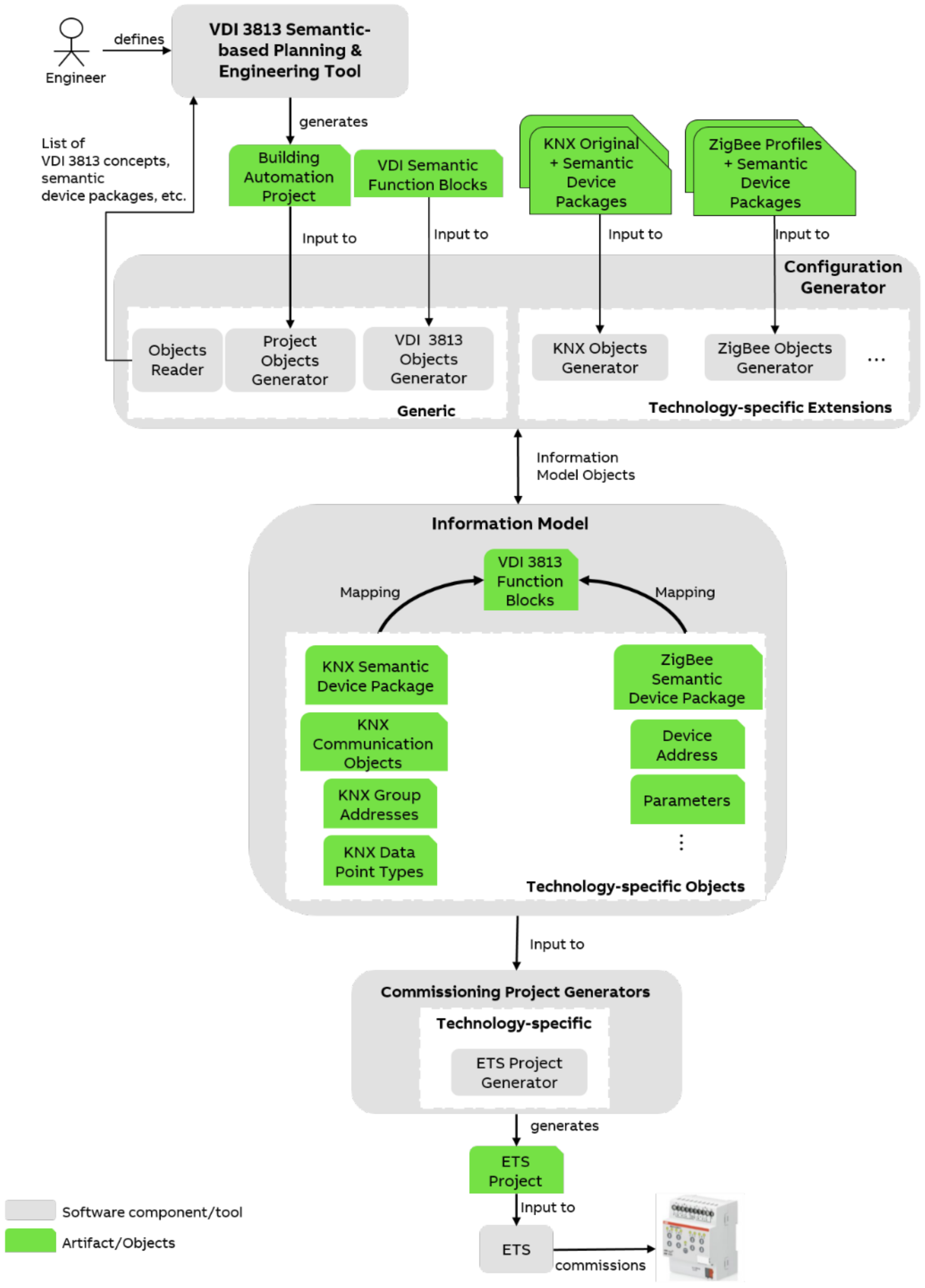

Engineer
defines
VDI 3813 Semantic-based Planning & Engineering Tool
generates
List of VDI 3813 concepts, semantic device packages, etc.
Building Automation Project
VDI Semantic Function Blocks
KNX Original + Semantic Device Packages
ZigBee Profiles + Semantic Device Packages
Input to
Input to
Input to
Input to
Configuration Generator
Objects Reader
Project Objects Generator
VDI 3813 Objects Generator
Generic
KNX Objects Generator
ZigBee Objects Generator
...
Technology-specific Extensions
Information Model Objects
Information Model
Mapping
VDI 3813 Function Blocks
Mapping
KNX Semantic Device Package
KNX Communication Objects
KNX Group Addresses
KNX Data Point Types
ZigBee Semantic Device Package
Device Address
Parameters
Technology-specific Objects
Input to
Commissioning Project Generators
Technology-specific
ETS Project Generator
generates
ETS Project
Input to
ETS
commissions
Software component/tool
Artifact/Objects

Figure 5.9: Semantic-based engineering architecture example [98]

5.5 Definition of a generalized process

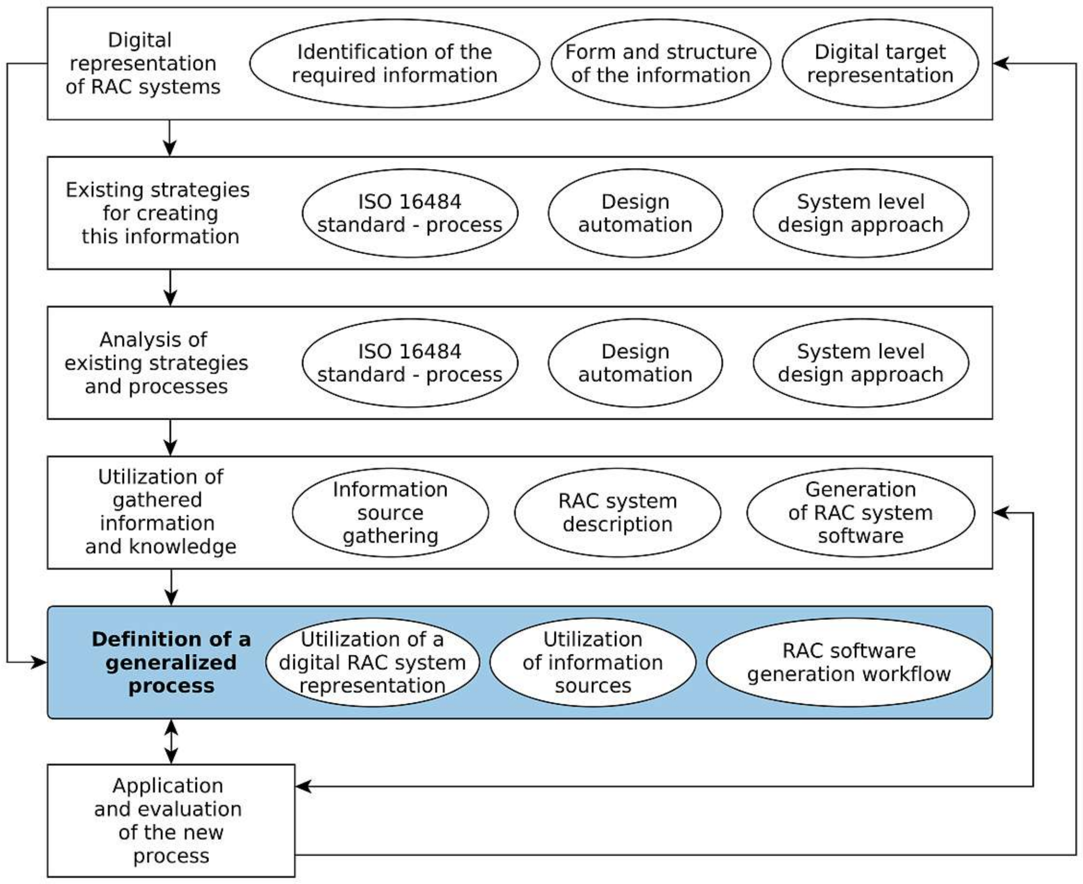

Digital representation of RAC systems
Identification of the required information
Form and structure of the information
Digital target representation
Existing strategies for creating this information
ISO 16484 standard - process
Design automation
System level design approach
Analysis of existing strategies and processes
ISO 16484 standard - process
Design automation
System level design approach
Utilization of gathered information and knowledge
Information source gathering
RAC system description
Generation of RAC system software
Definition of a generalized process
Utilization of a digital RAC system representation
Utilization of information sources
RAC software generation workflow
Application and evaluation of the new process

Figure 5.10: Part 5 of the methodological approach

In this section, a generally applicable process, which utilizes the previously identified information sources to create RAC system software is proposed. This corresponds with part 5 of the methodological approach as illustrated in Figure 5.10. The process is based on the adapted system level design approach described in Section 5.1 and combines the findings described in Sections 5.2, 5.3 and 5.4. The defined process is illustrated in Figure 5.11.

In the first part, *Information source gathering*, information sources which provide valuable knowledge for describing RAC systems and their functionality are identified and gathered. Promising information sources and considerations regarding their applicability are outlined in Section 5.2.
This information is processed and utilized in the second part of the process, *RAC system description.* In Section 5.3, a workflow is introduced, that allows to create an ontology

which represents the functionality and other required aspects of RAC systems to generate RAC system software. This ontology is compliant with the digital representation defined in Chapter 3. The workflow focuses on available information sources to automatically generate the RAC system description based on the required structure.
In the last step of the process, *RAC system software generation*, a RAC system software is generated on basis of the digital representation ontology created in the previously. Following rules and technology specific device packages, the software is generated using a semantic-based engineering approach. This task is described in more detail in Section 5.4.

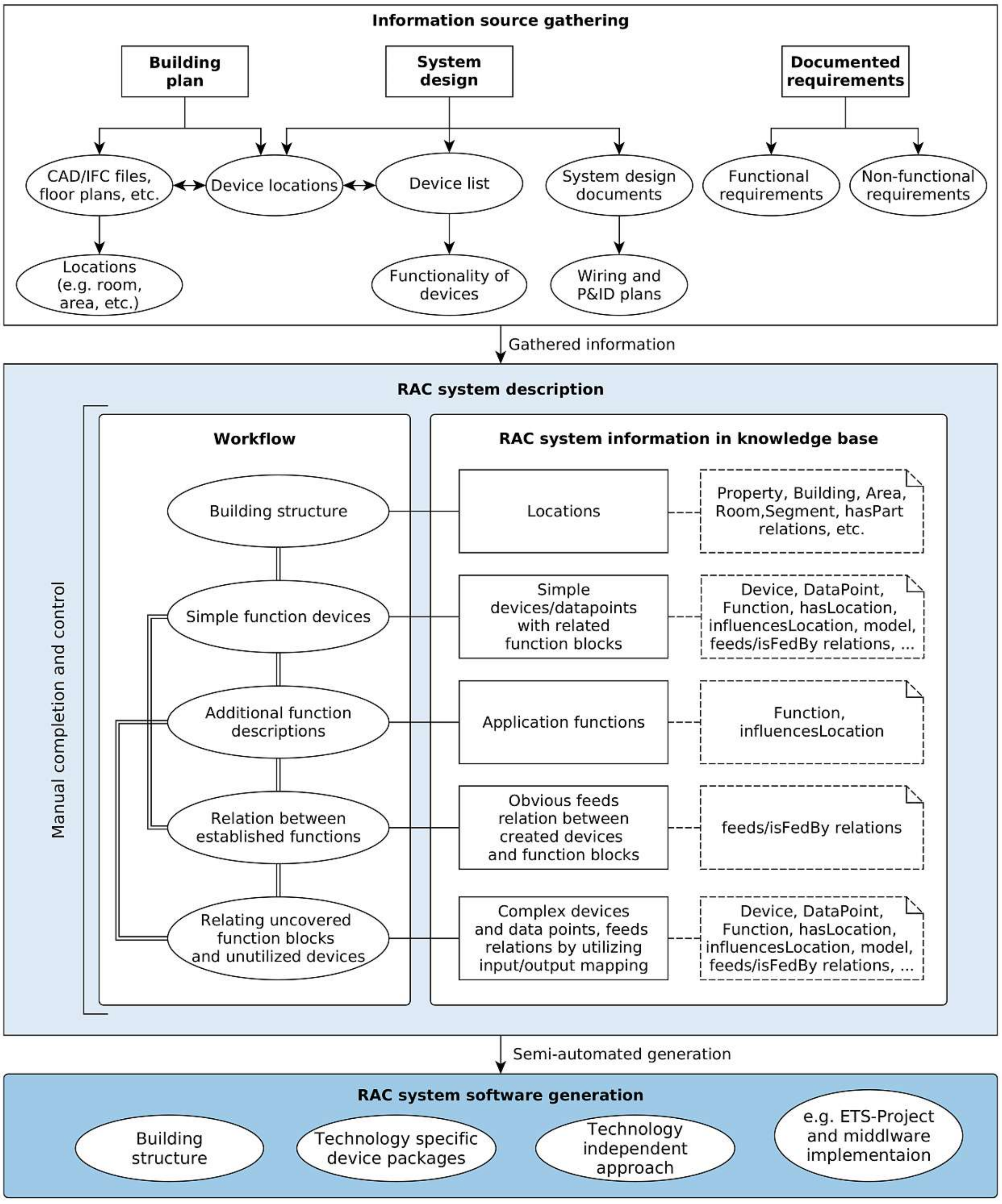

Information source gathering
Building plan
System design
Documented requirements
CAD/IFC files, floor plans, etc.
Device locations
Device list
System design documents
Functional requirements
Non-functional requirements
Locations (e.g. room, area, etc.)
Functionality of devices
Wiring and P&ID plans
Gathered information
RAC system description
Manual completion and control
Workflow
Building structure
Simple function devices
Additional function descriptions
Relation between established functions
Relating uncovered function blocks and unutilized devices
RAC system information in knowledge base
Locations
Property, Building, Area, Room,Segment, hasPart relations, etc.
Simple devices/datapoints with related function blocks
Device, DataPoint, Function, hasLocation, influencesLocation, model, feeds/isFedBy relations, ...
Application functions
Function, influencesLocation
Obvious feeds relation between created devices and function blocks
feeds/isFedBy relations
Complex devices and data points, feeds relations by utilizing input/output mapping
Device, DataPoint, Function, hasLocation, influencesLocation, model, feeds/isFedBy relations, ...
Semi-automated generation
RAC system software generation
Building structure
Technology specific device packages
Technology independent approach
e.g. ETS-Project and middlware implementaion

Figure 5.11: Generalized process for creating RAC system software

CHAPTER 6

Evaluation

In this chapter, the applicability of the generalized process defined in Section 5.5 is assessed based on fictional example. Furthermore, the process is evaluated in comparison with other, common approaches for RAC system software creation, which are introduced and discussed in Chapter 4. Deficiencies of the chosen digital representation, the defined generalized process as well as possible solutions are discussed. This chapter represents part 6 of the methodological approach as illustrated in Figure 6.1.

6.1 Applicability of the generalized process

An example RAC system is introduced in this section to evaluate the applicability of the process and workflow defined in Section 5.3. Therefore, each step of the workflow is executed and the resulting output examined. An exemplary digital representation as defined in Chapter 3 and a RAC system software based on KNX using the software tool *ETS* are created as a consequence.

6.1.1 VDI 3813-2 example

For proving the applicability of the newly defined process, an example RAC system description stated in part 2 of the VDI 3813 guideline is used. In this example, a function schematic and a function list describe the functionality of the RAC system (see Figures 3.5 and 3.6). This description serves as the target representation which should be created when executing the generalized workflow introduced in Section 5.3.

First, a new ontology which imports the digital representation knowledge base introduced in Chapter 3 is created. Only individuals and relations are added to this ontology. If entities are insufficient or missing, they are added to the base ontology which is imported. This strategy allows creating multiple ontologies for different projects and buildings based

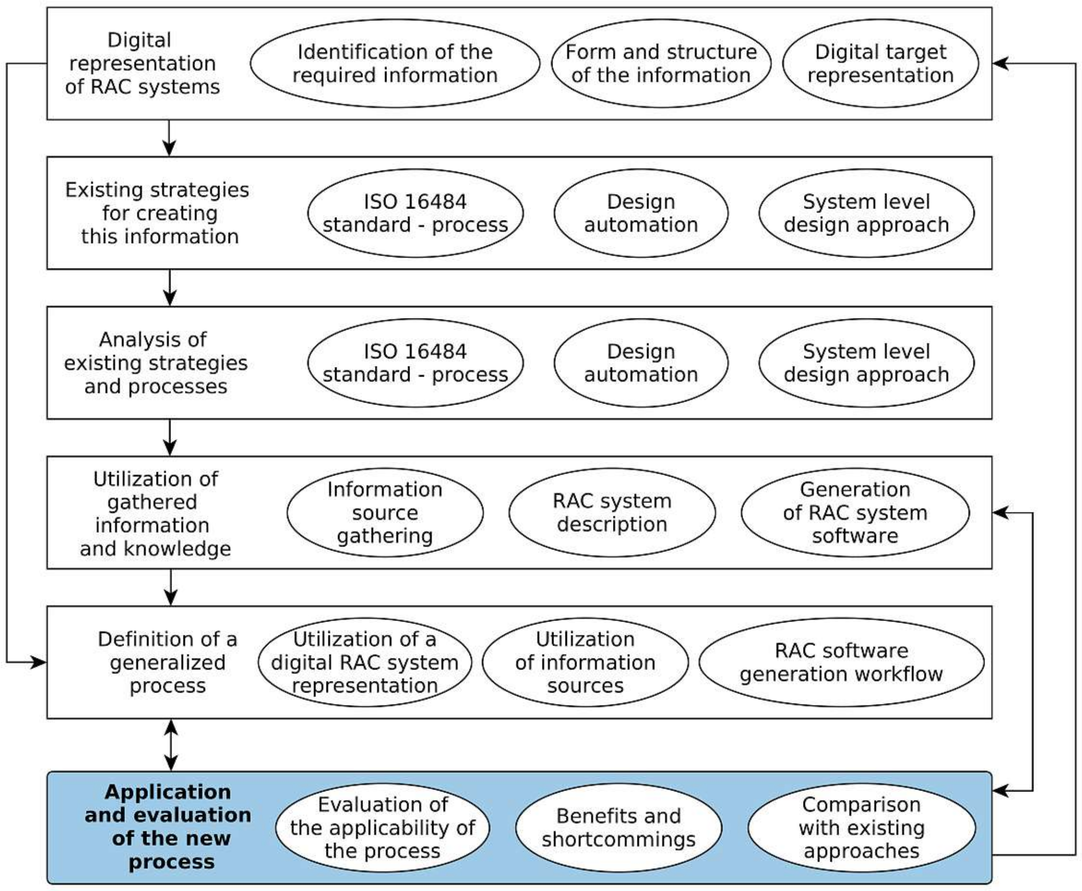

Digital representation of RAC systems
Identification of the required information
Form and structure of the information
Digital target representation
Existing strategies for creating this information
ISO 16484 standard - process
Design automation
System level design approach
Analysis of existing strategies and processes
ISO 16484 standard - process
Design automation
System level design approach
Utilization of gathered information and knowledge
Information source gathering
RAC system description
Generation of RAC system software
Definition of a generalized process
Utilization of a digital RAC system representation
Utilization of information sources
RAC software generation workflow
Application and evaluation of the new process
Evaluation of the applicability of the process
Benefits and shortcommings
Comparison with existing approaches

Figure 6.1: Part 6 of the methodological approach

on the same digital representation, keeping concrete implementations modular and as small as possible.

6.1.2 Information source gathering

The example stated in part 2 of the VDI 3813 guideline includes a rudimentary floor plan which gives an impression of how the office building and the rooms in the example are structured and divided (see Figure 6.2). This information is viable for deriving the structure of the building in subsequent tasks. No other building plan information source, which would be useful for application in this process, is available.
In the example stated in part 2 of the VDI 3813 guideline, no device list containing the models of devices is provided. A room equipment plan which visualizes the location and names of RAC equipment within a room is available, but not sufficient for deriving manufacturer and device-model specific information (see Figure 6.3). Therefore, an exemplary excerpt of a possible, reasonable list of devices suitable for this example, is

created and provided in Table 6.2.

Device label	**Manufacturer**	**Model name**
Multi-sensor	Siemens	UP 258D12
Room control panel	Siemens	RDG100KN
Sunshade actuator	ABB	SJR/S 4.24.2.1
Dewpoint monitor	Siemens	QXA2100
Window contact	Schratt	SKU: 200113
...		

Table 6.2: Exemplary excerpt of a device list based on the VDI 3813 example

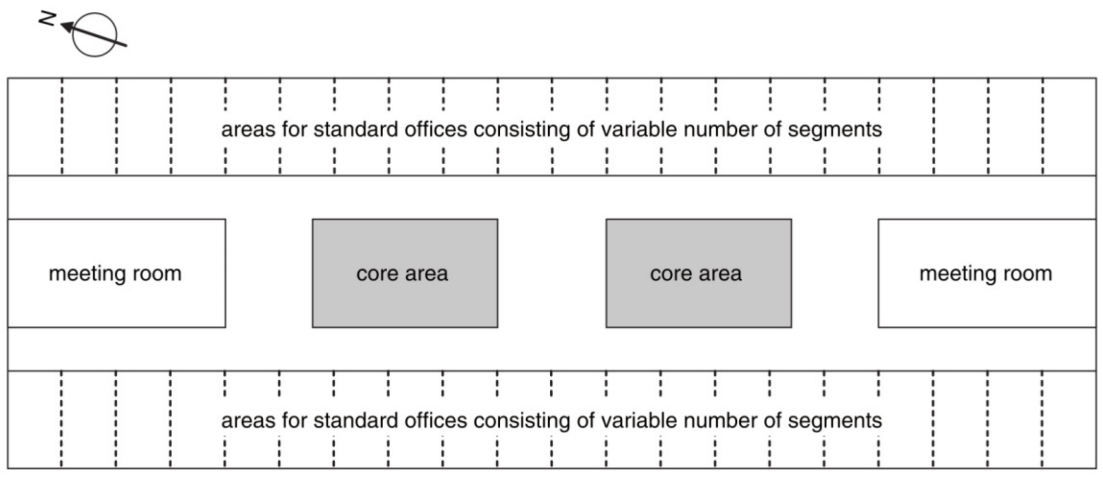

N
areas for standard offices consisting of variable number of segments
meeting room
core area
core area
meeting room
areas for standard offices consisting of variable number of segments

Figure 6.2: Example floor plan; Figure taken from part 2 of the VDI 3813 guideline [32]

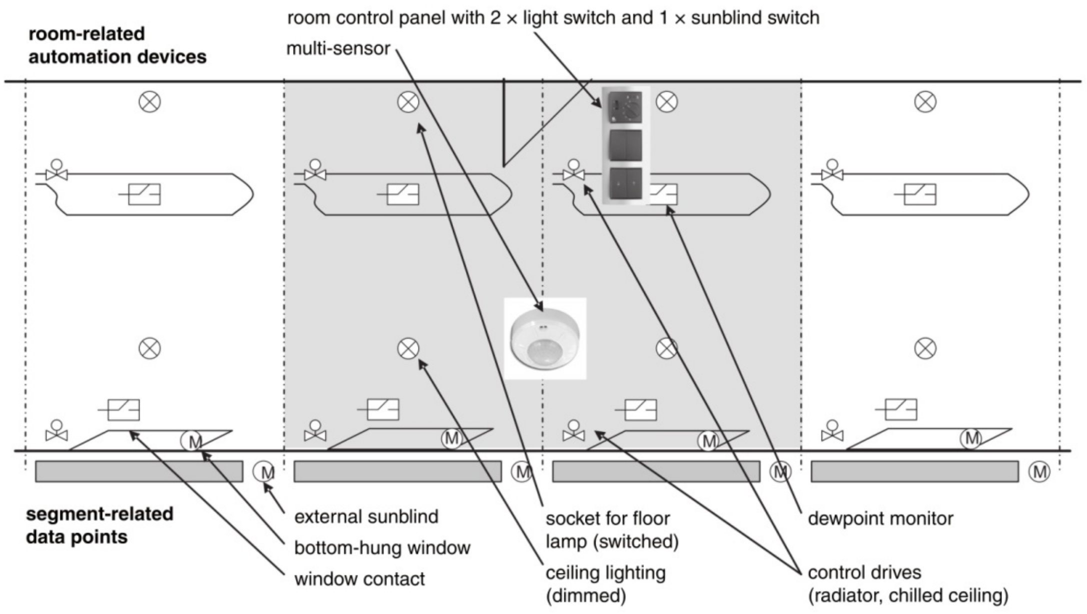


Figure 6.3: Example office area equipment plan; Figure taken from part 2 of the VDI 3813 guideline [32]

6.1.3 Building structure

In the first part of the workflow defined in Section 5.3, the available information about the structure of a building which is equipped with the RAC system is processed. This step is illustrated in Figure 6.4.

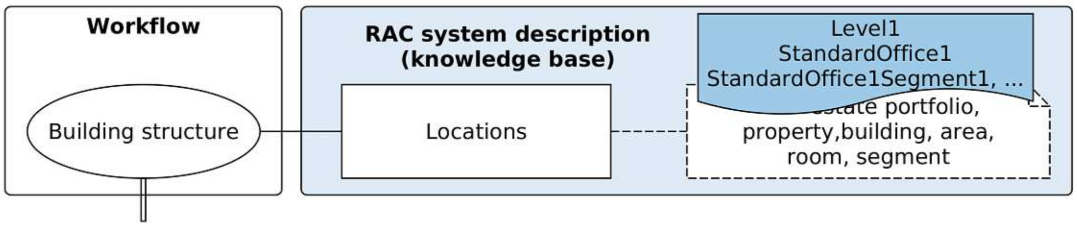


Figure 6.4: Part 1 of the workflow - building structure

Section 5.2.1 describes, how a BIM model available in the IFC format can be automatically converted and used in order to create the desired building structure up to the granularity of rooms based on the multilayer-model published in the VDI 3814 guideline. Unfortunately, the VDI 3813 example used in this evaluation chapter does not include a BIM model. Therefore, the room structure is manually created based on the given floor plan (see Figure 6.2). Such a floor plan could also be part of the documents given to system designers in real world projects. As this is an abstract example, only one room with

one segment has been modeled for demonstration, which is part of an exemplary area and building. The individuals are modeled using the corresponding subclasses of the class *Location*. For example, *Room* and *Segment* are associated with each other using the *hasPart/isPartOf* object property as defined in Section 3.3.2. Figure 6.5 illustrates the relation of modeled *Location* individuals in the ontology.
As mentioned above, if an IFC model is available, this structure could be automatically created up to the granularity of rooms (*StandardOffice1* in this example). Though, segments would still have to be added manually, as discussed in Section 5.2.1.

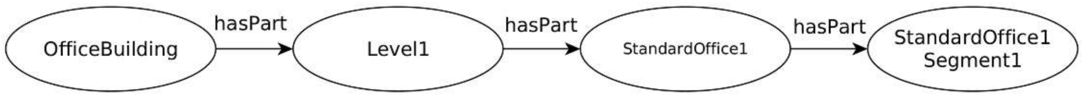


Figure 6.5: Building structure individuals and relations

6.1.4 Simple function devices

In this step, simple functions devices as defined in Section 5.3.2 are added to the digital representation knowledge base. This represents part 2 of the workflow for creating RAC system descriptions (see Section 5.3) and is illustrated in Figure 6.6.

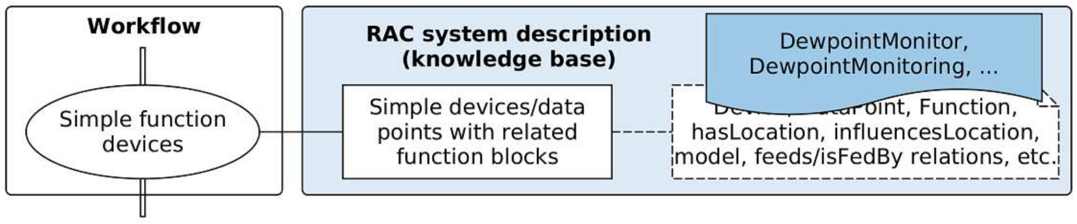

Workflow
Simple function devices
RAC system description (knowledge base)
Simple devices/data points with related function blocks
DewpointMonitor, DewpointMonitoring, ...
oint, Function, hasLocation, influencesLocation, model, feeds/isFedBy relations, etc.

Figure 6.6: Part 2 of the workflow - simple function devices

According to the device list outlined in Table 6.2, the dewpoint monitor would represent a simple function device, as it only supports one sensor function. Therefore, an individual *DewpointMonitor* of the class *Device* is added to the knowledge base. The data properties *manufacturer: Siemens* and *model: QXA2100 - Dewpoint monitor* are also associated with the device. Furthermore, corresponding connection types and points are linked to the devices using the *hasConnectionType* and *hasConnectionPoint* object properties. Using the *hasConnectionType* relation, the *DigitalInput* connection type is associated with the dewpoint monitor device. This information is derived from electronic or semantic device descriptions. The *CeilingDistributor* connection point is manually related with the dewpoint monitor, as none of the available information sources provides this insight. As next step, the only function supported by the dewpoint monitor device, *DewpointMonitoring*, is added as an individual to the ontology. As the class *DewpointMonitoring* is a

subclass of *SensorFunctions*, it can be derived that the dewpoint monitor device feeds the dewpoint monitoring function with information, hence the relation *DewpointMonitor feeds DewpointMonitoring* is implemented.
At last, the *influencesLocation* and optionally the *hasLocation* relations are associated with the newly created device. Both are not distinctively derivable from the given floor plan (see Figure 6.2) and the exemplary device list (see Table 6.2). As no other information sources provide this information, these relations are added manually. In this example, the relations *DewpointMonitor influencesLocation StandardOfficeSegment1* and *DewpointMonitor hasLocation StandardOffice1Segment1* are created. The model number of the device is added as data property.

Furthermore, the function block *DewpointMonitoring* is also manually associated with a location, e.g. using the *DewpointMonitoring influencesLocation StandardOfficeSegment1* relation. To simplify the identification of different dewpoint monitoring devices and functions according to their locations, the implemented individuals are renamed to *DewpointMonitorStandardOffice1Segment1* and *DewpointMonitoringStandardOffice1Segment1*. The workflow for the discussed dewpoint monitor example is illustrated in Figure 6.7. The same procedure is repeated for each simple function device. The resulting ontology entry for the dewpoint monitor device is outlined in Listing 6.1. A simplified illustration of the relation between a simple function device and a corresponding function block is given in Figure 6.8.

```
###  http://www.semanticweb.org/vdi-3813-example-ontology#
   DewpointMonitorStandardOffice1Segment1
:DewpointMonitorStandardOffice1Segment1 rdf:type owl:
   NamedIndividual ,
         <http://www.semanticweb.org/z003pwuk/ontologies/
            configuration-automation-ontology#Device> ;
     <http://www.semanticweb.org/z003pwuk/ontologies/
        configuration-automation-ontology#feeds> :
        DewpointMonitoringStandardOffice1Segment1 ;
     <http://www.semanticweb.org/z003pwuk/ontologies/
        configuration-automation-ontology#hasConnectionPoint> :
        CeilingDistributor ;
     <http://www.semanticweb.org/z003pwuk/ontologies/
        configuration-automation-ontology#hasConnectionType> :
        DigitalInput ;
     <http://www.semanticweb.org/z003pwuk/ontologies/
        configuration-automation-ontology#hasLocation> :
        StandardOffice1Segment1 ;
     <http://www.semanticweb.org/z003pwuk/ontologies/
        configuration-automation-ontology#influencesLocation> :
        StandardOffice1Segment1 ;
         <http://www.semanticweb.org/z003pwuk/ontologies/
```

```
            configuration-automation-ontology#manufacturer> "
            Siemens"^^xsd:string ;
    <http://www.semanticweb.org/z003pwuk/ontologies/
        configuration-automation-ontology#model> "QXA2100 -
        Dewpoint monitor"^^xsd:string .
```

Listing 6.1: Dewpoint monitor entry in ontology

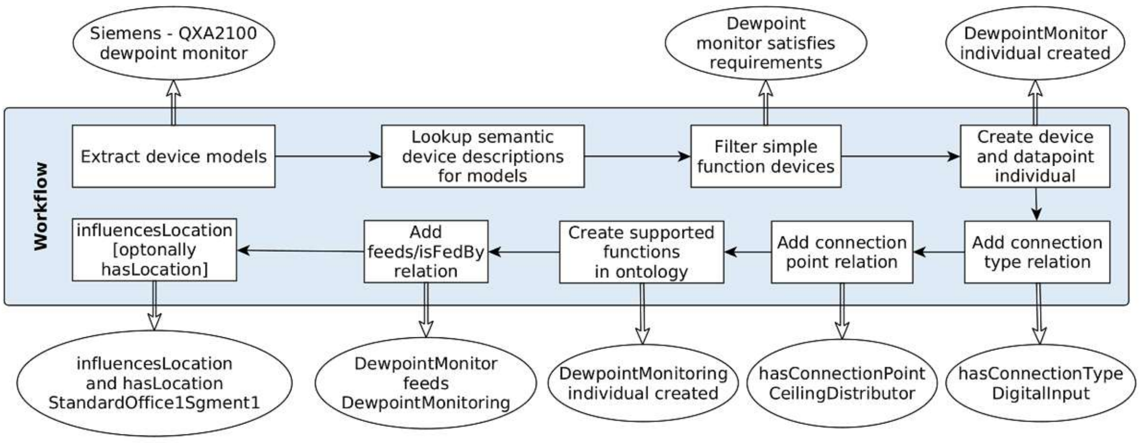

Siemens - QXA2100 dewpoint monitor
Dewpoint monitor satisfies requirements
DewpointMonitor individual created
Workflow
Extract device models
Lookup semantic device descriptions for models
Filter simple function devices
Create device and datapoint individual
influencesLocation [optonally hasLocation]
Add feeds/isFedBy relation
Create supported functions in ontology
Add connection point relation
Add connection type relation
influencesLocation and hasLocation StandardOffice1Sgment1
DewpointMonitor feeds DewpointMonitoring
DewpointMonitoring individual created
hasConnectionPoint CeilingDistributor
hasConnectionType DigitalInput

Figure 6.7: Workflow for creating a simple function device using a dewpoint monitor example

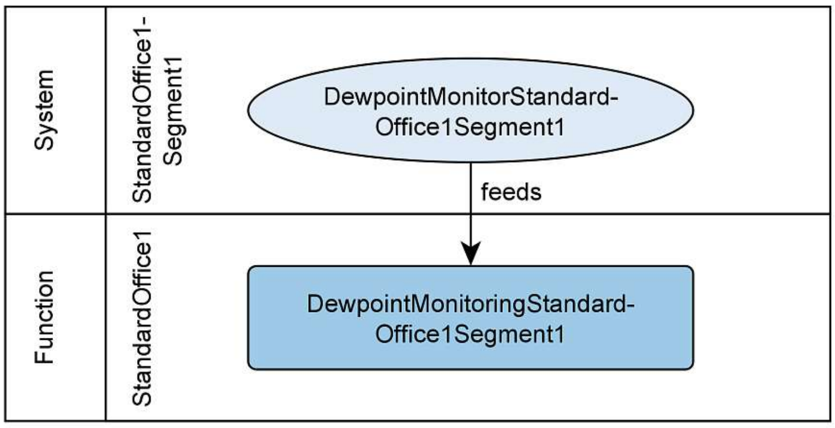

System
StandardOffice1-Segment1
DewpointMonitorStandard-Office1Segment1
feeds
Function
StandardOffice1
DewpointMonitoringStandard-Office1Segment1

Figure 6.8: Relation between the dewpoint monitor and the monitoring function

This procedure is repeated for each simple function device derived from all available device lists. Another example for such entities would be a window contact with a corresponding window monitoring function as stated in the VDI 3813 guideline example. [32]

6.1.5 Additional function descriptions

Part 3 of the workflow introduced in Section 5.3 is executed in this segment, as illustrated in Figure 6.9. As described in Section 5.3.3, application functions that describe the functionality of RAC systems are added to the digital representation.

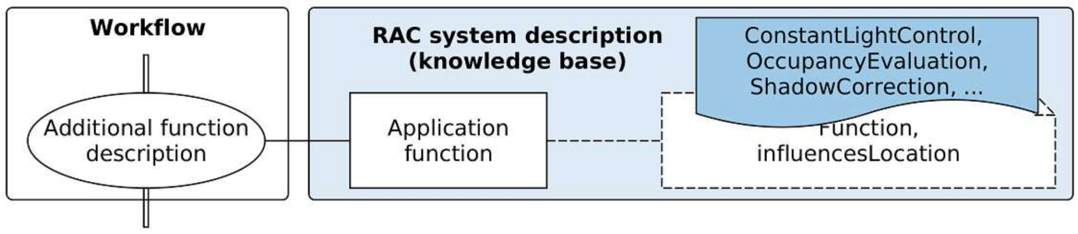

Workflow
Additional function description
RAC system description (knowledge base)
Application function
ConstantLightControl, OccupancyEvaluation, ShadowCorrection, ...
Function, influencesLocation

Figure 6.9: Step 3 of the workflow - Additional function descriptions

The VDI 3813 example provides a textual description containing information about required application functions. Though, in this example the text is used to complement the already available function schematic and list and is not comprehensive. Because a main focus of the introduced process as part of this thesis is to create such function descriptions based on a knowledge base, the text is not sufficient to implement the required application functions. Therefore, it is assumed that no textual documentation describing the functional and non-functional requirements is available.
Instead, the desired functional and non-functional requirements are added manually to the knowledge base in the form of application function blocks, as it would be done if no such documents are available. For example, a constant light control function block is implemented following this process. As with simple functions, application function blocks are associated with locations using the *influencesLocation* relation. Using the constant light control example, the function block is associated with the location entity *StandardOffice1*. At this point of the process, no *feeds* relations between devices and application functions are established.
Figure 6.10 includes all the application functions and the locations they influence. To improve readability, the informational location suffixes are omitted in the illustration. For example, *PriorityControl* is used instead of *PriorityControlStandardOffice1Segment1*, which can be easily derived from the figure. Required function blocks are manually added to the ontology in order to satisfy functional and non-functional requirements stated by stakeholders.

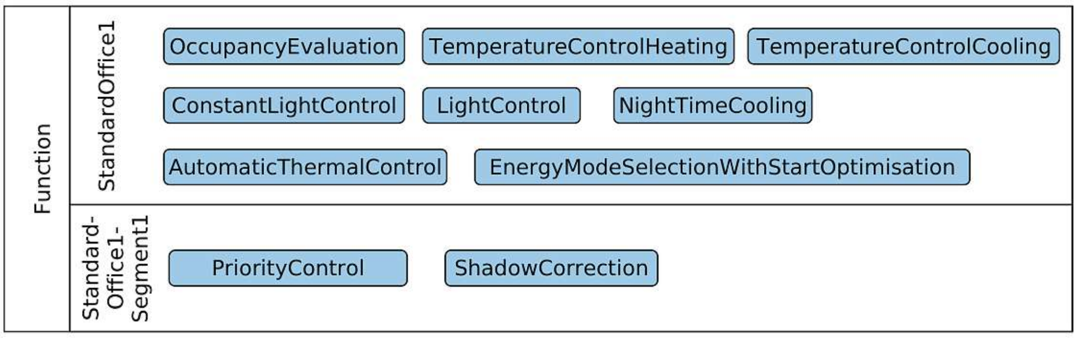


Figure 6.10: Additional functions added based on functional and non-functional requirements

6.1.6 Relation between established functions

In this section, part 4 of the workflow introduced in Section 5.3 is executed on basis of the VDI 3813 example, as illustrated in Figure 6.11.

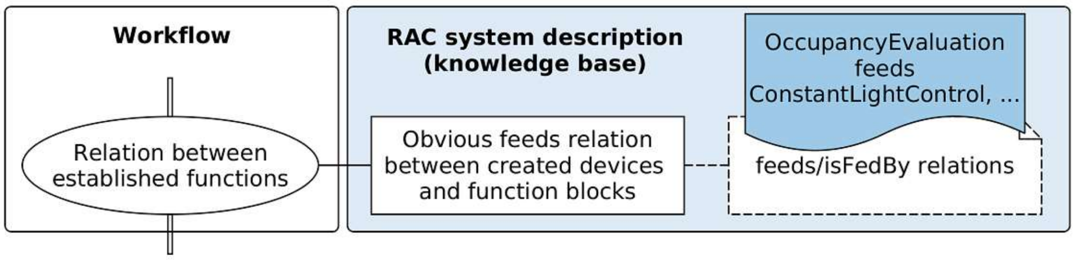

Workflow
Relation between established functions
RAC system description (knowledge base)
Obvious feeds relation between created devices and function blocks
OccupancyEvaluation feeds ConstantLightControl, ...
feeds/isFedBy relations

Figure 6.11: Step 4 of the workflow - Relating established functions and devices

As described in Section 5.3.4, input and output variables of function blocks related with simple function devices are examined first. For example, it is already established that the sensor function *WindowMonitoringStandardOffice1Segment1* is fed by the *WindowContactStandardOffice1Segment1* device. Furthermore, the function block description in part 2 of the VDI 3813 guideline states, that the *WindowMonitoring* function has only one binary output, which provides information about the *open/closed* state of a window (*B_WINDOW*). Using this information, other already established function blocks are checked if they rely on the input variable *B_WINDOW*.
Furthermore, the influenced locations associated with the two function blocks are checked. Starting with the smallest location level influenced by the function block, all other function blocks within the same area of influence are analyzed. If there is no function block which relies on the provided information, all other blocks within the next higher location of influence are examined. In this example, the *WindowMonitoringStandardOffice1Segment1* function block which provides the *B_WINDOW* information influences

the segment *StandardOffice1Segment1*. Though, no function block within this segment uses *B_WINDOW* as input variable. Therefore, the function blocks influencing the next higher location level, which is the room *StandardOffice1*, are analyzed based on the same requirements. Within this level, the block *EnergyModeSelectionWithStartOptimisationStandardOffice1* is identified to rely on the input *B_WINDOW*, and a *WindowMonitoringStandardOffice1Segment1 feeds EnergyModeSelectionWithStartOptimisation* relation can be established. This procedure is illustrated in Figure 6.12.

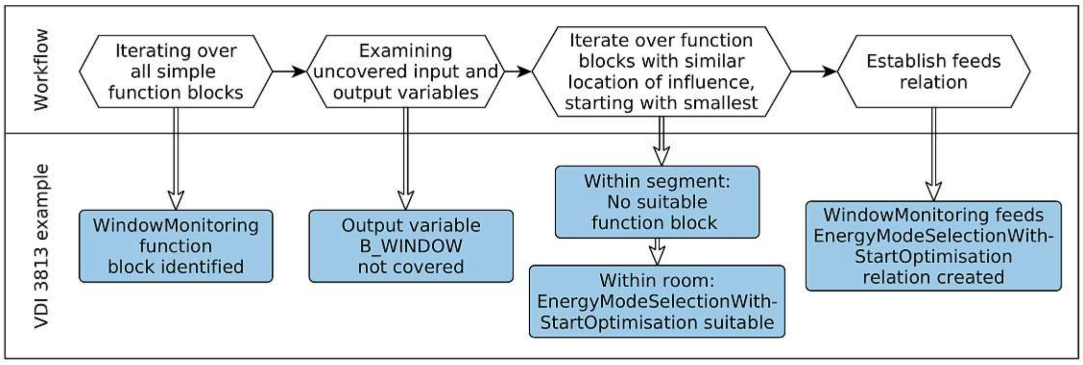

Workflow
Iterating over all simple function blocks
Examining uncovered input and output variables
Iterate over function blocks with similar location of influence, starting with smallest
Establish feeds relation
VDI 3813 example
WindowMonitoring function block identified
Output variable B_WINDOW not covered
Within segment: No suitable function block
Within room: EnergyModeSelectionWith-StartOptimisation suitable
WindowMonitoring feeds EnergyModeSelectionWith-StartOptimisation relation created

Figure 6.12: Example workflow for creating relations for simple function blocks

If multiple function blocks provide the same information within the same area of influence, a manual selection of the corresponding function block has to be made on based on some options provided to the user. This procedure is described in more detail in Section 6.1.7, using an example with simple pushbuttons providing identical functional capabilities.
Next, the same procedure is repeated for all established application functions. For example, a relation between the introduced occupancy evaluation and constant-light control function block can be established based on the *P_ACT* parameter, containing information about the current occupancy state of a room. The workflow for relating application function blocks using this example is illustrated in Figure 6.13.

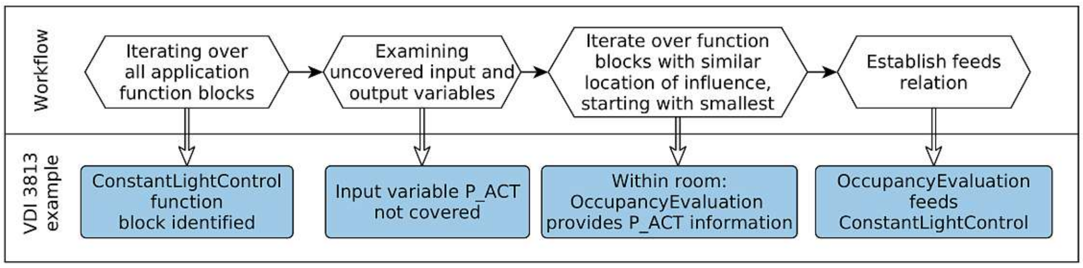

Workflow
Iterating over all application function blocks
Examining uncovered input and output variables
Iterate over function blocks with similar location of influence, starting with smallest
Establish feeds relation
VDI 3813 example
ConstantLightControl function block identified
Input variable P_ACT not covered
Within room: OccupancyEvaluation provides P_ACT information
OccupancyEvaluation feeds ConstantLightControl

Figure 6.13: Example workflow for relating application function blocks

6.1.7 Relating uncovered function blocks and unutilized devices

In this part, relations between uncovered function blocks and unutilized devices are established. This corresponds with part 5 of the workflow defined in Section 5.3 and is visualized in Figure 6.14.

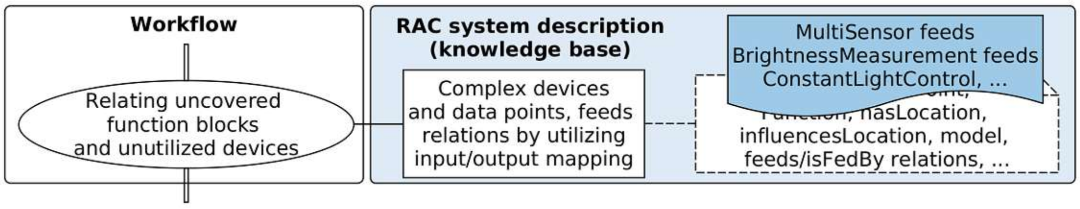
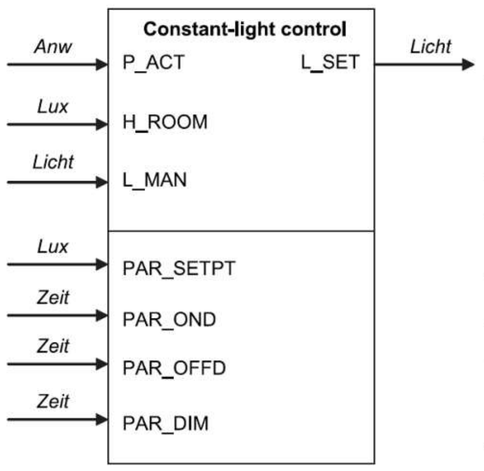
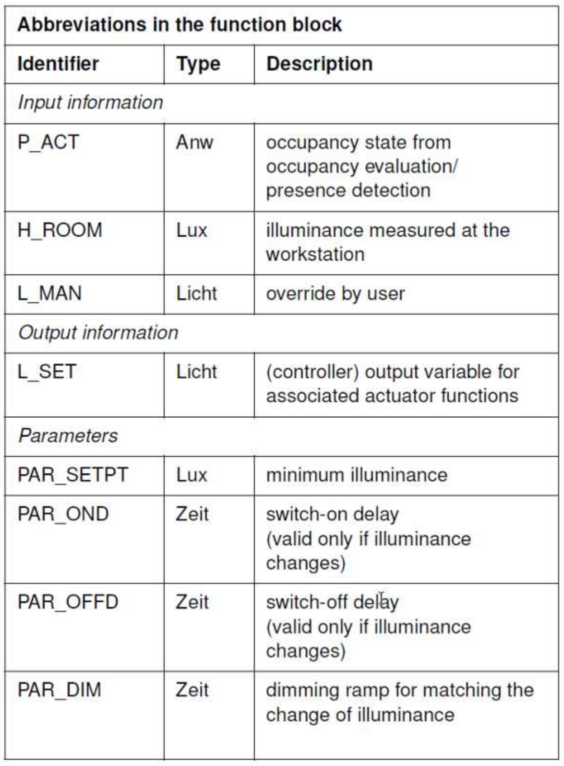

Workflow
Relating uncovered function blocks and unutilized devices
RAC system description (knowledge base)
Complex devices and data points, feeds relations by utilizing input/output mapping
MultiSensor feeds BrightnessMeasurement feeds ConstantLightControl, ...
Function, hasLocation, influencesLocation, model, feeds/isFedBy relations, ...

Figure 6.14: Step 5 of the workflow - Relating uncovered function blocks with unutilized devices

A constant-light control function block, which was created in Section 6.1.5, is used to describe this approach. In Figure 6.15, an overview of this function block and its parameters, as stated in part 2 the VDI 3813 guideline, is illustrated. [32]
The constant-light control function ensures, that the illuminance of an influenced location is always above a set minimum when occupied by automatically controlling associated light actuator functions. Therefore, input parameters containing information about the current occupancy state (P_ACT), illuminance of the influenced location (H_ROOM) and a possible override by a user (L_MAN) are required.
Furthermore, the function block provides configuration capabilities using the following parameters: PAR_SETPT for setting the minimum required illuminance, a switch-on delay used to only actuate lighting when the illuminance is below the minimum for a given time (PAR_OND), a switch-off delay used to delay switching off a lighting until the illuminance is above the minimum for a given time (PAR_OFFD), and a dimming ramp parameter, which is used to adjust the time required for changing brightness to match the preferred illuminance (PAR_DIM). In the VDI 3813 guideline, such parameters are not stated as input information that is fed by other function blocks or devices. Instead, they are regarded as informative and are intended to describe, how users can adjust functions during the operating phase without editing software programs. [32]
In the planning phase, such parameters can be specified to provide a preset functionality. This can be done based on experience of previous projects, templates, based on known user preferences or by creating them manually, e.g. by system integrators or users.
In this example, the parameters required by the constant-light control function block are set manually to realistic values. As already stated, they can be adjusted later in the operating phase in order to satisfy user preferences. The used parameter settings for the constant-light control function block are:

- PAR_SETPT: 800 Lux
- PAR_OND: 30 seconds





Abbreviations in the function block		
Identifier	**Type**	**Description**
Input information		
P_ACT	Anw	occupancy state from occupancy evaluation/ presence detection
H_ROOM	Lux	illuminance measured at the workstation
L_MAN	Licht	override by user
Output information		
L_SET	Licht	(controller) output variable for associated actuator functions
Parameters		
PAR_SETPT	Lux	minimum illuminance
PAR_OND	Zeit	switch-on delay (valid only if illuminance changes)
PAR_OFFD	Zeit	switch-off delay (valid only if illuminance changes)
PAR_DIM	Zeit	dimming ramp for matching the change of illuminance

Figure 6.15: Constant-light control function block taken from part 2 of the VDI 3813 guideline [32]

- PAR_OFFD: 30 seconds
- PAR_DIM: 5 seconds

Next, the input parameters of the constant-light control function block are examined one after the other, similar to the approach described in Section 6.1.6. A relation between the constant-light control and the occupancy evaluation function block using the occupancy state input variable (P_ACT) has already been established in the previous step of the process. The next required input variable H_ROOM, which contains information about the illuminance level measured at the workstation, is not provided by any of the already created function blocks. Therefore, remaining devices which support this functionality and have unutilized capacities, are identified by scanning available device lists. If a device provides the desired functionality, it is marked as a possible future relation partner. Thereby, only devices which influence the same location or sub-location as the related function block (i.e. constant-light control in this example) are considered.
Coming back to the VDI 3813 example, a multi-sensor device can be identified using this method. It is the only device which supports a brightness measurement functionality, which provides the required (H_ROOM) information as output variable. Therefore, a

brightness measurement function block is created and is related to the multi-sensor device using the relation *MultiSensorStandardOffice1 feeds BrightnessMeasurementStandardOffice1*. Furthermore, the newly created function block is connected with the constant-light control function using the relation *BrightnessMeasurement1StandardOffice1 feeds ConstantLightControlStandardOffice1*. This workflow is illustrated in Figure 6.16.

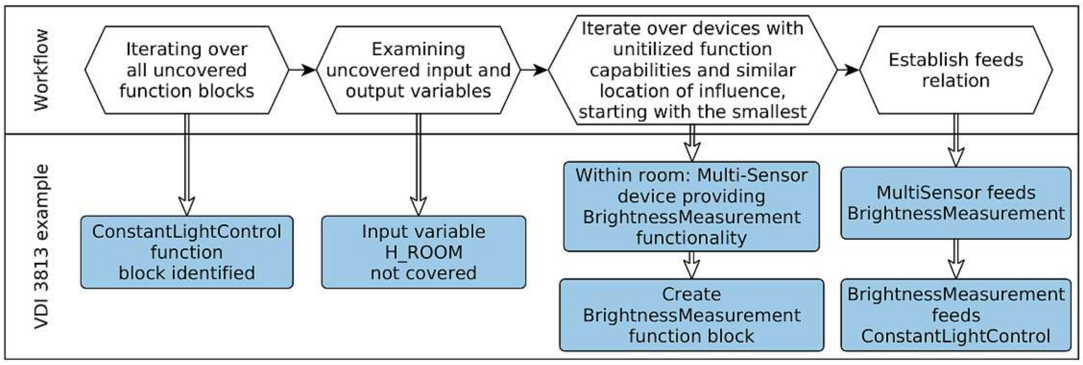

Workflow
Iterating over all uncovered function blocks
Examining uncovered input and output variables
Iterate over devices with unitilized function capabilities and similar location of influence, starting with the smallest
Establish feeds relation
VDI 3813 example
ConstantLightControl function block identified
Input variable H_ROOM not covered
Within room: Multi-Sensor device providing BrightnessMeasurement functionality
Create BrightnessMeasurement function block
MultiSensor feeds BrightnessMeasurement
BrightnessMeasurement feeds ConstantLightControl

Figure 6.16: Workflow for creating relations between uncovered function blocks and unutilized devices

In order to cover the last input parameter *L_MAN*, a device which either supports the *Actuate light* or *Control via room utilisation types* functionality has to be identified. In the VDI example, three pushbutton devices are available. Each can be used to implement the *Actuate light* function block. As they are influencing the same location, they are all equally suitable for this purpose. At this point, the user can be asked to select one of the devices by providing him the three options and all available information about them, or a random device can be selected. By adding meta-information about devices in device lists, such as the intended exact spatial location of a device, the decision process of the user can be simplified. For example, the information *top pushbutton left of entry door*, *bottom pushbutton left of entry door* and *pushbutton near window* as well as the intended room of installation, *Standard office 1*, could be displayed to the user. On basis of the digital representation introduced in Chapter 3, such meta-information about devices can be associated with them using the *rdfs:comment* annotation property.
After selecting one pushbutton, an *Actutate light* function block is created and associated with the device using the relation *PushButton1StandardOffice1 feeds ActuateLight1StandardOffice1*. Furthermore, the relation *ActuateLight1StandardOffice1 feeds ConstantLightControlStandardOffice1* is established. An illustration for such a workflow is shown in Figure 6.17.

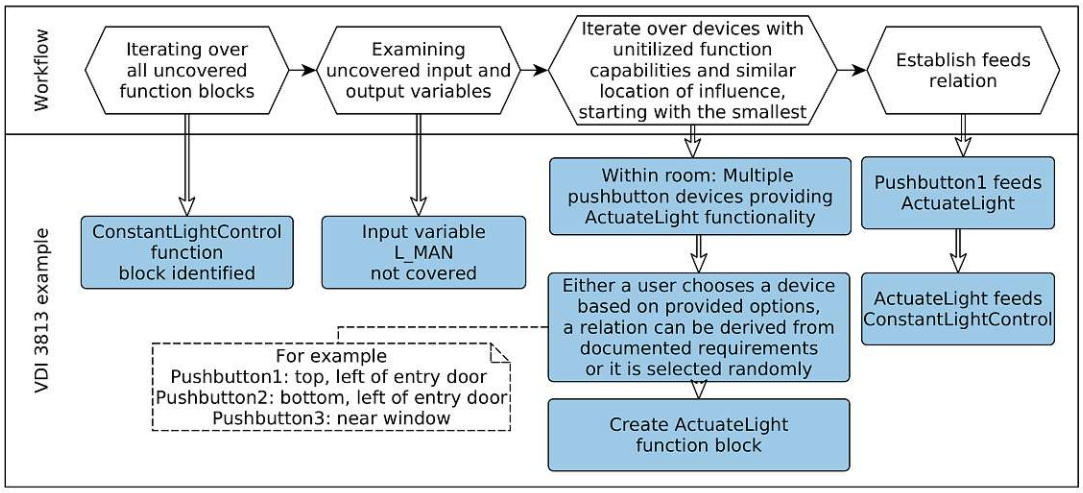

Workflow
Iterating over all uncovered function blocks
Examining uncovered input and output variables
Iterate over devices with unitilized function capabilities and similar location of influence, starting with the smallest
Establish feeds relation
VDI 3813 example
ConstantLightControl function block identified
Input variable L_MAN not covered
Within room: Multiple pushbutton devices providing ActuateLight functionality
Pushbutton1 feeds ActuateLight
Either a user chooses a device based on provided options, a relation can be derived from documented requirements or it is selected randomly
ActuateLight feeds ConstantLightControl
For example Pushbutton1: top, left of entry door Pushbutton2: bottom, left of entry door Pushbutton3: near window
Create ActuateLight function block

Figure 6.17: Workflow for creating relations between uncovered function blocks and multiple, suitable devices providing the required functionality

Depending on the means of operation, the operator panel and its design as well as its representation influenced by the manufacturer, the parameters required by the actuate light function block can differ. [32] This information should be provided in semantic device descriptions as discussed in Section 5.2.3.2.
All input parameters of the constant-light control function block are now connected to an information source. The controller output variable *L_SET* provided by this function block can be utilized by *Light actuator* functions. In the VDI 3813 example, two devices have the capability to process this information within the same room, a ceiling luminaire and a floor lamp. As both are equally appropriate recipients, a user can choose between the two devices similar to the selection of the pushbuttons earlier. It may also be possible, that this information is available through documented requirements, for example by stating that the ceiling luminaire should be regulated using a constant-light control functionality. By having this information, a link between the device and the application function constant-light control would have been already established during the creation of the function bock as described in Section 6.1.5. Independent of the decision approach, a *Light actuator* function block, which is associated with the ceiling luminaire device using the relation *LightActuator1StandardOffice1 feeds CeilingLuminaireStandardOffice1*, is introduced. This function block can be customized by means of parametrization, for example to allow creating definite output information states in case of automation system voltage failures or recovery. Furthermore, it is connected to the constant-light control block using the relation *ConstantLightControlStandardOffice1 feeds LightActuator1StandardOffice1*. This workflow is similar to the one shown in Figure 6.17.
All input and output variables of the constant-light control function block are now covered. The resulting relations for this example are illustrated in figure 6.18.

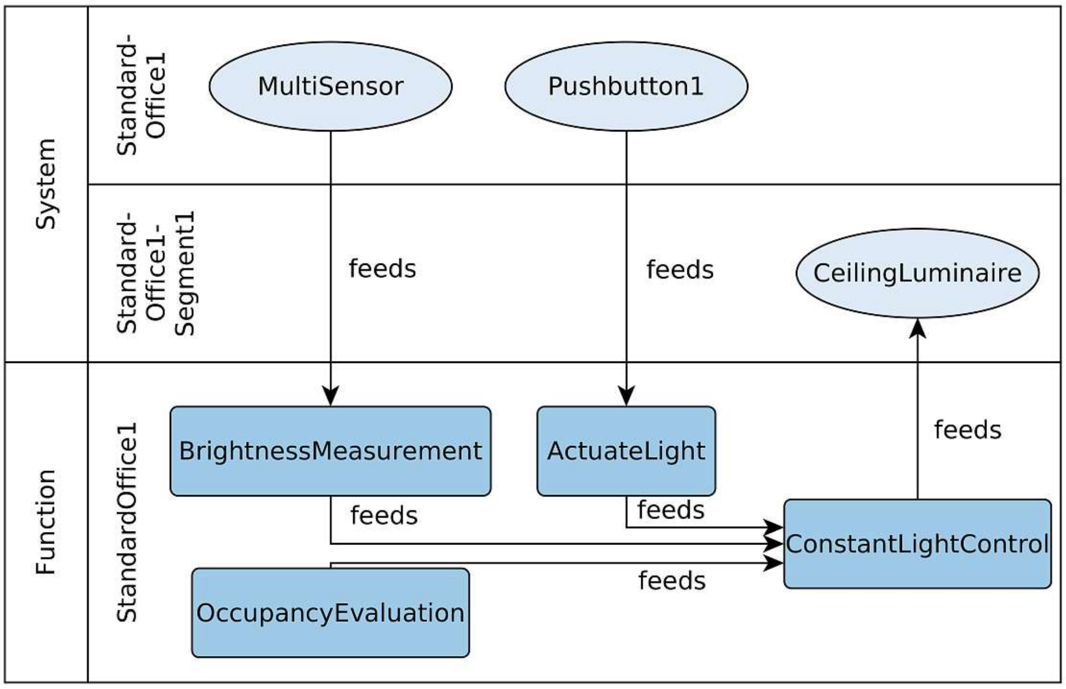


Figure 6.18: Resulting relations for the constant-light control application function block

This process is repeated for every, already created function block which has open connections and for all available devices, which provide and support suitable functions that are not yet utilized. Using this approach, most of the function blocks should be covered by the end of this step, leading to a schematic description of the functionality of the RAC system by using the digital representation introduced in Chapter 3.

6.1.8 Manual completion and control

The manual completion and control task is executed in parallel to all other steps of the workflows, as illustrated in Figure 5.5. A system integrator is asked to manually complete the digital representation for automated conversion to RAC system automation software. In the example provided in the VDI 3813 guideline, an additional, application-specific function *Solar altitude calculation* has to be introduced. This is necessary, as no device offers information about the azimuth and elevation angle of the sun, which are required as input by the *Shadow correction* function block implemented in this example. This lack of knowledge may either be covered by manually adding devices that provide this information or by creating a new function block, such as *Solar altitude calculation*, which processes existing information to cover missing requirements.
In order to manually create a function block, the new function has to be added as a class to the digital representation ontology. Furthermore, imported ontologies which

provide information about function blocks, their input and output requirements as well as parametrization have to be extended to support the newly introduced function. At last, a new individual, which is an instance of this function class, is created and relations are established in order to satisfy the required functionality of the system. This workflow is visualized in Figure 6.19 based on the example provided in the VDI 3813 guideline. This includes a *Solar altitude calculation* and a *Shadow correction* function block.

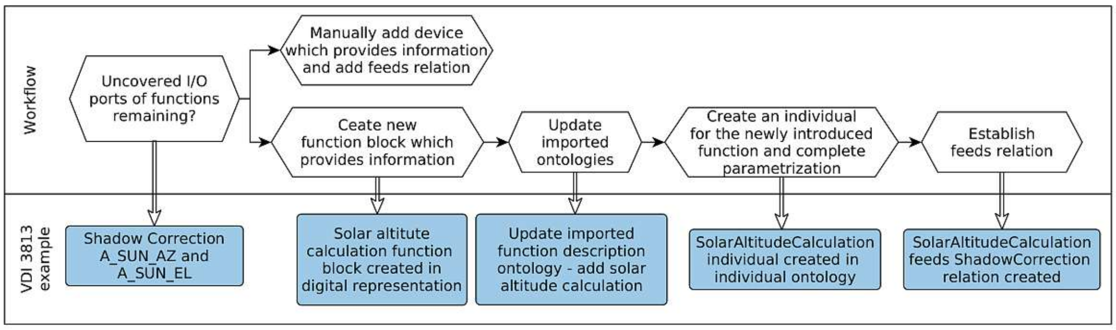

Workflow
Uncovered I/O ports of functions remaining?
Manually add device which provides information and add feeds relation
Ceate new function block which provides information
Update imported ontologies
Create an individual for the newly introduced function and complete parametrization
Establish feeds relation
VDI 3813 example
Shadow Correction A_SUN_AZ and A_SUN_EL
Solar altitute calculation function block created in digital representation
Update imported function description ontology - add solar altitude calculation
SolarAltitudeCalculation individual created in individual ontology
SolarAltitudeCalculation feeds ShadowCorrection relation created

Figure 6.19: Manual addition of a solar altitude calculation function block to create missing information

6.1.9 Generation of RAC system software

In this section, implementing RAC system software on basis of the digital representation created in the previous steps of the workflow is discussed. This represents the last task in the proposed process introduced in Section 5.3 as illustrated in Figure 5.5. The conversion is based on the process outlined in Section 5.4 by creating software for KNX [50] devices using the Engineering Tools Software (ETS) [99] software.
At first, the building structure is imported in ETS following the convention outlined in Section 5.4. The result of the simple building structure introduced in this chapter illustrated in Figure 6.20. The location *StandardOffice1Segment1* is not inherited by the ETS implementation, as creating segments is not supported.

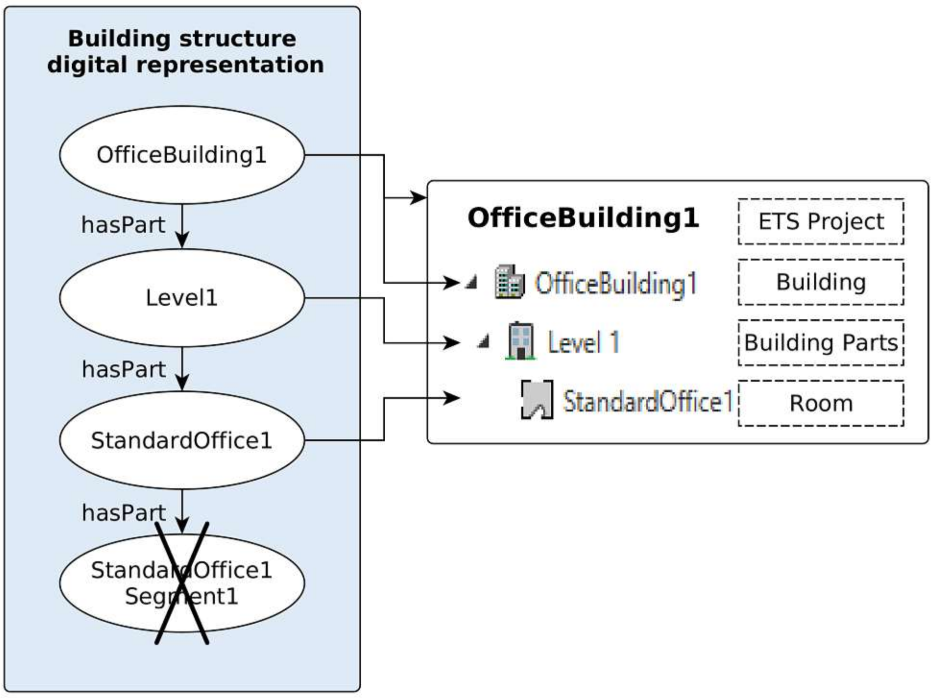


Figure 6.20: Building structure conversion example

Furthermore, the areas and rooms stated in the building structure are used to create a topology in ETS. Following the rules described in Section 5.4, the topology visualized in Figure 6.21 is implemented. In this simple example, only one line is created initially and added to an ETS topology area element by executing a depth first traversal, starting with segments. This approach is based on the method presented in [98]. All devices within a location are added to *lines* following this approach. In this example, the devices influencing *StandardOffice1Segment1* are associated with *Line1*. As the number of devices does not exceed 50, the devices within the next higher location level, *StandardOffice1* and *Level1* are added to the same line in this order. If the number of devices has exceeded 50, a new line would have to be defined and grouped to the same ETS area, i.e. *Area1*. On top of that, if devices associated with other areas within the defined building structure are existent, e.g. *Level2* with subsequent rooms, segments and devices, a new ETS line and area would have been introduced. As neither is the case in this example, one line within one area is sufficient.

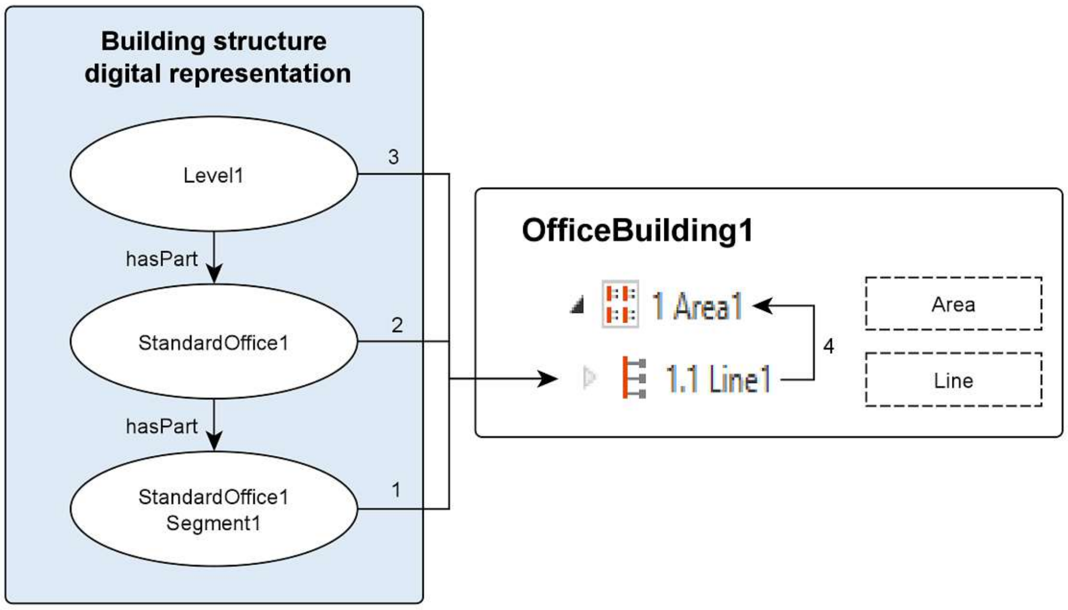


Figure 6.21: Topology creation example

As the next step, a physical KNX address is generated for each device. This is done following the pattern in Figure 6.22 using the association of devices with lines and areas. An algorithm can generate numbers for lines, areas and devices following this structure to assign physical addresses to them. [98]

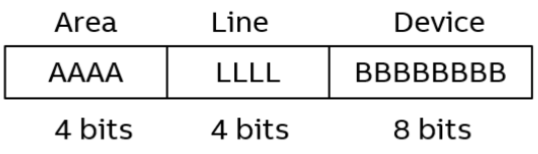


Figure 6.22: KNX physical address structure [98]

In order to use and connect the devices, which have been added via the ETS tool, connections between devices and function blocks that are provided by the digital representation have to be established. In the KNX technology, input and output ports of devices communicate with each other using so-called communication objects. These objects have a size and a type property, which often differ from the semantic data types used in function blocks presented in the VDI 3813 guideline. Type mapping algorithms have been created to overcome this problem. [98] This is discussed in Section 5.4.
Furthermore, semantic device packages have been introduced to allocate KNX devices to function blocks as presented in the VDI 3813 guideline. These packages describe, which function blocks are supported by devices and how the ports of function blocks

and devices are connected with each other to create the desired functionality. These connections are established using KNX group addresses.
In the following paragraphs, different approaches for connecting KNX devices with function blocks based on semantic device packages are shown.

Light control
It has to be checked, if function blocks needed in order to create a desired functionality, are supported by devices that have been added to the ETS project based on the digital representation. Figure 6.23 illustrates relations that have been established based on the VDI example in order to create a light control functionality. As can be seen in this figure, a feeds relation between the pushbutton and the *Actuate Light* function block has been established. Following the semantic device package description of the device, it can be established that the pushbutton itself supports the function block *Actuate light*. This means, the functionality is provided by means of correct parametrization of the device based on semantic device packages and the information provided by the digital representation ontology.
The *Actuate light* function block is used to feed the *Light control* function, which in turn feeds the *Light actuator* block according to the digital representation illustrated in Figure 6.18. In the VDI 3813 example, a floor lamp with a relay connection point is connected to the *Light actuator* function block. In the following, two different possibilities for implementing this functionality are discussed.
The first assumption is, that the floor lamp used in this example provides the *Light control* as well as the *Light actuator* functionality itself. This may be possible, if the switch actuator integrated in the lamp can be controlled via the KNX system. It may also be possible, that an extra switch actuator is installed and connected to the bus system, which itself is used to actuate the lamp via relay. If this is the case, this actuator has not been added to the example provided in the VDI 3813 guideline. It is assumed, that in the case of the utilization of such an additional switch actuator, the *Light control* and *Light actuator* functionality is provided by this device. Either way, there is no need for external function block implementations, as all function blocks shown in Figure 6.18 are supported by the corresponding devices, if they are parameterized correctly on basis of semantic device packages and the information provided by the digital representation. Using semantic device packages, group addresses are assigned to the communication object of the devices to create the desired functionality as shown in Figure 6.24.

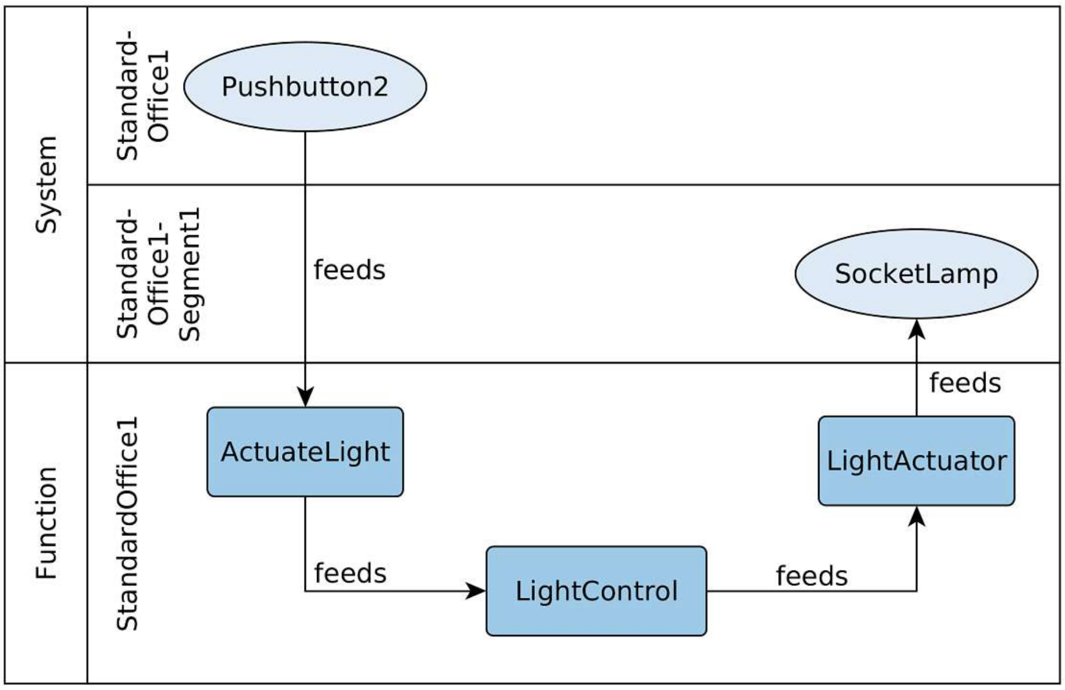


Figure 6.23: Light control schematics based on the VDI 3813 example

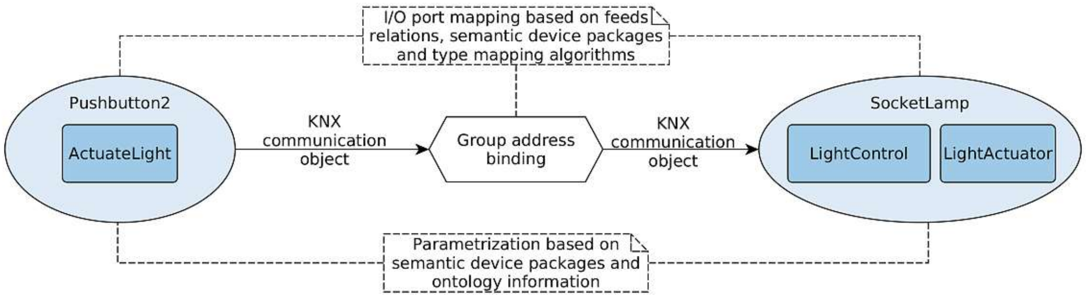

I/O port mapping based on feeds relations, semantic device packages and type mapping algorithms
Pushbutton2
ActuateLight
KNX communication object
Group address binding
KNX communication object
SocketLamp
LightControl
LightActuator
Parametrization based on semantic device packages and ontology information

Figure 6.24: Establishing communication between KNX devices - light control example

Another possible assumption is, that the floor lamp or a connected switch actuator do support the function *Actuate light* but not *Control light*. In this case, this functionality has to be provided using a middleware that implements the *Control light* function block. Therefore, a driver that connects the middleware to the KNX system has to be used. Appropriate group addresses are added to the devices as well as a driver in order to establish communication and the desired functionality. This leads to a communication sequence as illustrated in Figure 6.25. The function block *Control light* is assumed to be already implemented in the middleware, and is parameterized using the information provided by the digital representation ontology.

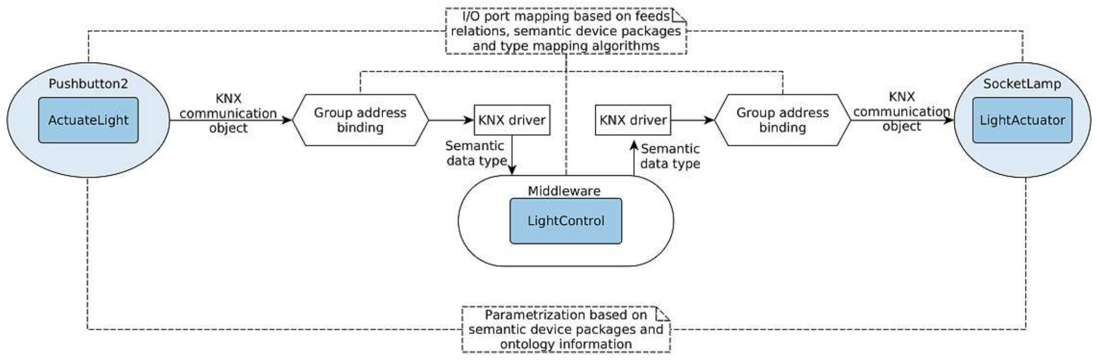


Figure 6.25: Establishing communication between KNX devices via middleware functionality and drivers - light control example

This workflow is repeated for each device and function block. If a function is supported by a device, there is no need for an additional implementation. Otherwise, the functionality is provided through a middleware which communicates with devices via technology specific drivers.

This RAC system software creation workflow is also applicable using other technologies and protocols than KNX, as long as a mapping and communication between devices and a technology independent middleware is provided through the use of drivers. For example, a ZigBee driver can be implemented, which supports devices that utilize the ZigBee wireless communication protocol. As with KNX devices, semantic device packages are used to map ZigBee devices to VDI 3813 function blocks. ZigBee provides profile-based concepts, in which profiles define commands and properties for specific device types, e.g. a light-switch profile. Instead of creating a mapping for each specific device type, the semantic device packages can relate to a set of profiles. This approach requires a standardization of profile-based concepts, which is currently not available. [98]
Using such semantic device packages, the function blocks which are supported by specific ZigBee devices can be discovered and related with each other the same way as it is done with KNX devices. Differences in the technology have to be supported by the digital representation ontology. For example, in ZigBee a central communication unit orchestrates the communication between devices instead of a direct communication via group addresses used in KNX. The digital representation has to support the creation of technology specific properties, e.g. storing addresses of ZigBee devices based on the IEEE 802.15.4 protocol. [79] In case of incomplete technology support, the digital representation ontology has to be manually updated and extended.
Through the use of semantic device packages to describe supported functions of devices and technology specific drivers, establishing a cross-technology RAC system is possible. The middleware implements VDI 3813 function blocks, which in turn are related and

connected with functions supported by devices via drivers, regardless of the technology and protocol of the device. [98] Using the light control functionality introduced in this section, a possible cross-technology communication between ZigBee and KNX devices is illustrated in Figure 6.26.

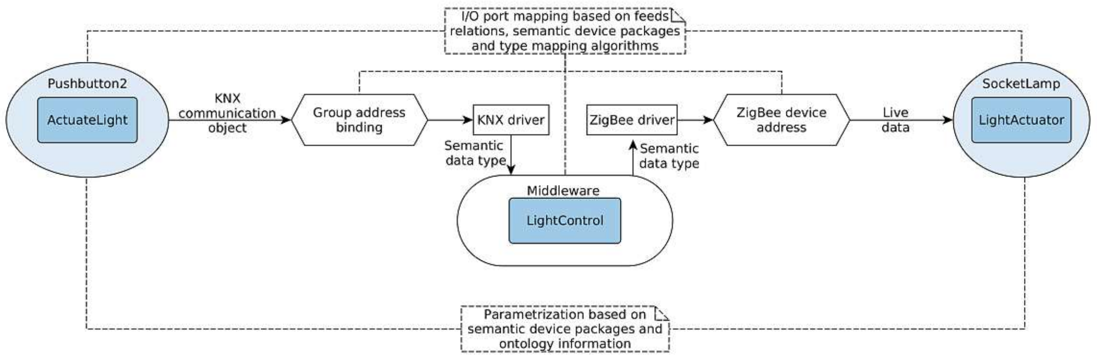


Figure 6.26: Light control example using multi-technology communication

6.2 Evaluation of the defined process

In the following, the process defined in Section 5.5 is evaluated by comparing it with existing RAC system design processes identified in Chapter 4. Advantages, deficiencies and possible problems within the process are discussed. Furthermore, solutions for solving the identified shortcomings are suggested.

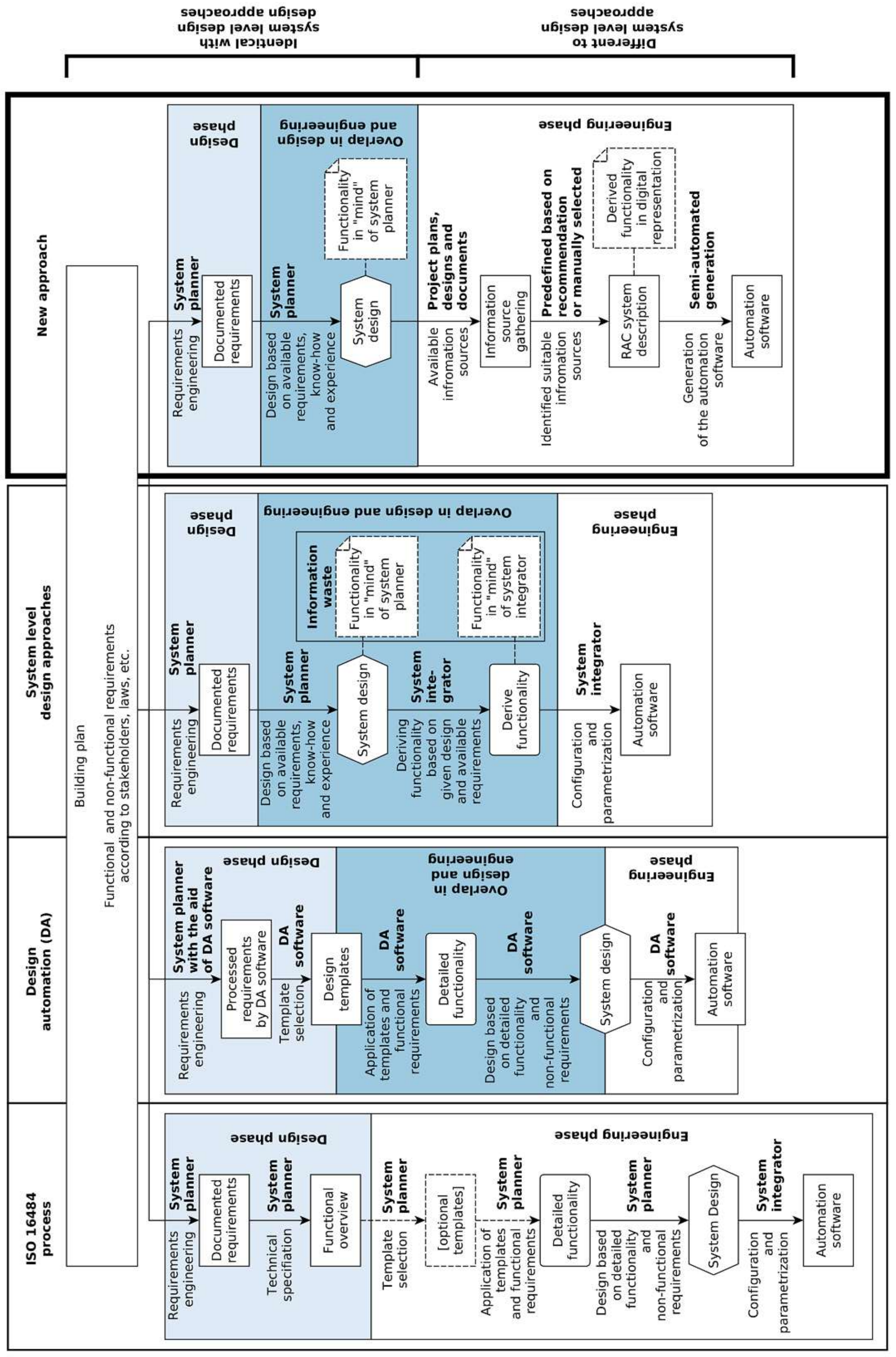

ISO 16484 process
Design automation (DA)
System level design approaches
New approach
Building plan
Functional and non-functional requirements according to stakeholders, laws, etc.
Identical with system level design approaches
Different to system level design approaches
Design phase
Engineering phase
Overlap in design and engineering
Requirements engineering
System planner
Documented requirements
Technical specification
Functional overview
Template selection
[optional templates]
Application of templates and functional requirements
Detailed functionality
Design based on detailed functionality and non-functional requirements
System Design
System integrator
Configuration and parametrization
Automation software
System planner with the aid of DA software
Processed requirements by DA software
DA software
Design templates
Design based on available requirements, know-how and experience
System design
System integrator
Deriving functionality based on design given design and available requirements
Derive functionality
Information waste
Functionality in "mind" of system planner
Functionality in "mind" of system integrator
Available infromation sources
Project plans, designs and documents
Information source gathering
Identified suitable infromation sources
Predefined based on recommendation or manually selected
RAC system description
Derived functionality in digital representation
Generation of the automation software
Semi-automated generation

Figure 6.27: Overview and comparison of existing BAS/RAC software creation approaches and the new process

Figure 6.27 visualizes the RAC system design approaches identified in Chapter 4 and the new process defined in Section 5.5. As can be seen in the figure, until the *System design* task, the generalized process defined in this thesis is identical with system level design approaches. Proceeding to the *Engineering phase*, the new process deviates from system level design approaches. In Table 6.4, the new process is compared with the existing approaches based on important characteristics for this thesis as identified in Section 4.2. These characteristics and findings are discussed in the following.

Applicability for buildings with rooms of various size and functionality
The proposed approach does not require the use of templates or specific tools in comparison to DA automation methods. As the RAC system is designed manually and the generalized process takes effect after the design, all kinds of building structures and layouts can be created by system designers, similar to system level design approaches.

Flexible sequence of tasks and their information output in the design phase
The design phase in the new process is identical with system level design approaches. The RAC system is designed manually during this phase. Time saving actions or business dependent deviations from the proposed sequence of tasks are therefore possible within this phase, as described in Section 4.3.
However, the proposed process requires, that the design phase is finished and the system design is available before the engineering phase is launched. Furthermore, most tasks within the engineering phase are not interchangeable, as they depend on each other.

Semi-automated RAC system software creation
Similar to DA approaches, an automated generation of RAC system software is possible if all required information is available in machine-readable and interpretable form, including functional and non-functional requirements. Having the information represented in an ontology as described in Chapter 3, a semi-automated generation of RAC system software can be conducted. Section 6.1.9 provides an example for this procedure.

Functionality not required before system design
In contrast to the ISO 16484 process and DA approaches, the new approach does not require planning the system functionality before designing the RAC system. The functionality is defined in parallel with the system design, as it is the case in system level design approaches. Later in the process, the functionality is derived from available information sources, as described in Chapter 5.

Applicable without tool-driven requirements engineering
The new approach does not require the utilization of specific tools in the requirements engineering phase of a project. This is due to the fact, that the system is defined manually by system engineers in contrast to DA approaches. Though, machine-readable requirements can be used to automate the creation of the digital representation ontology, as introduced in Chapter 3.

Characteristic	**ISO 16484 process**	**Design automation**	**System level design approach**	**New approach**	**Importance & relevance for this thesis**
Applicability for buildings with rooms of various size and functionality	✓	X	✓	✓	+++
Flexible sequence of tasks and their information output in the design phase	~	X	✓	✓	+++
Semi-automated RAC system software creation	X	✓	X	✓	++
Functionality not required before system design	X	X	✓	✓	+
Applicable without tool-driven requirements engineering	✓	X	✓	✓	+
Utilization of templates possible	✓	✓	X	✓	+
No specific software required for system functionality planning	✓	X	✓	✓	+

Table 6.4: Comparison of the new process with RAC system design approaches

Utilization of templates possible
Templates for common structures and configurations such as office rooms can be created within the proposed process. They can help to accelerate the functionality derivation and RAC system software generation by reusing knowledge and existing configurations of identical rooms. Though, it is not required to create templates in the new approach.

No specific software required for system functionality planning
The system functionality is not planned explicitly, but it is derived from available information sources after the RAC system design. Therefore, no specific tools or software are required for documenting or planning the system functionality in the design phase.

6.2.1 Advantages and benefits

Keeping system level design approach benefits while minimizing information waste
The newly defined process inherits important benefits of system level design approaches, as it is built upon them. Mainly, the flexibility in the early phase of projects, which characterizes system level design approaches, is retained. This is due to the fact, that the new approach only takes effect after the design phase. While keeping flexibility, information created in earlier phases of projects is reused to minimize information waste. This includes system design documents, documented requirements and building plans. Existing, available information is reused as input for deriving the functionality of already designed RAC systems.

Little effort for generating RAC system software after design phase
The efficiency of the new process is dependent on the quality, structure and form of information available from previous project steps. Most of the required information can be derived from information sources as defined in Section 6.1.2, which provide knowledge in machine-readable form. If such sources are available, little to no manual intervention is required to generate RAC system descriptions and software.

Reuse of generated configurations
In the newly defined process, systems are described in a machine-readable digital representation ontology (see Chapter 3). This knowledge is used to generate RAC system configurations, similar to DA approaches. Such system descriptions and configurations can be reused for identical building structures, e.g. office-rooms. This may help to reduce the necessary effort for creating configurations in contrast to manual implementations. Furthermore, it helps to reduce information waste by reusing already available knowledge.

Possible reduction of human errors
The newly created process emphasizes the reuse of already available information and knowledge. Even if information cannot be automatically reused, because it is not available in machine-readable form, the new approach leads system planners and integrators through the different tasks of the process. This can help to reduce failures and human errors,

which may occur more frequently when manually creating functional descriptions and system configurations.

6.2.2 Disadvantages and shortcomings

Possibly more effort required in case of insufficient information sources
The proposed approach is based on the reuse of information created in the design phase of building automation projects. If available information is limited or not machine-readable, it has to be manually added to the digital system representation defined in Chapter 3. This can lead to additional effort for creating the knowledge base in comparison to other approaches such as the ISO 16484 process or system level design approaches. Thus, after adding the system description to the digital representation ontology, reusing this information for similar or identical systems is possible with little to no effort. This can help to save time in later phases of BAS projects.

Dependent on semantic device descriptions
In order to automatically create system configurations based on the digital representation of a system, semantic device descriptions are utilized. These descriptions contain information about the functional capabilities of devices, as described in Section 5.2.3.2. The semantic description of the functionality is independent from the standard and protocol of the device.
To utilize this method, semantic device packages have to be provided e.g. by vendors or standardization groups, which is currently not the case. Such descriptions are also used in DA approaches, and it is suggested to make them a mandatory delivery component of BAS devices in the future. [98]

VDI 3813 functions can be insufficient
The new process is based on function blocks defined in the VDI 3813 standard. The standard does not claim to provide a complete set of BAS functions. Therefore, additional function blocks may be needed in order to describe specific BAS system characteristics. They can be added manually to a function block knowledge base and imported as needed in digital system representations used in the new process.
Though the approach defined in this thesis is based on VDI 3813 functions, it is not limited to this standard. For example, ISO 16484 or VDI 3814 function blocks can also be used when taking in the effort to make them compatible with each other.

Not fully automated process
Similar to the other approaches presented in this thesis, the newly defined process is not fully automated. Based on the available information and its structure, manual intervention may be necessary to create the desired output of a task. By reusing knowledge from previous steps, the effort required to complete subsequent tasks of the process is reduced as much as possible. The process is intended to minimize the burden on experts when creating RAC system descriptions and configuration software.

CHAPTER 7

Conclusion

7.1 Summary

In this thesis, a new approach for creating RAC system software has been presented. It incorporates concepts of established BAS design processes in order to minimize information waste and reuse available information sources. The key elements of the process defined in this thesis are summarized in the following.

In order to allow an automated creation of RAC system software, a description of the system has to be available in a digital, machine-readable form. In Chapter 3, the required information for describing RAC systems and their functionality are outlined. This includes considerations for representing functions, devices, locations and other aspects. A new digital representation ontology is introduced, which satisfies the defined requirements. Concepts of existing BAS representation ontologies have been implemented, adapted and extended for this purpose.
As the next step, common processes for creating RAC systems software are analyzed. This includes a process defined in part 1 of the ISO 16484, which comprises all phases of a BAS project, starting with the design of the system. Second, DA methods, which rely on machine-readable requirement descriptions, are analyzed. At last, a less structured approach, which is especially suitable for projects with time or cost constraints, is inspected. This approach is referred to as *system level design approach* in this work. The three selected processes are compared based on aspects which are relevant for automating the configuration of BAS system software, while keeping their applicability in mind. Disadvantages and possible information waste within the discussed approaches are outlined in Chapter 5. Based on the findings, the system level design approach was selected as basis for a new process for generating RAC system software presented in this thesis. However, advantages and possible information sources utilized in the other analyzed approaches are also considered to optimize the new method.

Possible information sources, which provide the knowledge required to create a digital representation of a RAC, are identified in Section 5.2. This includes the utilization of digital building structure documents such as BIM models and documented requirements of the envisioned RAC system. Furthermore, information available through system plan documents, including device lists, are gathered. Based on the identified information sources, a workflow for creating a digital RAC system description is introduced. This description is subsequently used to generate the desired RAC system software. In the software generation step, so-called *semantic device descriptions* are utilized to map the functionality described in the digital representation to the actual, installed devices of a RAC system. Such descriptions are also an important part of DA approaches, as described in Section 5.2.3.2. Based on the findings, a new generalized process for creating RAC system software is defined in Section 5.5. This process incorporates three tasks: information source gathering, RAC system description and RAC system software generation.
The applicability of the introduced process is evaluated in Chapter 6. Therefore, an example RAC system description taken from part 2 of the VDI 3813 guideline is used to generate RAC system software using the new process. Each task of the process is evaluated and possible problems, such as insufficient information sources, are highlighted. Advantages and possible drawbacks of the new process in comparison with the other approaches are described in Sections 6.2 and 6.2.1.
The process defined in this thesis improves upon common approaches for RAC system software creation and configuration. The main benefits of the identified processes are been implemented in the new approach, while possible drawbacks are mitigated. The flexibility in the design phase of RAC system projects, which is distinctive for system level design approaches, is still given. However, information waste which occurs in system level design approaches is reduced by reusing available knowledge and creating a digital representation of the desired system. Using the concepts of semantic device descriptions and function schematics, a semi-automated generation of RAC system software can be conducted on basis of the defined workflow.

7.2 Future work

The process presented in this thesis provides a basis towards automating BAS configurations. In the following section, future work and pending research questions are discussed.
In this thesis, the efficiency and possible cost and time savings by implementing the new process defined in Section 5.5 are not quantified. To compare these aspects with measurements of other existing approaches, research studies could be conducted in the future. For example, teams can be asked to develop configurations for identical RAC systems while utilizing different approaches such as DA, the ISO 16484 process, system level design approaches and the newly defined process. Thereby, cost and time differences in each project phase are documented in order to classify the viability and efficiency of the new process in comparison with established approaches.

As discussed in Section 6.2.1, the ability to reuse configurations created by utilizing the new process can be helpful to minimize information waste. In addition to that, reusing designs of common building structures in form of templates, e.g. office-rooms, may further be explored. The templates would be used as an additional input source for creating a digital RAC system representation as described in Chapter 3. This should help to save time and effort, while keeping the information waste for describing the functionality of RAC systems minimal.

In the future, the digital representation of a RAC system may be used to validate selected devices and system designs. For example, the ability to propose alternative or more energy efficient design solutions can be considered. Furthermore, cheaper or better fitting devices, which provide the same functional capabilities based on their semantic device descriptions, could be suggested to the system designer. It may also be possible to propose a device which includes several required functions to replace multiple devices which only support one, if so desired by a system planner or financially reasonable. In addition, possible interoperability problems of devices in the designed system may be detected and reported in course of the process application, before launching the engineering phase and creating a RAC system software configuration.

In case of failure, the digital RAC system description can be used to identify the cause of problems by checking known relations between devices and functions. Furthermore, the digital representation could be extended with error tracking capabilities in order to provide support during the operation of RAC systems. Knowledge about common failures in RAC system designs may also be utilized to implement a validation step of system designs in the new process.

List of Figures

List of Tables

Acronyms

AMAAD Adaptable Methodology for Automation Application Development. 20

API Application Programming Interface. 31

BA Building Automation. 44, 45, 59, 64

BACnet Building Automation and Control Networking. 14, 15

BACS Building automation and control systems. 10

BAS Building Automation System. 1, 2, 5, 10–16, 18–20, 23–26, 33, 34, 45, 47, 48, 50–53, 57, 59, 60, 62–64, 101, 103, 104

BIBB BACnet Interoperability Building Blocks. 15

BIM Building Information Modeling. 2, 3, 30, 59, 78, 104

CAAD Computer Aided Architectural Design. 58

CAD Computer Aided Design. 58, 59, 64

CPS Cyber-Physical System. 30

DA Design Automation. 19, 20, 47, 51–53, 57, 59, 60, 62–64, 98, 100, 101, 103, 104

DIN Deutsches Institut für Normung. 62

HVAC Heating, Ventilation and Air Conditioning. 2, 10, 14, 42

IEC International Electrotechnical Commission. 64

IFC Industry Foundation Classes. 58, 59, 78, 79

IMPROVE Integrated Method for PROcess Value Evaluation. 20, 21

IoT Internet of Things. 18, 30, 33

ISO International Organization for Standardization. 10, 13, 14, 24–26, 44–48, 50–53, 57, 59, 61–64, 98, 101, 103

KBE Knowledge Based Engineering. 20, 52

M-BACS Management Building Automation and Control Systems. 24, 25

MEP Mechanical, Electrical and Plumbing. 2

MOKA Methodology and tools Oriented to Knowledge-based Applications. 20

OWL Web Ontology Language. 17, 32

PICS Protocol Implementation Conformance Statement. 14, 15

RAC Room Automation and Controls. 1, 3, 4, 18, 23–27, 30, 32–34, 43, 45, 53, 55, 56, 62–65, 67–70, 72, 73, 75, 76, 78, 79, 82, 89, 90, 95, 96, 98, 100, 101, 103–105

RDF Resource Description Framework. 8, 10, 17, 32

RDFS Resource Description Framework Schema. 8, 17

SAC System Automation and Controls. 24, 25

SFPT Standard Functional Profile Templates. 16

SKOS Simple Knedge Organization System. 35

TTL Turtle - Terse RDF Triple Language. 17

URI Universal Resource Identifier. 16, 17

VDI Verein Deutscher Ingeneure (ger.: Society of German Engineers). 10, 12, 13, 24–26, 30, 35, 47, 51, 58, 60, 62, 66–69, 75, 76, 78, 81–83, 85–90, 92, 93, 95, 96, 101, 104

XML Extensible Markup Language. 10, 58

Bibliography

[1] J. Haase, G. Zucker, and M. Alahmad, “Energy efficient building automation: A survey paper on approaches and technologies for optimized building operation,” in *Industrial Electronics Society, IECON 2014-40th Annual Conference of the IEEE*. IEEE, 2014, pp. 5350–5356.

[2] B. Klauer, J. Haase, D. Meyer, and M. Eckert, “Wireless sensor/actuator device configuration by NFC with secure key exchange,” in *AFRICON, 2017 IEEE*. IEEE, 2017, pp. 473–478.

[3] J. Schein, “An information model for building automation systems,” *Automation in construction*, vol. 16, no. 2, pp. 125–139, 2007.

[4] A. Demeure, S. Caffiau, E. Elias, and C. Roux, “Building and using home automation systems: a field study,” in *International Symposium on End User Development*. Springer, 2015, pp. 125–140.

[5] VDI, “VDI 3813 1 - Building automation and control systems (BACS) - Fundamentals for room control,” Verein Deutscher Ingenieure, Düsseldorf, DE, Guideline, 2011.

[6] ISO, “ISO 16484-1 - Building automation and control systems (BACS) — Part 1: Project specification and implementation,” International Organization for Standardization, Geneva, CH, Standard, 2011.

[7] H. Dibowski, J. Ploennigs, and K. Kabitzsch, “Automated design of building automation systems,” *IEEE Transactions on Industrial Electronics*, vol. 57, no. 11, pp. 3606–3613, 2010.

[8] J. Ploennigs, H. Dibowski, U. Ryssel, and K. Kabitzsch, “Holistic design of wireless building automation systems,” in *2011 IEEE 16th Conference on Emerging Technologies & Factory Automation (ETFA 2010)*, 2011, pp. 1–9.

[9] M. Günther, “Durchgängiges modellbasiertes Engineering von Gebäudeautomationssystemen,” Helmut-Schmidt-Universität der Bundeswehr Hamburg, Dr.-Ing. Dissertation, 2018.

[10] J. Lu and K. Whitehouse, “Smart blueprints: automatically generated maps of homes and the devices within them,” in *International Conference on Pervasive Computing*. Springer, 2012, pp. 125–142.

[11] L. Schor, P. Sommer, and R. Wattenhofer, “Towards a zero-configuration wireless sensor network architecture for smart buildings,” in *Proceedings of the First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*. ACM, 2009, pp. 31–36.

[12] Z.-Z. Hu, P.-L. Tian, S.-W. Li, and J.-P. Zhang, “BIM-based integrated delivery technologies for intelligent MEP management in the operation and maintenance phase,” *Advances in Engineering Software*, vol. 115, pp. 1–16, 2018.

[13] L. Zhang, Z. Dong, L. Zhang, A. Zu, and L. Yang, “Cost Management of Mechanical and Electrical Engineering Project Based on BIM Technology,” *Smart Construction Research*, vol. 2, no. 4, 2018.

[14] J. R. Jupp, “Incomplete BIM implementation: Exploring challenges and the role of product lifecycle management functions,” in *IFIP International Conference on Product Lifecycle Management*. Springer, 2013, pp. 630–640.

[15] B. Detlor, “Viewpoint: Information management,” *International Journal of Information Management: The Journal for Information Professionals*, vol. 30, pp. 103–108, 04 2010.

[16] X. C. Lin, “Three Perspectives of Information Management,” in *2010 3rd International Conference on Information Management, Innovation Management and Industrial Engineering*, vol. 3, 2010, pp. 284–288.

[17] N. S. A. Karim and R. Hussein, “Managers’ perception of information management and the role of information and knowledge managers: The Malaysian perspectives,” *International Journal of Information Management*, vol. 28, no. 2, pp. 114–127, 2008.

[18] R. Blumberg and S. Atre, “The problem with unstructured data,” *Dm Review*, vol. 13, no. 42-49, p. 62, 2003.

[19] R. Sint, S. Schaffert, S. Stroka, and R. Ferstl, “Combining unstructured, fully structured and semi-structured information in semantic wikis,” in *CEUR Workshop Proceedings*, vol. 464, 2009, pp. 73–87.

[20] S. Abiteboul, “Querying semi-structured data,” in *International Conference on Database Theory*. Springer, 1997, pp. 1–18.

[21] T. Berners-Lee, J. Hendler, and O. Lassila, “The semantic web,” *Scientific american*, vol. 284, no. 5, pp. 34–43, 2001.

[22] ISO, "ISO 16484-2 - Building automation and control systems (BACS) — Part 2: Hardware," International Organization for Standardization, Geneva, CH, Standard, 2004.

[23] S. C. Hui, "Latest trends in building automation and control systems," in *CAI symposium on intelligent facility management and intelligent transport, Hong Kong, 28th March*, 2007.

[24] O. Maryasin, "Home Automation System Ontology for Digital Building Twin," in *2019 XXI International Conference Complex Systems: Control and Modeling Problems (CSCMP)*. IEEE, 2019, pp. 70–74.

[25] W. Kastner, G. Neugschwandtner, S. Soucek, and H. M. Newman, "Communication systems for building automation and control," *Proceedings of the IEEE*, vol. 93, no. 6, pp. 1178–1203, 2005.

[26] VDI, "VDI 3814 1 - Building automation and control systems (BACS) - Fundamentals," Verein Deutscher Ingenieure, Düsseldorf, DE, Guideline, 2019.

[27] R. Yang and L. Wang, "Multi-objective optimization for decision-making of energy and comfort management in building automation and control," *Sustainable Cities and Society*, vol. 2, no. 1, pp. 1–7, 2012.

[28] P. Domingues, P. Carreira, R. Vieira, and W. Kastner, "Building automation systems: Concepts and technology review," *Computer Standards & Interfaces*, vol. 45, pp. 1–12, 2016.

[29] W. Granzer and W. Kastner, "Information modeling in heterogeneous building automation systems," in *2012 9th IEEE International Workshop on Factory Communication Systems*. IEEE, 2012, pp. 291–300.

[30] R. Vieira, P. Carreira, P. Domingues, and A. A. Costa, "Supporting building automation systems in BIM/IFC: reviewing the existing information gap," *Engineering, Construction and Architectural Management*, 2020.

[31] VDI – Verein Deutscher Ingenieure e.V. Accessed on February 03, 2021. [Online]. Available: https://www.vdi.de/

[32] VDI, "VDI 3813 2 - Building automation and control systems (BACS) - Room control functions (RA functions)," Verein Deutscher Ingenieure, Düsseldorf, DE, Guideline, 2011.

[33] VDI, "VDI 3813 3 - Building automation and control systems (BACS) - Application examples for room types and function macros of room automation and control," Verein Deutscher Ingenieure, Düsseldorf, DE, Guideline, 2015.

[34] VDI, “VDI 3814 2.1 - Building automation and control systems (BACS) - Planning - Requirements planning, concept of operation and specifications sheet,” Verein Deutscher Ingenieure, Düsseldorf, DE, Guideline, 2019.

[35] VDI, “VDI 3814 2.2 - Building automation and control systems (BACS) - Planning - Planning content, system integration and interfaces,” Verein Deutscher Ingenieure, Düsseldorf, DE, Guideline, 2019.

[36] VDI, “VDI 3814 2.3 - Building automation and control systems (BACS) - Planning - Concept of operation and user interfaces,” Verein Deutscher Ingenieure, Düsseldorf, DE, Guideline, 2019.

[37] VDI, “VDI 3814 3.1 - Building automation and control systems (BACS) - BACS functions - Automation functions,” Verein Deutscher Ingenieure, Düsseldorf, DE, Guideline, 2019.

[38] VDI 3814 3.2 - Building Automation and Control Systems (BACS); Functions;. Accessed on February 03, 2021. [Online]. Available: https://www.vdi.de/richtlinien/details/vdi-3814-blatt-32-gebaeudeautomation-ga-funktionskatalog-makrofunktionen

[39] VDI, “VDI 3814 4.1 - Building automation and control systems (BACS) - Methods and tools for planning, building, and acceptance tests - Identification, addressing, and lists,” Verein Deutscher Ingenieure, Düsseldorf, DE, Guideline, 2019.

[40] VDI, “VDI 3814 4.2 - Building automation and control systems (BACS) - Methods and tools for planning, building acceptance tests - Requirements, content of planning, and system integration,” Verein Deutscher Ingenieure, Düsseldorf, DE, Guideline, 2020.

[41] VDI, “VDI 3814 6 - Building automation and control systems (BACS) - Graphical description of logic control tasks,” Verein Deutscher Ingenieure, Düsseldorf, DE, Guideline, 2008.

[42] ISO, “ISO 16484-3 - Building automation and control systems (BACS) — Part 3: Functions,” International Organization for Standardization, Geneva, CH, Standard, 2005.

[43] ISO, “ISO 16484-5 - Building automation and control systems (BACS) — Part 5: Data communication protocol,” International Organization for Standardization, Geneva, CH, Standard, 2017.

[44] ISO, “ISO 16484-6 - Building automation and control systems (BACS) — Part 6: Data communication conformance testing,” International Organization for Standardization, Geneva, CH, Standard, 2014.

[45] G. Mao, B. Fidan, and B. D. Anderson, “Wireless sensor network localization techniques,” *Computer networks*, vol. 51, no. 10, pp. 2529–2553, 2007.

[46] R. L. Moses, D. Krishnamurthy, and R. M. Patterson, "A self-localization method for wireless sensor networks," *EURASIP Journal on Advances in Signal Processing*, vol. 2003, no. 4, pp. 1–11, 2003.

[47] M. Shimosaka, O. Saisho, T. Sunakawa, H. Koyasu, K. Maeda, and R. Kawajiri, "ZigBee based wireless indoor localization with sensor placement optimization towards practical home sensing," *Advanced Robotics*, vol. 30, no. 5, pp. 315–325, 2016.

[48] M. W. Ahmad, M. Mourshed, D. Mundow, M. Sisinni, and Y. Rezgui, "Building energy metering and environmental monitoring–A state-of-the-art review and directions for future research," *Energy and Buildings*, vol. 120, pp. 85–102, 2016.

[49] J. Bhatt and H. Verma, "Design and development of wired building automation systems," *Energy and Buildings*, vol. 103, pp. 396–413, 2015.

[50] KNX standard. Accessed on February 02, 2021. [Online]. Available: https://www.knx.org/

[51] B. Aschendorf, *Energiemanagement durch Gebäudeautomation: Grundlagen-Technologien-Anwendungen*. Springer-Verlag, 2014.

[52] Y. Kitamura and R. Mizoguchi, "Ontology-based description of functional design knowledge and its use in a functional way server," *Expert Systems with Applications*, vol. 24, no. 2, pp. 153–166, 2003.

[53] I. Horrocks, P. F. Patel-Schneider, and F. Van Harmelen, "From SHIQ and RDF to OWL: The making of a web ontology language," *Journal of web semantics*, vol. 1, no. 1, pp. 7–26, 2003.

[54] W3C. (2014) RDF 1.1 Turtle - Terse RDF Triple Language. Accessed on April 29, 2020. [Online]. Available: https://www.w3.org/TR/turtle/

[55] S. C. for Biomedical Informatics Research. (2016) Protégé - A free, open-source ontology editor and framework for building intelligent systems. Accessed on April 29, 2020. [Online]. Available: https://protege.stanford.edu/

[56] M. Uschold and M. King, *Towards a methodology for building ontologies*. Citeseer, 1995.

[57] T. Qiu, X. Chen, H. Huang, and Q. Liu, "Ontology capture based on granular computing," in *Sixth international conference on intelligent systems design and applications*, vol. 1. IEEE, 2006, pp. 770–774.

[58] TNO. (2014) BrickSchema - A uniform metadata schema for buildings. Accessed on May 02, 2020. [Online]. Available: https://sites.google.com/site/smartappliancesproject/ontologies/reference-ontology

[59] TNO. (2015) SAREF4EE: the extension of SAREF for EEBus and Energy@Home. Accessed on May 02, 2020. [Online]. Available: https://ontology.tno.nl/saref4ee/

[60] S. Nagare. (2014) Smart Appliances REFerence (SAREF) ontology - Smart Appliances Project. Accessed on May 02, 2020. [Online]. Available: https://brickschema.org/

[61] T. G. Stavropoulos, D. Vrakas, D. Vlachava, and N. Bassiliades, "BOnSAI: a smart building ontology for ambient intelligence," in *Proceedings of the 2nd international conference on web intelligence, mining and semantics*, 2012, pp. 1–12.

[62] D. Bonino and L. D. Russis. (2015) DogOnt - Ontology. Accessed on May 02, 2020. [Online]. Available: http://iot-ontologies.github.io/dogont/

[63] C. Reinisch, M. J. Kofler, and W. Kastner, "ThinkHome: A smart home as digital ecosystem," in *4th IEEE International Conference on Digital Ecosystems and Technologies*. IEEE, 2010, pp. 256–261.

[64] B. Butzin, F. Golatowski, and D. Timmermann, "A survey on information modeling and ontologies in building automation," in *IECON 2017-43rd Annual Conference of the IEEE Industrial Electronics Society*. IEEE, 2017, pp. 8615–8621.

[65] M. Compton, P. Barnaghi, L. Bermudez, R. GarcíA-Castro, O. Corcho, S. Cox, J. Graybeal, M. Hauswirth, C. Henson, A. Herzog *et al.*, "The SSN ontology of the W3C semantic sensor network incubator group," *Journal of Web Semantics*, vol. 17, pp. 25–32, 2012.

[66] A. Gyrard, C. Bonnet, and K. Boudaoud, "Enrich machine-to-machine data with semantic web technologies for cross-domain applications," in *2014 IEEE World Forum on Internet of Things (WF-IoT)*. IEEE, 2014, pp. 559–564.

[67] L. Xue, Y. Liu, P. Zeng, H. Yu, and Z. Shi, "An ontology based scheme for sensor description in context awareness system," in *2015 IEEE International Conference on Information and Automation*. IEEE, 2015, pp. 817–820.

[68] M. Bermudez-Edo, T. Elsaleh, P. Barnaghi, and K. Taylor, "IoT-Lite: a lightweight semantic model for the Internet of Things," in *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*. IEEE, 2016, pp. 90–97.

[69] H. van der Schaaf and R. Herzog, "Mapping the OGC SensorThings API onto the OpenIoT Middleware," in *Interoperability and Open-Source Solutions for the Internet of Things*. Springer, 2015, pp. 62–70.

[70] R. Agarwal, D. G. Fernandez, T. Elsaleh, A. Gyrard, J. Lanza, L. Sanchez, N. Georgantas, and V. Issarny, “Unified IoT ontology to enable interoperability and federation of testbeds,” in *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)*. IEEE, 2016, pp. 70–75.

[71] W. J. Verhagen, B. de Vrught, J. Schut, and R. Curran, “A method for identification of automation potential through modelling of engineering processes and quantification of information waste,” *Advanced Engineering Informatics*, vol. 29, no. 3, pp. 307–321, 2015.

[72] AUTERAS - Room Automation Design Tool. Accessed on February 02, 2021. [Online]. Available: https://www.auteras.de/

[73] A. McGibney, S. Rea, M. Lehmann, S. Thior, S. Lesecq, M. Hendriks, C. Gardeux, L. T. Mai, F. Pacull, J. Ploennigs *et al.*, “A systematic engineering tool chain approach for self-organizing building automation systems,” in *IECON 2013-39th Annual Conference of the IEEE Industrial Electronics Society*. IEEE, 2013, pp. 7696–7701.

[74] K. Oldham, S. Kneebone, M. Callot, A. Murton, and R. Brimble, “MOKA-A Methodology and tools Oriented to Knowledge-based engineering Applications,” in *Proceedings of the Conference on Integration in Manufacturing*, 1998, pp. 198–207.

[75] C. Van der Velden, C. Bil, and X. Xu, “Adaptable methodology for automation application development,” *Advanced Engineering Informatics*, vol. 26, no. 2, pp. 231–250, 2012.

[76] VDI 3814 4.3 - Building automation and control systems (BACS) - Methods and tools for planning, building, and acceptance tests - BACS automation scheme, BACS function list, BACS functional description. Accessed on February 03, 2021. [Online]. Available: https://www.vdi.de/richtlinien/details/vdi-3814-blatt-43-gebaeudeautomation-ga-methoden-und-arbeitsmittel-fuer-planung-ausfuehrung-und-uebergabe-ga-automationsschema-ga-funktionsliste-ga-funktionsbeschreibung

[77] C. Steinmetz, A. Rettberg, F. G. C. Ribeiro, G. Schroeder, and C. E. Pereira, “Internet of Things Ontology for Digital Twin in Cyber Physical Systems,” in *2018 VIII Brazilian Symposium on Computing Systems Engineering (SBESC)*. IEEE, 2018, pp. 154–159.

[78] R. Rosen, G. Von Wichert, G. Lo, and K. D. Bettenhausen, “About the importance of autonomy and digital twins for the future of manufacturing,” *IFAC-PapersOnLine*, vol. 48, no. 3, pp. 567–572, 2015.

[79] S. Malakuti and S. Grüner, “Architectural aspects of digital twins in IIoT systems,” in *Proceedings of the 12th European Conference on Software Architecture: Companion Proceedings*, 2018, pp. 1–2.

[80] N. F. Noy, D. L. McGuinness *et al.*, “Ontology development 101: A guide to creating your first ontology,” 2001.

[81] D. Zhang, L. Wang, H. Xiong, and B. Guo, “4W1H in mobile crowd sensing,” *IEEE Communications Magazine*, vol. 52, no. 8, pp. 42–48, 2014.

[82] O. W. Group. (2012) Web Ontology Language - OWL. Accessed on April 29, 2020. [Online]. Available: https://www.w3.org/OWL/

[83] G. Bajaj, R. Agarwal, P. Singh, N. Georgantas, and V. Issarny, “A study of existing Ontologies in the IoT-domain,” *arXiv preprint arXiv:1707.00112*, 2017.

[84] N. Seydoux, K. Drira, N. Hernandez, and T. Monteil, “IoT-O, a core-domain IoT ontology to represent connected devices networks,” in *European Knowledge Acquisition Workshop*. Springer, 2016, pp. 561–576.

[85] D. Tsarkov and I. Palmisano, “Chainsaw: a Metareasoner for Large Ontologies,” in *ORE*, 2012.

[86] S. B. Alistair Miles. (2009) Simple Knowledge Organization System Namespace Document - SKOS. Accessed on May 05, 2020. [Online]. Available: https://www.w3.org/2009/08/skos-reference/skos.html

[87] U. Ryssel, H. Dibowski, and K. Kabitzsch, “Generation of function block based designs using semantic web technologies,” in *2009 IEEE Conference on Emerging Technologies & Factory Automation*. IEEE, 2009, pp. 1–8.

[88] M. Günther, P. Diekhake, A. Scholz, P. P. Schmidt, U. Becker, and A. Fay, “Seamless model-based engineering of building automation systems,” *at-Automatisierungstechnik*, vol. 64, no. 6, pp. 490–499, 2016.

[89] DIN, “DIN EN 15232 1 - Energy performance of buildings - Part 1: Impact of Building Automation, Controls and Building Management,” Deutsches Institut für Normung, Berlin, DE, Norm, 2017.

[90] S. Runde, A. Heidemann, A. Fay, and P. Schmidt, “Engineering of building automation systems—State-of-the-art, deficits, approaches,” in *2010 IEEE 15th Conference on Emerging Technologies & Factory Automation (ETFA 2010)*. IEEE, 2010, pp. 1–8.

[91] E. Rigger, K. Shea, and T. Stankovic, “Task categorisation for identification of design automation opportunities,” *Journal of Engineering Design*, vol. 29, no. 3, pp. 131–159, 2018.

[92] M. Günther, A. Scholz, P. P. Schmidt, A. Fay, P. Diekhake, D. E. D. Fuentes, and U. Becker, “Requirements engineering and modelling for building automation systems,” in *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2016, pp. 1–8.

[93] J. Ploennigs, B. Hensel, H. Dibowski, and K. Kabitzsch, "Basont-a modular, adaptive building automation system ontology," in *IECON 2012-38th Annual Conference on IEEE Industrial Electronics Society*. IEEE, 2012, pp. 4827–4833.

[94] DIN, "DIN EN 18386-C - VOB Vergabe- und Vertragsordnung für Bauleistungen - Teil C: Allgemeine Technische Vertragsbedingungen für Bauleistungen (ATV) – Gebäudeautomation," Deutsches Instititut für Normung, Berlin, DE, Standard, 2016.

[95] F. Rieckhof, H. Dibowski, and K. Kabitzsch, "Formal validation techniques for Ontology-based Device Descriptions," in *2011 IEEE 16th Conference on Emerging Technologies & Factory Automation (ETFA 2010)*. IEEE, 2011, pp. 1–8.

[96] H. Dibowski, "Semantic interoperability evaluation model for devices in automation systems," in *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2017, pp. 1–6.

[97] D. E. D. Fuentes, U. Becker, P. Diekhake, M. Günther, A. Scholz, P. P. Schmidt, and A. Fay, "Evaluation and simulation of building automation systems based on their AutomationML description," in *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2016, pp. 1–6.

[98] S. Malakuti, J. Schmitt, and T. Gamer, "From Heterogeneity to Uniformity in Building Automation Systems via Semantic-based Engineering," in *2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1. IEEE, 2018, pp. 115–122.

[99] ETS standard. Accessed on February 02, 2021. [Online]. Available: https://www.knx.org/knx-de/fuer-fachleute/software/ets-5-professional/