MIoTA: Modeling IoT applications for air conditioning facilities with ADOxx

Benjamin Nast[1], Kurt Sandkuhl[1,2], Stefan Paulus[3] and Heiko Schiller[4]

[1] *Rostock University, Albert-Einstein-Str. 22, 18059 Rostock, Germany*
[2] *Jönköping University, Gjuterigatan 5, 55111 Jönköping, Sweden*
[3] *Dr. Diestel GmbH, Schonenfahrerstr. 6, 18057 Rostock, Germany*
[4] *schiller engineering, Ulmenstr. 50A, 22299 Hamburg, Germany*

Abstract

The development of Internet of Things (IoT) applications is a multifaceted and demanding exercise that requires developers to navigate through several complicated tasks and challenges. Primary issues in this area stem from the absence of standardization and the complexity and heterogeneity inherent to the IoT landscape. Model-Driven Development (MDD) is considered an effective technique for IoT application development. We argue that there is a need for methodologies of MDD that encompass system development and integration, as well as organizational aspects. To compensate for this lack, this work introduces MIoTA (Modeling IoT Applications for Air Conditioning Facilities), a modeling method supporting IoT application development in the field of air conditioning facilities. The ADOxx-based MIoTA tool was developed in an industrial use case. It contains a Domain-Specific Modeling Language (DSML) and various functionalities to support the modeling and development processes, so no specific IT skills are required at the application level. The approach has been evaluated practically within the case study and with quality criteria in previous work.

Keywords

Modeling Method, Domain-Specific Modeling Language, Model-Driven Development, Internet of Things, Digital Twin, MIoTA

1. Introduction

The development of IoT applications is a multifaceted and demanding exercise, mandating developers to navigate through an array of intricate tasks and challenges. A lack of standardization and the complexity and heterogeneity of the IoT are stated as significant issues in this field [1,2]. There is a substantial body of knowledge regarding technological and developmental dimensions encompassing the specification, design, and implementation of IoT solutions [3,4]. Model-Driven Development (MDD) is considered an effective technique for IoT application development [5].

Our observations show, however, that many enterprises, especially small and medium-sized enterprises (SMEs), are reluctant to invest in IoT initiatives. This hesitation is due to the perceived complexity of development processes and the challenges of devising effective organizational integration strategies [6].

The motivation for this work stems from the realization that existing model-based approaches for IoT solutions tend to prioritize software and system perspectives, revealing a gap in terms of integrating organizational and business model aspects. We argue that methodologies for MDD of IoT solutions need to be more tailored for effective application within SMEs, requiring comprehensive incorporation of organizational integration. This stance aligns with the findings of Fortino et al. [7], who emphasize that prevailing IoT development products, such as

BIR-WS 2023: BIR 2023 Workshops and Doctoral Consortium, 22nd International Conference on Perspectives in Business Informatics Research (BIR 2023), September 13-15, 2023, Ascoli Piceno, Italy
benjamin.nast@uni-rostock.de (B. Nast); kurt.sandkuhl@uni-rostock.de (K. Sandkuhl); stefan.paulus@dr-diestel.de (S. Paulus); info@schiller-engineering.de (H. Schiller)
0000-0003-4659-9840 (B. Nast); 0000-0002-7431-8412 (K. Sandkuhl)

methodologies, frameworks, platforms, and tools, predominantly focus on the software components of IoT systems.

This work introduces MIoTA (Modeling IoT Applications for Air Conditioning Facilities), a modeling method supporting IoT application development in the field of air conditioning facilities. It adopts the definition of modeling methods by Karagiannis and Kühn [8] and thus consists of a modeling language, which contains a notation and a meta-model, and a modeling procedure, which describes steps and guidelines for using the language to create valid models.

The ADOxx-based MIoTA tool contains a Domain-Specific Modeling Language (DSML) for air conditioning facilities and was developed in an industrial use case. Various functionalities support the modeling and development processes, so no specific IT skills are required at the application level. The DSML allows the employees to model the facilities with their domain knowledge. A familiar interface also allows one to enter the required configuration data, to classify the facilities. They can be exported (for example, sent directly to the AWS cloud) to configure services and prepare visualizations and subsequent data analysis, enabling energy optimizations. Within the scope of the industrial case, we practically validated the modeling language as such and also the way it can be used and supported by the tool. Quality criteria for evaluation were also applied in previous work.

The paper is structured as follows: The use case and resulting requirements for the development of the MIoTA tool are described in section 2. Section 3 describes the meta-model and visual notation of the modeling language. Section 4 is about modeling with the tool and how to use the functionalities. Furthermore, the evaluation of the tool is briefly described. Section 5 presents the experiences and recommendations for meta-model development. A conclusion of this work, as well as an outlook, is given in section 6.

2. The MIoTA modeling method

This section describes the case study and resulting requirements for the development of the MIoTA modeling method.

2.1. Use case

The MIoTA modeling method was developed in an industrial use case in collaboration with an SME that builds, operates, and maintains air conditioning facilities. Our goal was to improve the energy efficiency of air conditioning facilities by equipping them with as few sensors as possible.

An analysis of inspection reports showed that energy savings of up to 30% can be achieved in the majority of facilities using low-cost technologies. Other research (e.g., [9,10]) has confirmed these potential savings, which are made possible by simply identifying malfunctions in these systems.

Industrial air conditioning facilities are not mass-produced but rather designed individually for each client. Numerous components must work together in these facilities to operate effectively, and monitoring their internal operations requires a variety of sensors. However, interpreting the signal from these sensors is not straightforward. They often measure only an analog signal from zero to ten volts, which then needs to be converted into a physical unit of measurement.

Additionally, measurement arrangements vary according to the configurations of air conditioning facilities. Some include a cooling unit, and it may be used only for cooling or for both cooling and dehumidification. These differing configurations also impact our proposed solution.

With the growing automation of control technology, the volume of data is increasing rapidly. To operate facilities in an energy-efficient way, it is necessary to use intelligent data processing for both direct and indirect processes. Therefore, there is a significant need for system solutions for self-recognition, self-organized learning, and system control. In our use case, the existing facilities are equipped with minimal use of sensors. This results in an IoT-based system that not only creates the foundation for this purpose but also provides the basis for novel business

services. The planned IoT solution is intended to provide diagnostic assistance for potential improvements within the air conditioning facility, as well as for the operating processes of the enterprise under investigation in the case study. This solution must be able to process a large amount of data from different sources. It must also fit seamlessly into the day-to-day operations of the case study company and enable the delivery of innovative service offerings.

2.2. Requirements

In our previous work [11], we derived requirements for the overall system in collaboration with the use case company. The primary functionalities required of the system are maintenance support, detecting malfunctions and operational errors, predictive maintenance, and support the process of energetic inspections. Energetic inspections are legally mandated procedures (in Germany) to assess the energy efficiency of industrial-sized air conditioning facilities. Frequent inspections of operating facilities reveal deviations from determined energy efficiency and can only identify faults occurring at the inspection time. The system should therefore allow the evaluation of historical data. Using this data, maintenance intervals should be determined dynamically in accordance with the intensity of facility use, rather than conducting them at fixed intervals as was done previously. Based on the requirements, the necessary data, its origin or recording time, and the required measurement accuracy have been determined.

The implemented system should be user-friendly for the employees of our case study company, without presuming any IoT- or analysis-specific skills. Moreover, it should facilitate easy monitoring through alarms, diagrams, and clear (written) reports on data analysis. To achieve this goal, interfaces for configuring, operating, and maintaining are necessary.

The MIoTA modeling method must, therefore, provide a familiar interface (*requirement 1*) and enable the graphical representation of facilities (*requirement 2*). This should help technicians in the design of IoT solutions or in presentations to customers to explain issues such as upgrades, maintenance, or replacements. This is to be enabled by a DSML, which has a notation familiar to the employees.

One of our project partners has pointed out that there is significant potential for energy savings when comparing similar facility configurations. To achieve this and to address the heterogeneity, essential data must be collected from each facility. This so-called configuration data contains, e.g., information about the operator, components and their respective values, and sensors. Table 1 shows, as an example, a little excerpt from the configuration data. A data type, value range, and corresponding unit are specified. Sometimes, default values are provided, which can be used when data is missing to enable meaningful data analysis. The involved experts compiled the configuration data in a manner that allows inference of the necessary sensors for each facility. This data, when combined with the corresponding sensor data, allows the derivation of relevant energy-saving measures.

Table 1
Excerpt from configuration data

	Data Type	Value Range	Default Value	Unit
Operator				
Customer	String			
Object	String			
Supply Area	String			
Owner	String			
…				
Cal. Pr. Electricity	Integer	[0..3000]	400	[EUR/MWh]
Preheater				
Existing	Boolean			
Generation	Enumeration	Thermal/Electrical		
Power (if electrical)	Integer	[0..1000]		[kW]

This data not only informs the type of a facility. It also allows the application of appropriate calculation methods, according to the included components, for subsequent energy optimizations at a later stage. Energy consumption or costs, for example, can be determined from these values in conjunction with sensor data. The tool should have the function of collecting (*requirement 3*) and, exporting this data (*requirement 4*), and assigning unique IDs to the facilities, components, and sensors (*requirement 5*).

We have developed a tool that includes a DSML for air conditioning facilities to meet the requirements. This tool is intended to provide comprehensive support to technicians in designing IoT solutions. The development of this tool was done using the ADOxx meta-modeling platform [12]. It is a development and configuration platform for the implementation of modeling methods. With this platform, it is possible to realize a full-fledged modeling software that contains procedures and functionalities in the form of mechanisms and algorithms in addition to the modeling language.

3. Conceptualizing MIoTA

This section introduces the MIoTA meta-model, covering the concepts and relationships. The purpose is to expose the meta-model and facilitate the application and adoption of the method. We also describe the needed constraints to ensure the integrity of MIoTA models. Section 3.1 describes the abstract notation of our method based on the meta-model. The visual notation of the meta-model concepts is shown in section 3.2.

3.1. The MIoTA meta-model

The MIoTA meta-model is presented in Figure 1. It contains the concepts and relations. All abstract classes and methods are italicized and cannot be instantiated. The abstract class *MIoTA Object* is the most general, providing each concept with a *name*. An *Air Conditioning Facility* consists of *MIoTA Objects*. The *MIoTA Object* is a representation of *Air Type*, *Component*, *Sensor*, *Notes*, *Picture*, *Room*, *Configuration Data*, or *MIoTA Relation*. The representation of context is supported by the *MIoTA Object Attribute* element. It has a *name* and *type* and provides information about the *MIoTA Objects*. The relation classes that can be instantiated are connected to the relations through a line to demonstrate which can relate to which classes. A *Component* can be connected with *Components* and *Air Type* using the *Air Flow Direction* relation. A *Component* can also be connected with a *Sensor* using the *Connector* relation.

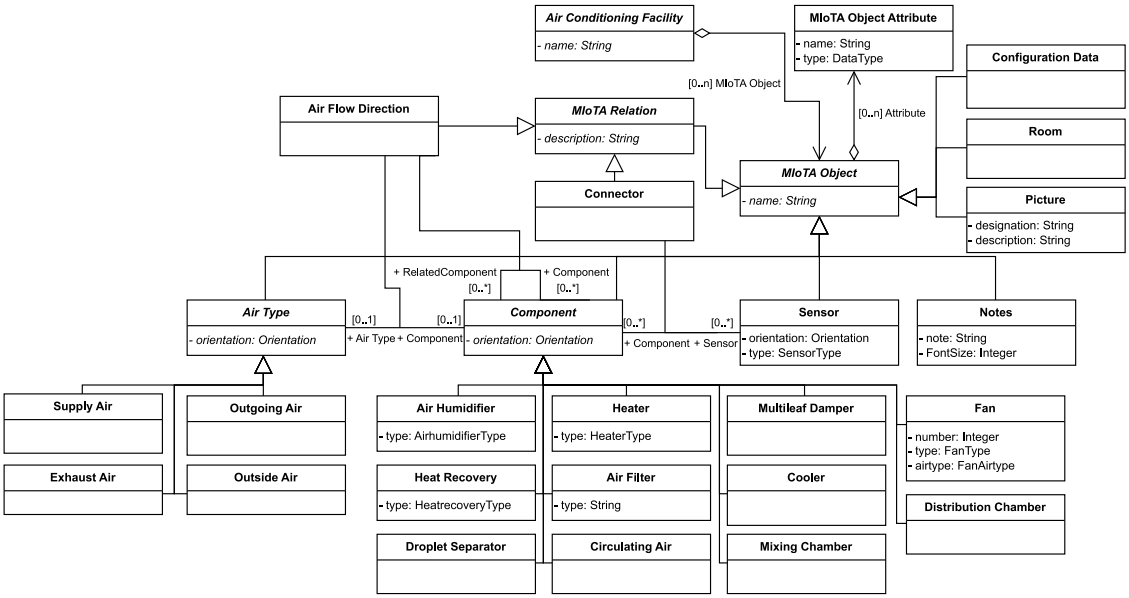

Figure 1: MIoTA meta-model

Air Type, *Component*, and *MIoTA Relation* are superclasses that have subclasses that can be instantiated. *Air Type* has *Supply Air*, *Exhaust Air*, *Outgoing Air*, or *Outside Air*. *Component* has an *Air Humidifier*, *Heat Recovery*, *Droplet Separator*, *Heater*, *Air Filter*, *Circulating Air*, *Multileaf Damper*, *Cooler*, *Mixing Chamber*, *Fan*, and *Distribution Chamber*. *MIoTA Relation* has an *Air Flow Direction* or *Connector*.

There are also some enumeration class attributes: *SensorType* (Temperature/Relative Humidity/Absolute Humidity/CO_2 Content/Pressure/Volume Flow/Power), *Orientation* (Left/Right/Up/Down), *DataType* (String/Boolean/Integer), *HeatrecoveryType* (Plate Heat Exchanger/Rotation), *HeaterType* (Preheater/Reheater), *FanAirtype* (Supply Air/Exhaust Air), and *AirhumidifierType* (Steam/Evaporation).

The *Configuration Data* contains all the needed information of the model (more on this in section 4). For reasons of space, the attributes are not shown in the meta-model.

3.2. The MIoTA visual notation

Based on the meta-model, we developed a visual notation to support the modeling of air conditioning facilities (see Figure 2). It is available inside the MIoTA library for the ADOxx platform. According to the superclasses in the meta-model, we also have *Component*, *Air Type*, and *MIoTA Relation* as groups here. The class *Sensor* has different visual representations based on the *SensorType*. The visual notation allows to model the elements, including their Attributes and relationships (using *MIoTA Relation*). There are also some additional objects (*Others*). The *Configuration Data* object enables representation and modification of the configuration data through the modeling process. *Notes* facilitate the recording of comments or tasks. The element *Room* allows to visually group components into a room. *Picture* is not shown in Figure 2 because the graphical representation is determined by the upload of an individual image file.

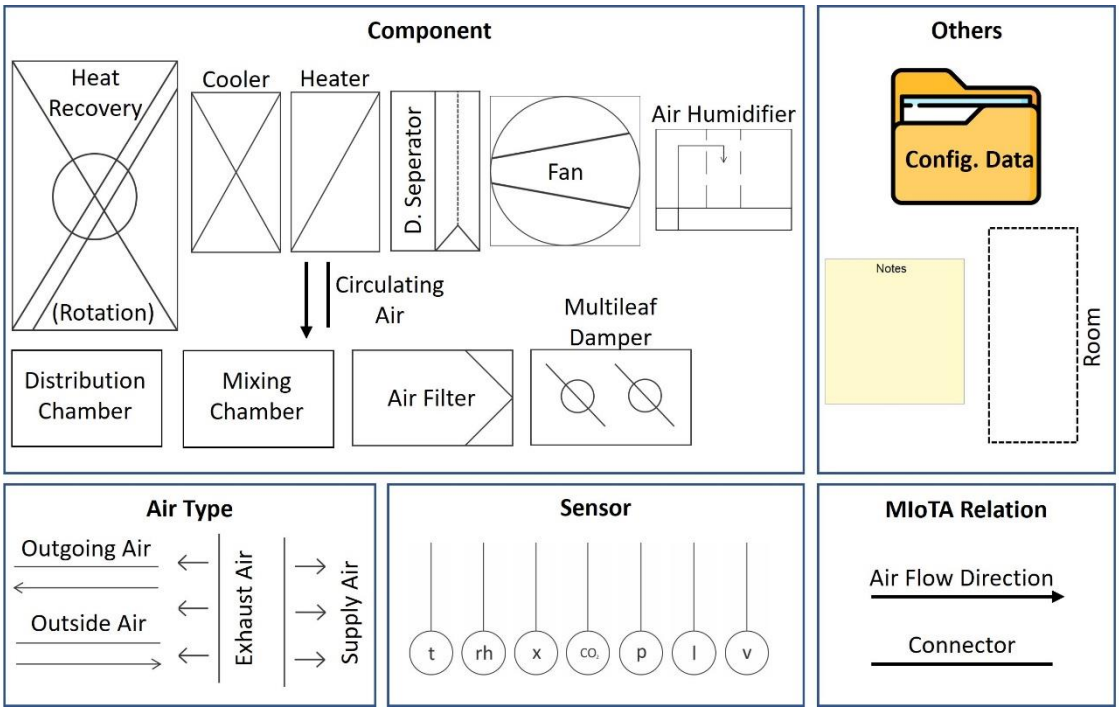

Figure 2: MIoTA visual notation

The graphical representations are mainly based on the European standard DIN EN 12792 and the knowledge of domain experts engaged in the project. The current version is (as with the development of the meta-model) the result of an iterative and incremental development process (see section 5). Depending on the attribute values, some objects have different graphical representations (e.g., *Sensor* depending on the *SensorType*). Some attribute values are also visible

in the model (e.g., *HeatrecoveryType*). A *MIoTA Relation* not only connects the different elements. It also shows the air flow direction of a facility and which *Sensor* is connected to which *Component*.

The visual notation allows domain experts to model air conditioning facilities containing the configuration data. Graphical objects can be connected to each other considering the rules defined in the meta-model. The configuration data can then be transformed into a JSON file and, e.g., deployed on AWS using functionalities of the MIoTA tool (more on this in section 4).

4. Working with the MIoTA modeling tool

This section gives an insight into the MIoTA Tool. It starts with a description of how to create models for air conditioning facilities, and then an overview of implemented functionalities is given. Finally, the evaluation of the tool is briefly discussed.

4.1. Modeling

The process of using the MIoTA tool for modeling is shown in Figure 3. After creating a model in the tool, the first step is to *Add Configuration Data* (**1.**). To speed up the input, dialog boxes are used, and corresponding queries are only made if the respective components are included in the facility (**2.**). For example, if heat recovery exists, further questions about it follow (e.g., type). Value ranges are checked during input, and default values are shown in the modeling tool. Fictitious values are adopted for some attributes (default values) if they cannot be measured or found in the documents to enable the necessary calculation. Only for certain attributes is this possible, which nevertheless allows meaningful data evaluations with fictitious values determined by domain experts.

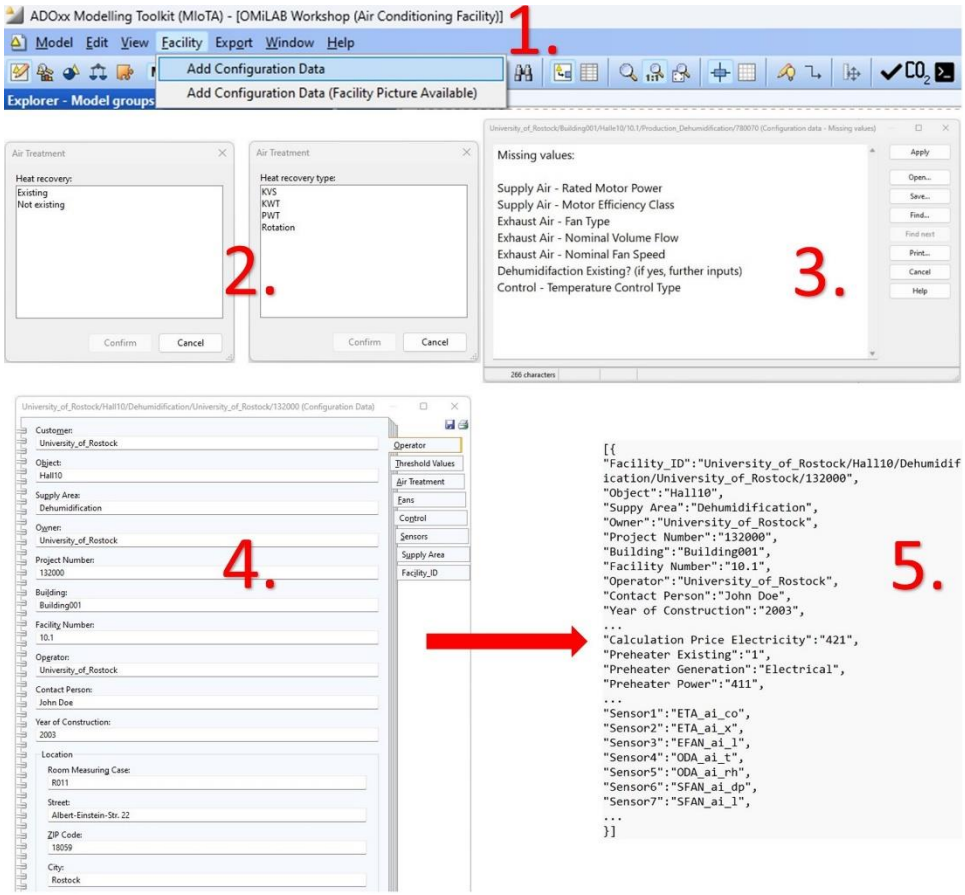

Figure 3: Process of entering configuration data

The modeling objects and the corresponding required sensors are automatically created based on the configuration data. Sometimes, there is already an illustration of the facility. For this case, we have the *Add Configuration Data (Facility Picture Available)* function, where no objects are created automatically. Based on the data necessary for meaningful analyses for energy optimizations, we have developed two distinct measurement kits. There is a standard set of sensors for facilities that do not have humidification and dehumidification (seven sensors). Facilities with either humidification or dehumidification (or both) automatically added nine sensors, as additional energy savings calculations are possible in these when humidity is monitored. Based on this distinction, two different kinds of measurement cases have been developed, which will then be installed in the facilities. Depending on the facility or the objective of the analysis, CO_2 content sensors can automatically be added.

The designation of a sensor (*position_type_physical size*) is unique in each facility and in conjunction with the Facility_ID, names the MQTT topic, so the MQTT Broker in the AWS cloud knows on which topic new data is expected. This way, the time series database, including visualizations of the raw data, is prepared. The Facility_ID is also created automatically based on the configuration data and is composed as follows: *customer_object_supply area_owner_project number*.

Once the configuration data has been inputted, a confirmation of completeness is issued, or a comprehensive list of any missing values is generated (**3.**). A distinction is made between necessary and unnecessary configuration data as far as further data processing is concerned. This output can be personalized and exported either as a text file (txt) or a Portable Document Format (PDF), also capable of being printed. The tool automatically appends the Facility_ID to these documents. Subsequently, users have the flexibility to tailor the model by arranging, interconnecting, and incorporating or removing components as needed. For example, additional sensors can be added beyond the standard equipment to collect more data.

Each modeling entity is accompanied by a notebook, providing a display of its attributes and their corresponding values. This interface facilitates the editing and exportation of attribute data (**4.**). The notebook of the *Configuration Data* incorporates the entered configuration data, sensor designations, and the assigned Faciliy_ID. Accessing this notebook is as simple as double-clicking the corresponding configuration data modeling entity. The right-hand side is divided into sections, delineating distinct segments of the configuration data. Figure 3 also shows an excerpt from the JSON representation of the configuration data as an example (**5.**).

Figure 4 shows a fully customized model of an air conditioning facility. This example shows the capability of using a real facility picture as the background on which modeling elements are positioned. This augmentation allows for a visually enriched representation of the facility configuration. The model contains seven sensors, distributed as follows: two for monitoring the exhaust air (CO_2 content and absolute humidity) and two for outside air (relative humidity and temperature). Additionally, the two fans are each equipped with a sensor for power, and the fan of the supply air also has a sensor for pressure.

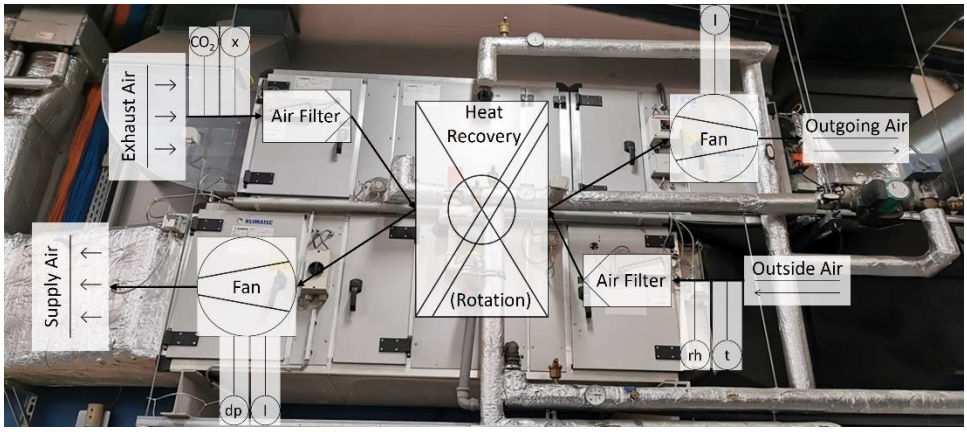

Figure 4: Model example of an air conditioning facility

This section presents that the tool was developed with consideration given to a user-friendly interface (*requirement 1*) and the ability to create visualizations (i.e., visual models) of air conditioning facilities (*requirement 2*). There are many facilitations in the whole modeling process. These requirements were addressed through the meta-model, and the visual notation.

4.2. Functionalities

MIoTA offers several functionalities to check the constraints defined in the meta-model, to ease the modeling process, and to export data from the tool. Figure 5 shows where to find and how to use the menu icons and items. There are three menu icons (**6.**):

- **Check Sensor Duplicates:** Checks if a sensor designation appears more than once in a model.
- **Add CO_2 Content Sensors:** Automatically adds CO_2 content sensors with a given designation.
- **Execute AdoScript:** Allows to execute AdoScript code in the modeling tool.

Once the model has been tailored and validated, the data is capable of being dispatched in the form of a JSON file through an HTTP request directed toward an Application Programming Interface (API) gateway situated within the AWS cloud infrastructure (**7.**). Subsequently, this data is stored in a document-centric database. We have two different menu items to upload the configuration data. With *Upload Configuration Data (Initial)*, the upload only takes place if a plausible Facility_ID is available, no duplicate sensors are in the model, and the value ranges of the entered data are fine. A list of missing data is then output, on which necessary values are marked. *Upload Configuration Data (Final)* only allows the upload if Facility_ID, sensors, and value ranges are fine and all necessary configuration data is entered. A list of missing (not necessary) data is then output.

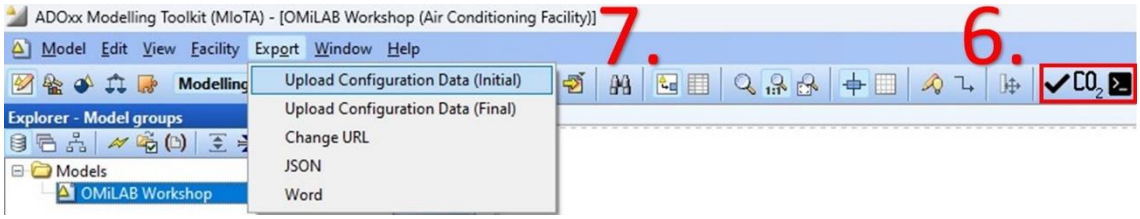

Figure 5: Process of entering configuration data

If the upload is successful, there is always an infobox "Upload successful!". The two types of data upload are needed because, in most cases, not all information about the facilities is available when the initial model is created, but the corresponding configurations in the cloud should already be done. *Change URL* allows to edit and show the current endpoint to which the data is sent. It is possible to export the data as a JSON file (*JSON*) or to export a documentation of the facility (*Word*) containing all the configuration data and a picture of the model. Such documentation is invaluable for various applications, including presentations, as well as for internal use within the case study company. This resource allows customers to understand their air conditioning facilities, track the results of energy-related inspections, and even identify the need for repairs or upgrades.

The collected data saved in the database can now be used in a variety of ways. It tells the MQTT broker on which topic new sensor data is expected. This allows us to create dashboards automatically and alarms for the raw values of a facility. As the configuration data classify the facility, it also supports further data analysis, e.g., in which it says what kind of algorithms can be used based on the contained component in order to derive energy optimization measures. The uploaded data also facilitates the practical setup of sensors. They provide information about the required sensor types and their intended installation locations within the facility.

This section describes the functionalities that meet the following requirements: The tool enables the ingestion of the configuration data (*requirement 3*), subsequently exporting it (*requirement 4*). The devices, components, and sensors are automatically assigned unique IDs (*requirement 5*). Meeting the requirements for the MIoTA modeling method is essential to support the development of IoT applications. Whether the successful implementation of the requirements was achieved will be practically examined in the next section.

4.3. Evaluation

In our prior work [13], we applied quality criteria to assess the MIoTA modeling method. To prove the effectiveness of the modeling language itself, as well as its compatibility and support within the tool, we conducted a practical validation process involving a real-world scenario followed by an interview. The lessons learned from this played an important role in the subsequent improvements. The following is a brief summary of the evaluation.

We selected the semiotic model quality framework (SEQUAL) [14] for the evaluation. SEQUAL provides the capability to differentiate between quality characteristics and means to achieve these (goals). This allows us to examine the requirements of the model from the air conditioning area and how they were implemented. Quality is addressed in multiple semiotic levels, such as pragmatics, semantics, and syntax. This aspect holds significance in encompassing the modeling language itself, as well as the utilization within the case study and its support through the tool.

SEQUAL further acknowledges that models frequently emerge through collaborative efforts involving individuals engaged in modeling, whose comprehension of the modeling domain evolves during the modeling process. The development of IoT applications, in our experience, involves different stakeholders in such a collaboration process.

The coherence with Syntactic, Semantic, and Deontic Quality was confirmed in [15]. To do this, we evaluated the coherence of the DSML and its implementation in the modeling tool and the coherence of the concept of meaning in the domain and the concept in the meta-model. We also gathered requirements for the design in interviews with domain experts. In [13], we evaluated the Physical, Perceived Semantic, and Technical Pragmatic Quality. The application of the criteria of Moody's "Physics of Notations" showed that Semiotic Clarity, Semantic Transparency, Visual Expressiveness, and Dual Coding have been fulfilled.

The principles that have not been implemented or were only partly fulfilled indicated improvement potential. The principle of Graphic Economy advises limiting the usage to a maximum of six symbols. However, our DSML diverges from this guideline, as we decided to follow the stakeholder requirement to implement the symbol set used in the domain, which has more than six symbols. One possible way for enhancement would be to distinguish between different groups of users of the DSML (principle of Cognitive Fit) and tailor the symbol sets available to each group accordingly. Nevertheless, even after this adaption, the symbol count for each group would still exceed six. This requires the involvement of different groups, a change of the DSML, and, most crucially, conducting an exhaustive assessment in conjunction with our practical case study company if this really results in positive effects. Such an effort would also facilitate a more comprehensive evaluation of both Empirical Quality (Can different actors comprehend the model?) and Social Quality (Do actors agree on the interpretation of the model?).

The coherence with Perceived Semantic Quality was confirmed through a modeling session that was followed by a guided expert interview with the participants. This session involved two employees from the case study company who will use the tool in the future.

The comprehensive practical assessment showed the significant value of the tool. Modeling is easily possible for domain experts without modeling experience, and the tool functionalities speed up the processes. The tool introduces numerous simplifications, enabling the digitization of (existing) facilities and eliminating the need for manual model sketching. The consistent procedure for model creation, coupled with the aid of support functionalities and the possibility for dynamic adjustments, streamlines user efforts, saves time, and avoids mistakes. According to the participants, explaining the tool's usage to colleagues is perceived as easy. Additionally, several suggestions for improvement could be derived from the interview, with a primary focus

on refining collaborative modeling capabilities in future iterations. Furthermore, there is potential for improving the tool's overall look, as well as enhancing the process of querying configuration data.

5. Experiences of the MIoTA meta-model development

The meta-model in its current version resulted from an iterative and incremental development process that we consider both suitable for the development task and essential for achieving a model that meets the requirements of the application domain well. There were three major versions of the meta-model, all validated with the end users and anchored in requirements.

For the first meta-model version, we followed our intuition to use the existing symbols in the area of air conditioning facilities. We identified the underlying concepts starting from these symbols and represented them in the meta-model. However, there are many more symbols than required by the use cases at hand. The total set would include symbols for mechanical components, control equipment, valves, and fittings that represent more than 50 concepts in various variants (see, e.g., the European standard DIN EN 12792, [16] and [17]). We narrowed the number of concepts (and symbols) down to what the domain experts identified as essential. The challenge in this context was that the relations between the concepts seemed to have no other meaning than being "connected to".

The second version of the meta-model was initiated by experiences in modeling actual examples of air conditioning facilities in cooperation with the users. The main change made in the second version was a refinement of the actual symbols and a few additional concepts.

The third version was triggered by the results of the evaluation of the notation, as reported in [13]. Here, we discovered the need to improve the notation by introducing new relationship types. Our experience during this improvement process is that the participation of the domain experts in concept identification, design of symbols, handling of the modeling tool, and specifying functionality was essential.

6. Conclusion and outlook

The MIoTA modeling method supports the development of IoT applications for air conditioning facilities. This work focuses on the modeling language with its meta-model and the tool support provided by the ADOxx-based MIoTA toolkit. By experimenting with the ADOxx environment, we have succeeded in implementing the toolkit that satisfies the requirements of our use case. The meta-modeling environment was a great help, allowing us to quickly modify the toolkit, test hypotheses, and get the toolkit to the point where it began to be useful in practice. Extending the MIoTA library or functionalities is easily possible. The main challenges were dealing with interfaces and creating individual, graphical representations of the objects.

The initial evaluation showed that the tool is of great importance and will also be used in the future in the case study company. It is not feasible to make broad generalizations from this particular case to multiple others. Further case studies are needed to ensure validity. Furthermore, the suggestions for improvement gained in the evaluation are to be realized (e.g., refining collaborative modeling capabilities or enhancing the process of querying the configuration data). In the future, it is planned to display the status of sensors and visualizations of their data directly in the tool. The control of actuators, or the display of changes in the physical facility that require changes in the model, should also be made possible.

Tool Download https://www.omilab.org/MIoTA/

Acknowledgements

Part of the research presented in this paper was supported by grant no. TBI-V-1-426-VBW-145 of the Ministry of Economics, Infrastructure, Tourism and Labour of the State of Mecklenburg-Vorpommern using funds from the European Regional Development Fund.

References

[1] A. Farahzadi, P. Shams, J. Rezazadeh, R. Farahbakhsh, Middleware technologies for cloud of things: a survey, Digital Communications and Networks 4 (2018) 176–188.

[2] M. A. Chaqfeh, N. Mohamed, Challenges in middleware solutions for the internet of things, in: International Conference on Collaboration Technologies and Systems (CTS), IEEE, 2012, pp. 21–26.

[3] G. Fortino, A. Guerrieri, W. Russo, C. Savaglio, Towards a development methodology for smart object-oriented IoT systems: A metamodel approach, in: IEEE International Conference on Systems, Man, and Cybernetics, IEEE, 2015, pp. 1297–1302.

[4] A. M. Alberti, M. A. S. Santos, R. Souza, H. D. L. Da Silva, J. R. Carneiro, V. A. C. Figueiredo, J. J. P. C. Rodrigues, Platforms for smart environments and future internet design: A survey, IEEE Access 7 (2019) 165748–165778.

[5] C. Sosa-Reyna, E. Tello-Leal and D. Lara-Alabazares, Methodology for the model-driven development of service oriented IoT applications, in: Journal of Systems Architecture, pp. 15–22.

[6] K. Sandkuhl, U. Seigerroth, Digital transformation of enterprises: Case studies and transformation paths, in: Twenty-fifth Pacific Asia Conference on Information Systems (PACIS 2021), 2021.

[7] G. Fortino, C. Savaglio, G. Spezzano, M. Zhou, Internet of things as system of systems: A review of methodologies, frameworks, platforms, and tools, IEEE Trans. Syst. Man Cybern, Syst. 51 (2021) 223–236.

[8] D. Karagiannis, H. Kühn, Metamodelling platforms, in: EC Web, p. 182.

[9] W. Kim, S. Katipamula, A review of fault detection and diagnostics methods for building systems, Science and Technology for the Built Environment 24 (2018) 3–21.

[10] S. P. Melgaard, K. H. Andersen, A. Marszal-Pomianowska, R. L. Jensen, P. K. Heiselberg, Fault detection and diagnosis encyclopedia for building systems: A systematic review, Energies 15 (2022) 4366.

[11] B. Nast, K. Sandkuhl, Methods for model-driven development of IoT applications: requirements from industrial practice, in: Proceedings of the 18th International Conference on Evaluation of Novel Approaches to Software Engineering, SCITEPRESS - Science and Technology Publications, 2023, pp. 170–181.

[12] OMiLAB, The ADOxx Metamodelling Platform, 2023. URL: https://www.adoxx.org/live/home.

[13] B. Nast, K. Sandkuhl, Quality evaluation of a DSML supporting model-driven IoT development for air conditioning facilities, Unpublished, 2023. http://dx.doi.org/10.13140/RG.2.2.34167.39841/1

[14] J. Krogstie, Quality of business process models, in: J. Krogstie (Ed.), Quality in Business Process Modeling, Springer International Publishing, Cham, 2016, pp. 53–102.

[15] B. Nast, K. Sandkuhl, Meta-model and tool support for the organizational aspects of Internet-of-Things development methods, in: Proceedings of the 3rd International Conference on Advanced Information Science and System, ACM, New York, NY, USA, 2021, pp. 1–6.

[16] Siemens, Grafische symbole der anlagenelemente für lüftung und klima, 1999, https://sid.siemens.com/v/u/20004.

[17] M. Ashraf, Ashrae handbook fundamentals inch-pound edition, 2005, https://www.academia.edu/15194757/2005_ASHRAE_HANDBOOK_FUNDAMENTALS_I_P_Edition.