UMLsec4Edge: Extending UMLsec to model data-protection-compliant edge computing systems

Sven Smolka[1], Jan Laufer[1], Zoltán Ádám Mann[2], Klaus Pohl[1]
[1]*paluno – The Ruhr Institute for Software Technology, University of Duisburg-Essen, Essen, Germany*
[2]*Complex Cyber Infrastructure (CCI) group, University of Amsterdam, Amsterdam, The Netherlands*

***Abstract*—Edge computing enables the processing of data – frequently personal data – at the edge of the network. For personal data, legislation such as the European General Data Protection Regulation requires data protection *by design*. Hence, data protection has to be accounted for in the design of edge computing systems whenever personal data is involved. This leads to specific requirements for modeling the architecture of edge computing systems, e.g., representation of data and network properties.**

To the best of our knowledge, no existing modeling language fulfils all these requirements. In our previous work we showed that the commonly used UML profile UMLsec fulfils some of these requirements, and can thus serve as a starting point.

The aim of this paper is to create a modeling language which meets all requirements concerning the design of the architecture of edge computing systems accounting for data protection. Thus, we extend UMLsec to satisfy all requirements. We call the resulting UML profile UMLsec4Edge. We follow a systematic approach to develop UMLsec4Edge. We apply UMLsec4Edge to real-world use cases from different domains, and create appropriate deployment diagrams and class diagrams. These diagrams show UMLsec4Edge is capable of meeting the requirements.

***Index Terms*—edge computing, data protection, UMLsec, fog computing**

I. Introduction

Motivation: Edge computing enables real-time data processing with low network communication latency and satisfactory quality of service at the same time. Due to these positive characteristics, edge computing systems become increasingly popular [1]. So-called edge nodes can perform tasks that most end devices (e.g. IoT devices) are not capable of due to insufficient computing power [2]. Edge nodes can also preprocess data, for example to reduce the amount of data that is sent to the cloud for further processing [1]. However, processing (personal) data at the edge of the network leads to new challenges. For example, edge nodes may use different types of communication (e.g., 5G or WLAN) which offer different levels of data protection. Regulations such as the General Data Protection Regulation (GDPR) of the European Union [3] prescribe the protection of personal data. In particular, the GDPR stipulates the need for data protection *by design*. To ensure data protection *by design* in edge computing systems, requirements for modeling the architecture of edge computing systems emerge [4] which can be sorted into categories: "network properties", "devices", "actors", and "data properties".

Problem statement & Aim of the paper: Existing modeling languages used in system design focus either on security (e.g. [5], [6]), or on privacy (e.g. [7], [8]), or none of the above. Although the GDPR requires appropriate security under Article 32, the definition of data protection goes beyond that of security and privacy. Existing modeling languages cannot capture all aspects of edge computing systems related to data protection, for example, the different levels of data protection stemming from different types of communication technologies [4]. To the best of our knowledge, there is no modeling language which covers all requirements on the modeling of the architecture of data-protection-compliant edge computing systems. We identified UMLsec (an extension of the well-known modeling language UML) as a promising starting point to model data protection concerns in edge computing systems, because UMLsec is capable of modeling information security aspects for software systems. UMLsec addresses the "data properties" requirement category as well as parts of the categories "network properties" and "devices".

The aim of this paper is to create a modeling language which satisfies the identified requirements concerning the design of the architecture of edge computing systems, accounting for data protection.

Contribution & Approach: To achieve this goal, this paper provides the following contributions.

- A new modeling language called UMLsec4Edge that supports the modeling of data-protection-compliant edge computing systems. UMLsec4Edge is an extension of UMLsec, focusing on deployment and class diagrams. We focus on these diagram types because the architecture of systems and dependencies between system components have great impact on data protection *by design*.
- A formalization of UMLsec4Edge as well as an UML profile created with the modeling platform Papyrus.
- Deployment and class diagrams of three real-world use cases which employ edge computing in diverse application domains. The use cases originate from the EU-funded research project FogProtect [9]. The diagrams confirm that UMLsec4Edge meets the requirements by addressing the shortcomings of UMLsec.

The creation of UMLsec4Edge follows a systematic approach based on the work of Lagarde et al. [10], including a literature

Work partially funded by the European Union's Horizon 2020 research and innovation programme under grant agreement no. 871525 (FogProtect). Useful discussions with project partners are gratefully acknowledged.

This paper was published in: *48th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pp. 418-425, 2022.

search focusing on UML profiles covering security, privacy, or data protection to assess alternative extension options.

Outline: Sec. II defines terminology and introduces the requirements for modeling the architecture of edge computing systems accounting for data protection. Sec. III describes our systematic development approach and as a result the UMLsec4Edge profile. Sec. IV shows extracts from UMLsec4Edge diagrams and discusses threats to validity. Sec. V examines related work, while Sec. VI concludes the paper. Additional material, including the full UMLsec4Edge profile, complete diagrams, and further information on the systematic approach and literature review, can be found online[1].

II. Requirements to model data protection in edge computing systems

A. Data protection in edge computing systems

The term *data protection* refers to the protection of personal data. Regulations such as the GDPR [3] prescribe this protection. The GDPR requires the enforcement of technical and organizational measures to prevent so called *personal data breaches*. Personal data breaches occur, e.g., whenever an unauthorized actor accesses personal data. Any system, and therefore also an edge computing system, which processes personal data must ensure the absence of personal data breaches.

Ensuring data protection in edge computing systems faces new challenges compared to more established computing paradigms like cloud computing. End devices and edge nodes could differ in hardware configuration, preventing the implementation of uniform data protection mechanisms such as hardware enclaves across all devices. Moreover, end devices and edge nodes can be deployed almost anywhere. Thus, they may not be protected by sufficient physical security measures, so there is a threat of attackers physically compromise them.

Since the GDPR requires systems processing personal data to ensure data protection *by design*, such data protection challenges need to be considered when developing an edge computing system, already starting with the architectural design of the system. In our previous work [4], we identified four data-protection-related requirements on modeling languages for modeling the architecture of edge computing system:

(R-1) Network properties: It must be possible to model different communication types between devices in an edge computing system as well as possible threats posed by them.
(R-2) Devices: It must be possible to model different device types in an edge computing system as well as threats posed by their use.
(R-3) Actors: It must be possible to model actors within an edge computing system, as well as their trust in each other, their relationship to data, and their data-specific roles.
(R-4) Data properties: It must be possible to model data protection requirements specifying whether data must not be disclosed, manipulated, or deleted.

[1]See https://git.uni-due.de/fogprotect/umlsec4edge

B. Modeling edge computing architectures with UMLsec

In our previous work [4], we analyzed to what extent UMLsec [5] supports modeling data protection requirements and threats to data protection during the design of edge computing systems. We concluded that UMLsec provides a reasonable basis for satisfying the identified requirements:

(R-1): UMLsec introduces stereotypes such as `<<wire>>` or `<<LAN>>`, which can be used to assign a connection type to a communication path between nodes in deployment diagrams. In addition, UMLsec introduces the adversary model, which represents the threat of unauthorized reading, insertion, and deletion of data during data exchange over a communication path of a certain connection type. UMLsec is limited in having only stereotypes representing wired connection types. In edge computing systems, however, data exchange between nodes often takes place wirelessly. Consequently, UMLsec only partially addresses **(R-1)**.

(R-2): UMLsec allows assigning device types to nodes by stereotypes like `<<POS device>>`. The adversary model then allows the representation of the threat of unauthorized physical access to these types of devices. However, there are no stereotypes for device types common in edge computing systems. Accordingly, the threat of unauthorized physical access to them cannot be modeled in the adversary model. In addition, UMLsec cannot model threats occurring when data is exchanged between components placed on the same node. Overall, UMLsec fulfills a part of **(R-2)**.

(R-3): UMLsec has no stereotypes or tags to model actors within an edge computing system as well as their trust in each other, their relationship to the data, and their data-specific roles. Accordingly, UMLsec does not address **(R-3)**.

(R-4): UMLsec introduces stereotypes and tags to model the security requirements of data during data exchange in terms of confidentiality, integrity, and availability. For example, UMLsec introduces the stereotype `<<secrecy>>`, which can be attached to dependencies between nodes and components in a deployment diagram to define the security requirement of disallowing data to be read by an attacker during data exchange. In combination with the adversary model (which represents the threats of data transmission over a communication channel with a specific connection type), it is possible to evaluate whether security objectives are met by the system design. Thus, UMLsec addresses **(R-4)**.

Table I summarizes the restrictions that make UMLsec not fulfill the requirements. The table also lists the stereotypes and tags of our UMLsec4Edge profile (presented in the following section) leading to the fulfillment of the requirements.

III. UMLsec4Edge

A. Systematic Approach towards UMLsec4Edge

To create our UMLsec4Edge profile that satisfies all the requirements, we conduct a systematic extension of UMLsec. Since UMLsec already partially addresses **(R-1)** and **(R-2)** in deployment diagrams, it is reasonable to extend UMLsec with respect to deployment diagrams to fully address the

TABLE I
RESTRICTIONS OF UMLSEC RESULTING IN NON-FULFILMENT OF THE REQUIREMENTS, AS WELL AS SOLUTIONS PROVIDED BY UMLSEC4EDGE

Data protection-related requirement	Restriction of UMLsec leading to non-fulfilment of the requirement	Solution provided by UMLsec4Edge	
		Stereotype	Tag
(R-1)	Lack of ability to model wireless data transmission	`<<Wireless>>`, `<<3G>>`, `<<4G>>`, `<<5G>>`, `<<RFID>>`, `<<NFC>>`, `<<Bluetooth>>`, `<<WLAN>>`	
(R-2)	Lack of ability to model the threat of unauthorized physical access to device types in edge computing systems	`<<ComputingContinuum-Device>>`, `<<EndDevice>>`, `<<EdgeNode>>`, `<<Cloud>>`	
	Lack of ability to model threats between components on the same node	`<<internal>>`	
(R-3)	Lack of ability to model relationships between actors and data	`<<Actor>>`, `<<DataTraceability>>`	`rights`, `obligations`
	Lack of ability to model trust relationships between actors	`<<Actor>>`	`trusts`
	Lack of ability to model actors with multiple data-specific roles	`<<Actor>>`	`roles`
(R-4)	Already covered by UMLsec		

requirements. As UMLsec does not address **(R-3)**, we extend UMLsec with respect to class diagrams, since the data of a system is often modeled in class diagrams. Fig. 1 shows our systematic approach of extending UMLsec. We first create a temporary UML profile (called secEdge) following the systematic approach for creating UML profiles according to Lagarde et al. [10]. Their systematic approach has already been successfully applied by other authors [11], [12]. In the first three phases a *domain model*, *solution model* and *UML profile skeleton* are created, and in the last phase consistency preservation and optimization are performed.

Phase 1: The domain model describes the problem, i.e. the required concepts and their relationships to each other.

Phase 2: The solution model is created based on the understanding that emerges from the domain model and contains technical solutions to the problem. To create the solution model, we also conducted a systematic literature search with the goal to assess different existing solutions. Hence, we investigated existing UML profiles regarding security, privacy, and data protection. The literature search was conducted on 23 September 2021 in the Scopus database using the following search string: TITLE-ABS-KEY (“UML profile” AND (“data protection” OR “security” OR “privacy”)). The search resulted in 129 papers, which we systematically filtered for relevance by applying predefined exclusion and inclusion criteria. Included were all papers which contain a UML profile with references to security, privacy, or data protection. Excluded were all findings which are proceedings, which have not been undergone a peer review process, which are not written in German or English, which do not reference a UML profile, and which do not present UMLsec. In the end, 47 papers remained, which we analyzed in detail.

The decision tree in Fig. 2 shows how we proceeded in creating the solution model. We first investigated whether the literature provides solutions to the requirements. If the literature provides solutions to the requirements, we reviewed to what extent these solutions are compliant with the concepts of the GDPR and UMLsec. If the proposed solutions are compliant with the GDPR and UMLsec, we have included them in the solution model - otherwise, we have derived our own solutions from the proposed solutions taking into account the concepts of the GDPR and UMLsec. In case the literature does not provide solutions for the requirements, we derived our own solutions considering the concepts of the GDPR, UMLsec and the EU-funded research project FogProtect.

Phase 3: After the solution model is created, the UML profile skeleton is constructed. This is done by converting all classes of the solution model to stereotypes in the skeleton, as well as by adding metaclasses to the skeleton to which the stereotypes are mapped.

Phase 4: After the creation of the skeleton, a validation of the consistency of the profile on the UML meta level is performed as well as an optimization of the profile, i.e. a reduction of the number of stereotypes.

Having systematically created the preliminary secEdge profile, we integrated it with UMLsec, resulting in the final UMLsec4Edge profile. We also formalized the profile in the Papyrus modeling tool. Papyrus enables tool support, precluding modeling errors. In the following, we present how we realized **(R-1)**, **(R-2)** and **(R-3)** in UMLsec4Edge and discuss possible alternatives. Thereby, we first describe the creation of the solution and subsequently explain where the solution comes from. In the additional material (see Sec. I) the following artifacts can be found: The models created according to the phases of Lagarde et al. (domain model, solution model, UML profile skeleton, secEdge profile), an extended description on how we realized the requirements, the complete UMLsec4Edge profile, the technical formalization of the profile in Papyrus, and the exclusion and inclusion criteria used in the literature search.

B. Addressing Requirement (R-1)

In order to fulfill **(R-1)**, it is necessary to have the means to mark wireless communication channels as such and to model threats posed by such communication channels. One possibility is to introduce a stereotype such as `<<Wireless>>`, which represents wireless data transmission between nodes

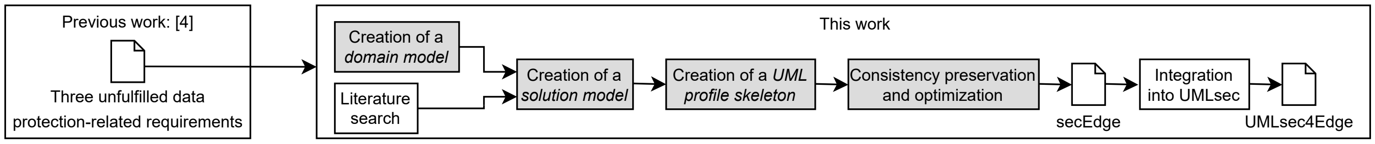
Previous work: [4]
Three unfulfilled data protection-related requirements
This work
Creation of a *domain model*
Literature search
Creation of a *solution model*
Creation of a *UML profile skeleton*
Consistency preservation and optimization
secEdge
Integration into UMLsec
UMLsec4Edge

Fig. 1. UMLsec4Edge creation process (boxes marked in gray indicate a phase according to the approach of Lagarde et al. [10])

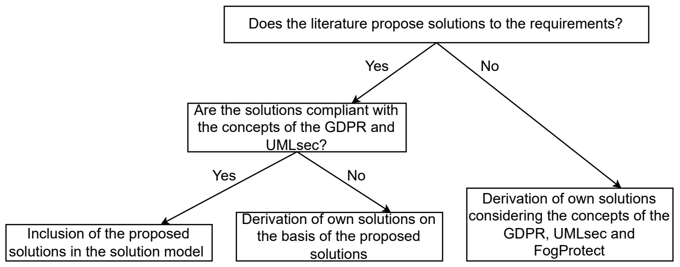
Does the literature propose solutions to the requirements?
Yes
No
Are the solutions compliant with the concepts of the GDPR and UMLsec?
Yes
No
Inclusion of the proposed solutions in the solution model
Derivation of own solutions on the basis of the proposed solutions
Derivation of own solutions considering the concepts of the GDPR, UMLsec and FogProtect

Fig. 2. Procedure for the creation of the solution model

(analogous to the `<<wire>>` stereotype in UMLsec for a wired data transmission). Since the communication path between nodes is described by the metaclass "communicationpath", the stereotype is mapped to it.

Due to the heterogeneity of devices, several different data transmission types can be used in edge computing systems, posing different security risks. Accordingly, it is appropriate to consider such different wireless data transmission types as well. One possibility would be to assign a tag (for example, of the type of a string or array of an enumeration) to the stereotype `<<Wireless>>`, which can be used to represent a specific wireless data transmission type. However, this does not allow the threats of unauthorized data reading, insertion, and deletion, to be mapped to these wireless data transmission types in the adversary model of UMLsec, since threats in the adversary model can only be mapped to stereotypes.

Since the adversary model allows modeling of threats resulting from the use of certain communication types, it is suitable to extend the adversary model since threats must also be represented to fulfill the requirement. The extension of the adversary model can be made by representing the wireless data transmission modes as stereotypes. Thus, an inheritance hierarchy with the stereotype `<<Wireless>>` as superclass and the stereotypes of the wireless data transmission types as subclasses is appropriate. Such an inheritance hierarchy can later be extended to include other wireless data transmission types. Since the UMLsec adversary model is formalized, we have also formally extended it to include the above stereotypes. This formal extension can be found in the online material.

Origin of the solution: The literature recommends that the following types of wireless data transmission should be considered: 3G, 4G, 5G [13], Bluetooth [13]–[15], WLAN [13], [16]–[19]. Furthermore, based on the case studies of the EU-funded research project FogProtect, we also consider RFID and NFC. These wireless data transmission types are modeled as an inherited stereotype from `<<Wireless>>`, allowing the mapping to communication paths.

C. Addressing Requirement (R-2)

For the realization of **(R-2)**, it must be possible to represent the threat of unauthorized physical access to device types in an edge computing system. Since the adversary model of UMLsec already provides a mapping of this threat to nodes, it is reasonable to extend the adversary model to map the threat to device types used in edge computing systems. For this purpose, stereotypes must be introduced, which allow to assign such device types to nodes. Since various device types could be found in edge computing systems, covering the whole computing continuum from edge to cloud, we introduce a general stereotype `<<ComputingContinuumDevice>>` to describe a general type of device. In order for nodes to be labeled with this stereotype, it must be mapped to the metaclass "node". This stereotype can further act as a superclass and associated subclasses can represent specific device types.

To represent devices in an edge computing system, we introduce stereotypes `<<EndDevice>>`, `<<EdgeNode>>`, and `<<Cloud>>`, which inherit from the stereotype `<<ComputingContinuumDevice>>`. Alternatively, the specific device types can be further specified by additional inheritance hierarchies. For example, `<<EndDevice>>` could be further refined with subclasses such as "Smartphone", "Smart Vehicle", or "CCTV Camera". However, since the type of device depends on the scenario in which it is used, this is problematic (e.g., a smartphone could be an end device in one scenario and an edge node in another). Accordingly, such a further specification would not be appropriate. Another alternative would be to resolve the inheritance hierarchy into an enumeration. However, this does not allow the threat of unauthorized physical access to device types to be represented in the adversary model, as this requires stereotypes.

To fulfill **(R-2)**, it is also necessary to model threats in the data exchange between two components placed on the same node. The UMLsec adversary model can represent threats in data transmission over a communication path, and thus between components on different nodes. However, this is not applicable due to the lack of a communication path if the components are placed on the same node. One solution could be to map data exchange threats in such a case to the node on which the components are placed. The stereotypes introduced above, which allow the assignment of device types to nodes, could be used for this purpose. However, this would couple threats to device types, so that labeling a node with the stereotype of a device type would implicitly represent threats in the exchange of data between components on the node.

For a better separation of concerns, such threats should be modeled independently of the device type. For this purpose, we introduce a stereotype `<<internal>>`, which is mapped to the metaclass "node" and to which the threats during data transmission between components on the same node can be mapped. Moreover, we formally extended the UMLsec adversary model to include the above stereotypes. This formal extension can be found in the additional online material.

Origin of the solution: From the literature search, it was found that no papers directly address this requirement. Thus, the device types in edge computing systems, are based on the labels used in the EU research project FogProtect. The modeling of threats between components on the same node is based on the adversary model of UMLsec.

D. Addressing Requirement (R-3)

To fulfill **(R-3)**, it is necessary to represent relationships between actors and data. Since data is typically represented as attributes of classes, this requires actors to be represented in class diagrams. One way to do that is to model actors as classes and introduce a stereotype such as `<<Actor>>` which makes it apparent that a class represents an actor. This stereotype is mapped to the metaclass "class". To represent the relationship between actors and data, a stereotype such as `<<DataTraceability>>` can be attached to the data. Subsequently, a tag can be introduced for this stereotype, in which the names of the actors are represented, for example, in the form of a string. Attaching the stereotype to the attributes of a class and modeling the tag, e.g., in the form of a comment outside the class (multiple stereotypes and associated tags result in multiple comments), increases the size of the class diagram, which decreases readability.

An alternative is to map the stereotype `<<DataTraceability>>` to the class itself and thus to the metaclass "class". This results in only one tag for all attributes within a class. Since a class can have multiple attributes and an attribute can have a relationship to multiple actors, the value of this tag must have a well-defined structure. Such a value can be a string with the following structure: The relationship between an attribute of a class and a actors is a (1+a)-tuple, which starts with the name of the attribute, followed by the names of the actors' classes. Such a tuple is created for each attribute for which a relationship to actors exists. The value of the string is finally a concatenation of the tuples, where the tuples are separated by a comma.

Since the GDPR provides rights (e.g. the right of a data subject to obtain information about its data) and obligations (e.g. the obligation of a data processor to inform the data subject when collecting personal data) for an actor with respect to data, it is rational to introduce two tags `rights` and `obligations`. These tags must each be formatted according to the string structure above to allow representing the relation between actors and data in terms of rights and obligations.

Furthermore, in order to fulfill **(R-3)**, it is necessary to model a trust relationship between actors. For this purpose, the previously introduced stereotype `<<Actor>>` can be used and extended by another tag like `trusts`. This tag may be of type string. The string could contain the names of the actors (i.e. their class names). An alternative approach for the type would be a Boolean, which, when taking the value "true", states all other actors of the edge computing system trust the actor. However, this construct is rarely found in reality, since an actor is usually not trusted by all other existing actors [20]. Actors differentiate between actors whom they (do not) trust.

Finally, in order to meet **(R-3)**, it must be possible to assign – possibly multiple – data-specific roles to actors. Since roles are tied to actors, it is again a suitable option to extend the stereotype `<<Actor>>` with another tag, such as `roles`. The type of the tag can be a string or an array of an enumeration containing the data-specific roles that actors can take. However, this approach can cause syntactical errors in the role assignment when using a modeling tool which does not support checking the contents of strings. In contrast, an enumeration allows the designer to select data-specific roles from a predefined set, thus preventing syntactical mistakes.

An alternative approach would be to represent the roles in an inheritance hierarchy. Here a stereotype, which represents a role in general, acts as superclass and stereotypes which represent concrete roles are associated subclasses. The stereotype acting as a superclass can be mapped to the metaclass "class", whereupon all stereotypes in the inheritance hierarchy can be used to label classes. However, in their systematic approach, Lagarde et al. propose a pattern for transforming such an inheritance hierarchy into an enumeration to reduce the number of stereotypes. Accordingly, we introduce an enumeration called "RoleType" which contains the data-specific roles an actor can take. The type of the tag `roles` is therefore an array of type "RoleType". In our case, the enumeration includes the roles "DataSubject", "DataController", "DataProcessor" and "ThirdParty", since these roles are defined by the GDPR.

Origin of the solution: Many papers introduce the stereotype `<<Actor>>` ([21]–[28]) to represent actors. Since we did not find solutions in the literature for modeling trust relationships between actors and traceability between actors and data, we derive solutions with respect to the GDPR. The representation of roles has often been realized in the literature also by an inheritance hierarchy [25]–[36].

IV. Evaluation and Discussion

A. Application of UMLsec4Edge to real-world use cases

To illustrate UMLsec4Edge's capability of satisfying **(R-1)**, **(R-2)**, and **(R-3)**, we extended the deployment diagrams, class diagrams, and adversary models of three real-world use cases from the domains of Smart Manufacturing, Smart Media, and Smart Cities using UMLsec4Edge. To ensure representatives of the regarding systems the use cases originate from the EU-funded research project FogProtect. To ensure reliability we had frequent exchange with the responsible use case owners. Due to space limitations, we provide excerpts of the diagrams and the adversary model of one of the use cases in this paper. The complete extended diagrams and adversary models of the three use cases can be found in the additional material (see Sec. I). Overall, the use cases represent a basis of an edge computing system. They consist of cloud centres, edge nodes, and end devices. Edge nodes and end devices in particular can exist in a multitude of identical versions up to many thousands of units. To reduce complexity the use cases are kept to a core that can be scaled up. However, the core includes all threats to data protection and covers all requirements raised. Further information can be found online[2].

Use Case: The Smart Manufacturing use case describes a scenario in which a mobile manufacturing facility called "Factory in a Box" (FiaB) [37] is installed inside a multimodal transport container. Inside the container there are end devices such as cameras and robots as well as edge nodes which, e.g., (pre-)process the camera video feed. FiaB is capable of manufacturing customer-specific orders while in transit. The inside of FiaB is monitored using cameras. Authorized personnel can then access monitoring and production data via a terminal (in this case, the personnel is an operator). In terms of data-related roles, the customer and authorized personnel are in the role of a data subject. The operator is in the role of both data subject (as he/she may be recorded) and data processor (as he/she can access the customer's personal data and the video recordings).

Example for requirement (R-1): Fig. 3 shows an excerpt of the Smart Manufacturing deployment diagram extended with the help of the UMLsec4Edge profile. Table II shows an excerpt of the associated adversary model. In the deployment diagram, a robot and an edge node are modeled as nodes which are connected to each other through a communication path. One software component (called "Robot Service") is placed on the robot and two software components (called "Robot Control" and "Data Hub (Edge)") are placed on the edge node. Software components "Robot Service" and "Robot Control" communicate with each other through the communication path, while software components "Robot Control" and "Data Hub (Edge)" communicate with each other within the edge node.

The UMLsec4Edge profile allows marking the communication path with the stereotype `<<5G>>`, which enables modeling wireless data transmission between nodes. In addition, the threats of unauthorized data reading ("read"), data insertion ("insert"), and data deletion ("delete") can be mapped to the stereotype in the adversary model. By enabling the modeling of a wireless data transmission along with the resulting threats, UMLsec4Edge satisfies **(R-1)**.

Example for requirement (R-2): In the deployment diagram (see Fig. 3), it is also possible to mark the node "Robot" with the stereotype `<<EndDevice>>` and the node "FiaB Edge Node" with the stereotype `<<EdgeNode>>`. Furthermore, the threat "access" can be mapped to these stereotypes in the adversary model (see Table II). This allows modeling the threat of unauthorized physical access.

[2]see https://fogprotect.eu/results/

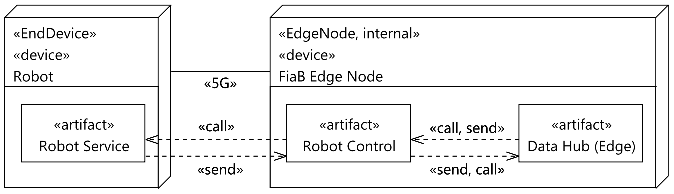

Fig. 3. Excerpt from the Smart Manufacturing deployment diagram

TABLE II
EXCERPT FROM THE SMART MANUFACTURING ADVERSARY MODEL

Stereotype	Threats
`<<5G>>`	{read, insert, delete}
`<<EdgeNode>>`	{access}
`<<EndDevice>>`	{access}
`<<internal>>`	{read, insert, delete}

Moreover, it is possible to mark the node "FiaB Edge Node" with the stereotype `<<internal>>`. Hence, the threats "read", "insert", and "delete" can be mapped to this stereotype in the adversary model. This allows modeling the threat of an attacker being able to read and delete data exchanged between the "Robot Control" and the "Data Hub (Edge)" as well as insert new data. By being able to represent both threats of unauthorized physical access to devices and threats in data exchange between components placed on the same device, UMLsec4Edge satisfies **(R-2)**.

Example for requirement (R-3): Fig. 4 shows an excerpt of the Smart Manufacturing class diagram created using the UMLsec4Edge profile. In the diagram, the class "Dashboard (Edge)" represents a software component and the class "Operator" represents an actor. The "Operator" accesses the "Dashboard (Edge)", which contains the data "Recorded Video" and "Customer Data". With UMLsec4Edge, it is possible to mark the class "Operator" with the stereotype `<<Actor>>` to show this class represents an actor. Furthermore, it is possible to mark the class "Dashboard (Edge)" with the stereotype `<<DataTraceability>>`. Subsequently, the values of the tags `rights` and `obligations` of the stereotype can be used to define which actor has rights or obligations with respect to which attribute of the class. The string of tag `rights` means that actor "Authorized Personnel" has rights regarding attribute "Recorded Video" and actor "Customer" has rights regarding attribute "Customer Data". The string `obligations` denotes that actors "FiaB-Container Owner" and "Operator" have obligations towards attribute "Customer Data" (Note: Actors "Authorized Personnel", "Customer" and "FiaB-Container Owner" are not presented in the extract of the diagram). Thus, it is possible to model relationships (specified in terms of rights and obligations) between actors and data.

The tag `trusts` of the stereotype `<<Actor>>` is used to model the trust of the actor "Operator" towards other actors. The "Operator" trusts the "Authorized Personnel" and the "FiaB Container Owner". Correspondingly, trust relationships between actors can be modeled.

In addition, the stereotype `<<Actor>>` enables the use of

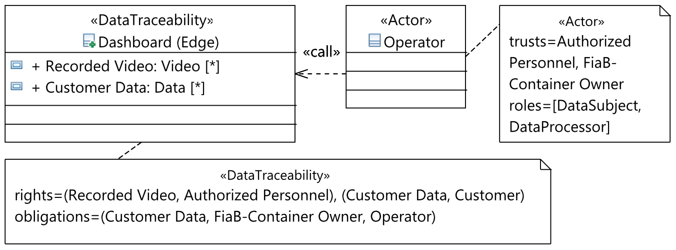
«DataTraceability»
Dashboard (Edge)
+ Recorded Video: Video [*]
+ Customer Data: Data [*]
«call»
«Actor»
Operator
«Actor»
trusts=Authorized Personnel, FiaB-Container Owner
roles=[DataSubject, DataProcessor]
«DataTraceability»
rights=(Recorded Video, Authorized Personnel), (Customer Data, Customer)
obligations=(Customer Data, FiaB-Container Owner, Operator)

Fig. 4. Excerpt from the Smart Manufacturing class diagram

the tag `roles` to assign data-related roles to an actor. The "Operator" is assigned the roles "DataSubject" and "DataProcessor". Accordingly, the operator can simultaneously take on the role of data subject and data processor. By allowing relations between actors and data, trust relationships between actors, and assignment of multiple data-specific roles to an actor, UMLsec4Edge satisfies **(R-3)**.

B. Threats to Validity

Internal validity: An internal validity risk exists due to possible syntactic and semantic errors which may have been made when creating the UMLsec4Edge profile. In order to avoid errors, the creation process of the UMLsec4Edge profile was based on the systematic UML profile creation approach by Lagarde et al. including a consistency preservation of the created UML profile on meta-level [10]. The usage of the Papyrus modeling tool prevented syntactic and semantic errors.

Furthermore, there may be the risk that the extensions to UMLsec only focus on solving the limitations identified for three use cases in previous work which were co-created by the authors. To address this risk, we focused on meeting requirements instead of solving limitations. Additionally, we conducted a systematic literature review to find ideas for possible realizations of the requirements. We investigated 47 papers, about half of which provided fruitful ideas.

Reliability: A threat to reliability exists for the application of the UMLsec4Edge profile because it was done by the same group of authors who defined the profile. To minimize the risk we have distributed the creation and the evaluation of UMLsec4Edge within our team.

External validity: Although UMLsec4Edge fulfills all data protection-related requirements in the three real-world use cases used in our evaluation, there are other use cases in edge computing which have not been considered. It is therefore not possible to conclude whether the data protection-related requirements are also fulfilled by UMLsec4Edge in all other use cases. However, this risk exists with any modeling language, since complete coverage of all possible use cases in edge computing is not possible.

V. Related work

Related work covers several modeling languages which are based on UML and deal with modeling security and / or privacy aspects. Privacy-focused UML profiles like [7], [8], [38], [39] support the development of privacy-aware applications. Modeling languages that focus on modeling security aspects [5], [6], [40] especially focus on confidentiality, integrity, and availability of data. However, both privacy and security modeling languages lack important contributions in regard to data protection and thus do not address our requirements.

Ahmadian et al. [41] present two UML profiles regarding data protection. One of the profiles allows the modeling of personal data including their sensitivity, while the other profile, which is an extension of UMLsec, allows the modeling of access to data based on data-specific roles. However, both profiles are only used in UML behavior diagrams. Thus, their approach is complementary to UMLsec4Edge. In the future, both approaches could be combined to address both, modelling of system architecture as well as modelling of system behavior.

Modeling languages such as [40] and [42] extend UMLsec to model secure systems in the Internet of Things (IoT). Ficco et al. also extend UMLsec by introducing new stereotypes to model a secure deployment of cloud applications [43]. These approaches do not consider modeling IoT and cloud elements as part of an edge computing system. Thus, they are not capable of meeting our requirements.

When modeling secure business processes, several extensions to the Business Process Model and Notation (BPMN) can be used [44]–[46]. While the main focus is on security, some data protection aspects can also be modeled. However, modeling business processes only partially supports the development of the architecture of (edge computing) systems which take data protection into account.

Approaches like [20] allow to model data protection in cloud systems. Yet, these approaches do not cover characteristics of edge computing, nor are they as extensive as UMLsec. Furthermore, they are not as commonly known and easy to use as a modeling language that is based on the UML.

VI. Conclusions and future work

This paper addressed the problem of accounting for data protection while modeling the architecture of edge computing systems. We presented UMLsec4Edge as an extension to UMLsec. UMLsec4Edge covers deployment and class diagrams. With the help of UMLsec4Edge, it is possible to model data protection aspects in the design of edge computing systems. The extensions to UMLsec we have developed represent a coherent construct and complement each other.

We assume UMLsec4Edge can also be applied to related domains such as the Internet of Things and cloud computing. Further research could verify this hypothesis. This work can serve as a basis for investigating whether our extensions also contribute to the modeling of data protection aspects in combination with other UMLsec diagram types. Furthermore, UMLsec4Edge could serve as a communication tool for developers as well as an input for data protection compliance analysis. An evaluation including domain experts to investigate whether UMLsec4Edge fulfils this purposes is planned as future work. An automated analysis of UMLsec4Edge diagrams could be implemented by using for example the Object Constraint Language. In the future, UMLsec4Edge could serve as a basis to model edge computing systems at runtime to evaluate data protection whenever the system changes.

References

[1] A. Yousefpour, C. Fung, T. Nguyen, K. Kadiyala, F. Jalali, A. Niakanlahiji, J. Kong, and J. P. Jue, "All one needs to know about fog computing and related edge computing paradigms: A complete survey," *Journal of Systems Architecture*, vol. 98, pp. 289–330, 2019.

[2] S. Yi, C. Li, and Q. Li, "A Survey of Fog Computing: Concepts, Applications and Issues," in *Mobidata '15*, 2015, pp. 37–42.

[3] General Data Protection Regulation, "Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC," *Official Journal of the European Union*, p. L119, 2016.

[4] J. Laufer, Z. Á. Mann, and A. Metzger, "Modelling data protection in fog computing systems using umlsec and sysml-sec," in *2021 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*. IEEE, 2021, pp. 777–786.

[5] J. Jürjens, "UMLsec: Extending UML for secure systems development," in *International Conference on The Unified Modeling Language*. Springer, 2002, pp. 412–425.

[6] L. Apvrille and Y. Roudier, "SysML-sec: A SysML environment for the design and development of secure embedded systems," in *APCOSEC*, 2013.

[7] J. El Mokhtari, A. Abou El Kalam, S. Benhadou, and H. Medroumi, "PrivUML: A privacy metamodel," *Procedia Computer Science*, vol. 151, pp. 53–60, 2019.

[8] T. Basso, L. Montecchi, R. Moraes, M. Jino, and A. Bondavalli, "Towards a UML profile for privacy-aware applications," in *IEEE International Conference on Computer and Information Technology*, 2015, pp. 371–378.

[9] D. Ayed, E. Jaho, C. Lachner, Z. Á. Mann, R. Seidl, and M. Surridge, "FogProtect: Protecting sensitive data in the computing continuum," in *European Conference on Service-Oriented and Cloud Computing*. Springer, 2020, pp. 179–184.

[10] F. Lagarde, H. Espinoza, F. Terrier, C. André, and S. Gérard, "Leveraging patterns on domain models to improve UML profile definition," in *International Conference on Fundamental Approaches to Software Engineering*. Springer, 2008, pp. 116–130.

[11] G. Zoughbi, L. Briand, and Y. Labiche, "Modeling safety and airworthiness (RTCA DO-178B) information: Conceptual model and UML profile," *Software & Systems Modeling*, vol. 10, no. 3, pp. 337–367, 2011.

[12] H. Stallbaum and M. Rzepka, "Toward DO-178B-compliant test models," in *2010 Workshop on Model-Driven Engineering, Verification, and Validation*. IEEE, 2010, pp. 25–30.

[13] S. Mili, N. Nguyen, and R. Chelouah, "Transformation-based approach to security verification for cyber-physical systems," *IEEE Systems Journal*, vol. 13, no. 4, pp. 3989–4000, 2019.

[14] M. Decker, "An UML profile for the modelling of mobile business processes and workflows," in *Proceedings of the 5th International ICST Mobile Multimedia Communications Conference*, 2009, pp. 1–7.

[15] A. Ferrante, I. Kaitovic, and J. Milosevic, "Modeling requirements for security-enhanced design of embedded systems," in *SECRYPT 2014, Vienna, Austria*, M. S. Obaidat, A. Holzinger, and P. Samarati, Eds. SciTePress, 2014, pp. 315–320.

[16] K. Beckers, S. Faßbender, M. Heisel, and R. Meis, "A problem-based approach for computer-aided privacy threat identification," in *Annual Privacy Forum*. Springer, 2012, pp. 1–16.

[17] K. Beckers, "Supporting common criteria security analysis with problem frames," in *Pattern and Security Requirements*. Springer, 2015, pp. 195–228.

[18] K. Beckers, D. Hatebur, and M. Heisel, "A problem-based threat analysis in compliance with common criteria," in *ARES*. IEEE, 2013, pp. 111–120.

[19] Y. Mahmoodi, S. Reiter, A. Viehl, O. Bringmann, and W. Rosenstiel, "Model-guided security analysis of interconnected embedded systems." in *MODELSWARD*, 2018, pp. 602–609.

[20] Z. Á. Mann, F. Kunz, J. Laufer, J. Bellendorf, A. Metzger, and K. Pohl, "RADAR: Data protection in cloud-based computer systems at run time," *IEEE Access*, vol. 9, pp. 70 816–70 842, 2021.

[21] M. Alshammari and A. Simpson, "A UML profile for privacy-aware data lifecycle models," in *Computer Security*. Springer, 2017, pp. 189–209.

[22] R. Bouaziz, S. Kallel, and B. Coulette, "An approach for security patterns application in component based models," in *ICCSA*. Springer, 2014, pp. 283–296.

[23] R. Bouaziz and B. Coulette, "Secure component based applications through security patterns," in *GreenCom*. IEEE, 2012, pp. 749–754.

[24] ——, "Applying security patterns for component based applications using UML profile," in *2012 IEEE 15th International Conference on Computational Science and Engineering*. IEEE, 2012, pp. 186–193.

[25] M. A. Neri, M. Guarnieri, E. Magri, S. Mutti, and S. Paraboschi, "A model-driven approach for securing software architectures," in *SECRYPT 2013*. IEEE, 2013, pp. 1–8.

[26] A. Salem, S. Triki, H. Ben-Abdallah, N. Harbi, and O. Boussaid, "Verification of security coherence in data warehouse designs," in *International Conference on Trust, Privacy and Security in Digital Business*. Springer, 2012, pp. 207–213.

[27] P. Colombo and E. Ferrari, "Towards a modeling and analysis framework for privacy-aware systems," in *2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*. IEEE, 2012, pp. 81–90.

[28] Ç. Cirit and F. Buzluca, "A UML profile for role-based access control," in *Proceedings of the 2nd international conference on Security of information and networks*, 2009, pp. 83–92.

[29] C. Blanco, E. Fernández-Medina, J. Trujillo *et al.*, "An MDA approach for developing secure OLAP applications: Metamodels and transformations," *Computer Science and Information Systems*, vol. 12, no. 2, pp. 541–565, 2015.

[30] C. Blanco, I. G.-R. de Guzmán, E. Fernandez-Medina, and J. Trujillo, "An architecture for automatically developing secure OLAP applications from models," *Information and Software Technology*, vol. 59, pp. 1–16, 2015.

[31] C. Blanco, I. G. R. de Guzmán, E. Fernández-Medina, and J. Trujillo, "Showing the benefits of applying a model driven architecture for developing secure OLAP applications." *J. Univers. Comput. Sci.*, vol. 20, no. 2, pp. 79–106, 2014.

[32] S. Triki, H. Ben-Abdallah, J. Feki, and N. Harbi, "Modeling conflict of interest in the design of secure data warehouses," in *KEOD 2010*, 2010, p. 445.

[33] R. Villarroel, E. Soler, E. Fernández-Medina, J. Trujillo, and M. Piattini, "Representing levels of abstraction to facilitate the secure multidimensional modeling," in *ARES'06*. IEEE, 2006, pp. 7–pp.

[34] R. Villarroel, E. Fernández-Medina, M. Piattini, and J. Trujillo, "A UML 2.0/OCL extension for designing secure data warehouses," *Journal of Research and Practice in Information Technology*, vol. 38, no. 1, pp. 31–43, 2006.

[35] R. Villarroel, E. Soler, E. Fernández-Medina, J. Trujillo, and M. Piattini, "Using UML packages for designing secure data warehouses," in *ICCSA*. Springer, 2006, pp. 1024–1034.

[36] H. Wada, J. Suzuki, and K. Oba, "A service-oriented design framework for secure network applications," in *COMPSAC'06*, vol. 1. IEEE, 2006, pp. 359–368.

[37] Z. Mann, A. Metzger, J. Prade, R. Seidl, and K. Pohl, "Cost-optimized, data-protection-aware offloading between an edge data center and the cloud," *IEEE Transactions on Services Computing*, 2022.

[38] T. Basso, L. Montecchi, R. Moraes, M. Jino, and A. Bondavalli, "PrivAPP: An integrated approach for the design of privacy-aware applications," *Software: Practice and Experience*, vol. 48, no. 3, pp. 499–527, 2018.

[39] J. L. Cánovas Izquierdo and J. Salas, "A UML profile for privacy enforcement," in *STAF*. Springer, 2018, pp. 609–616.

[40] P. J. Escamilla-Ambrosio, D. A. Robles-Ramírez, T. Tryfonas, A. Rodríguez-Mota, G. Gallegos-García, and M. Salinas-Rosales, "IoTsecM: A UML/SysML extension for internet of things security modeling," *IEEE Access*, vol. 9, pp. 154 112–154 135, 2021.

[41] A. S. Ahmadian, D. Strüber, V. Riediger, and J. Jürjens, "Model-based privacy analysis in industrial ecosystems," in *ECMFA*. Springer, 2017, pp. 215–231.

[42] K. K. Chambwe, "Model-based secure software engineering using UMLsec applied to assisted living and home care," Master thesis, Institute for Informatik, University of Oslo, 2018.

[43] M. Ficco, F. Palmieri, and A. Castiglione, "Modeling security requirements for cloud-based system development," *Concurrency and*

Computation: Practice and Experience, vol. 27, no. 8, pp. 2107–2124, 2015.

[44] S. Agostinelli, F. M. Maggi, A. Marrella, and F. Sapio, "Achieving GDPR compliance of BPMN process models," in *CAiSE*. Springer, 2019, pp. 10–22.

[45] M. Salnitri, F. Dalpiaz, and P. Giorgini, "Designing secure business processes with SecBPMN," *Software & Systems Modeling*, vol. 16, no. 3, pp. 737–757, 2017.

[46] M. E. A. Chergui and S. M. Benslimane, "A valid BPMN extension for supporting security requirements based on cyber security ontology," in *MEDI*. Springer, 2018, pp. 219–232.