Semantic Interoperability as Key to IoT Platform Federation

Michael Jacoby[1], Aleksandar Antonić[2], Karl Kreiner[3], Roman Łapacz[4], and Jasmin Pielorz[3]

[1] Fraunhofer IOSB, Karlsruhe, Germany
michael.jacoby@iosb.fraunhofer.de
[2] Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia
aleksandar.antonic@fer.hr
[3] Austrian Institute of Technology (AIT), Austria
{karl.kreiner, jasmin.pielorz}@ait.ac.at
[4] Poznan Supercomputing and Networking Center, Poznan, Poland
romradz@man.poznan.pl

Abstract. Semantic interoperability is the key technology to enable evolution of the Internet of Things (IoT) from its current state of independent vertical IoT silos to interconnected IoT platform federations. This paper analyzes the possible solution space on how to achieve semantic interoperability and presents five possible approaches in detail together with a discussion on implementation issues. It presents the H2020 symbIoTe project as an example on how semantic interoperability can be achieved using semantic mapping and SPARQL query re-writing. We conclude that the found approaches together with the proposed technologies have the potential to act as corner stone technologies for achieving semantic interoperability.

Keywords: Semantic Interoperability, Internet of Things, IoT Platform federation, Semantic Mapping, symbIoTe, SPARQL Query Re-Writing

1 Introduction

Semantic Interoperability is the key to "data exchange and service creation across large vertical applications" as seen as next step of evolution of the IoT [8]. In order to enable building new innovative applications which make use of data from multiple existing vertical IoT silos these systems must not only be able to exchange information but also have a common understanding of the meaning of this data. This means, even if today's IoT systems are willing to expose their data and resources to others their semantically incompatible information models become an issue to dynamically and automatically inter-operate as they have different descriptions or even understandings of resources and operational procedures. To enable dynamic and automated interoperability new features like semantic annotation, well-defined semantic mapping, unified resource discovery and federated authentication and authorization are required which cannot solely

be provided by the existing platforms on their own but rather need to be offered by some kind of interoperability framework mediating between the platforms. Moreover, in the era of virtualization and Any-as-a-Service models there is a need to federate independent infrastructures and introduce simplified methods to provide virtual resources of different types and owners in a dynamic and consistent manner. Because semantic interoperability is the basis for building services addressing sophisticated requirements across heterogeneous vertical IoT platforms, in this paper we present our thoughts and concepts how semantic interoperability between IoT platforms can be achieved.

The remainder of the paper is organized as follows. Section 2 provides a definition of semantic interoperability as it is used in this paper as well as some background on semantic technologies. In Section 3 possible approaches to achieve semantic interoperability are presented on a rather abstract level and Section 4 gives some detailed insights on what to keep in mind when trying to realize them. Section 5 presents how the symbIoTe project is approaching semantic interoperability picking up one of the possible approaches introduced in Section 3 and showing how it is realized in symbIoTe. The paper closes with conclusions and future work in Section 6.

2 Related Work

In this section we will provide a definition of the term semantic interoperability as used in this paper. As semantic interoperability is a compound word we will first analyze existing definitions for each of the terms and from this conclude a definition of the whole term. Semantics, as seen in linguistics and philosophy, refers to the study of meaning which means the relation of signifiers like words, symbols or signs and their denotation [1]. In computer science, the meaning of semantics is basically the same but here the relations of signifiers and their denotation need to be understandable and processable by machines. The most common way to achieve this is by using an ontology which is "an explicit specification of a [shared] conceptualization"[10] and can be imaged like a formally-defined information model.

In 2001, Tim Berners-Lee introduced the idea of the Semantic Web[2], proposing the evolution of the internet from a web of documents to a web of machine-readable and -understandable data, which is becoming more and more reality. The corner stone technologies of the Semantic Web are the Resource Description Format[1] (RDF), a lightweight (meta data) data model for describing ontologies, and SPARQL Protocol and RDF Query Language[2] (SPARQL), a query language for data in RDF format, which both are standardized by the World Wide Web Consortium (W3C).

"Broadly speaking, interoperability can be defined as a measure of the degree to which diverse systems, organizations, and/or individuals are able to work together to achieve a common goal" [12]. As this definition of interoperability is

[1] https://www.w3.org/RDF/
[2] https://www.w3.org/TR/rdf-sparql-query/

to broad in this context we also refer to the *Levels of Conceptual Interoperability Model* (LCIM) [21] depicted in Figure 1 which were created in the context of simulation theory but have a much broader applicability. In the scope of this

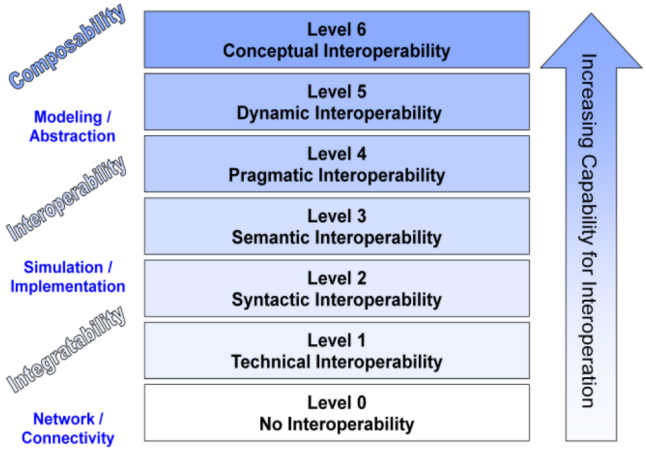


Fig. 1. The Levels of Conceptual Interoperability Model (from [21]).

paper we see interoperability only up to Level 3 of LCIM where Level 1 refers to the low-level technical connectivity of platforms, Level 2 to using a common data format or protocol like XML or HTTP and Level 3 to having a unified understanding of the shared data. Therefore, semantic interoperability is defined as "the ability of computer systems to exchange data with unambiguous, shared meaning" [14] within this paper.

3 Approaches to Semantic Interoperability of IoT platforms

The question discussed in this section is what possible approaches a system can take to achieve semantic interoperability between multiple IoT platforms. Figure 2 visually depicts the current situation where multiple platforms, in this case IoT platform A and B, having their own internal information model exist in parallel. To enable interoperability between those platforms they need to have a mutual understanding of things, i.e. some unification of their internal information models must somehow be defined.

As depicted in Figure 3 the solution space to this problem can be thought of as a line between the two most radical approaches which are using a single core information model every platform must comply to on the one side and to not provide one at all and all platforms provide only their own information model which need to be aligned using semantic mapping on the other side. In between

there exists a large, not clearly defined number of intermediate solutions from which three are representatively presented in the following together with the two radical approaches. These approaches are motivated by and in line with concepts presented by Wache and Choi et al. [24, 3].

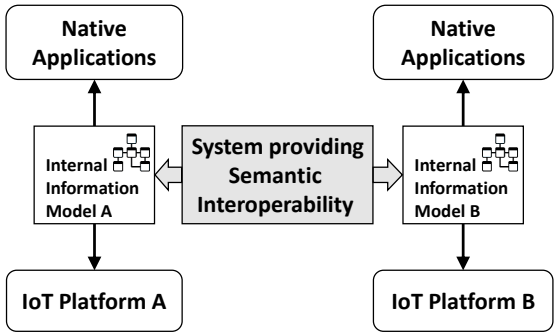


Fig. 2. Schematic representation of the problem of semantic interoperability between different IoT platforms.

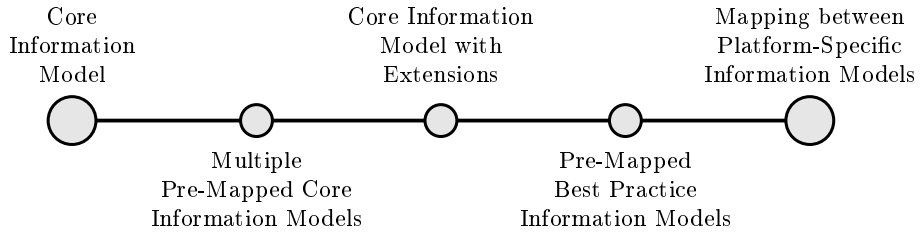


Fig. 3. Solution space for possible approaches to semantic interoperability.

3.1 Core Information Model

The most widespread approach amongst existing platforms is to use a single core information model that all platforms must comply with. This means that a platform can only expose data that fits into this core information model as custom extensions are not permitted. If a platform needs to expose data that does not fit into the core information model the platform cannot expose this data and cannot inter-operate with others.

Pros

- easy to implement and use since the data from all platforms follows the same information model
- resulting system easy to use for app developers who only need to know one information model

Cons

- finding/defining an information model all platforms can agree upon may be difficult
- information model tends to become complex as it must comprise all data that should be exchangeable between platforms
- will always exclude some platforms whose internal information model does not fit the core information model
- no way to integrate future platforms with information models not compatible to the core information model without breaking the existing system

3.2 Multiple Pre-Mapped Core Information Models

Based on the single core information model approach this one tries to make it more easy and convenient for platform owners to integrate their internal information model by supporting not only a single core information model but multiple ones. To achieve that a large number of existing platforms can easily participate it would be a good idea to choose well-established information models (e.g. the Semantic Sensor Network Ontology [4] or the oneM2M ontology [15]) as core information models. To ensure interoperability between platforms using different core information models the supported core information models are already mapped to each other. As it will not always be possible to map two core information models completely there will be some degree of information loss if platforms conform to different core information models but if they conform to the same one they will be fully interoperable.

Pros

- flexible approach as further core information models and mappings can be added over time
- does not enforce use of one single core information model which excludes less platforms from participating

Cons

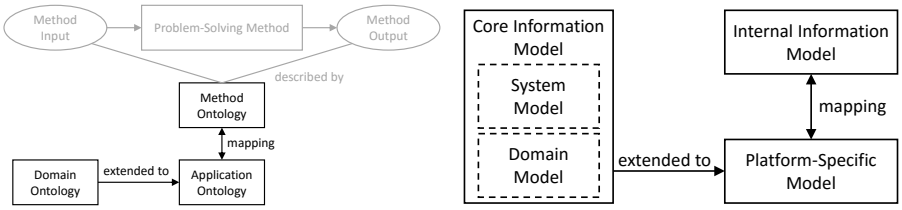
- may still exclude some platforms whose information model does not match any of the core information models

3.3 Core Information Model with Extensions

This approach is based on an information model that is designed to be as abstract as possible but at the same time as detailed as needed. Therefore, the core

information model should try to only define high-level classes and their interrelations which act as extension points for platform-specific instantiations of this information model. These platform-specific instantiations either use the provided classes directly or they can define a subclass which can hold any platform-specific extensions to the core information model, e.g. additional properties. Besides the high-level classes the core information model may also contain properties the system needs which will be very general properties like *ID* or *name* in most of the cases.

This approach resembles an approach for a model-driven knowledge engineering system (KES) presented by Studer et al. shown in Figure 4a where a domain ontology is extended to an application ontology which is mapped to a method ontology that is finally used to define in- and output of a method used to solve a problem. The core information model with extensions can be very closely matched to this approach as depicted in Figure 4b. The main difference is, that there exists not only a domain ontology that is extended but rather the core information model which contains the domain model and the system model (which can be seen as a platform-specific extension of the domain model to the system that provides the interoperability). The application ontology corresponds to the platform-specific model which is a platform-specific extension to the core information model and the method ontology corresponds to the internal information model of the platform as depicted in Figure 4.



(a) Ontology-based approach for a model-driven KES (based on Figure 6 from[20]).

(b) Conceptual structure of the Core Information Model with Extensions approach.

Fig. 4. Structural similarity between an ontology-based model-driven KES and the Core Information Model with Extensions.

This results in an information model that has a minimalistic core that all platforms must conform to and extension points to realize custom requirements. Two platforms using different extensions can directly understand each other in terms of the core information model and when they need also to understand the custom extensions they must define a semantic mapping between their extensions.

Pros

- provides basic interoperability between platforms by defining minimalistic core information model
- provides full flexibility by custom extensions, i.e. no platforms are excluded
- high acceptance from adopter-side as it combines basic out-of-the-box interoperability (by the core information model) with support for complex scenarios (through extensions and semantic mapping)

Cons

- requires semantic mapping when custom extensions need to be understood by different platforms
- defining a semantic mapping can be a complex task and requires additional work from developers/platform owners
- design of the core information model is a complex task

3.4 Pre-Mapped Best Practice Information Models

Essentially, this is the same approach as *Multiple Core Information Models* but with one small but significant modification: the provided information models are no longer seen as *core* information models but rather as *best practice* information models. Hence, platforms must not be compliant to any of the provided information models as in the previous approach but can choose their information model freely. If they choose to re-use one of the provided best practice information models they will gain instant interoperability to other platforms also aligned with one of the best practice information models.

Pros

- no limitations on information model, hence does not exclude any platform
- best practice information models make usage for inexperienced platform owner more easy
- better and broader interoperability due to already aligned best practice information models

Cons

- no initially interoperability between platforms as long as no mapping is defined when no pre-mapped information model is used
- defining a semantic mapping can be a complex task and requires additional work from developers/platform owners

3.5 Mapping between Platform-Specific Information Models

In this approach, there isn't anything like one or more core information models. Instead, every platform independently provides its own information model. Interoperability is only achieved through mapping between these platform-specific information models.

Pros

- not limited only to a fixed set of information models but rather supports all possible information models
- mappings can be added iteratively increasing the degree of interoperability

Cons

- no initially interoperability between platforms as long as no mapping is defined
- defining a semantic mapping can be a complex task and requires additional work from developers/platform owners
- the system does not understand any of the data it is processing

4 Considerations About Realizing the Approaches

When thinking about realizing one of the above approaches there are multiple things that need to be considered. On the one hand there are design decisions to make regarding the concrete specification of the information model(s) and on the other hand there are practical issues to take into account like what kind of software is needed to implement the chosen approach and what tools do already exist. This chapter will first present two issues relevant for the approaches using semantic mapping and after that some approach-specific issues.

4.1 Semantic Mapping

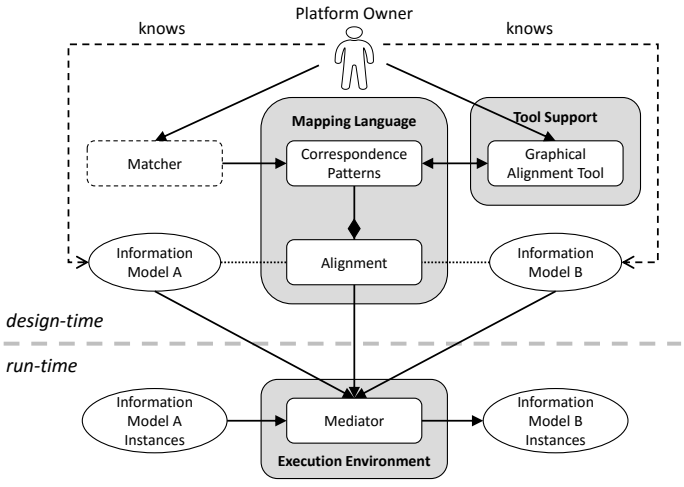


Fig. 5. Schematic representation of an example usage of semantic mapping and the included tools.

Semantic mapping is used in three out of the five approaches as an alternative to defining a common information model all platforms can agree on. Figure 5 depicts a schematic representation how this could look like when implemented. At first, a platform owner must know that another platform he would like to inter-operate with exists. This issue is discussed in detail in the next section. For now, we assume that the platform owner of platform A knows that platform B exists and that he wants to define a mapping between the information models of the two platforms. To define such a mapping, which consists of multiple correspondence patterns and is called an alignment, he essentially needs a mapping language to express the alignment in. As defining such a mapping is not a trivial task some tool support in form of a graphical alignment tool, i.e. a visual editor for the mapping language, is desirable. Optionally, to further ease the complexity of the task, a matcher could be up-streamed to automatically provide an initial mapping of the two information models. At run-time the mapping/alignment together with both information models is used by some kind of mediator to translate instances between the two information models.

When trying to realize any of the approaches which include semantic mapping these are areas that need to be analyzed for existing tools that fit the requirements. The mapping language is the most important component in this tool stack as all the other components need to be able to understand or generate mappings expressed in that language. The main criteria for choosing a language is its expressiveness and support for defining complex mappings.

4.2 Finding Other Platforms of Interest

A key problem when trying to make multiple platforms exposing their data in (partially) different information models interoperable using semantic mapping is the following: How to know that other platforms exist and that they provide data that is of such an interest that it justifies the effort to define a semantic mapping between the platforms? As the parts of the information models that need to be mapped are platform-specific there can be no semantic-based discovery but only a syntactical one as only the platform itself understands its information model. This implies that there is a need to have humans in the loop to close the semantic and syntactic gap.

Therefore, a suitable candidate for enabling a platform owner to find other platforms that might be of interest to him would be some kind of search functionality. Such a search could be quite primitive, e.g. a simple full-text search on the terms defined in the information model, or more sophisticated using concepts like phonetic search, natural language processing or translations to enable finding of relevant terms even if they do not syntactically match the search term or are expressed in another language.

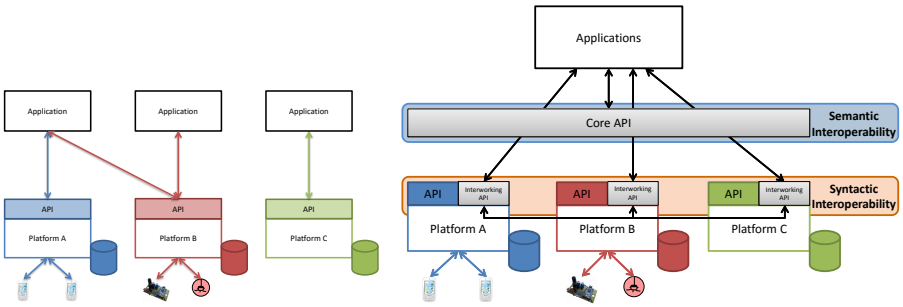
Keeping in mind that mapping information models is the key to enable interoperability between multiple heterogeneous platforms this is an essential part of any system that aims to achieve this kind of interoperability and therefore needs to be solved appropriately.

4.3 Approach-Specific Issues

Single Core Information Model The main challenge of this approach is to define a single core information model that does contain every information that any platform that should be made interoperable needs but at the same time to not make design decisions that prevent integration of upcoming platforms in the future. As this is an impossible task since these two goals are contradicting the single core information model approach is not suitable to provide interoperability between various kind of existing and upcoming IoT platforms. However, in a scenario where the domain is more narrow this approach could still be feasible.

Core Information Model With Extensions Realizing this approach is essentially a trade-off between defining a quite detailed and easy-to-use core information model and finding the right level of abstraction to not make design decisions that exclude some platforms. This task is quite hard as, referring to the definitions provided by [20], the core information model is a hybrid between a domain ontology, defining only the very abstract structure of the IoT domain, and an application ontology, because it is especially tailored to be used with a single system as depicted in 4b. Therefore, this approach needs special attention on modeling the core information model to not bring platform-specific concepts, relations and properties into the core information model.

5 SymbIoTe's Approach to Semantic Interoperability



(a) without symbIoTe (b) with symbIoTe

Fig. 6. IoT ecosystem with and without symbIoTe.

SymbIoTe (**Symb**iosis of smart objects across **IoT** **e**nvironments) [19] is an EU project and part of the European Union's Horizon 2020 research and innovation programme. Its main objective is to provide an interoperability and mediation framework for collaboration and federation of vertical IoT platforms

thus enabling creation of cross-domain applications using multiple heterogeneous IoT platforms in a unified way.

Figure 6a depicts the IoT ecosystem as it exists now. It consists of multiple IoT platforms that each represent a vertical silo which is tailored to a specific domain. If an application wants to integrate more than one platform it has to do additional implementation work to make use of another platform-specific API. The vision of symbIoTe is to enable platform interoperability and creation of cross-platform apps between the existing vertical IoT silos with minimal integration effort for the platform owners which is shown in Figure 6b. Semantic Interoperability is realized by the Core API providing a query functionality for meta data on available platforms and their resources. Syntactic interoperability is achieved by the Interworking API which provides a uniform access to resources of all platforms and can be seen as some kind of adapter that a platform owner needs to implement to be symbIoTe-compliant.

5.1 General Approach

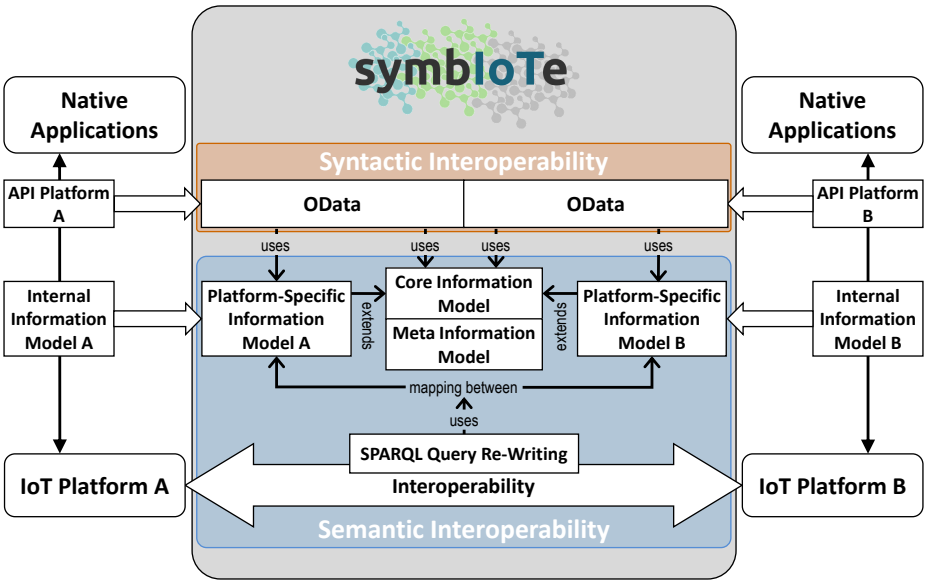


Fig. 7. High-level diagram showing how symbIoTe approaches syntactic and semantic interoperability.

Figure 8 shows how symbIoTe achieves semantic interoperability by implementing the Core Information Model with Extensions approach as presented in Section 3.3. On the left and the right, we see two existing IoT platforms exposing platform-specific APIs based on an internal information model to

applications. Between those two vertical IoT stacks we see symbIoTe with the Core Information Model in its center. As proposed in Section 3.3 and shown in Figure 4b symbIoTe uses two central information models. The Core Information model describes domain specific information and therefore matches the Domain Model in Figure 4b and the Meta information Model describes symbIoTe internal meta information about platforms and resources and matches the System Model. For a platform to become symbIoTe-compliant it must expose its data using a platform-specific information model which is basically the Core Information Model with platform-specific extensions to it. The main part of the actual interoperability happens via what is depicted at the arrow connection the two platform-specific information models: semantic mapping. This allows to define how the platform-specific extension of one platform can be translated into the platform-specific extensions of another platform and therefore allows to define an arbitrary degree of interoperability between two platforms. When an app or a platform queries the Core API to find resources of interest on all available platforms symbIoTe uses these mappings to re-write the query to fit the platform-specific information model of each platform and execute it against the meta data it has stored about each of them. Details in the symbIoTe Information Model as well as semantic mapping and SPARQL query re-writing are provided in the following sections.

5.2 symbIoTe Information Model

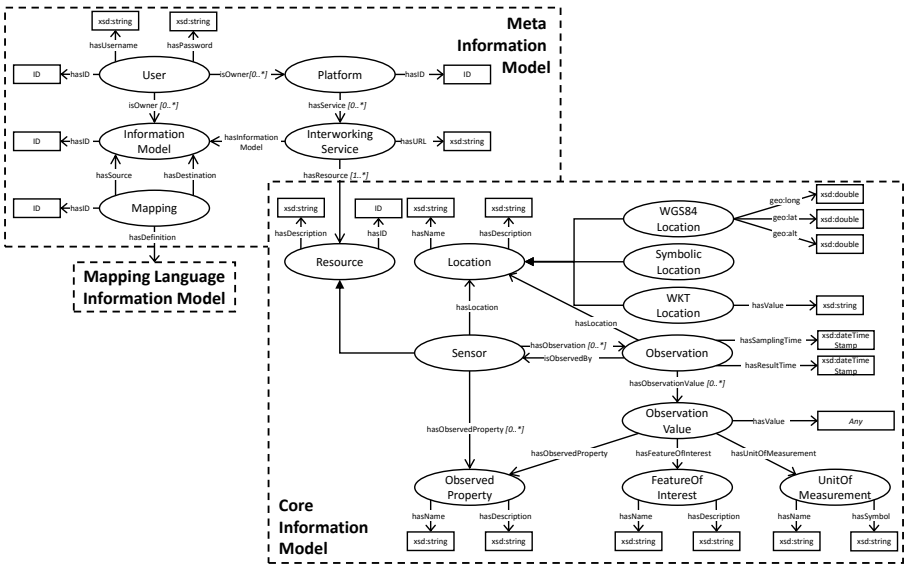


Fig. 8. SymbIoTe Information Model.

The symbIoTe Information Model is comprised of two parts as depicted in Figure 8 The first part is the Meta Information Model which covers all meta data about platforms that symbIoTe needs to store internally such as which platforms uses which Information Model and the URL of the Interworking API endpoint of a platform. Furthermore, it keeps track of the mappings between different information models that are described using any Mapping Language Information Model which will be explained in detail in the following section.

The second part is the Core Information Model. Its design was driven by the trade-off to keep it is abstract as possible to not unnecessarily exclude any platform (as it may use another information model that doesn't fit the Core Information Model) but to include all information that symbIoTe needs to understand from the platforms. This is due to the fact that symbIoTe internally can only understand the information coming from the platforms that is modeled within the Core Information Model but not the platform-specific extensions defined in the platform-specific information models. A good example for that is how locations are modeled. Initially, there was only Location defined and no sub-classes (which was thought to be defined in the platform-specific information models). But as symbIoTe needs to be able to internally understand the location to be able to provide location-based query capabilities for platform resources three sub-classes were defined to be used in the platform-specific information models so that symbIoTe can make use of the location information as it now is able to understand the meaning of this data.

Because this trade-off leaves only a very narrow solution space symbIoTe cannot just re-use any existing ontology because their scope was either to abstract like the Semantic Sensor Network (SSN) ontology [4] thus resulting in symbIoTe not being able to understand the minimal information needed to work properly, or to narrow like the oneM2M Base Ontology [15] which would result in unnecessarily excluding platforms by over-specifying the information model.

For this reason, symbIoTe defines its own domain ontology for the IoT especially tailored to that narrow solution space to satisfy the trade-off between a desired high-level of abstraction and a needed concretization of some terms.

5.3 Semantic Mapping and SPARQL Query Re-Writing

Semantic mapping and SPARQL query re-writing are closely related and together the most essential parts of providing semantic interoperability in symbIoTe. In this section we narrow down the general term *information model* used so far in this paper to refer to an information model realized as an ontology. Semantic mapping can therefore also be called ontology mapping in this section and is motivated by the fact that different IoT platforms may use different ontologies to describe their available resources covering (partially) the same domain and therefore could generally be (partially) interoperable. Even if these platform-specific ontologies essentially cover the same domain they can describe this domain quite differently, e.g. use a taxonomy with another scope or granularity or use another terminology. These differences are called ontology mismatches and there exist multiple classifications for them [22, 23, 13, 18, 16]. Based on which of these types

of mismatches a system should be able to resolve it needs to choose or develop a mapping language that offers language constructs to resolve these mismatches. At the current state, symbIoTe is using the Expressive and Declarative Ontology Alignment Language (EDOAL) [7, 9] which supports quite a lot mismatch types [18, 17].

Having defined mappings/alignments between the different platform-specific information models/ontologies the platforms use to describe their available resources we don't instantly gain anything. Rather we need to implement some logic to access data presented in these different models in a unified way. Therefore we need some execution environment with some kind of mediator like depicted in Figure 5. As shown in Figure 7 this is done in symbIoTe by SPARQL query re-writing based on the mapping definitions between two information models. In order to perform SPARQL query re-writing we are using the Mediation Toolkit [3] by Correndo which functionality is explained in detail here [5, 6].

The overall algorithm for executing a SPARQL query formulated based on the platform-specific ontology of one IoT platform against the data of all platforms is comprised of three steps:

1. find all platforms and their interworking services for which a mapping exists from the ontology the query is formulated in (see Meta Information Model depicted in Figure 8)
2. for each of the found mappings
   - re-write original SPARQL query based on the mapping
   - execute it against stored information about available resources
   - transform results back to match the ontology the query was originally formulated in
3. collect and return results

As symbIoTe is currently work in progress and these are still areas of research the chosen mapping language and SPARQL query re-writing framework might change in the future.

6 Conclusions & Future Work

In this paper we introduced five possible approaches to achieve semantic interoperability along with detailed considerations on problems and risks to keep in mind when trying to implement them. We further presented the symbIoTe project as an example how to achieve not only internal but external interoperability as introduced in [11]. Moreover, the details of how symbIoTe realizes the Core Information Model with Extensions approach using semantic mapping and SPARQL query re-writing as core technologies was shown.

As the symbIoTe project is currently work in progress please note that the implementation details and used frameworks are subject to change. Furthermore, the following issues regarding the symbIoTe Information Model will be addressed in the future:

[3] https://github.com/correndo/mediation

- add support for other resource types (actuators, services),
- revise modeling of Observations with focus on Location and FeatureOfInterest with regard to mobile sensors, and
- better user management.

Analyzing symbIoTe's approach we conclude that further research in the area of mapping languages and SPARQL query re-writing is essential for creating an interoperability framework for IoT platforms as these techniques are needed in three out of five possible approaches introduced in Section 3.

Acknowledgement

This work is supported by the H2020 symbIoTe project, which has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688156. The authors would like to cordially thank the entire symbIoTe consortium for their valuable comments and discussions.

References

1. Merriam-webster dictionary. http://www.merriam-webster.com/dictionary/semantics, accessed: 2016-09-23
2. Berners-Lee, T., Hendler, J., Lassila, O., et al.: The semantic web. Scientific american 284(5), 28–37 (2001)
3. Choi, N., Song, I.Y., Han, H.: A survey on ontology mapping. ACM Sigmod Record 35(3), 34–41 (2006)
4. Compton, M., Barnaghi, P., Bermudez, L., GarcíA-Castro, R., Corcho, O., Cox, S., Graybeal, J., Hauswirth, M., Henson, C., Herzog, A., et al.: The ssn ontology of the w3c semantic sensor network incubator group. Web Semantics: Science, Services and Agents on the World Wide Web 17, 25–32 (2012)
5. Correndo, G., Salvadores, M., Millard, I., Glaser, H., Shadbolt, N.: Sparql query rewriting for implementing data integration over linked data. In: Proceedings of the 2010 EDBT/ICDT Workshops. p. 4. ACM (2010)
6. Correndo, G., Shadbolt, N.: Translating expressive ontology mappings into rewriting rules to implement query rewriting. In: International Semantic Web Conference, Ontology Matching workshop. CEUR-WS (2011)
7. David, J., Euzenat, J., Scharffe, F., Trojahn dos Santos, C.: The alignment api 4.0. Semantic web 2(1), 3–10 (2011)
8. European Commission: Accompanying the document communication from the commission to the european parliament, the council, the european economic and social committee and the committee of the regions digitising european industry: Reaping the full benefits of a digital single market. Comission Staff Working Document (2016), http://www.kowi.de/Portaldata/2/Resources/fp/2016-SWD-Internet-of-Things.pdf
9. Euzenat, J., Scharffe, F., Zimmermann, A.: Expressive alignment language and implementation. Knowledge Web project report, KWEB/2004/D2.2.10/1.0 (2007)
10. Gruber, T.R., et al.: A translation approach to portable ontology specifications. Knowledge acquisition 5(2), 199–220 (1993)

11. Herzog, R., Jacoby, M., Podnar Žarko, I.: Semantic interoperability in iot-based automation infrastructures. AT - Automation Technology: Methods and Applications of Control, Regulation, and Information Technology (2015)
12. Ide, N., Pustejovsky, J.: What does interoperability mean, anyway? toward an operational definition of interoperability for language technology. In: Proceedings of the Second International Conference on Global Interoperability for Language Resources. Hong Kong, China (2010)
13. Klein, M.: Combining and relating ontologies: an analysis of problems and solutions. In: IJCAI-2001 Workshop on ontologies and information sharing. pp. 53–62. USA. (2001)
14. Network-Centric Operations Industry Consortium: Systems, capabilities, operations,programs, and enterprises (scope) model for interoperability assessment (2008)
15. oneM2M Partners Type 1: oneM2M base ontology. Tech. Rep. TS-0012-V2.0.0, oneM2M (Aug 2016)
16. Rebstock, M., Janina, F., Paulheim, H.: Ontologies-based business integration. Springer Science & Business Media (2008)
17. Scharffe, F.: Correspondence patterns representation. Ph.D. thesis, University of Innsbruck (2009)
18. Scharffe, F., Zamazal, O., Fensel, D.: Ontology alignment design patterns. Knowledge and Information Systems 40(1), 1–28 (2014)
19. Soursos, S., Žarko, I.P., Zwickl, P., Gojmerac, I., Bianchi, G., Carrozzo, G.: Towards the cross-domain interoperability of iot platforms. In: 2016 European Conference on Networks and Communications (2016)
20. Studer, R., Benjamins, V.R., Fensel, D.: Knowledge engineering: principles and methods. Data & knowledge engineering 25(1), 161–197 (1998)
21. Tolk, A., Wang, W., Wang, W.: The levels of conceptual interoperability model: applying systems engineering principles to m&s. In: Proceedings of the 2009 Spring Simulation Multiconference. p. 168. Society for Computer Simulation International (2009)
22. Visser, P.R., Jones, D.M., Bench-Capon, T.J., Shave, M.J.: Assessing heterogeneity by classifying ontology mismatches. In: Proceedings of the FOIS. vol. 98 (1998)
23. Visser, P.R., Jones, D.M., Bench-Capon, T.J., Shave, M.: An analysis of ontology mismatches; heterogeneity versus interoperability. In: AAAI 1997 Spring Symposium on Ontological Engineering, Stanford CA., USA. pp. 164–72 (1997)
24. Wache, H., Voegele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., Hübner, S.: Ontology-based integration of information-a survey of existing approaches. In: IJCAI-01 workshop: ontologies and information sharing. vol. 2001, pp. 108–117. Citeseer (2001)