ISO/IEC 21823-3

Edition 1.0 2021-09

INTERNATIONAL STANDARD

colour inside

Internet of Things (IoT) – Interoperability for IoT systems – Part 3: Semantic interoperability

IEC Central Office
3, rue de Varembé
CH-1211 Geneva 20
Switzerland

Tel.: +41 22 919 02 11
info@iec.ch
www.iec.ch

About the IEC
The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications
The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigendum or an amendment might have been published.

IEC publications search - webstore.iec.ch/advsearchform
The advanced search enables to find IEC publications by a variety of criteria (reference number, text, technical committee, …). It also gives information on projects, replaced and withdrawn publications.

IEC Just Published - webstore.iec.ch/justpublished
Stay up to date on all new IEC publications. Just Published details all new publications released. Available online and once a month by email.

IEC Customer Service Centre - webstore.iec.ch/csc
If you wish to give us your feedback on this publication or need further assistance, please contact the Customer Service Centre: sales@iec.ch.

IEC online collection - oc.iec.ch
Discover our powerful search engine and read freely all the publications previews. With a subscription you will always have access to up to date content tailored to your needs.

Electropedia - www.electropedia.org
The world's leading online dictionary on electrotechnology, containing more than 22 000 terminological entries in English and French, with equivalent terms in 18 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

ISO/IEC 21823-3

Edition 1.0 2021-09

INTERNATIONAL STANDARD

colour inside

Internet of Things (IoT) – Interoperability for IoT systems – Part 3: Semantic interoperability

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

ICS 35.020; 35.110

ISBN 978-2-8322-1019-4

Warning! Make sure that you obtained this publication from an authorized distributor.

CONTENTS

INTERNET OF THINGS (IoT) – INTEROPERABILITY FOR IoT SYSTEMS –

Part 3: Semantic interoperability

FOREWORD

1) ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

2) The formal decisions or agreements of IEC and ISO on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC and ISO National bodies.

3) IEC and ISO documents have the form of recommendations for international use and are accepted by IEC and ISO National bodies in that sense. While all reasonable efforts are made to ensure that the technical content of IEC and ISO documents is accurate, IEC and ISO cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.

4) In order to promote international uniformity, IEC and ISO National bodies undertake to apply IEC and ISO documents transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC and ISO document and the corresponding national or regional publication shall be clearly indicated in the latter.

5) IEC and ISO do not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC and ISO marks of conformity. IEC and ISO are not responsible for any services carried out by independent certification bodies.

6) All users should ensure that they have the latest edition of this document.

7) No liability shall attach to IEC and ISO or their directors, employees, servants or agents including individual experts and members of its technical committees and IEC and ISO National bodies for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this ISO/IEC document or any other IEC and ISO documents.

8) Attention is drawn to the Normative references cited in this document. Use of the referenced publications is indispensable for the correct application of this document.

9) Attention is drawn to the possibility that some of the elements of this ISO/IEC document may be the subject of patent rights. IEC and ISO shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 21823-3 has been prepared by subcommittee 41: Internet of Things and related technologies, of ISO/IEC joint technical committee 1: Information technology. It is an International Standard.

The text of this International Standard is based on the following documents:

Draft	Report on voting
JTC1-SC41/233/FDIS	JTC1-SC41/244/RVD

Full information on the voting for its approval can be found in the report on voting indicated in the above table.

The language used for the development of this International Standard is English.

This document was drafted in accordance with ISO/IEC Directives, Part 2, and developed in accordance with ISO/IEC Directives, Part 1, available at: www.iec.ch/members_experts/refdocs and www.iso.org/directives.

IMPORTANT – The 'colour inside' logo on the cover page of this publication indicates that it contains colours which are considered to be useful for the correct understanding of its contents. Users should therefore print this document using a colour printer.

INTRODUCTION

The use of the Internet of Things (IoT) is increasing every year, in application areas such as manufacturing, healthcare, and new cross-domain applications related to smart cities (e.g. water, energy, transport, or health). Most IoT systems want to share information, which can be done by interoperability. Mechanisms are therefore needed on how to exchange information and use associated data and data description.

IoT interoperability is described as a successful interaction among entities specified in ISO/IEC 30141 [1] [1], for instance between IoT services provided by different IoT service providers. It can be achieved using the interoperability facet model defined in ISO/IEC 21823-1, which defines five facets: transport, syntactic, semantic, behavioural and policy interoperability.

IoT semantic interoperability is the facet which enables the exchange of data between IoT systems using understood data information models (or semantic meanings). According to a recently published white paper [2]:

> "Semantic interoperability is achieved when interacting systems attribute the same meaning to an exchanged piece of data, ensuring consistency of the data across systems regardless of individual data format. This consistency of meaning can be derived from pre-existing standards or agreements on the format and meaning of data or it can be derived in a dynamic way using shared vocabularies either in a schema form and/or in an ontology-driven approach."

As shown in Figure 1,

- semantic interoperability means that information in different data information models can be translated into understandable meaning and exchanged between applications;
- semantic interoperability provides the capability for applications to understand exchanged information;
- semantic interoperability for IoT is achieved by invoking services, and by using specific knowledge and concepts of IoT.

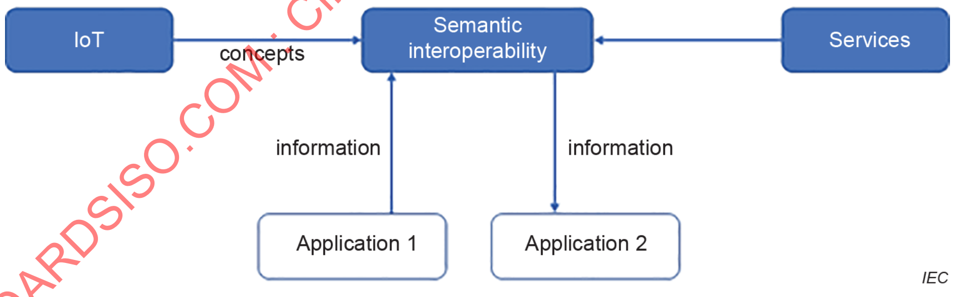


Figure 1 – Semantic interoperability facet for IoT

Semantic interoperability is achieved through the use of metadata, or descriptions of data. The approach of providing data and descriptions has been widely used in IT systems. Two examples are:

a) conceptual schemas have been used to describe database content;

b) record layouts have been used to display the content of a database record.

As shown in Figure 2, many services invoked by semantic interoperability involve metadata, thus enabling their discovery, understanding and (re)usability.

[1] Numbers in square brackets refer to the Bibliography.

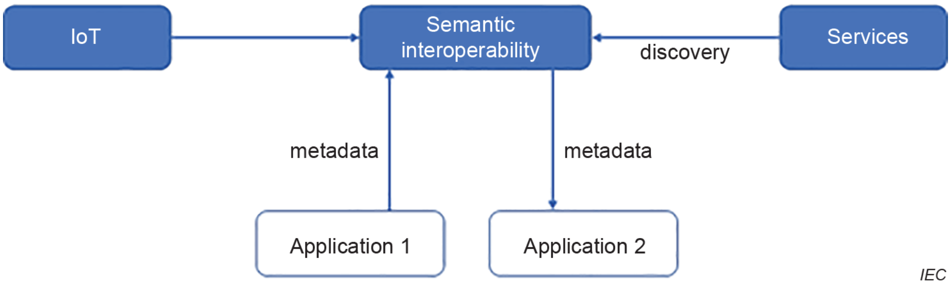


Figure 2 – Using metadata in semantic interoperability

Metadata provides IoT systems with a common understanding of exchanged data. Figure 3 shows how the meaning of data is defined by the metadata to a specific room temperature (left column) and how it is described with metadata (right column).

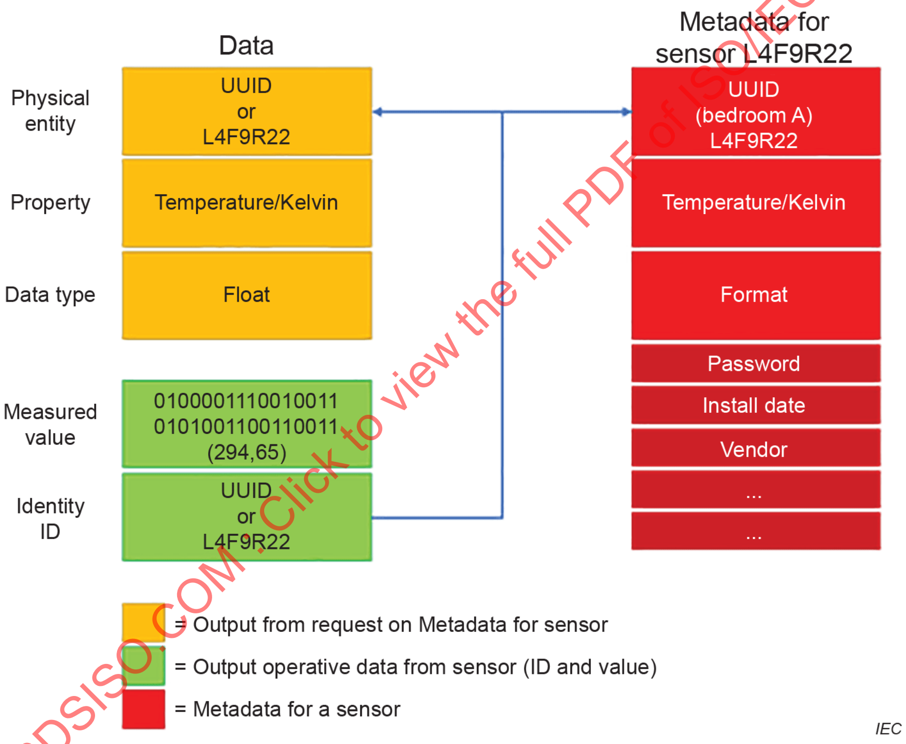


Figure 3 – Meaningfulness of the data, described with metadata

Knowledge that metadata represents can be described using ontologies. In other words, semantic interoperability needs shared, unambiguous, machine-understandable metadata, to be able to perform exchange of information using metadata. The application of semantics in IoT has still been limited because most metadata are developed independently, making it difficult for IoT entities or applications to interoperate semantically. In this document, an ontology-driven approach for semantic interoperability is specified to design and specify metadata, so that the sensors, devices, systems and services can express metadata information and data by applying the ontologies to achieve semantic interoperability. Stakeholders targeted by this document include ontology engineers and IoT system engineers who are building semantic interoperability capabilities for IoT systems.

This document also specifies methods and techniques to build semantic interoperability for IoT systems. Clause 5 focuses on the IoT semantic interoperability process. Clause 6 focuses on the IoT semantic interoperability life cycle management.

Informative annexes provide additional information and guidance. Annex A, Annex B and Annex C provide guidance on how to learn IoT semantic interoperability, develop IoT semantic interoperability, and manage IoT semantic interoperability life cycle, respectively. Annex D provides ontological specification of the IoT Reference Architecture specified in ISO/IEC 30141 [1]. Annex E provides related existing ontologies that are applicable for IoT semantic interoperability.

INTERNET OF THINGS (IoT) – INTEROPERABILITY FOR IoT SYSTEMS –

Part 3: Semantic interoperability

1 Scope

This document provides the basic concepts for IoT systems semantic interoperability, as described in the facet model of ISO/IEC 21823 -1, including:

- requirements of the core ontologies for semantic interoperability;
- best practices and guidance on how to use ontologies and to develop domain-specific applications, including the need to allow for extensibility and connection to external ontologies;
- cross-domain specification and formalization of ontologies to provide harmonized utilization of existing ontologies;
- relevant IoT ontologies along with comparative study of the characteristics and approaches in terms of modularity, extensibility, reusability, scalability, interoperability with upper ontologies, and so on;
- use cases and service scenarios that exhibit necessities and requirements of semantic interoperability.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 20924, *Internet of Things (IoT) – Vocabulary*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 20924 and the following apply. ISO and IEC maintain terminological databases for use in standardization at the following addresses:

- IEC Electropedia: available at http://www.electropedia.org/
- ISO Online browsing platform: available at http://www.iso.org/obp

3.1
semantic interoperability
interoperability so that the meaning of the data model within the context of a subject area is understood by the participating systems

[SOURCE: ISO/IEC 19941:2017, 3.1.5, modified – In the term, "data" has been deleted.]

3.2
metadata
data that defines and describes other data

[SOURCE: ISO/IEC 11179-3:2013, 3.2.74]

3.3
ontology
specification of concrete or abstract things, and the relationships among them, in a prescribed domain of knowledge

Note 1 to entry: The specification should be computer processable.

[SOURCE: ISO/IEC 19763-3:2020, 3.1.1.1]

4 Abbreviated terms

ICT	information and communication technology
IoT	Internet of Things
PKI	public key infrastructure

5 IoT semantic interoperability process

5.1 Overview

IoT semantic interoperability enables the exchange of data between IoT systems using understood data information models (or semantic meanings). Such models are based on ontologies, which are processable specifications of concepts and relations concerning IoT systems. The benefit of IoT semantic interoperability is to contribute to meet interoperability, reusability, scalability or trustworthiness properties. IoT semantic interoperability has the following stakeholders and concerns:

a) ontology engineers, who focus on ontology development in a concern;

b) IoT system engineers, who focus on IoT system development and integration of semantic interoperability capabilities in a concern.

As shown in Figure 4, this document provides ontology engineers and IoT system engineers with requirements and specification of methods to prepare and build semantic interoperability.

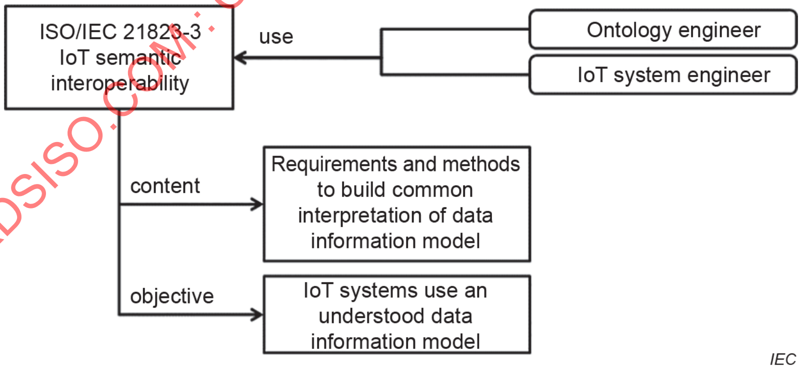

ISO/IEC 21823-3
IoT semantic
interoperability
use
Ontology engineer
IoT system engineer
content
Requirements and methods to build common interpretation of data information model
objective
IoT systems use an understood data information model
IEC

Figure 4 – Objective of semantic interoperability standard

Two types of requirements are identified:

1) IoT semantic interoperability requirements, which focus on how to create data information models, how to develop and integrate interpretation capability in IoT systems;

2) life cycle requirements, which focus on the management of data information model perimeters, of data information model design, and of data information model maintenance.

5.2 IoT semantic interoperability process requirements

The following processes shall be carried out by IoT system engineers and ontology engineers:

- data information model creation;
- semantic interoperability capability integration;
- semantic interoperability engineering support.

Table 1 provides for each process a list of requirements with a rationale and a resulting work product that can be used as evidence for conformance.

Table 1 – IoT semantic interoperability process requirements

Processes	Requirements	Rationale	Resulting work product
Data information model creation	Capture common knowledge using common ontologies	IoT systems rely on common architecture, capabilities and communication means, e.g. IoT sensors, actuators, gateways, devices.	Data information model based on the gathered and documented common knowledge on the IoT system
	Capture domain-specific knowledge using domain ontologies	IoT systems can focus on specific domains such as transport, energy, health.	Domain specific data information model on the IoT system
	Capture cross-domain knowledge using cross-domain ontologies	IoT systems can cover cross-domains. For instance, an electric-mobility IoT system belongs to the transport and the energy domain.	Cross-domain specific data information model on the IoT system
Semantic interoperability capability integration	Develop an IoT system capability to exchange information about IoT device interoperability	IoT systems can use various interoperability parameters including technical parameters (e.g. drivers, protocols) and semantic parameters (e.g. application capabilities). Exchange of such parameters can take place using metadata prior to device nominal operation. Metadata can be exposed in the form of schema or subschema or by OWL/XML or JSON-LD. The exchange can involve a proxy when an IoT device cannot provide the capability directly.	IoT system capability to exchange information on IoT device interoperability
	Develop an IoT system capability to negotiate interoperability profiles	IoT systems can use multiple knowledge representations. Agreement is needed on the representation to be used for information exchange. Negotiation is needed to agree on the level of semantic knowledge that is at stake, e.g. using representations of parameters and agreeing on interoperability profiles. The negotiation can involve a proxy when an IoT device cannot provide the capability directly.	IoT system capability to negotiate interoperability profiles
	Validate and integrate the various capabilities to enable IoT device interoperability	IoT systems integrate multiple technologies to enable interoperability (e.g. drivers, protocols, middleware) corresponding to different interoperability facets. They also integrate multiple operations (e.g. discovery, negotiation) to enable interoperability.	IoT device providing overall interoperability capability
Semantic interoperability engineering support	Provide and apply tools to achieve semantic interoperability	Engineers need supporting tools such as ontology discovery and selection, mapping, alignment, merging and integration.	Selection of tools to achieve semantic interoperability

5.3 IoT semantic interoperability models

IoT semantic interoperability can be described with two models: a process model and a usage model. The process model focuses on how semantic interoperability capability is created. The usage model focuses on how semantic information is used and exchanged.

Figure 5 provides the process model for IoT semantic interoperability. It is based on the assumption that interoperability knowledge is made available in processable forms such as the knowledge graph, constructed through ontologies. The entities in Figure 5 will be further described in 5.4.1.

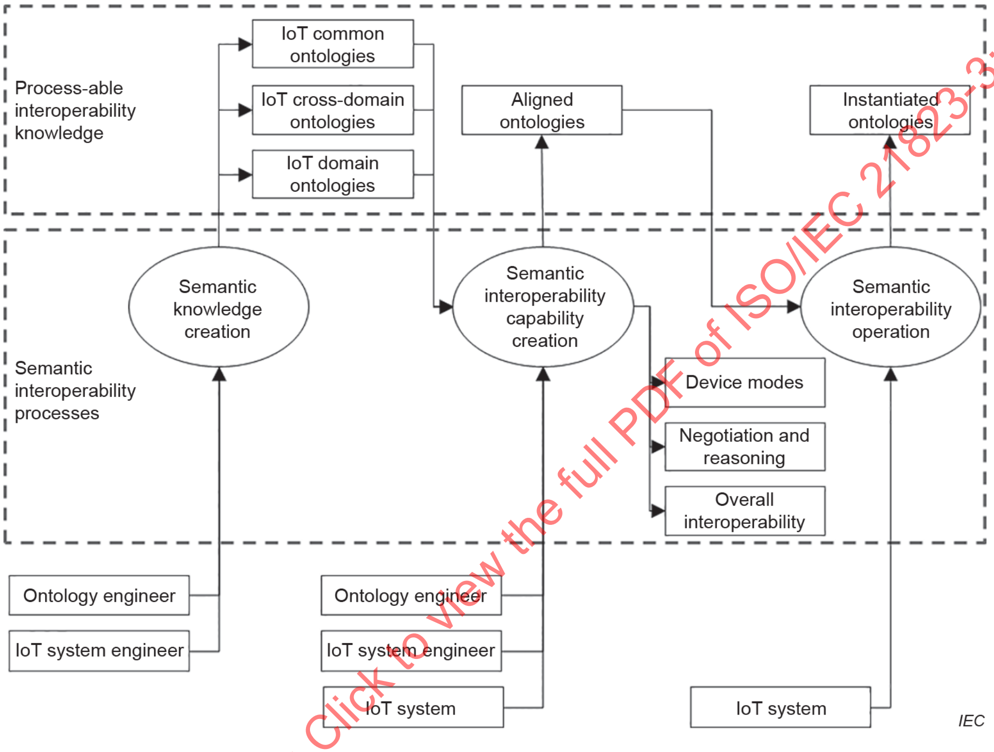

IoT common ontologies
Process-able interoperability knowledge
IoT cross-domain ontologies
IoT domain ontologies
Aligned ontologies
Instantiated ontologies
Semantic knowledge creation
Semantic interoperability capability creation
Semantic interoperability operation
Semantic interoperability processes
Device modes
Negotiation and reasoning
Overall interoperability
Ontology engineer
IoT system engineer
Ontology engineer
IoT system engineer
IoT system
IoT system
IEC

Figure 5 – IoT semantic interoperability process model

IoT semantic interoperability involves the following stakeholders:

- ontology engineers, who focus on ontology development;
- IoT system engineers, who focus on the development of IoT systems and the integration of interoperability capabilities.

IoT semantic interoperability includes the following processes.

a) Semantic knowledge creation: the purpose of this process is to create semantic information for interoperability. This process is carried out by ontology engineers and IoT system engineers. The input to this process is the knowledge of the engineers. The outcome of this process is a consistent digital representation of the semantic information used for interoperability, represented by ontologies, including IoT common ontologies, IoT cross-domain ontologies and IoT domain ontologies.

b) Semantic interoperability capability creation: the purpose of this process is to create semantic interoperability capabilities and to adapt semantic information to a desired semantic information representation. The capabilities are the following:

   1) device mode capability;

   2) negotiation and reasoning capability;

   3) overall interoperability.

This process is carried out by ontology engineers, IoT system engineers as well as the IoT system itself if it has semantic information adaptation processing capabilities. The input to this process is the IoT domain, IoT cross-domain and IoT common ontologies. The outcome of this process is aligned ontologies.

This process is assisted by ontology engineering support tools, as well as reasoning capabilities such as rule-based reasoning, ontology-based reasoning or machine learning.

c) Semantic interoperability use: the purpose of this process is to exchange semantic information with other IoT systems. This process is carried out by the IoT system. The input to this is the aligned ontologies. The outcome of this process is instantiated ontologies.

This process is assisted by ontology engineering support tools, as well as reasoning capabilities such as rule-based reasoning, ontology-based reasoning or machine learning.

The use of semantic information is based on the model shown in Figure 6.

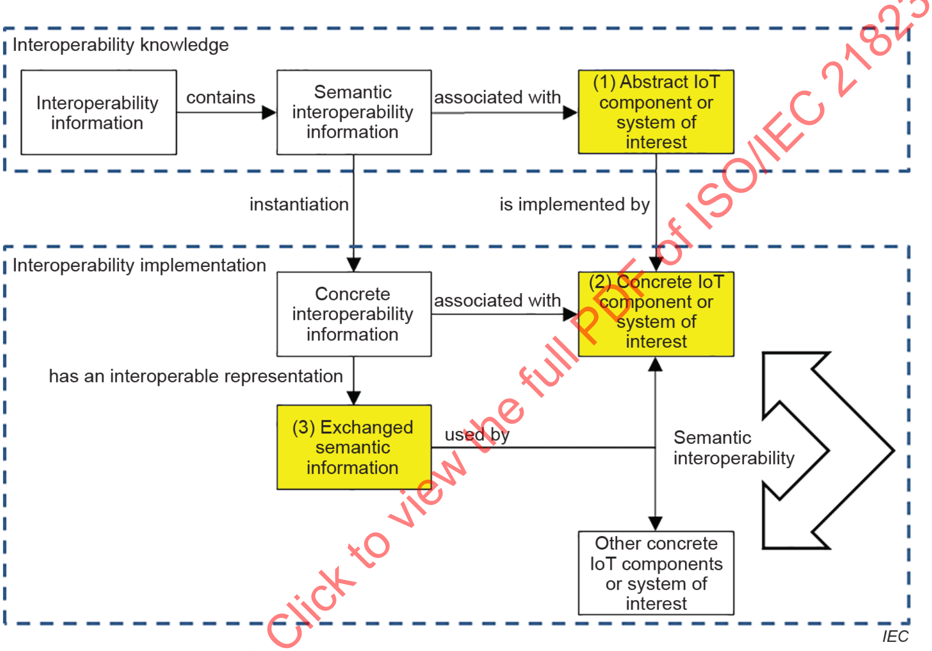

Interoperability knowledge
Interoperability information
contains
Semantic interoperability information
associated with
(1) Abstract IoT component or system of interest
instantiation
is implemented by
Interoperability implementation
Concrete interoperability information
associated with
(2) Concrete IoT component or system of interest
has an interoperable representation
(3) Exchanged semantic information
used by
Semantic interoperability
Other concrete IoT components or system of interest
IEC

Figure 6 – Semantic information usage model

The semantic information usage model includes the following.

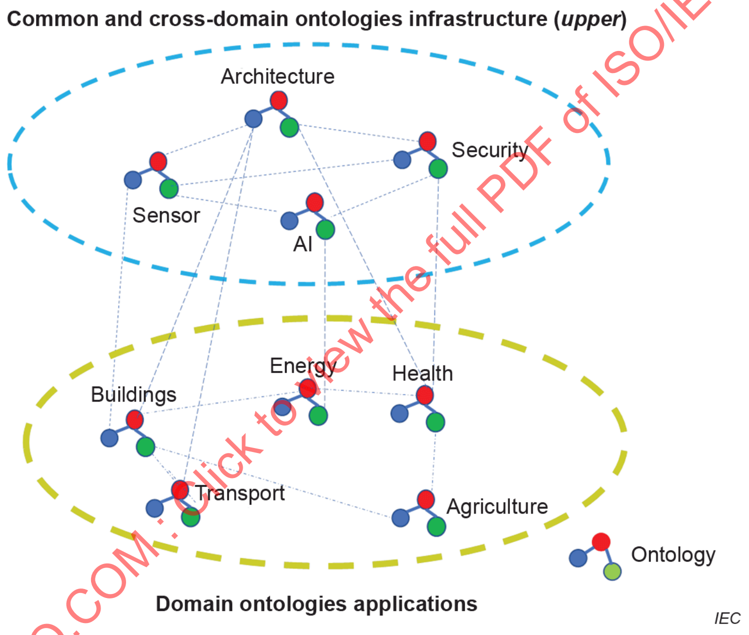
- An abstract model of an IoT component or system-of-interest: interoperability information contains semantic interoperability information which is associated with an abstract IoT component or system-of-interest in a domain. An example is a temperature model associated with an abstract IoT temperature sensor.
- A concrete model of the IoT component of system-of-interest: concrete interoperability information is associated with the concrete IoT component or system-of-interest in operation. An example is the instantiated temperature model associated with a physical temperature sensor.
- Exchanged semantic information: a computer processable representation of semantic information that can be used by other concrete IoT components of systems. An example is the interoperable representation of the temperature information provided by the physical temperature sensor.

5.4 IoT semantic interoperability guidelines

5.4.1 Guidelines on the capture of semantic meaning

The capture of semantic meaning is achieved through the semantic knowledge creation process as described in Figure 5. It includes the use of the following knowledge.

- IoT common ontologies. They refer to common concepts such as architecture or framework concepts. Ontologies can be based on IoT standards such as ISO/IEC 30141:2018 [1], ISO/IEC 20924:2018 [2], or on mapping specifications such as [3].
- IoT cross-domain ontologies. They refer to shared concepts between domains, at the application level (e.g. electric mobility ontologies which mix energy and transport knowledge), or at the cross-cutting concern. For instance, privacy cross-cutting concerns can be based on privacy standards such as ISO/IEC 29100:2011 [4].
- IoT domain ontologies. For instance, in the energy domain, a common ontology for energy management can be used [5]. Another example could be in the smart city domain [6], in the smart manufacturing domain [7], or in the agriculture domain [8].


Common and cross-domain ontologies infrastructure (*upper*)
Architecture
Security
Sensor
AI
Energy
Health
Buildings
Transport
Agriculture
Ontology
Domain ontologies applications
IEC

NOTE Ontologies are often represented by triples. They are visually displayed in the figure using three connected circles.

Figure 7 – Example of structured knowledge representation

Figure 7 shows an example of how semantic interoperability knowledge can be structured. It includes:

- common and cross-domain ontologies, such as an architecture ontology, a sensor ontology, a security or privacy ontology;
- domain-specific ontologies related to applications, such as a health ontology, a transport ontology, an energy ontology, or an agriculture ontology. There are many domain-specific ontologies that can be used to structure semantic interoperability knowledge. See, for instance, LOV4IoT [9], the Linked Open Vocabulary Initiative for the Internet of Things.

Ontologies are built using semantic web languages such as RDF [10], RDFS [11] and OWL [12], and queried with SPARQL [13].

- RDF is a basic language based on triples. A triple comprises a subject, a predicate, and an object. For instance, "temperature is 39": "temperature" is the subject, "is" is the predicate, and "39" is the object.

- RDFS is an extension of RDF which provides additional representation capabilities such as describing hierarchies (e.g. a body thermometer is a "subclass of" a sensor).
- OWL is a language which enriches RDFS by allowing the creation of new classes and new relationships between classes to describe a specific domain (e.g. add a new "sensor" concept in the IoT domain).
- Triples can be accessed or queried with SPARQL, a query language to interact with RDF, RDFS, and OWL graphs.

A hierarchy of languages, often called semantic web cake, is available from the W3C [14].

A representation for information exchanged is called a serialized dataset. It is a computer processable representation of the semantic dataset. Serialization refers to the translation of a structure into a series of bytes that can be stored and/or transmitted. Examples of methods for such serialization are OWL and JSON-LD [15].

5.4.2 Guidelines on the integration of semantic interoperability capability

The integration of semantic interoperability capability is achieved through the semantic interoperability capability creation process as described in Figure 5. It includes the integration of the following capabilities.

a) An IoT system capability to exchange information about IoT device interoperability. IoT systems have several modes such as operation mode, maintenance mode, or out of service mode:

   1) The set-up mode objective is to define the operational capability of an IoT device. The description of this capability is based on ontologies. The resulting description can be used by devices to inform other IoT systems on their capabilities. This includes technical level parameters (e.g. drivers, protocols, security, quality of service, quality of data) as well as semantic level parameters (e.g. processing capabilities, access rights, application capabilities). The resulting device ontology is used to inform other connected IoT systems on how it will interpret data. For example, a temperature sensor will have attributes on range, accuracy, sensing time and so forth. There are numerous sensor ontology descriptions in public repositories such as LOV4IoT [9] or the Semantic Sensor Network Ontology from W3C [16], [17].

   2) Other modes include the operation mode or the maintenance mode. In the operation mode, IoT systems have agreed on the semantic interoperability conditions and the device can therefore interact, e.g. receive sensing information as well as issue actuating information. In the maintenance mode, IoT system can modify the interoperability parameters.

b) An IoT system negotiation and reasoning capability. Negotiation is needed as there are multiple knowledge representations. Ontologies are subjective representations reflecting different views and requirements of diverse designers. For instance, several ontology descriptions for a given sensor profile can be available, possibly stored in an on-line repository. Operations can be necessary including alignment, mapping, or merging. They can further include reasoning capabilities, such as rule-based reasoning, ontology-based reasoning or machine learning.

c) An IoT device overall interoperability capability. Prior to their operations, IoT devices communication subsystem must have been configured to include all the various components negotiated to achieve interoperability. During operations, semantic descriptions can be exchanged. As shown in Figure 6, there is a hierarchy of three representations: the abstract representation, the concrete representation and the exchanged information. Figure 8 shows the example of an IoT temperature sensor which has been implemented using the ISO/IEC 30141 IoT Reference Architecture [1]:

1) An abstract IoT temperature sensor representation is used, relying on an agreed smart home ontology and on the IoT Reference Architecture based on ISO/IEC 30141. The smart home ontology will contain concepts such as comfort, rooms, energy savings. It contains a model of an abstract IoT temperature sensor which includes a description of capabilities (e.g. read temperature) and refers to the architecture ontology based on ISO/IEC 30141, therefore using concepts such as service, network, IoT device and IoT gateway.
2) A concrete IoT temperature sensor representation is used. This representation contains the actual information. An example of representation format is the RDF graph representation. RDF (Resource Description Framework) is a semantic web language specified by W3C. RDF graphs are structured into triples. They can be accessed or queried with SPARQL.
3) A representation for information exchanged is called serialized dataset. It is a computer processable representation of the semantic dataset. Serialization refers to the translation of a structure into a series of bytes that can be stored and/or transmitted. Examples of methods for such encoding are OWL/XML and JSON-LD.

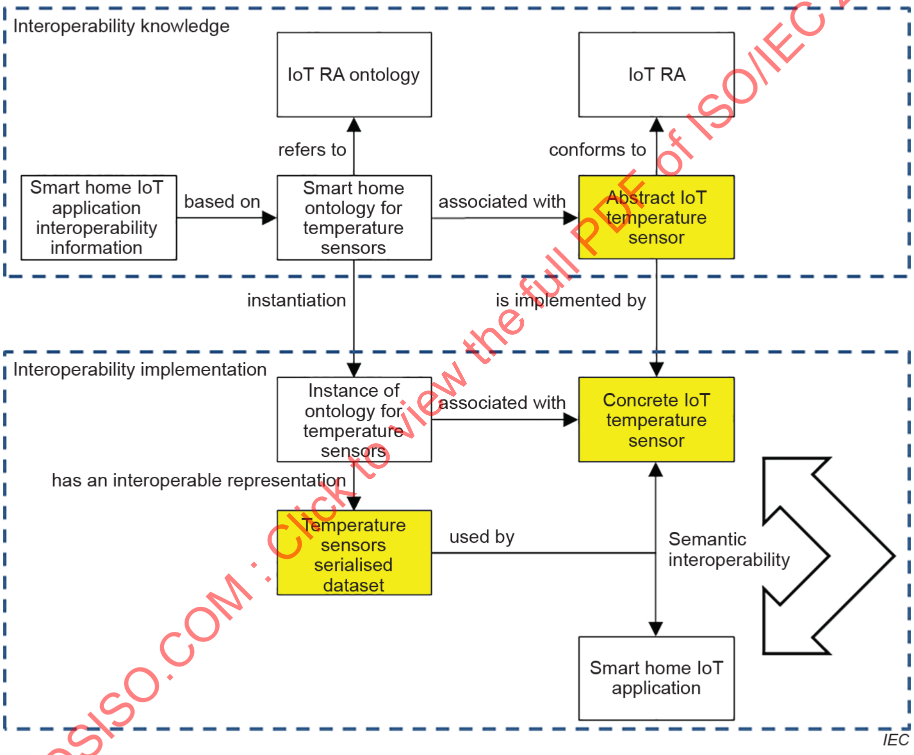

Interoperability knowledge
IoT RA ontology
IoT RA
refers to
conforms to
Smart home IoT application interoperability information
based on
Smart home ontology for temperature sensors
associated with
Abstract IoT temperature sensor
instantiation
is implemented by
Interoperability implementation
Instance of ontology for temperature sensors
associated with
Concrete IoT temperature sensor
has an interoperable representation
Temperature sensors serialised dataset
used by
Semantic interoperability
Smart home IoT application
IEC

Figure 8 – Example of semantic information usage for a temperature sensor

5.4.3 Guidelines on the support of semantic interoperability engineering

5.4.3.1 General

Semantic interoperability engineering requires ontology matching expertise and support for the manipulation of ontologies. Mismatches between ontologies can happen at language-level or ontology-level. The creation of semantic information can therefore include ontology mediation activities in order to reconcile differences between separate ontologies. This can include alignment, mapping, merging and other required operations. For instance, IoT application can use these operations to derive (or formulate) the commands that are sent to an actuator.

NOTE 1 These processes are assisted by ontology engineering support tools and by appropriate guidelines to manage and merge ontologies. Different communities are likely to have somewhat differing requirements, to take different starting points and to use different terminology; there will be challenges when the semantic models evolve over time such is the allocation of identifiers for different versions, or the provision of a model for the changes between subsequent versions; the effort for development and maintenance of ontologies can be eased through the use of machine learning from a corpus of examples and counter examples, thus allowing for automated revisions to data and metadata to track changes to the ontology.

NOTE 2 Whilst the use of the same ontology can be practical in a constrained environment, more generally, one needs to address interoperability between parties using different ontologies. This can involve mapping data and metadata, something that is likely to be context sensitive. One challenge is what to do with missing details when mapping from a weaker ontology to a stronger one. This could be solved in the future by exploiting statistical approaches that characterize prior knowledge and past experience.

The following functions are identified and covered in subsequent subclauses:

- ontology discovery (5.4.3.2);
- ontology-based data integration (5.4.3.3);
- ontological mediation such as ontology mapping, ontology alignment, ontology merging, and ontology integration (5.4.3.4).

5.4.3.2 Ontology discovery

Ontology discovery is the process of accessing operational ontologies and using them for semantic interoperability. As shown in the semantic interoperability model process in Figure 5, this can take place when semantic knowledge is created, as well as later, during device mode set-up. Repositories for ontologies are available, for instance [9], [18], [19]. They can use different ontology (semantic web) languages such as RDFS [11] or OWL [12].

5.4.3.3 Ontology-based data integration

Ontology-based data integration uses ontologies to provide interoperability of data by combining data or information from multiple heterogeneous sources. Data from multiple sources are characterized by the following types of heterogeneity [21].

- Syntactic heterogeneity: the differences in syntax such as formats and structures of data.
- Structural heterogeneity: the differences in the native model or structure to store data.
- Semantic heterogeneity: the differences in interpretation of the meaning of data.
- System heterogeneity: the differences originating from different operating systems and hardware platforms.

Ontologies, as representations of explicitly defined concepts, and of relationships between these concepts, are used to address the issue of semantic heterogeneity in data sources. All information is linked and accessible through a universal identification scheme provided by the ontology (e.g. URL/URI). Ontologies enable the unambiguous identification of entities in heterogeneous information systems and assertion of applicable named relationships that connect these entities together. Specifically, ontologies play the following roles.

a) Content and/or context explication: The ontology enables accurate interpretation of data from multiple sources through the explicit definition of terms and relationships in the ontology.

b) Query model: In some systems, the query is formulated using the ontology as a global query schema. Query formulation is a process during which the original keyword query issued by the user is transformed into a structured query representation that is consumed by the search engine. Commonly the information is queried using SPARQL [13].

c) Verification: The ontology can be used to verify the mappings used to integrate data from multiple sources. These mappings can either be specified manually or generated by a system (i.e. ontology matching tools).

There are three main approaches in ontology-based data integration applications as follows.

1) Single ontology approach: A single ontology is used as a global reference model in the system.
2) Multiple ontologies: Multiple ontologies, each modelling an individual data source, are used in combination for integration. Although this approach is more flexible than the single ontology approach, it requires creation of mappings between the multiple ontologies. Such mappings are likely to be context sensitive, i.e. to depend on the data and metadata.
3) Hybrid approaches: The hybrid approach involves the use of multiple ontologies that are linked to a common, top-level vocabulary. The top-level vocabulary defines the basic terms of the domain. Thus, the hybrid approach makes it easier to use multiple ontologies for integration in presence of the common vocabulary.

5.4.3.4 Ontological mediation

5.4.3.4.1 General

Effective semantic interoperability can be achieved by utilizing an ontology as an integration schema for content interpretation and as a tool for detection and resolution of content inconsistency. Since ontologies are shared specifications, the use of ontology should enable a certain degree of semantic interoperability (e.g. if two systems are designed by two different entities sharing the same ontology then the semantic interoperability is ensured). However, complete interoperability is not guaranteed because ontology has subjective representation reflecting different views and requirements of diverse designers. Such mismatches between ontologies can happen at the language-level or ontology-level. Language-level mismatches come from the difference in expressiveness or semantics of ontology language (e.g. an orange could be a fruit or a colour). Ontology-level mismatches come from the difference in the structure of semantics of the ontology (e.g. an apple can be represented as a concept, and therefore we can describe different varieties of apple, or an apple could just be an instance of the fruit concept).

Ontology mediation is a process that reconciles differences between separate ontologies to achieve semantic interoperability by performing alignment, mapping, merging and other required operations. Ontology matching is the process of finding pairs of resources coming from different ontologies which can be considered equal in meaning. The similarity value typically in the range of [0,1] is used as an input to the different approaches for ontology mediation.

5.4.3.4.2 Ontology mapping

Ontology mapping is the process of relating similar concepts or relations of two or more information sources using equivalence relations or order relations. Figure 9 shows an example of ontology mapping.

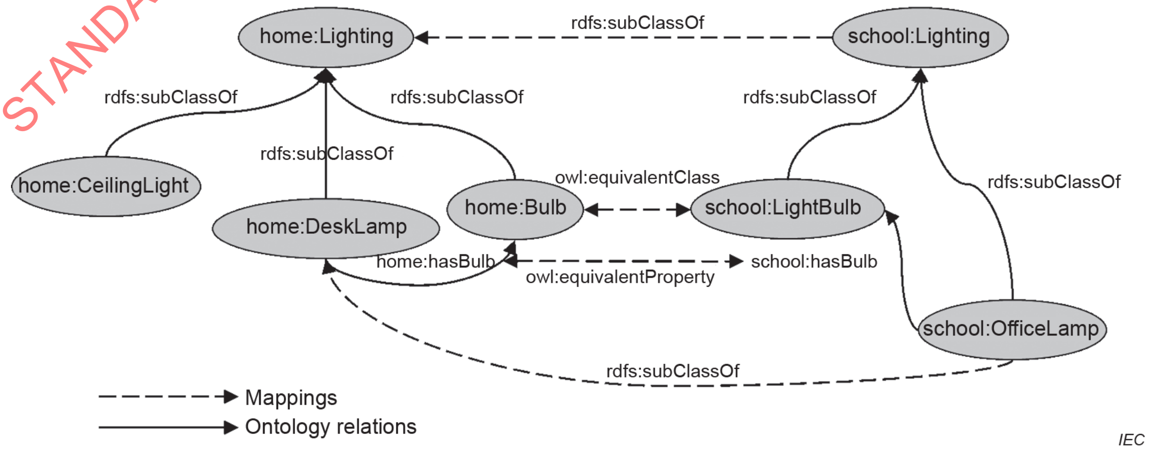


Figure 9 – Example of ontology mapping

5.4.3.4.3 Ontology alignment

Ontology aligning is the process of bringing two or more ontologies into mutual agreement, making them consistent and coherent. Ontology aligning is similar to ontology mapping but it makes use of more expressive relations between ontology concepts such as "partOf" and "subsumes". Figure 10 shows an example of ontology alignment.

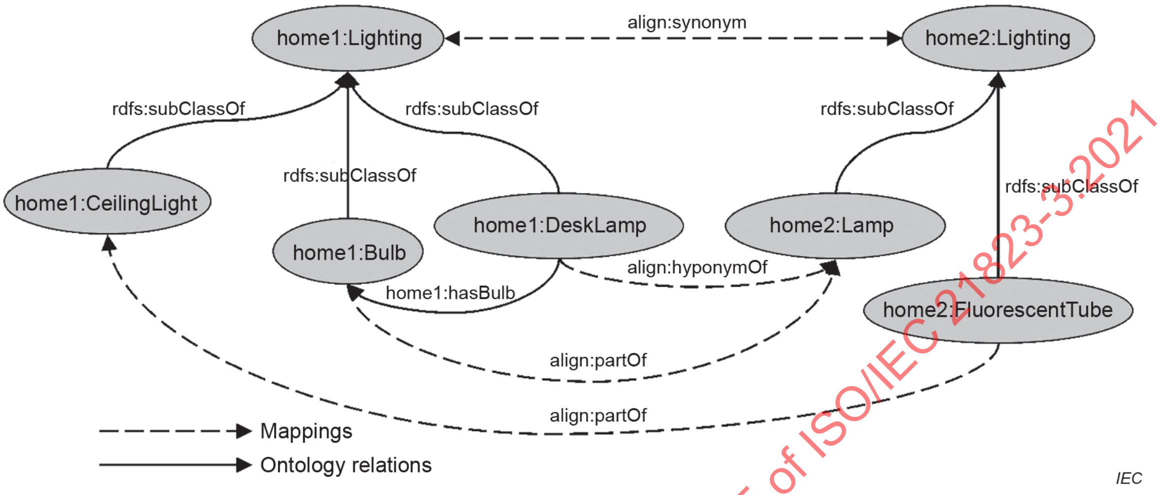


Figure 10 – Example of ontology alignment

5.4.3.4.4 Ontology merging

Ontology merging is the process of creating one ontology from two or more source ontologies with overlapping concepts or definitions. In the merging process, the merged ontology is created from scratch, unifying all the source ontologies. Figure 11 illustrates an example of ontology merging.

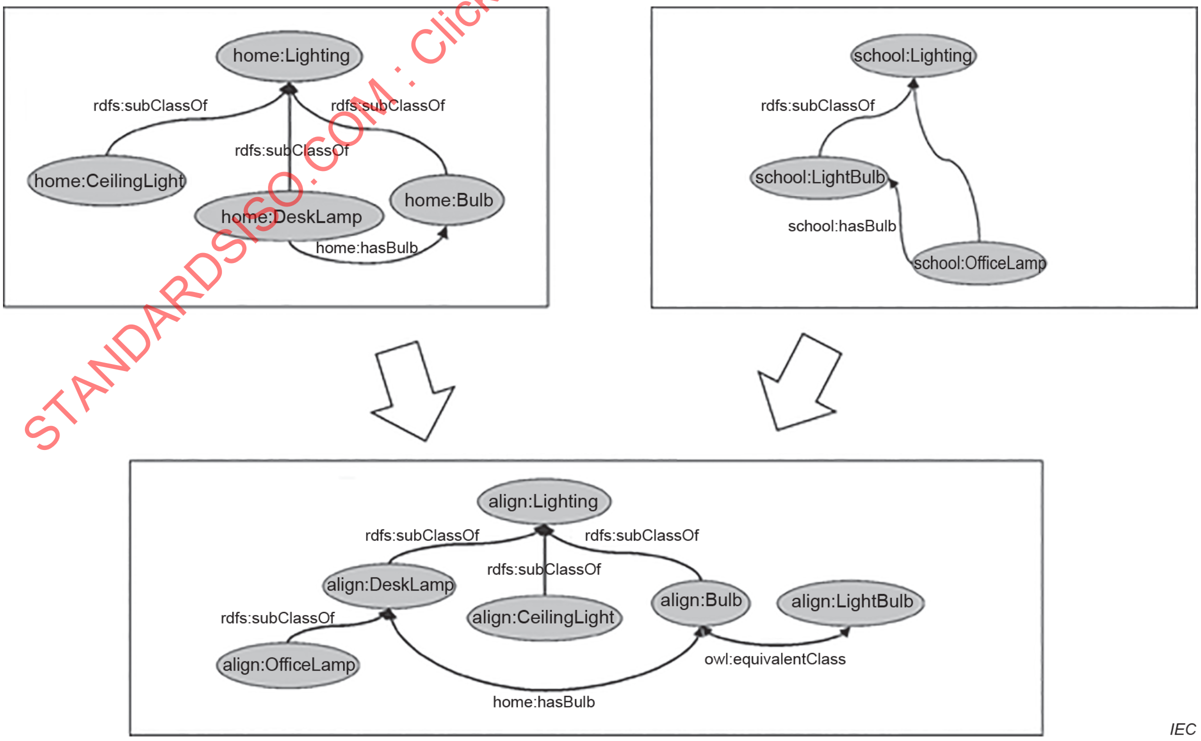


Figure 11 – Example of ontology merging

5.4.3.4.5 Ontology integration

Ontology integration is similar to ontology merging, but the integrated ontology is created reusing parts of source ontologies as they are. Figure 12 shows an example of ontology integration.

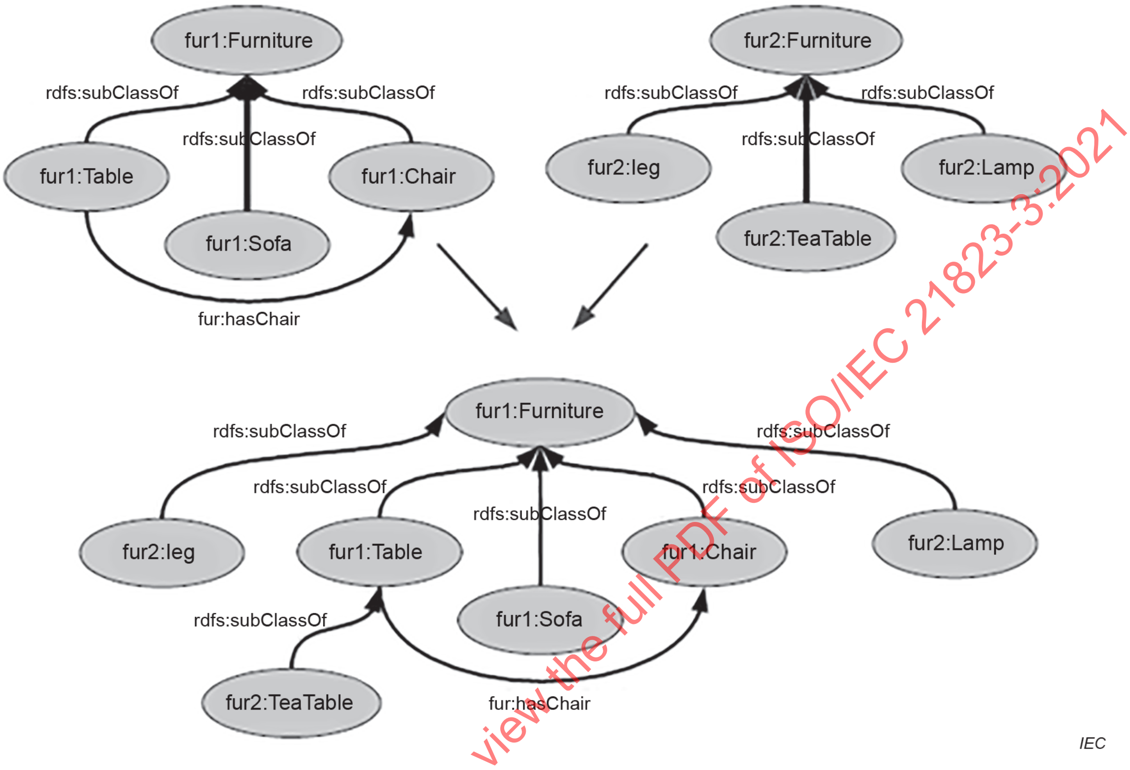


Figure 12 – Example of ontology integration

6 IoT semantic interoperability life cycle

6.1 Life cycle requirements

The following activities shall be carried out by organizations developing IoT systems:

- manage semantic interoperability perimeter, i.e. the amount of knowledge that is needed;
- manage semantic interoperability design;
- manage resulting semantic interoperability work products, i.e. the output of the semantic interoperability process.

Table 2 provides for each activity a list of requirements with a rationale and a resulting work product that can be used as evidence for conformance.

Table 2 – Llife cycle requirements

Activity	Requirements	Rationale	Resulting work products
Manage semantic interoperability perimeter	Define knowledge perimeter	The practice of semantic interoperability requires a clear definition of the knowledge perimeter, i.e. the set of concepts and relationships that are used in a semantic interoperability specification. If the selected knowledge perimeter is too large, then many concepts that are defined in the ontology might not be used. It could also be counter effective. For instance, when cross-domain ontologies are used, it is important not to access an entire domain ontology, but rather the subset that is useful to build a cross-domain ontology. If the selected perimeter is too small then needed concepts in the specification are missing, resulting in an incomplete semantic specification.	Description of method for knowledge perimeter specification. Specification of the knowledge perimeter of the ontologies to be used
Manage semantic interoperability design	Co-create ontologies using the separation of concern principle	Co-creation is a design approach that brings humans with different expertise and viewpoints together – for instance, a domain expert and an ontology expert – in order to jointly produce a mutually valued outcome. Stakeholders separation of concerns is a design principle for separating an item to design into distinct elements, so that each element addresses a separate concern. The practice of creating ontologies for interoperability requires both: – co-creation between domain engineering and semantic interoperability experts. Depending on the domain, other categories of experts might also be needed such as security and privacy experts, or user-centric design experts; and – separation of concerns between domain experts and semantic interoperability experts. The former focuses on functional interoperability specification while the latter focuses on ontology creation expertise. For instance, the development of a healthcare semantic interoperability specification might include ICT experts of health, semantic interoperability experts, security and privacy experts, and users (patients, caregivers).	Description of method for co-creation of ontologies ensuring separation of concern and modular design. Ontologies resulting from the co-creation process and modular design

Activity	Requirements	Rationale	Resulting work products
	Apply modular design of ontologies	The practice of modular design focuses on the structuring of a wider concept into multiple and simpler sub-concepts that can be handled separately. In terms of ontologies, sub-concepts are described by sub-ontologies – that are using self-contained knowledge, – that can be designed, used and maintained in a stand-alone way, – that are loosely coupled among themselves through well-defined relationships, thus enabling the preservation of the semantic richness of the wider ontology, and – that are consequently reusable. Figure 13 shows an example of modular design where – the ontologies of the energy and electric vehicle ontologies are structured into sub-ontologies (not shared and shared), – a cross-domain ontology reuses the shared ontologies of each domain, and – the resulting semantic interoperability specification uses two existing modules and one new module.	
Manage resulting semantic interoperability work products	Evaluate maturity of ontologies	The practice of evaluating the maturity of a specification includes two dimensions: – ontology readiness which measures maturity of the knowledge specification; – specification readiness which measures the maturity of the interoperability specification. Figure 14 illustrates how the maturity of a specification can be evaluated. It will start with an ontology readiness and a specification readiness level that are low. As sub-ontologies are reused, the ontology readiness might move quickly to a higher level, while specification readiness could stay low for a longer time as it necessitates more multi-domain consensus discussions.	Description of method used for ontology maturity. Ontology maturity evolution history
	Document and maintain ontologies	The practice of using and maintaining interoperability specifications can require further needs: – Provision for profiles and discovery. Profiles are service descriptions. They allow for the support of specific optional features. For instance, a high-end device can support additional interoperability features which a low-end device would not offer. Service discovery is an interoperability mechanism that can be used to discover a profile. – Support for version management. Semantic interoperability specifications can include considerations on specification evolutions or enhancements, to match the needs for different generations of products (e.g. a new version of a consumer device). Rules for version management could be needed (e.g. upward compatibility). – Provision for ontology sharing and reuse. Semantic interoperability specifications can be developed more effectively through openly accessible documented ontologies. The concepts of profiles, service discovery and support for version management can be described through specific ontologies.	Description of documentation of maintenance approach. Repository of maintained ontologies

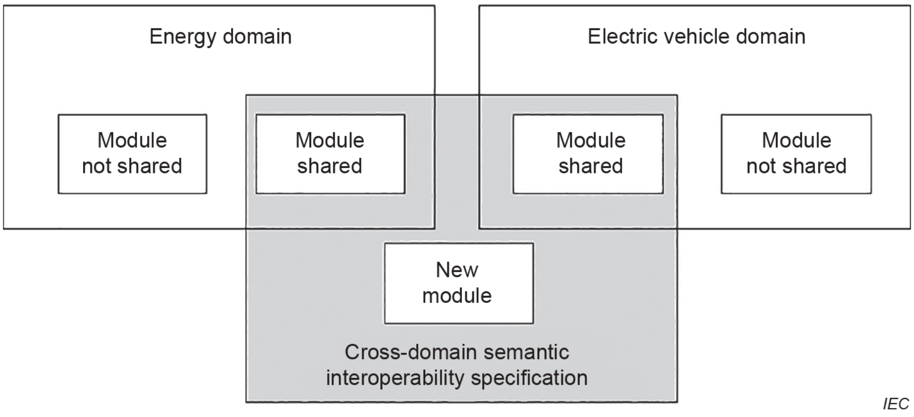


Figure 13 – Example of modular design

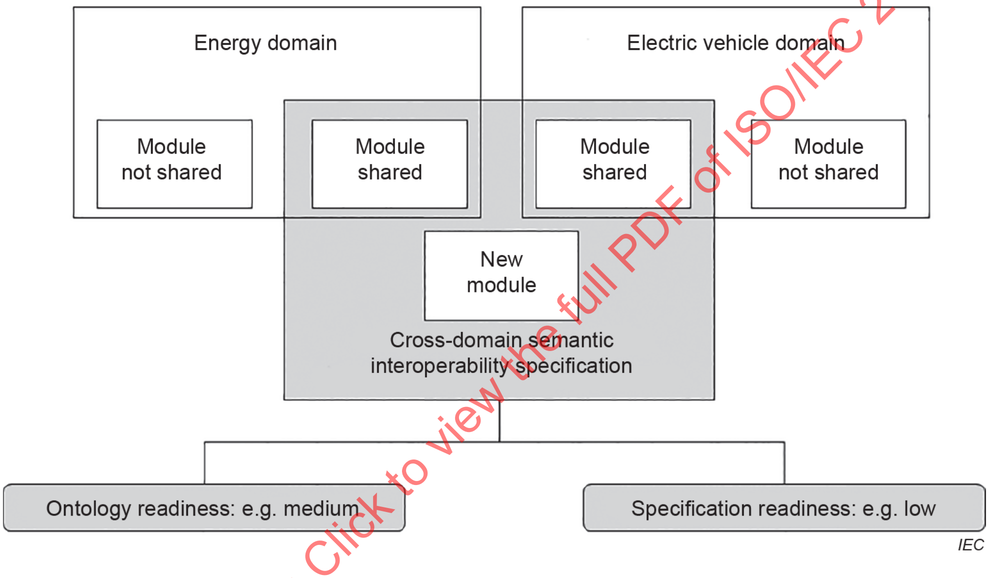

Energy domain
Electric vehicle domain
Module not shared
Module shared
Module shared
Module not shared
New module
Cross-domain semantic interoperability specification
Ontology readiness: e.g. medium
Specification readiness: e.g. low
IEC

Figure 14 – Example of interoperability maturity evaluation result

6.2 Life cycle model

Figure 15 describes the resulting life cycle model, integrating the ontology life cycle, showing the relationship between the life cycles. An example is the following.

- The ontology engineer develops an energy IoT architecture ontology, an energy management capability ontology and an energy information ontology.
- The IoT system engineer
  - uses the energy information ontology to develop an energy data exchange semantic interoperability specification, and
  - uses the energy data exchange semantic interoperability specification to develop an energy application which involves data exchange between components.

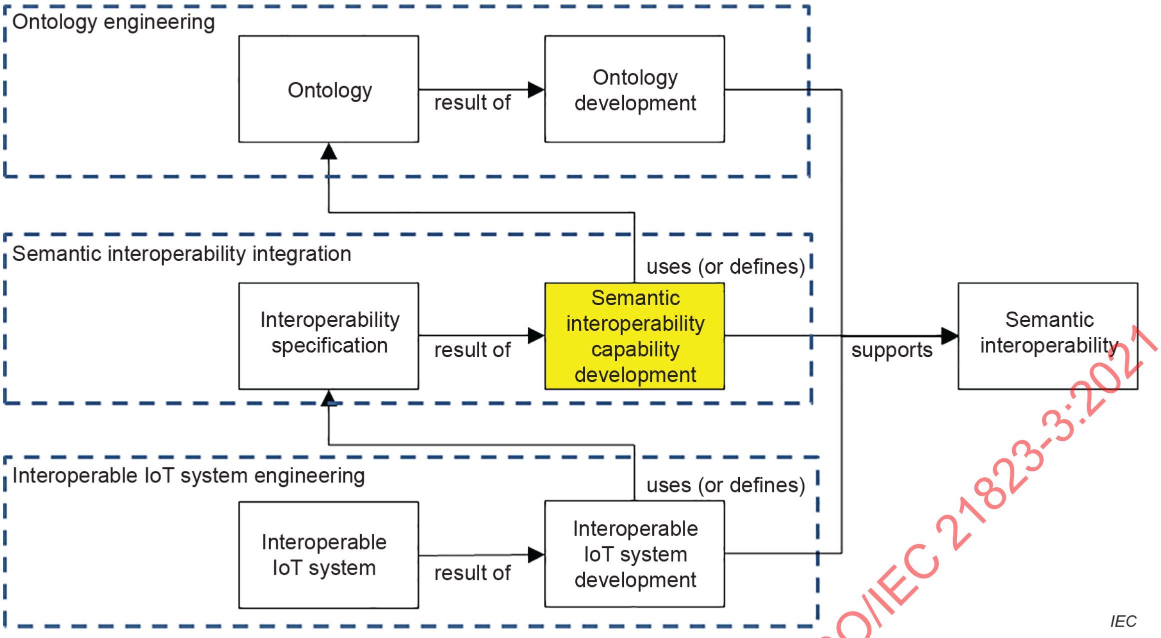

Ontology engineering
Ontology
result of
Ontology development
Semantic interoperability integration
uses (or defines)
Interoperability specification
result of
Semantic interoperability capability development
supports
Semantic interoperability
Interoperable IoT system engineering
uses (or defines)
Interoperable IoT system
result of
Interoperable IoT system development
IEC

Figure 15 – Semantic interoperability life cycle model

6.3 Life cycle implementation guidelines

6.3.1 Guidelines on ontology life cycle

Ontology engineering is a practice that has its own life cycle. Figure 16 shows an example of ontology life cycle consisting of the following phases:

- ontology requirements definition phase: define the requirements of the ontology to create;
- ontology co-creation phase: co-create the ontology, including for instance a domain specific expert and an ontology expert;
- ontology consistency validation phase: validation that an ontology is well-formed and consistent;
- ontology consensus phase: ensuring that domain experts agree with the concepts and relationships as captured by the ontology;
- ontology publication phase: publish, use and maintain the ontology.

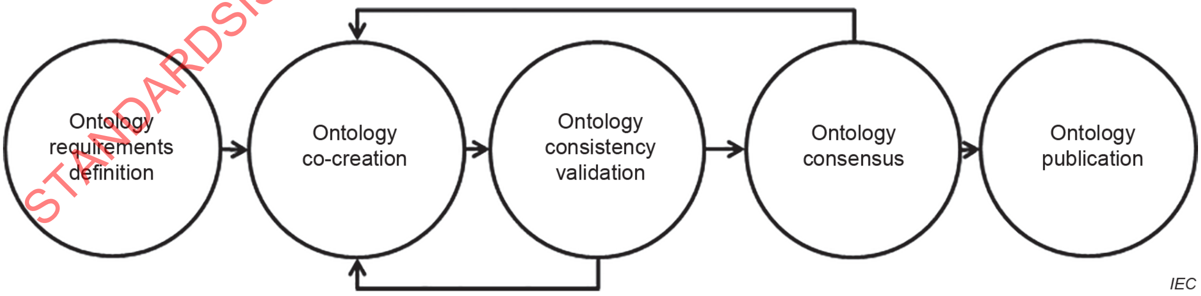

Ontology requirements definition
Ontology co-creation
Ontology consistency validation
Ontology consensus
Ontology publication
IEC

Figure 16 – Example of ontology life cycle model

6.3.2 Guidelines on semantic interoperability life cycle

Semantic interoperability engineering is a practice that focuses on integrating semantic descriptions in the life cycle processes of an interoperability specification. Figure 17 shows an example of an interoperability specification life cycle consisting of the following phases.

– Requirements phase: define the requirements of the interoperability specification.

– Specification phase: provide the specification.

– Consensus validation: reach consensus on the specification.

– Publication: publish the interoperability specification.

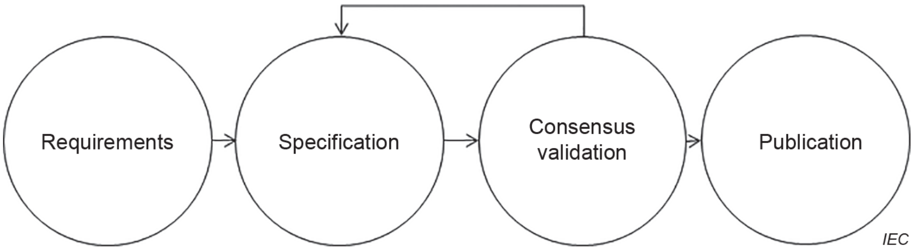


Figure 17 – Example of interoperability specification life cycle

The integration of semantic descriptions in an interoperability specification is based on ontologies:

a) a semantic description can use an existing ontology or create a new ontology; and

b) a semantic description can focus on different semantic items:

   1) information ontologies (e.g. the semantics of the information contained in a protocol data unit),

   2) interaction ontologies (e.g. the semantics of a protocol),

   3) capabilities ontologies (e.g. the semantic description of a profile),

   4) patterns ontologies (e.g. the semantic description of a publish-subscribe paradigm, the semantic description of a service discovery mechanism), or

   5) architecture ontologies (e.g. the semantic description of architecture features).

The difference between an ontology life cycle and an interoperability specification life cycle can be explained in the following example: an interoperability specification for a health sensor that uses an architecture, a protocol and a health ontology involves the following life cycles:

– the interoperability specification life cycle,

– the architecture ontology life cycle,

– the protocol ontology life cycle, and

– the health ontology life cycle.

6.3.3 Guidelines on IoT system life cycle

Interoperable IoT system engineering is a practice that focuses on integrating interoperability concerns in the life cycle processes of an IoT system.

ISO/IEC/IEEE 15288:2015 [22] defines a system life cycle as "an abstract functional model representing the conceptualization of a need for the system, its realization, utilization, evolution and disposal". A system life cycle is described as a set of processes. A process is described according to its purpose, the outcome it creates, and its activities. ISO/IEC/IEEE 15288:2015 describes thirty processes structured into four categories:

a) agreement processes which focus on activities related to supplier agreements,

b) organizational project-enabling processes which focus on activities related to improvement of the organization's business or undertaking,

c) technical management processes which focus on managing the resources and assets allocated to the engineering of a system, and

d) technical processes which focus on technical actions throughout the life cycle.

The integration of interoperability concerns depends on the context:

1) an IoT system can be based on an existing interoperability specification or a new interoperability specification that has been created for it;
2) the specification can be at a syntactic level or at a semantic level; and
3) supporting elements such as reference implementations or test tools are available or not.

The difference between an interoperability specification life cycle and an IoT system life cycle can be explained in the following example. An IoT system which remotely controls home appliances involves the following life cycles:

- the remote control IoT system life cycle, and
- the home appliance interoperability specification life cycle.

Annex A
(informative)

Guidance on how to learn IoT semantic interoperability

IoT semantic interoperability requires specific competence that is not straightforward to acquire. Table A.1 provides an example of a syllabus that can be used by instructors to teach IoT semantic interoperability to IoT system engineers.

Table A.1 – Syllabus example on IoT semantic interoperability practice

Aim	This course provides an understanding on how ontologies can be used to achieve semantic interoperability for IoT systems. Topics to be covered include ontology engineering (principles, ontology representations, ontology usage) and reasoning capabilities.
Learning outcomes	Upon completing the course, the IoT system engineer should be able – to represent IoT system concepts using ontologies (e.g. sensors), – to represent IoT system interoperability concepts using ontologies (e.g. temperature metadata), – to develop IoT system interoperability capabilities using ontologies, including reasoning capabilities (e.g. when temperature exceeds a threshold, air conditioning is activated), and – to integrate the resulting work products in the IoT system life cycle processes (e.g. suitable documentation is generated on the capability to activate air conditioning).
Format	Two learning formats are provided: – IoT engineers develop IoT systems with the support of ontology engineers, and – IoT engineers gain competence on ontology engineering.
Prerequisite	The IoT system engineer should be familiar with IoT system engineering. For instance, the engineer could be familiar with engineering practices related to the use of ISO/IEC 20924 [23], of ISO/IEC 30141 [1] and of ISO/IEC/IEEE 15288 [22].

Table A.2 provides an example of course content based on the syllabus.

Table A.2 – Course content for semantic interoperability practice

Sessions	Description	
IoT semantic interoperability objectives	Content	Explain IoT semantic interoperability objectives: – capture semantic meaning, – integrate semantic capabilities, and – practice semantic interoperability engineering.
	Example of teaching material	This document (Clause 5). White papers and recommendation documents [2], [24], [25], [26].
Use case and course format selection	Content	Select an IoT system use case in which an interoperability capability is be developed (e.g. a temperature sensor triggers the activation of the heating system). Select the course format approach: – co-development practice, i.e. the IoT system engineer works with an ontology engineer; – development practice, i.e. the IoT system engineer develops an ontology.
	Example of teaching material	Material from the IoT system engineer domain, or [27].
The first ontology	Content	This session is manual. The IoT system engineer is invited to specify the concepts and relations of his or her use case.
	Example of teaching material	Material from the IoT system engineer domain.
The first represented ontology	Content	Ontologies representation is explained (e.g. triples). In the case of co-development practice, the instructor presents the ontology specification environment and develops the representation of the ontology. In the case of development practice, the ontology specification environment is explained, the IoT system engineer gets familiar with its use and develops the representation of the ontology with the support of the instructor.
	Example of teaching material	Representation schemes such as RDF [10] and RDFS [11], and OWL [12]. 101 ontology development [28]. Using the WebProtégé software [29].
The first rule	Content	This session is manual. The IoT system engineer is invited to specify the rules that he or she wants to implement (e.g. if the temperature is below 10 °C, then the heating system is activated).
	Example of teaching material	Material from the IoT system engineer domain.
The first represented rule	Content	Rules representations are explained (e.g. rules, inference API). In the case of co-development practice, the instructor presents the rule-based development environment and implements the representation of the rule. In the case of development practice, the rule-based development environment is explained, the IoT system engineer gets familiar with its use and develops the representation of the rule with the support of the instructor.
	Example of teaching material	Development frameworks such as Apache Jena™ [30].[a]

Sessions	Description	
The first semantic action	Content	This session is manual: the IoT system engineer implements the rule using its standard development practice (i.e. the rule is implemented by hardwired code).
	Example of teaching material	Information on the development environment used by the IoT system engineer.
The first implemented semantic action	Content	Reasoning capability is explained, i.e. querying the ontology representation (e.g. using SPARQL). In the case of co-development practice, the instructor presents the reasoning capability development environment and implements the rule based on the reasoning capability. In the case of development practice, the reasoning capability development environment is explained, the IoT system engineer gets familiar with its use and implements the semantic action associated with the rule with the support of the instructor.
	Example of teaching material	Apache Jena inference engine [31].
The first semantic annotation for interoperability	Content	The need for semantic annotation to ensure interoperability is explained. This session is manual: the IoT system engineer specifies the metadata associated with physical data provided by a concrete IoT device (e.g. external temperature, unit in Celsius).
	Example of teaching material	Information on typical IoT devices used by IoT system engineer.
The first implementation of semantic annotation for interoperability	Content	The need to create dynamic mapping capabilities between the ontology and data produced by heterogeneous devices is explained. In the case of co-development practice, the instructor presents the mapping programming interface in the reasoning capability development environment and implements the semantic annotation. In the case of development practice, the mapping interface is explained, the IoT system engineer gets familiar with its use and implements the semantic annotation with the support of the instructor.
	Example of teaching material	Documentation on semantic annotation work products used in the development framework (such as Jena). Tutorials on semantic annotation [32].
Integration into the IoT system life cycle processes	Content	Methodologies for ontology development are explained, and their integration in the IoT system life cycle processes is explained. The relation between the ontology life cycle, the interoperability life cycle, and the IoT system life cycle is explained. The work products resulting from IoT semantic interoperability practice are presented (e.g. ontologies, interoperability specification, interoperability tests).
	Example of teaching material	101 ontology development methodology [29], NeOn [33]. See [34] for ontology methodology mindmap. Concerning life cycle matters, this document (Clause 6, Annex B, Annex C) and ISO/IEC/IEEE 15288 [22].

[a] Apache Jena™ is the trade name of a product supplied by the Apache Software Foundation. This information is given for the convenience of users of this document and does not constitute an endorsement by IEC or ISO of the product named.

Annex B
(informative)

Guidance on how to develop IoT semantic interoperability

B.1 Developing semantic interoperability capabilities

If the necessary semantic interoperability capabilities for a particular IoT system do not exist, they need to be developed. This development is driven by the IoT system engineer desiring to achieve the interoperability and is often done in collaboration with an ontology engineer and/or by using the ontology engineering support tools.

When developing interoperability capabilities, the IoT system engineer can be in different positions depending on whether he or she works on the IoT device supplier side (as IoT device developer) or as an IoT system integrator for an organization that wants to build and deploy the IoT system by using off-the-shelf IoT devices. Both have an interest to create interoperable systems.

The IoT system engineer working as an IoT device developer has more freedom to affect the way the device works. Thus, it is possible to change the properties of the device to match a public ontology, or to develop a vendor-specific ontology (or the missing pieces of the ontology) to match the capabilities of the device.

The IoT system engineer working as a system integrator is mostly concerned with selecting the right kind of devices for the system he or she is building. The devices will often come from multiple vendors, and the IoT engineer relies on the semantic interoperability capabilities to provide interoperability.

If the semantic interoperability capabilities (e.g. ontology) are provided by the IoT device supplier, they can be used by all the different IoT system integrators (customers) that will use those devices. However, if the semantic interoperability capabilities for a particular IoT device are not available from the IoT device supplier, the IoT system integrator could still want to use the ontology approach to achieve semantic interoperability by collaborating with an ontology engineer to develop the missing ontology (or missing ontological entries). In this scenario, the work is carried out separately by each IoT system integrator unless the developed ontological entries are contributed to a public ontology.

B.2 Building steps

Table B.1 provides an example which can be used by the primary stakeholder (IoT system engineer) to build semantic interoperability. It can also apply to secondary stakeholders; IoT system integrators, IoT device developers, system integrators.

The steps in Table B.1 can be viewed as a guide on how to use this document when developing IoT semantic interoperability.

Table B.1 – Building steps for IoT semantic interoperability

Steps	Description
Step 1	What is the purpose and requirement for this interoperability?
Step 2	Who are the parties involved (e.g. organizations)? Which systems are involved (e.g. IT endpoints, IoT endpoints)? Which mechanisms (e.g. certificates and PKI) are used to identify and authenticate the digital parties?
Step 3	Which information is to be transferred (e.g. format, frequency, quality, identity)?
Step 4	Should semantic interoperability be applied? And what level of semantic information is available from the other party?
Step 5	If the answer in step 4 is Yes, then find out the semantic interoperability and connectivity parameters (e.g. ontology based on OneM2M). Then develop the capability to be added to the sending party to test connectivity with the other party.
Step 6	Start testing. The first phase is to check that the two parties agree on the right identity (e.g. using a PKI).
Step 7	The second phase of testing is to check that the two parties can exchange messages. For instance, send test messages with large variance on contents so that the two parties analyse and run quality checks. The results need to be verified by the two parties to ensure interoperability.
Step 8	When the test procedures have run successfully, the two parties agree that the semantic interoperability is assured and interact using the ontology.

Annex C
(informative)

Guidance on how to manage IoT semantic interoperability life cycle

C.1 Interoperability specification life cycle that supports ontologies

Table C.1 describes an interoperability specification life cycle that is ontology-based.

Table C.1 – Example of interoperability specification life cycle

Interoperability specification life cycle	Description	
Requirements	Identify ontologies to be created or used	– If needed, get familiar with the creation and use of ontologies, using tutorials such as [28], [29]. – Identify the type of knowledge that needs to be captured in the ontology (domain, cross-domain and transversal, e.g. health, transport and security). – Identify whether existing ontologies can be reused or extended. Catalogues such as LOV4IoT [9] can be used. – Define the semantic descriptions that will use ontologies: information, interaction, capabilities, patterns, architectures, etc. – Define the operational requirements (e.g. compatibility) – Identify an ontology version management scheme.
Specification	Co-create ontologies	– Set up a collaboration scheme between domain and ontology experts to create the structure and content of ontologies. Guidance is provided in [25]. Best practices are provided in [35] – Validate that ontology is semantically consistent. – Validate level of modularity. – Establish a consensus process between domain and ontology experts.
	Specify the relationship between interoperability specification and ontologies	– List concepts and relations that need to be supported by the ontologies.
	Validate interoperability specification with the help of ontologies	– List dependencies that can be verified. – Identify consistency constraints expressed in ontologies. – Verify constraints in the interoperability specification.
Consensus validation	Semantic interoperability commissioning and deployment integrating associated ontologies	– Update ontologies and specification if needed. – Evaluate maturity by ecosystem (e.g. community that will use the ontology). – Integrate in the ontology version management.
Publication	Semantic interoperability ontology maintenance	– Publish the created or extended ontology in existing catalogues – Update and enhance the exposed ontology. – Validate the updated ontology.
	Semantic interoperability ontology decommissioning	– Update the ontology version management.

C.2 IoT system life cycle supporting interoperability

Table C.2 describes an IoT system life cycle that supports interoperability.

Table C.2 – Example of IoT system life cycle

IoT system life cycle	Interoperability integration activities
Stakeholder needs and requirements definition	Identifying the interoperability specification to be used or created
System requirements definition process	Integrating the requirements of the interoperability specification in the system requirements
Architecture definition process	No specific activity
Design definition process	Design the features needed to meet the requirements of the interoperability specification
System analysis process	No specific activity
Implementation process	Implementing the features needed to meet the requirements of the interoperability specification. If needed, develop the interoperability test capabilities.
Integration process	No specific activity
Verification process	Testing the interoperability of the IoT system (in-house)
Transition process	Testing the interoperability of the IoT system (third party, plug test)
Validation process	Deployment validation
Operation process	No specific activity
Maintenance process	Integrating modifications of the interoperability specification
Disposal process	No specific activity

Annex D
(informative)

Ontological specification of the IoT Reference Architecture

D.1 General

ISO/IEC 30141 [1] specifies a general IoT Reference Architecture in terms of defining system characteristics, and architecture views for IoT. Annex D provides an ontological specification of the reference architecture. The ontological specifications are authored in the Web Ontology Language (OWL) [12] and depicted in the Visual Notation for OWL Ontologies (VOWL) [36].

An ISO/IEC 30141 ontology is needed when IoT systems exchange information that includes references to IoT architecture artefacts, e.g. during sensor and application discovery phases. The fact that ISO/IEC 30141 is a standard ensures that the ontology uses standardized concepts. This is important because different ontologies can use concepts with the same name. For instance, the SSN ontology [16] defines the concept of "device". Likewise, the OWL-S ontology [20] defines the "service" concept. If these ontologies are used at the same time as the IoT RA ontology, then it should be determined whether these concepts have the same meaning.

D.2 Service, network, IoT device and IoT gateway

Service is an abstract concept. A service is implemented by one or more components. There could be multiple alternative implementations of the same service. Entities which interact via networks do so by exposing one or more endpoints on a network. A network connects endpoints. A service exposes one or more endpoints by which it can be invoked. An endpoint has one or more network interfaces. Services, which are located remotely, can be reached by endpoints through network interfaces across a communication network. Endpoints exist on one or more networks. Data associated with services, with IoT devices and with IoT gateways can be held in a data store used by one or more entities. Figure D.1 and Figure D.2 describe the overall IoT entities related with service, network, IoT device and IoT gateway.

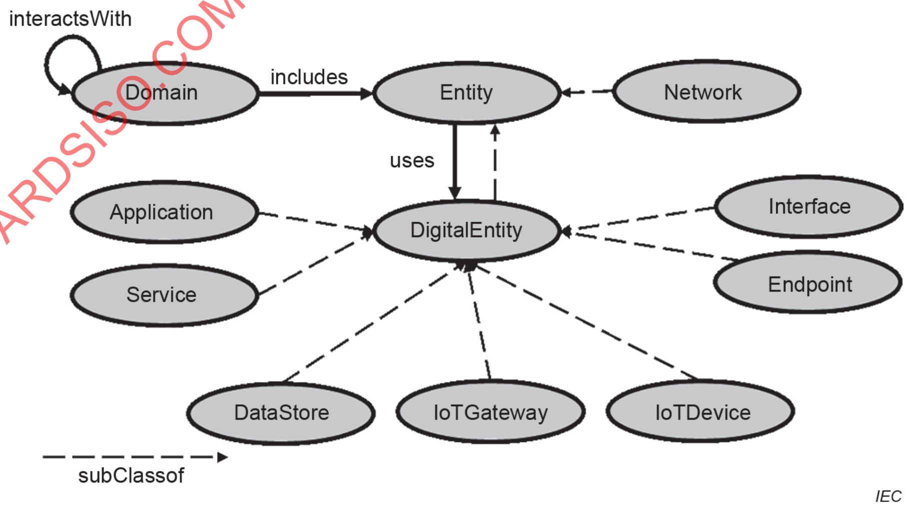


Figure D.1 – IoT entity

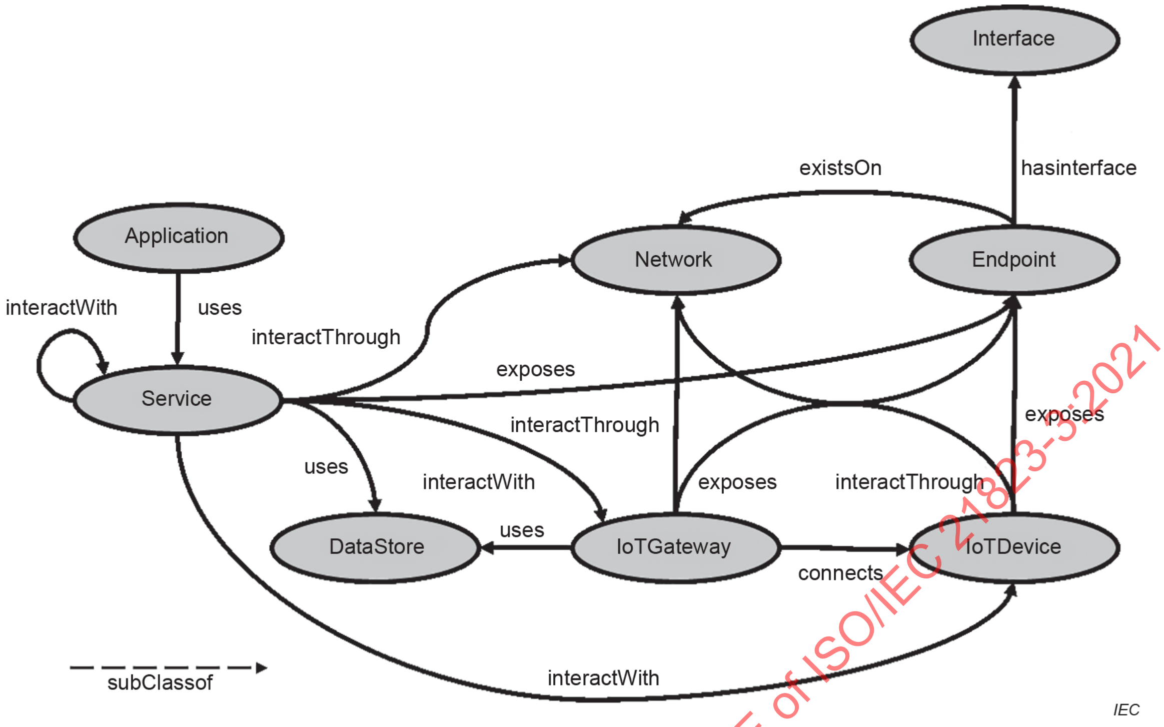


Figure D.2 – Service, network, IoT device and IoT gateway

D.3 IoT-User

An IoT-User can be either human (human user) or digital (digital user). A digital user includes automation services that act on behalf of human users, for example in machine-to-machine interactions. A digital user interacts with one or more services directly or indirectly through the service endpoint. A human user interacts through one or more applications. Figure D.3 depicts IoT-User as the super class of human user and digital user.

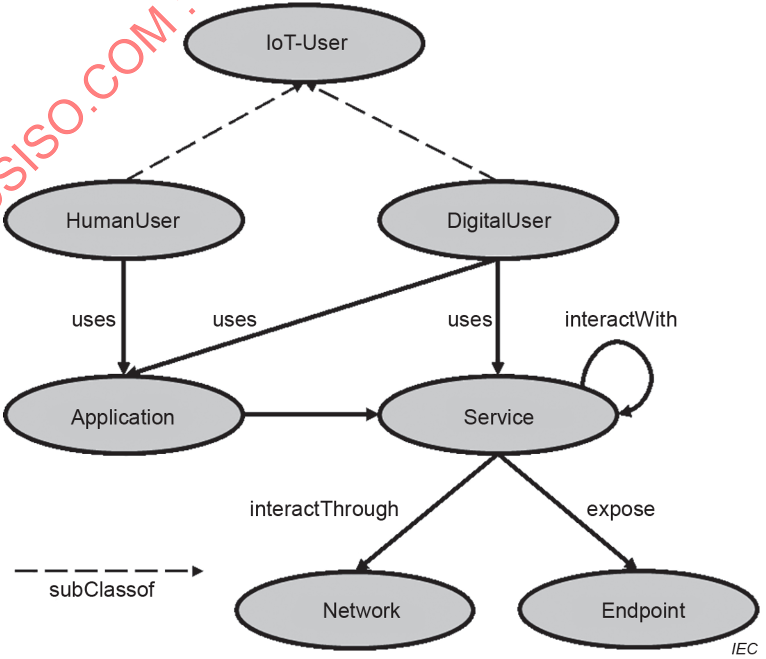


Figure D.3 – IoT-User

D.4 Virtual entity, physical entity and IoT device

Actuator and sensor are IoT devices which have direct or indirect contact with a physical entity. An actuator operates on received digital information to act on (change) some property of a physical entity. A sensor perceives certain characteristics of a physical entity and transforms them into a digital representation which can be communicated. A physical entity can have one or more tags attached to it and sensors can monitor the tag rather than the physical entity itself. Actuator and sensor are two kinds of IoT device, which converts variations in one physical quantity, quantitatively into variations in another. A single IoT device can hold multiple sensors in a single device. Figure D.4 illustrates the relations between virtual entity, physical entity and IoT device.

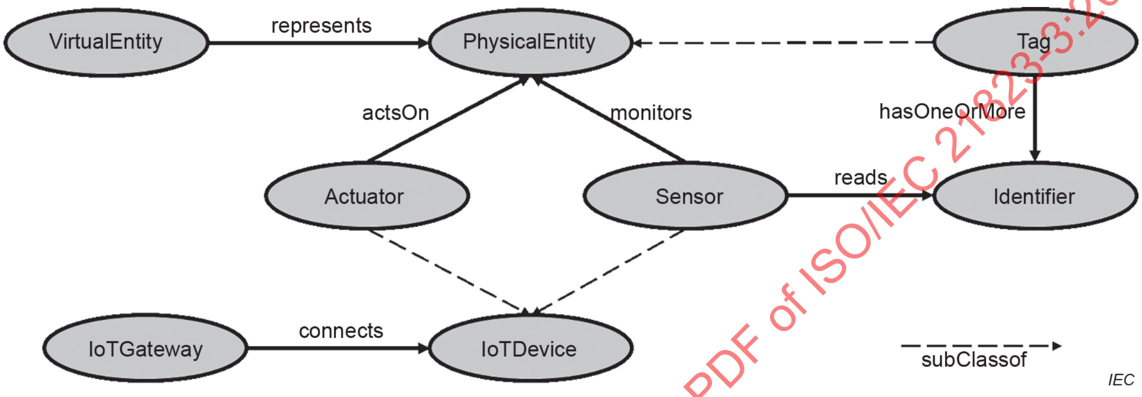


Figure D.4 – Virtual entity, physical entity, and IoT device

D.5 Domain-based Reference Model (RM)

The domains help the designer to focus on the various tasks that need to be performed, by allowing a logical (and sometimes physical) subdivision. Mainly, domains are used to sort functions in areas of responsibility; these areas (functional areas) are then typically deployed in separate sub-systems. The identified domains are: User Domain (UD), Operations & Management Domain (OMD), Application & Service Domain (ASD), Resource Access & Interchange Domain (RAID), Sensing & Controlling Domain (SCD), and Physical Entity Domain (PED). Each identified domain is mutually exclusive from all other domains. The relationship between entity-based RM and domain-based RM is given in Figure D.5. From different views, each domain has different kinds of entities.

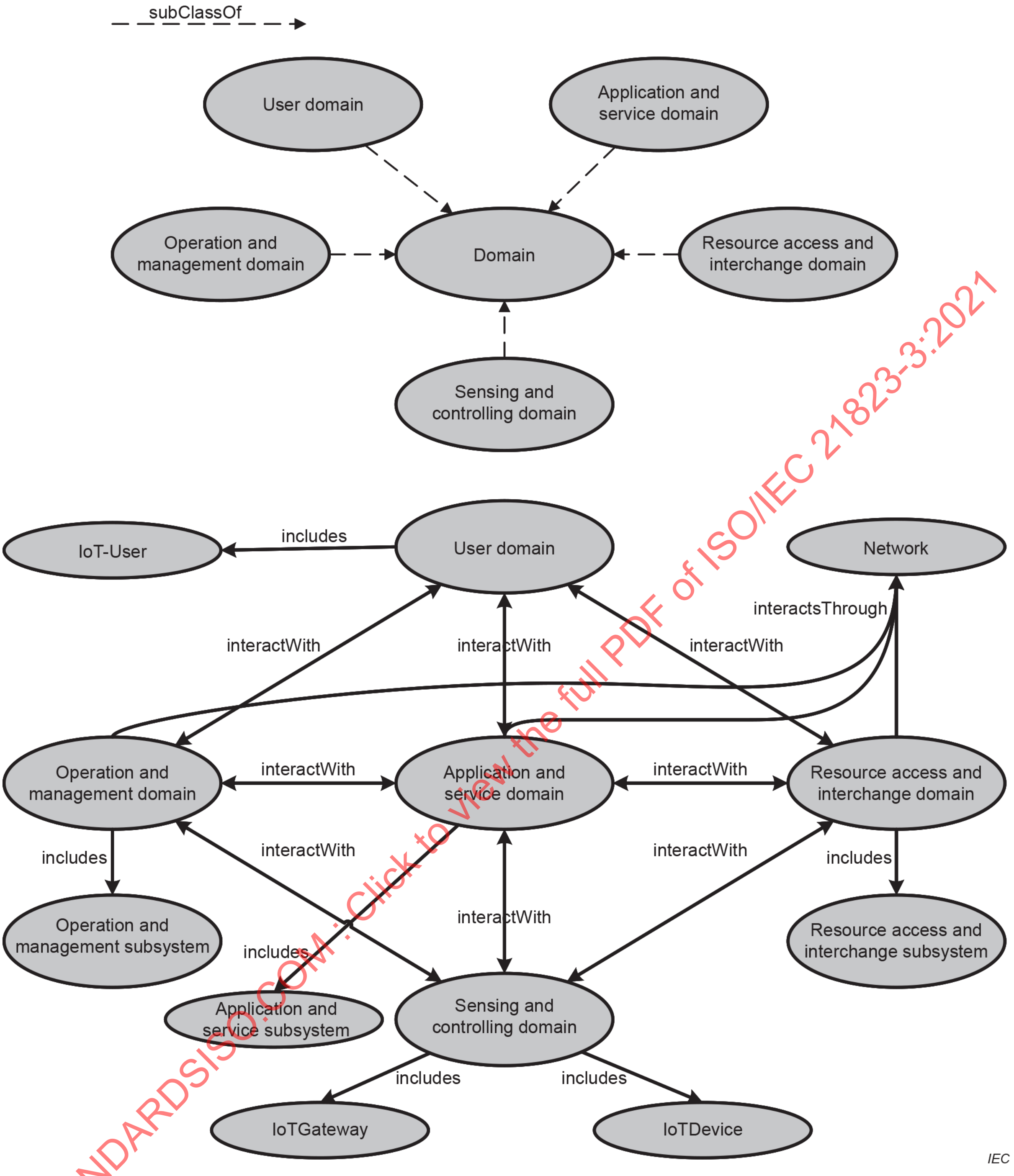

subClassOf
User domain
Application and service domain
Operation and management domain
Domain
Resource access and interchange domain
Sensing and controlling domain
IoT-User
includes
User domain
Network
interactsThrough
interactWith
interactWith
interactWith
Operation and management domain
interactWith
Application and service domain
interactWith
Resource access and interchange domain
includes
interactWith
interactWith
includes
interactWith
Operation and management subsystem
includes
Resource access and interchange subsystem
Application and service subsystem
Sensing and controlling domain
includes
includes
IoTGateway
IoTDevice
IEC

Figure D.5 – Domain-based Reference Model

Annex E
(informative)

Related existing ontologies

E.1 W3C Semantic Sensor Network ontology

The World Wide Web Consortium (W3C) Spatial Data on the Web Working Group (WG) developed the SSN ontology [16], which provides a high-level schema to describe sensors and their observations, the involved procedures, the studied features of interest, the samples used to do so, and the observed properties, as well as actuators. SSN follows a horizontal and vertical modularization architecture by including a lightweight but self-contained core ontology called SOSA (Sensor, Observation, Sample, and Actuator) for its elementary classes and properties.

The SSN ontology is based around concepts of systems, processes, and observations. It supports the description of the physical and processing structure of sensors. Sensors are not constrained to physical sensing devices: rather a sensor is anything that can estimate or calculate the value of a phenomenon, so a device or computational process or combination could play the role of a sensor. The representation of a sensor in the ontology links together what it measures (the domain phenomena), the physical sensor (the device) and its functions and processing (the models).

The SSN ontology revolves around the central Stimulus-Sensor-Observation pattern. Here, stimuli are detectable changes in the physical world. Sensors are physical objects that perform observations. In addition, observations act as the nexus between incoming stimuli, the sensor, and the output of the sensor.

The SSN ontology has several conceptual modules to build on the pattern that is to cover key sensor concepts.

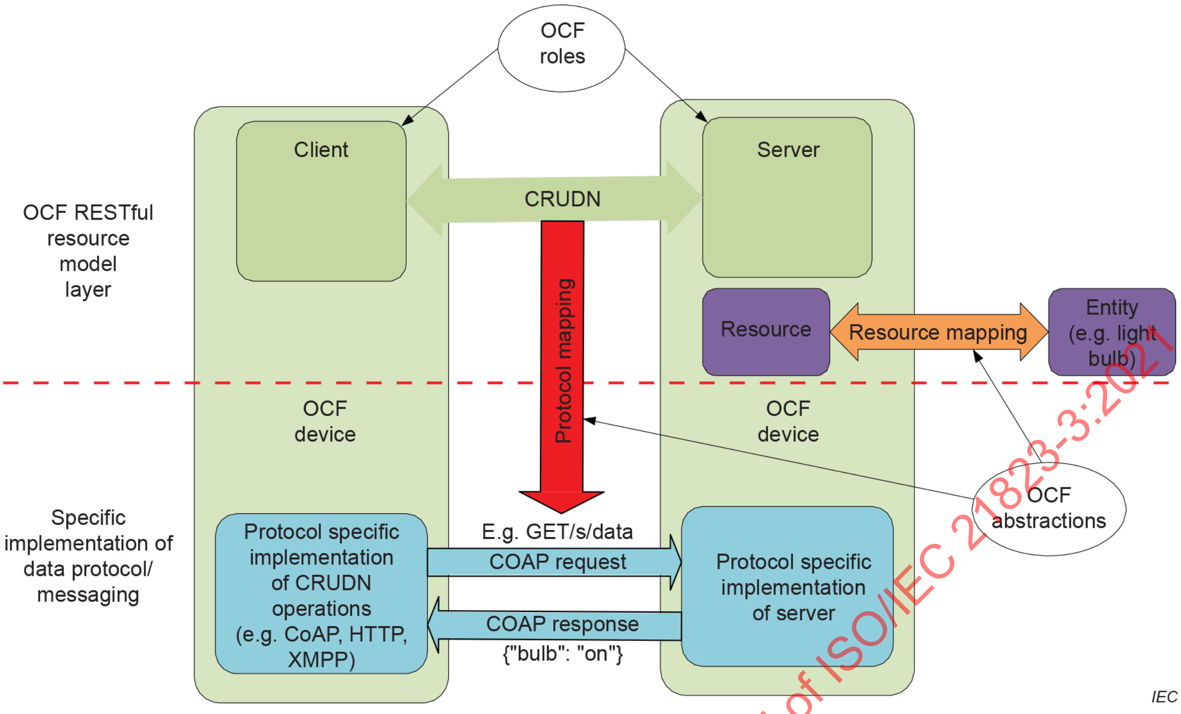
E.2 IoT-Lite

The IoT-Lite ontology [37] allows representing Internet of Things (IoT) resources, entities and services. In addition, the IoT-Lite ontology, which is a lightweight instantiation of the SSN ontology, can describe the key IoT concepts that allow interoperability and discovery of sensory data in heterogeneous IoT platforms.

The IoT-Lite ontology describes IoT concepts in three classes: objects, system or resources, and services. IoT devices are classified into, although not restricted to, three classes: sensing devices, actuating devices and tag devices. The IoT-Lite ontology is focused on sensing, although it has a high-level concept on actuation that allows any future extension on this area. Services are described with a coverage. This coverage represents the 2D-spatial covered by the IoT device [37].

E.3 Open Connectivity Foundation (OCF) ontology

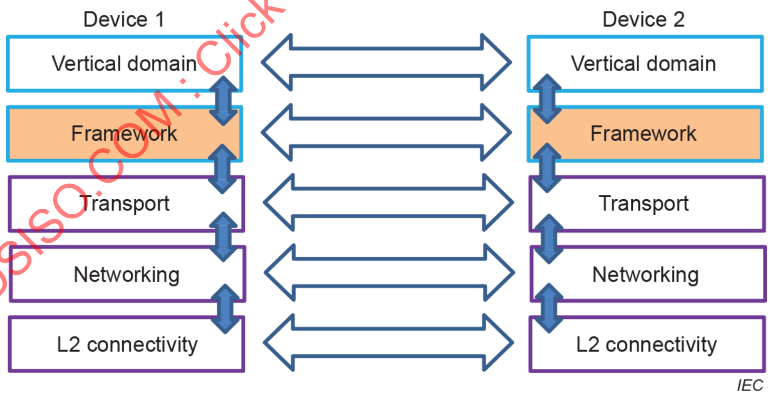
The OCF architecture [38] describes a client server architecture as described in Figure E.1. The server side of the architecture is made up of physical sensors and actuators. OCF is built on RESTful concepts, i.e. it is driven using a small set of generic operations: CREATE, RETRIEVE, UPDATE, DELETE and NOTIFY (CRUDN).


OCF roles
OCF RESTful resource model layer
Client
Server
CRUDN
Protocol mapping
Resource
Resource mapping
Entity (e.g. light bulb)
OCF device
OCF device
OCF abstractions
Specific implementation of data protocol/ messaging
Protocol specific implementation of CRUDN operations (e.g. CoAP, HTTP, XMPP)
E.g. GET/s/data
COAP request
COAP response
{"bulb": "on"}
Protocol specific implementation of server
IEC

SOURCE: ISO/IEC 30118-1.

Figure E.1 – Architecture – concepts

The architecture is organized conceptually into three major aspects that provide overall separation of concern: resource model, RESTful operations, and abstractions. When two Devices communicate with each other, each functional block in a Device interacts with its counterpart in the peer Device as shown in Figure E.2.



SOURCE: ISO/IEC 30118-1.

Figure E.2 – Communication layering model

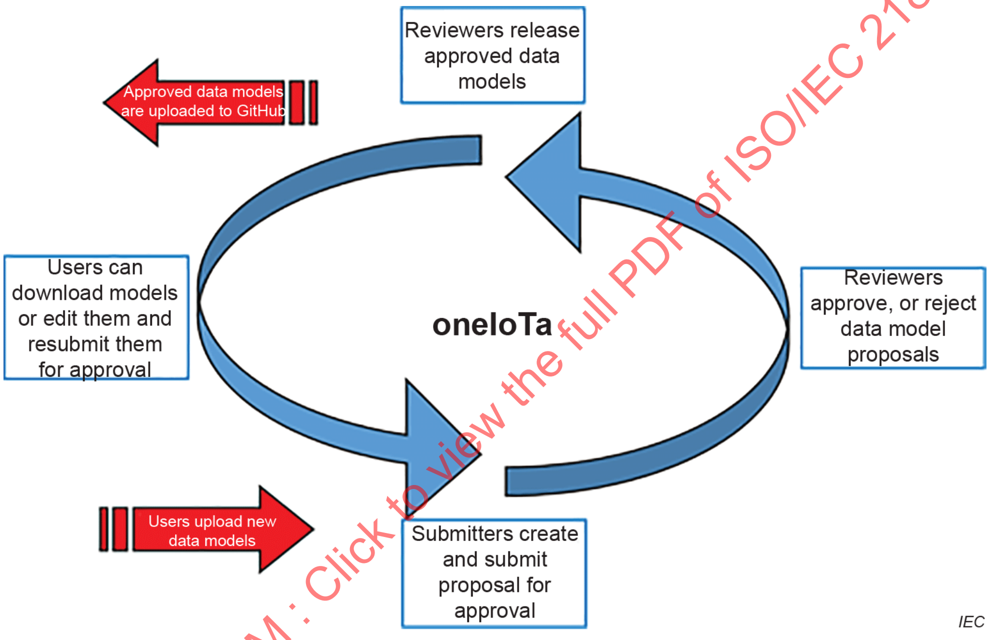
The vertical domains are described by the resource models. The resource models are described by RESTful API design language OpenAPI 2.0 (swagger). The resource modelling is built on the architectural concepts that an OCF Device hosts resources. Each OCF Device has a device type (e.g. Refrigerator, oic.d.refrigerator). Associated with each device type is a minimal mandated set of resources to be implemented by that device type, additional resources may optionally be implemented. The following different types of resources exist:

- resources needed to describe the OCF architecture, including resources needed for security;
- resources describing sensors;

- – resources describing actuators;
- – resources describing configuration aspects of sensors/actuators;
- – resources describing service enabling concepts such as Collections, Scenes, and Rules.

The resource descriptions conveying values are described in a manner that is agnostic of whether the resource models a sensor or actuator. The resources are a virtual description of what is physically implemented on the device. In this sense the architecture follows Figure E.1.

The separation of the resource modelling and the OCF framework also means that the development of the resource models is decoupled from the architecture. To aid the development of the data models, the Open Connectivity Foundation created a tool called oneIoTa (www.oneIOTa.org) as shown in Figure E.3. The oneIoTa tool is an integrated development environment (IDE) that supports the development of data models and has an integrated process for acceptance and publication of the data models.


Reviewers release approved data models
Approved data models are uploaded to GitHub
Users can download models or edit them and resubmit them for approval
oneIoTa
Reviewers approve, or reject data model proposals
Users upload new data models
Submitters create and submit proposal for approval
IEC

SOURCE: oneIoTa User Guide [39] . Reproduced with permission.

Figure E.3 – oneIoTa

The oneIoTa tool also supports derived models. The Derived modelling is a mechanism to describe translation between OCF models and models from other ecosystems. An example of such a derived model is the translation between OCF and Zigbee™[2] models.

Figure E.4 describes the OCF ontology. All OCF Servers and Clients are instantiations of an `OCFDevice`. The ontology additionally describes both simple and complex resources, i.e. those including instantiations of `CollectionInterface` and `OCFLink`. The CRUDN operations are represented in the Method definition which is exposed by `OCFResource` and defined by `OCFInterfaceType`. The representation of devices is provided by `OCFProperty` values.

[2] Zigbee™ is is the trade name of a product supplied by the Zigbee Alliance. This information is given for the convenience of users of this document and does not constitute an endorsement by IEC or ISO of the product named.