ETSI TS 103 264 V4.1.1 (2025-03)

SmartM2M;
Smart Applications;
Reference Ontology and oneM2M Mapping

Reference
RTS/SmartM2M-103264v411

Keywords
data, IoT, M2M, oneM2M, ontology, SAREF, semantic

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° w061004871

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI IPR online database.

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™**, **LTE™** and **5G™** logo are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM®** and the GSM logo are trademarks registered and owned by the GSM Association.

BLUETOOTH® is a trademark registered and owned by Bluetooth SIG, Inc.

Foreword

This Technical Specification (TS) has been produced by ETSI Technical Committee Smart Machine-to-Machine communications (SmartM2M).

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

1 Scope

The present document provides a standardized framework for the Smart Applications REFerence ontology based on the results of a European Commission Study Group on Smart Appliances ontologies and of different Specialist Task Forces that have supported the maintenance and evolution of the ontology taking into account all the interest of the relevant stakeholders. This reference ontology contains concepts and patterns that can be specialized in several domains and is a basis for extensions in particular domains.

The present document also defines the equivalent mapping between the Smart Applications REFerence Ontology and the oneM2M Base Ontology.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found in the ETSI docbox.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are necessary for the application of the present document.

[1] ETSI EN 303 760: "SmartM2M; SAREF Guidelines for IoT Semantic Interoperability; Develop, apply and evolve Smart Applications ontologies".

[2] ETSI TS 118 112: "oneM2M; Base Ontology (oneM2M TS-0012)".

[3] ETSI TS 103 267: "SmartM2M; Smart Applications; Communication Framework".

[4] ETSI TS 103 673: "SmartM2M; SAREF Development Framework and Workflow, Streamlining the Development of SAREF and its Extensions".

[5] ETSI TS 103 548: "SmartM2M; SAREF reference ontology patterns".

[6] W3C® Recommendation 18 August 2009: "SKOS Simple Knowledge Organization System Reference".

[7] W3C® Recommendation 19 October 2017:"Semantic Sensor Network Ontology", OGC® and W3C® Spatial Data on the Web working Group.

[8] W3C® Candidate Recommendation Draft 15 November 2022: "Time Ontology in OWL", OGC® and W3C® Spatial Data on the Web working Group.

[9] OGC® IS 22-047r1 (V1.1): "GeoSPARQL - A Geographic Query Language for RDF Data".

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] European Commission and TNO: "Study on Semantic Assets for Smart Appliances Interoperability", final report, April 2015.

[i.2] ETSI SAREF: "The Smart Applications REFerence ontology (SAREF)".

[i.3] ETSI TR 103 411: "SmartM2M Smart Appliances SAREF Extension Investigation".

[i.4] European Commission and TNO D-S4 - SMART 2013-0077: "Smart Appliances - Mapping SAREF to short list assets.xlsx", February 2015.

[i.5] ETSI TS 103 264 (V1.1.1): "SmartM2M; Smart Appliances; Reference Ontology and oneM2M Mapping".

[i.6] ETSI TS 103 264 (V2.1.1): "SmartM2M; Smart Appliances; Reference Ontology and oneM2M Mapping".

[i.7] ETSI TS 103 264 (V3.1.1): "SmartM2M; Smart Applications; Reference Ontology and oneM2M Mapping".

[i.8] ETSI TS 103 264 (V3.2.1): "SmartM2M; Smart Applications; Reference Ontology and oneM2M Mapping".

[i.9] ETSI TS 103 410-1: "SmartM2M; Extension to SAREF; Part 1: Energy Domain".

[i.10] ETSI TS 103 410-2: "SmartM2M; Extension to SAREF; Part 2: Environment Domain".

[i.11] ETSI TS 103 410-3: "SmartM2M; Extension to SAREF; Part 3: Building Domain".

[i.12] ETSI TS 103 410-4: "SmartM2M; Extension to SAREF; Part 4: Smart Cities Domain".

[i.13] ETSI TS 103 410-5: "SmartM2M; Extension to SAREF; Part 5: Industry and Manufacturing Domains".

[i.14] ETSI TS 103 410-6: "SmartM2M; Extension to SAREF; Part 6: Smart Agriculture and Food Chain Domain".

[i.15] ETSI TS 103 410-7: "SmartM2M; Extension to SAREF; Part 7: Automotive Domain ".

[i.16] ETSI TS 103 410-8: "SmartM2M; Extension to SAREF; Part 8: eHealth/Ageing-well Domain".

[i.17] ETSI TS 103 410-9: "SmartM2M; Extension to SAREF; Part 9: Wearables Domain".

[i.18] ETSI TS 103 410-10: "SmartM2M; Extension to SAREF; Part 10: Water Domain".

[i.19] ETSI TS 103 410-11: "SmartM2M; Extension to SAREF; Part 11: Lift Domain".

[i.20] ETSI TS 103 410-12: "SmartM2M; Extension to SAREF; Part 12: Smart Grid Domain".

[i.21] ETSI TR 103 549: "SmartM2M; Guidelines for consolidating SAREF with new reference ontology patterns, based on the experience from the ITEA SEAS project".

[i.22] ETSI TR 103 781: "SmartM2M; Study for SAREF ontology patterns and usage guidelines".

[i.23] ISDA: "ISDA Taxonomy 2.0".

[i.24] European Environment Agency: "Vocabulary: Standard International Energy Product Classification (SIEC)".

[i.25] International Renewable Energy Agency: "Energy taxonomy: Classifications for the energy transition". ISBN: 978-92-9260-583-4.

[i.26] Wikidata.

[i.27] QUDT.org Quantity Kind: "QUDT Quantity Kind Vocabulary Version 2.1".

[i.28] QUDT.org Unit: "QUDT Unit Vocabulary Version 2.1".

[i.29] OM 2.0: "Ontology of units of Measure (OM)".

[i.30] British Oceanographic Data Centre: "BODC Parameter Usage Vocabulary - P01".

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms given in ETSI TS 103 673 [4] and the following apply:

ontology: formal specification of a conceptualization, used to explicitly capture the semantics of a certain reality

smart application: any application in an IoT system

NOTE: Smart applications should make use of the SAREF ontology as specified in the present document and make use of the SAREF communication framework as specified in ETSI TS 103 267 [3].

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

CBOR	Concise Binary Object Representation
CoAP	Constrained Application Protocol
DP	Datatype Property
DUL	DOLCE+DnS Ultralite
IoT	Internet of Things
IRI	Internationalized Resource Identifier
ITEA	Information Technology for European Advancement
M2M	Machine to Machine
oneM2M	oneM2M Partnership Project
OP	Object Property
OWL	Web Ontology Language
RDF	Resource Description Framework
SAREF	Smart Applications REFerence ontology
SAREF4AGRI	SAREF for the Smart Agriculture and Food Chain domain
SAREF4AUTO	SAREF for the Automotive domain
SAREF4BLDG	SAREF for the Building domain
SAREF4CITY	SAREF for the Smart Cities domain
SAREF4EHAW	SAREF for the eHealth/Ageing well domain
SAREF4ENER	SAREF for the Energy domain
SAREF4ENVI	SAREF for the Environment domain
SAREF4GRID	SAREF for the Smart Grid domain
SAREF4INMA	SAREF for the Industry and Manufacturing domain

SAREF4LIFT	SAREF for the Lift domain
SAREF4WATR	SAREF for the Water domain
SAREF4WEAR	SAREF for the Wearables domain
SEAS	Smart Energy Aware Systems
SIEC	Standard International Energy product Classification
SKOS	Simple Knowledge Organization System
SOSA	Sensor, Observations, Sample, Actuator
SSN	Semantic Sensor Network
STF	Specialist Task Force
SUMO	Suggested Upper Merged Ontology
W3C®	World Wide Web Consortium
WGS84	World Geodetic System 1984

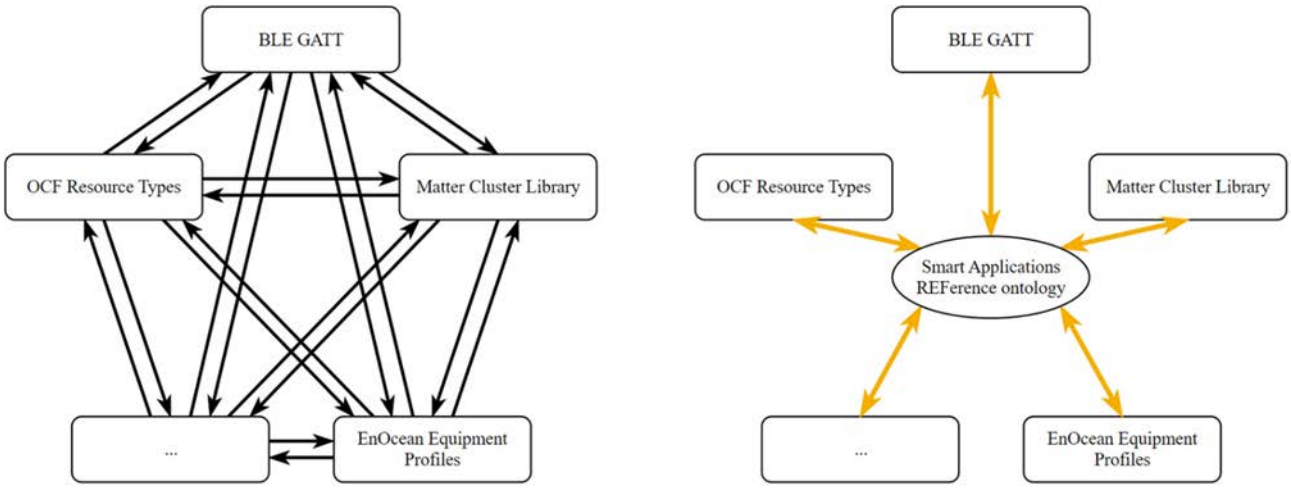
4 History, Design Principles, Extensions

4.1 Introduction and Overview

The Smart Applications REFerence ontology (SAREF) is intended to enable interoperability between solutions from different providers and among various activity sectors from the Internet of Things (IoT), thus contributing to the development of the global digital market.

SAREF explicitly specifies the recurring core concepts in the Smart Applications domain, the main relationships between these concepts, and axioms to constrain the usage of these concepts and relationships. SAREF is based on the fundamental ontology engineering principles of **reuse and alignment** of concepts and relationships that are defined in existing assets, **modularity** to allow separation and recombination of different parts of the ontology depending on specific needs, **extensibility** to allow further growth of the ontology, **maintainability** to facilitate the process of identifying and correcting defects, accommodate new requirements, and cope with changes in (parts of) SAREF. In addition, a principle of **generic versus specific entity distinction** ensures that SAREF can be useful both for editors of online catalogues, taxonomies, and vocabularies, and developers of smart applications.

Mappings to other concepts used by different semantic assets allow translation from the reference ontology to specific assets, reducing the effort of translating from one asset to another, since the reference ontology requires one set of mappings to each asset, instead of a dedicated set of mappings for each pair of assets. Figure 1 shows the role of the reference ontology in the mapping by means of sample assets. The mappings of SAREF to various assets are available in [i.4].



NOTE 1: Bluetooth® Low Energy, Matter™, EnOcean® are examples of suitable products available commercially. This information is given for the convenience of users of the present document and does not constitute an endorsement by ETSI of these products.

NOTE 2: EnOcean®, EnOcean Alliance logo, EnOcean Alliance technology and promoter/member logo are registered trademarks of EnOcean GmbH and EnOcean Alliance Inc. All other product or service names are the property of their respective owners.

NOTE 3: Matter™ is developed by the Connectivity Standards Alliance™. This brand, related logos, and marks are trademarks of the Alliance, all rights reserved.

Figure 1: The role of SAREF in the mapping among different assets

4.2 A Brief History of SAREF

The SAREF initiative started in 2014/2015 with a study requested by the European Commission on "Available Semantics Assets for the Interoperability of Smart Appliances: Mapping into a Common Ontology as a M2M Application Layer Semantics" [i.1]. Such study acknowledged that the energy utilization of Smart Appliances can be reduced if they are managed and controlled on a system level. The system needs standardized interfaces to ensure interoperability. Many of the required standards already exist, but a common architecture does not, resulting in a market which is too fragmented and powerless. Therefore, a reference ontology of consensus was designed to cover the needs of all appliances relevant for energy efficiency. The study consisted of three tasks:

- **Task 1:** Take stock of existing semantic assets and use case assets.
- **Task 2:** Perform a translation exercise of each model (or use case) to a common ontology language and a mapping or matching exercise between all the models.
- **Task 3:** Propose a reference ontology and document the ontology into the ETSI M2M architecture.

NOTE: The ETSI M2M architecture has evolved into the oneM2M architecture, therefore the latter one was considered.

About 50 different semantic assets (i.e. standards, protocols, data models, ontologies) had been identified that describe various properties of Smart Appliances in residential environments. After translating half of these semantic assets into Web Ontology Language (OWL) (https://sites.google.com/site/smartappliancesproject/ontologies), 20 recurring concepts were used as initial building blocks for creating the Smart Applications REFerence ontology. The concepts were mapped from the semantic assets to SAREF to allow for translations between different semantic assets [i.4].

In November 2015, SAREF was transformed into a Technical Specification and published by ETSI SmartM2M as ETSI TS 103 264 [i.5].

In 2016, ETSI SmartM2M requested a Specialist Task Force (STF) to identify and create possible extensions of SAREF, and provide input to update SAREF according to the requirements collected from the stakeholders that have used SAREF since its first release in April 2015. This led to a new release SAREF V2.1.1 [i.6], which incorporated the feedback both from the STF that created the first SAREF extensions, and from the stakeholders that provided their input for improving SAREF. This feedback can be found in ETSI TR 103 411 [i.3] and was used to create SAREF V2.1.1.

The scope of the first release of SAREF was limited to an indoor managed domain, such as a building managed by a building manager or an apartment managed by a user. This scope also included the outdoor premises that belong to the considered indoor managed domain, in other words, a pergola that is part of the building is also within the scope, as well as a sensor located under that pergola. Note that the smart city domain was not originally considered, i.e. if the same sensor that is under the pergola is also in a street, then the sensor in the street was out of the scope of SAREF. After extending SAREF to different domains, it was clear the need for broadening the scope of SAREF from home appliances and buildings to any device that can be found in smart applications; this motivated the change of name of the ontology from "Smart Appliances REFerence ontology" to "Smart Applications REFerence ontology".

In June 2018, another STF started in SmartM2M with the goal (among others) of consolidating SAREF with new reference ontology patterns, based on the experience from the EUREKA ITEA SEAS project. As a result of this STF, 37 different issues were identified and discussed in ETSI TR 103 549 [i.21], proposing and agreeing on resolutions for most of them. Furthermore, it was identified the need for moving some transversal terms used in several extensions of SAREF to the core SAREF ontology, because of the broad applicability of such terms. This led to the release of SAREF ETSI TS 103 264 V3.1.1 [i.7].

SAREF and its different extensions were developed quite independently by different teams of experts, sometimes in parallel. Sometimes different modelling decisions were made, with the result that SAREF extensions had important discrepancies. SmartM2M started to identify ontology patterns that may be used to homogenise the structure of SAREF extensions. In 2022, two new STFs started in SmartM2M with the goal to homogenise and facilitate the use of SAREF and existing 11 SAREF domain mapping by using common ontology patterns. A total of 91 different issues were identified and discussed in ETSI TR 103 781 [i.22], and a set of SAREF Core reference ontology patterns was specified in ETSI TS 103 548 [5]. Many terms were deprecated in V3.2.1 [i.8], and have been deleted in V4.1.1 (the present document). One notable addition to V4.1.1 is that SAREF explicitly targets two main types of usage:

1) SAREF in online taxonomies, vocabularies, and catalogues, can be used identify and describe generic or prototypical entities than can then be reused in many verticals and applications, thus facilitating semantic interoperability.

2) SAREF in smart application, can be used to identify and describe specific entities of the real world for a specific application.

In October 2024, the first ETSI EN about SAREF has been published. ETSI EN 303 760 [1] gives guidance and provisions for making IoT smart applications and products interoperable at the semantic level in compliance to the SAREF framework. It contains provisions about how to use SAREF, and specifies a methodology to follow for showing SAREF compliance. Further on, it describes how to contribute optionally to a new SAREF extension.

Annex A of the present document contains a summary of the most relevant changes made in SAREF V2.1.1 [i.6], V3.1.1 [i.7], V3.2.1 [i.8], and V4.1.1 (the present document). Annex B contains a migration guide from SAREF V3.1.1 to V4.1.1.

4.3 SAREF Design Principles

The Smart Applications REFerence ontology (SAREF) is conceived as a shared model of consensus that facilitates the matching of existing semantic assets for building smart applications, reducing the effort of translating from one asset to another, since SAREF requires one set of mappings to each asset, instead of a dedicated set of mappings for each pair of assets.

Different semantic assets share some recurring, core concepts, but they often use different terminologies and adopt different data models to represent these concepts. Using SAREF, different assets can keep using their own terminology and data models, but still can relate to each other through their common semantics. In other words, SAREF enables semantic interoperability in smart applications through its shared, core concepts.

SAREF explicitly specifies recurring core concepts in smart applications, the main relationships between these concepts, and axioms to constrain the usage of these concepts and relationships. SAREF has been created based on the following fundamental principles:

- **Reuse and alignment** of concepts and relationships that are defined in existing assets. Since a large amount of work was already being done in the smart appliances and in the Internet of Things domains, nothing has been re-invented, but harmonized and aligned what was already there. SAREF is based on the core concepts that were identified as especially relevant to describe the existing semantic assets for smart applications and is aligned to the main classes and properties of the oneM2M base ontology [2].

  SAREF reuses the following resources:

  - oneM2M base ontology [2];
  - W3C® SKOS ontology [6];
  - OGC® and W3C® SOSA/SSN ontology [7];
  - OGC® and W3C® Time ontology [8];
  - OGC® GeoSPARQL vocabulary [9].

  SAREF currently does not contain explicit references to upper ontologies such as DUL or SUMO. The use of upper ontologies is a best practice in ontology engineering, but the industrial world - main user of SAREF - is very pragmatic and is not acquainted with high-level upper ontologies. Introducing DUL would have unnecessarily complicated the understanding and, consequently, the adoption of SAREF by the industry. Anyway, SAREF has been built on a solid ontological foundation and can be related to DUL, but this was not explicitly done in order not to confuse industry users. Furthermore, SAREF currently has mappings to the OGC® and W3C® SOSA/SSN ontology, which is in turn related to DUL. Therefore, SAREF currently includes an indirect reference to DUL through the OGC® and W3C® SOSA/SSN ontology.

- **Modularity** to allow separation and recombination of different parts of the ontology depending on specific needs. SAREF provides building blocks that can be combined to accommodate different needs and points of view. The starting point is the concept of **device**, which is actually common to all the semantic assets considered in the study, although some assets may refer to it with different names, such as **resource** or **product,** but mappings for that are provided in [i.4]. A device is always designed to perform one or more **functions**, therefore, SAREF offers a list of basic functions that can be eventually combined in order to have more complex functions in a single device. Each function has some associated **commands**, which can also be selected as building blocks from a list. Depending on the function(s) it performs, a device can be found in some corresponding **states** that are also listed as building blocks, so that it is easy and intuitive to combine devices, functions and states. SAREF also provides a list of **properties** that can be used to further specialize the functioning of a device.

- **Extensibility** to allow further growth of the ontology. Different stakeholders can specialize the SAREF concepts according to their needs and points of view, add more specific relationships and axioms to refine the general (common) semantics expressed in the reference ontology, and create new concepts, as long as they explicitly link these extensions to at least one existing concept and/or relationship in SAREF. The minimum requirement is that any extension/specialization shall comply with SAREF. Examples of extensions of SAREF in different domains are SAREF4ENER (energy domain) [i.9], SAREF4ENVI (environment domain) [i.10] and SAREF4BLDG (building domain) [i.11]. SAREF and extensions are based on patterns that are used in different domains. SAREF extension developers should reuse SAREF reference ontology patterns as specified in ETSI TS 103 548 [5].

- **Maintainability** to facilitate the process of identifying and correcting defects, accommodate new requirements, and cope with changes in (parts of) SAREF. According to the extensibility criterion mentioned above, a new module/ontology can be created to further extend/specialize concepts of SAREF. The party that creates the extension should also be responsible for the maintenance of this extension and its evolution over time. SAREF extension developers shall comply with the SAREF Development Framework and Workflow as specified in ETSI TS 103 673 [4]. For an initial strategy proposed in ETSI to extend, maintain and evolve SAREF (and its extensions), see ETSI TR 103 411 [i.3].

- **Generic versus specific entity distinction**. SAREF is designed to model both generic and specific entities. Firstly, it provides a standardized way to define generic or prototypical entities in taxonomies, vocabularies, and catalogues, ensuring broad semantic interoperability across domains. Secondly, it enables the description of real-world specific entities within smart applications, ensuring precise and context-aware implementation. This dual-purpose approach enhances SAREF's adaptability across different levels of abstraction and application scenarios. Online catalogue and taxonomy editors will populate classes such as **DeviceKind**, **FeatureKind, Property**, **State**, **Function**, **Command**, while application developers will instantiate classes such as **Device** and **FeatureOfInterest**.

4.4 SAREF Extensions

SAREF is the reference ontology for smart applications and contains recurring concepts that are used in several domains. SAREF has a close relation with the oneM2M Base Ontology, for which a mapping is defined in clause 6. As smart applications are not restricted to only one domain, it is possible that specific concepts for a certain domain are not part of SAREF. To be able to handle these additional concepts and provide different domains with a proper ontology that reflects the specific needs of that domain, extensions to SAREF should be created. Figure 2 shows SAREF as the core model to be used as basis for creating extensions in different domains, which are represented as rectangles. Each domain can have one or more extensions, depending on the complexity of the domain and the different needs. Extensions of SAREF have been created for:

- SAREF4ENER for the Energy domain in ETSI TS 103 410-1 [i.9].
- SAREF4ENVI for the Environment domain in ETSI TS 103 410-2 [i.10].
- SAREF4BLDG for the Building domain in ETSI TS 103 410-3 [i.11].
- SAREF4CITY for the Smart City domain in ETSI TS 103 410-4 [i.12].
- SAREF4INMA for the Industry and Manufacturing domain in ETSI TS 103 410-5 [i.13].
- SAREF4AGRI for the Agrifood domain in ETSI TS 103 410-6 [i.14].
- SAREF4AUTO for the Automotive domain in ETSI TS 103 410-7 [i.15].
- SAREF4EHAW for the eHealth and Ageing Well domain in ETSI TS 103 410-8 [i.16].
- SAREF4WEAR for the Wearables domain in ETSI TS 103 410-9 [i.17].
- SAREF4WATR for the Water domain in ETSI TS 103 410-10 [i.18].
- SAREF4LIFT for the Smart Lifts domain in ETSI TS 103 410-11 [i.19].
- SAREF4GRID for the Smart Grids domain in ETSI TS 103 410-12 [i.20].

Other extensions can be created for new domains and, if needed, also for the same domains for which extensions already exist.

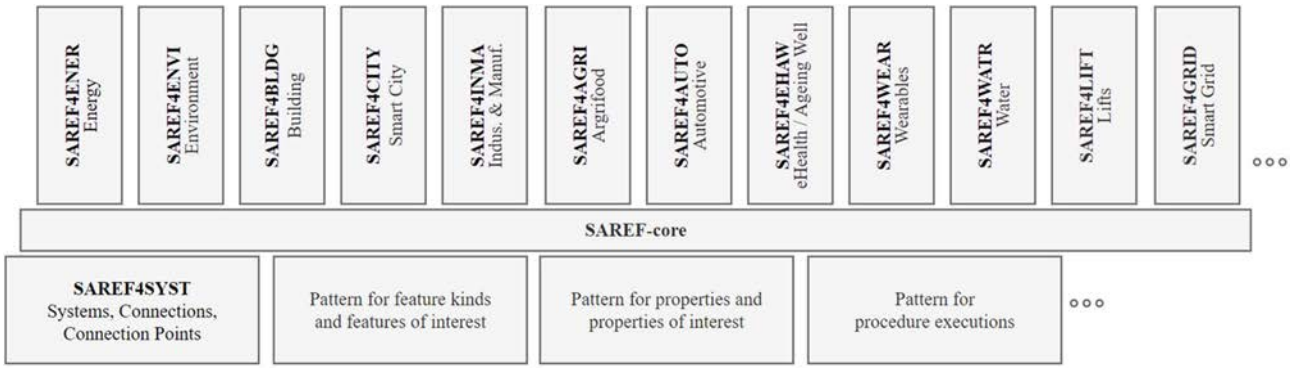


Figure 2: SAREF and its extensions

4.5 Develop, apply and evolve SAREF ontologies

The provisions given in ETSI EN 303 760 [1] and the following apply to SAREF:

Provision 1: SAREF Core and SAREF Extension development shall conform to ETSI TS 103 673 [4].

Provision 2: Whenever appropriate, SAREF Extensions should reuse and extend SAREF Reference ontology patterns as specified in ETSI TS 103 548 [5].

Provision 3: SAREF shall use the SAREF Communication framework as defined in ETSI TS 103 267 [3].

Some examples of device built using SAREF can be found in ETSI TR 103 411 [i.3] and in the different SAREF extensions ([i.9] to [i.20]).

5 Specification of SAREF

5.0 Prefixes and Namespaces

The prefixes and namespaces used in SAREF and in the present document are listed in Table 1.

Table 1: Prefixes and namespaces used within the SAREF ontology

Prefix	Namespace
saref	https://saref.etsi.org/core/
dcterms	http://purl.org/dc/terms/
owl	http://www.w3.org/2002/07/owl#
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#
rdfs	http://www.w3.org/2000/01/rdf-schema#
xsd	http://www.w3.org/2001/XMLSchema#
skos	http://www.w3.org/2004/02/skos/core#
sosa	http://www.w3.org/ns/sosa/
ssn	http://www.w3.org/ns/ssn/
time	http://www.w3.org/2006/time#
geo	http://www.opengis.net/ont/geosparql#

NOTE: Some examples in the present document make use of additional prefixes. Prefix `s4abcd:` is used on entities that would be expected to be part of some SAREF extension such as SAREF4ABCD. Prefix `ex:` is used on entities that would be expected to be part of an application-specific extension of SAREF.

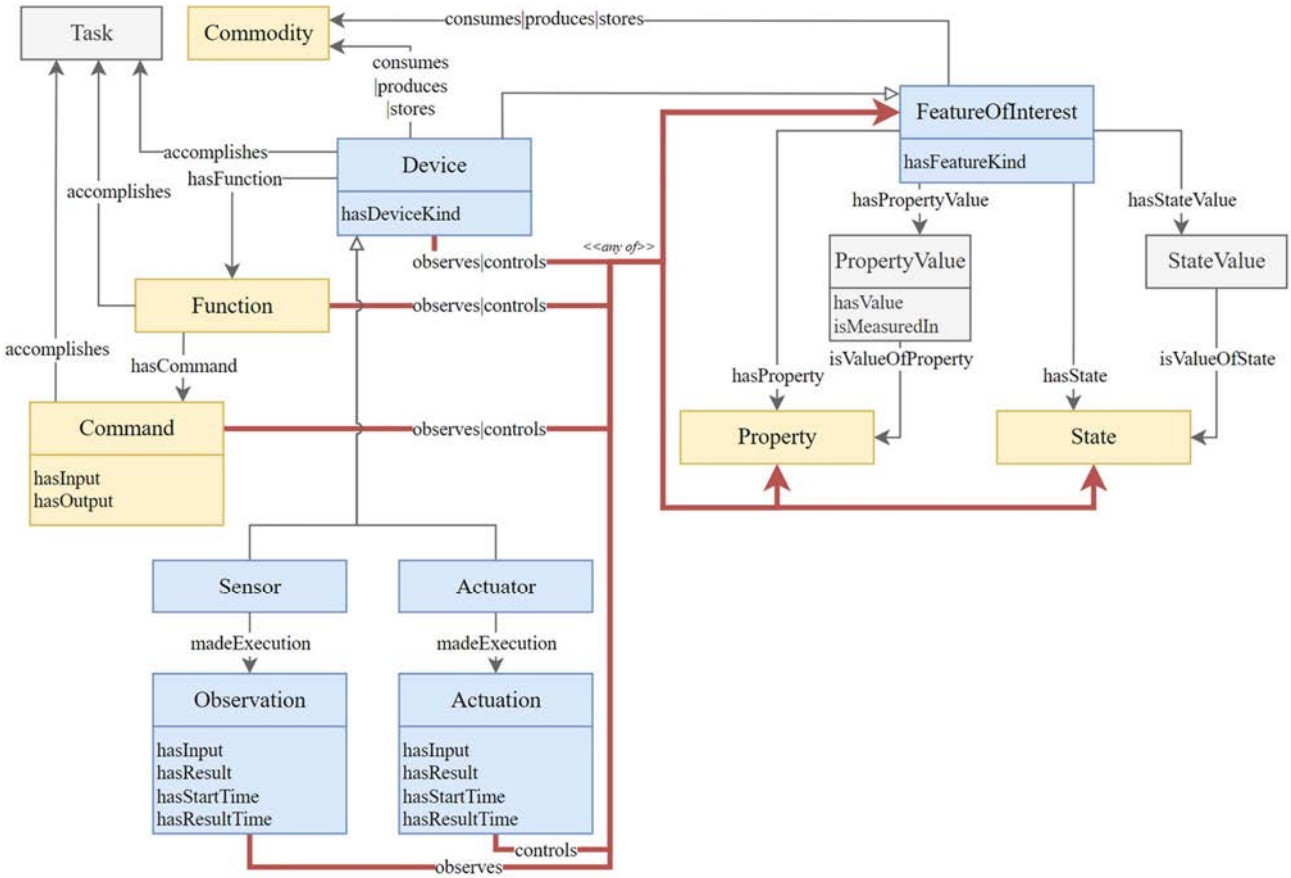
5.1 General Overview

5.1.1 Main classes and properties of SAREF Core

SAREF is designed to model both generic and specific entities.

Figure 3 shows an overview of some of the main classes and properties of SAREF Core.

A detailed explanation of each class is presented in clause 5.2 to clause 5.14.



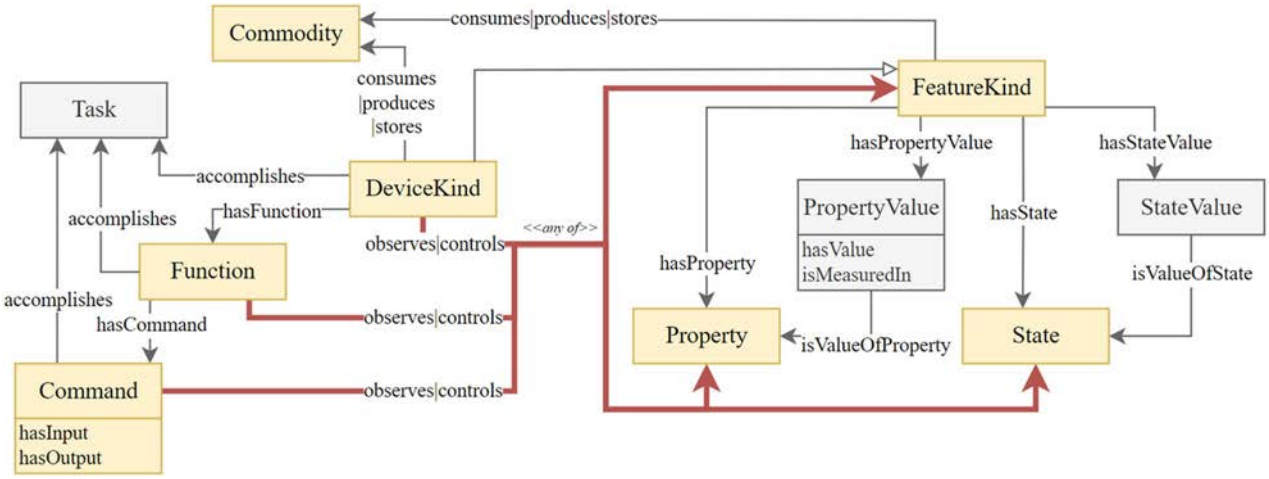
NOTE: Classes represented in yellow should be instantiated only in taxonomies, vocabularies, and catalogues.
Classes represented in blue should be instantiated only in smart applications.
Classes represented in light grey can be instantiated in either type of usage of SAREF.
An instance of any class from which originates the red lines can be linked to an instance of any class where the red lines end.

Figure 3: Overview of the main classes and properties of SAREF Core

5.1.2 Classes and properties for generic entities

SAREF provides a standardized way to define generic or prototypical entities in taxonomies, vocabularies, and catalogues, ensuring broad semantic interoperability across domains.

Figure 4 shows an overview of the classes and properties for generic entities.



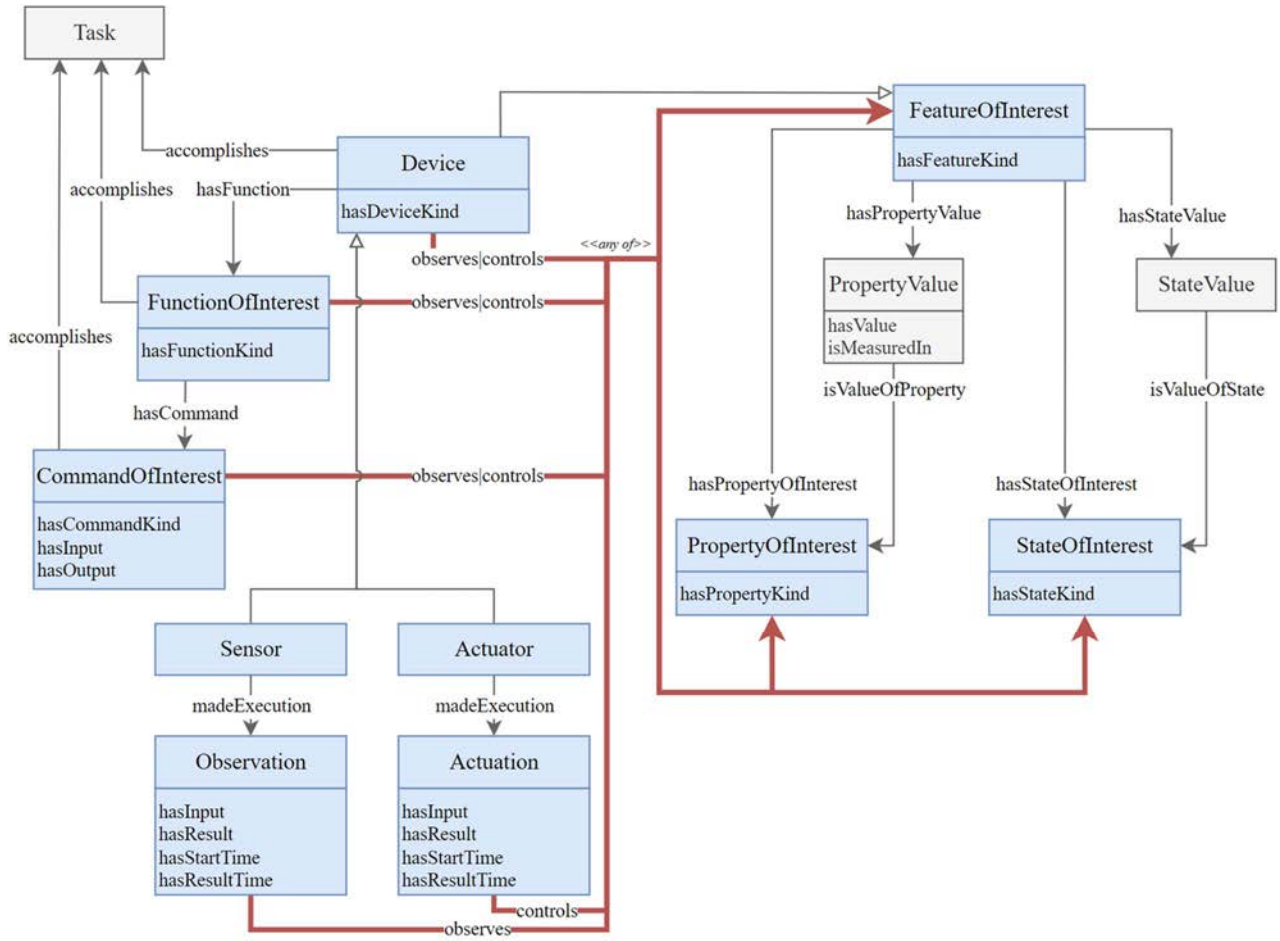
NOTE: Classes represented in yellow should be instantiated only in taxonomies, vocabularies, and catalogues. An instance of any class from which originates the red lines can be linked to an instance of any class where the red lines end.

Figure 4: Classes and properties for generic entities

5.1.3 Classes and properties for specific entities

SAREF also enables the description of real-world specific entities within smart applications, ensuring precise and context-aware implementation.

Figure 5 shows an overview of the classes and properties for specific entities.



NOTE: Classes represented in blue should be instantiated only in smart applications.
An instance of any class from which originates the red lines can be linked to an instance of any class where the red lines end.

Figure 5: Classes and properties for specific entities

5.1.4 Relation to other documents

ETSI EN 303 760 [1] provides guidelines to apply, extend, and contribute to the evolution of SAREF Core. The SAREF Communication framework as defined in ETSI TS 103 267 [3]. SAREF Core development conforms to ETSI TS 103 673 [4]. SAREF Core reuses and extends SAREF Reference ontology patterns as specified in ETSI TS 103 548 [5].

5.2 Device kinds and devices

Figure 6 illustrates the main classes and properties for devices. SAREF extensions and applications may create, specialize, and categorize devices as specified in ETSI TS 103 548 [5].

NOTE 1: SAREF V4.1.1 introduced the class `saref:DeviceKind` for the prototypical description of devices that can populate online taxonomies, vocabularies, and catalogues. Considering `saref:Device` was mostly instantiated for smart applications, it was decided to keep this identifier for the specific class (instead of introducing some class like "DeviceOfInterest"), and introduce `saref:DeviceKind`.

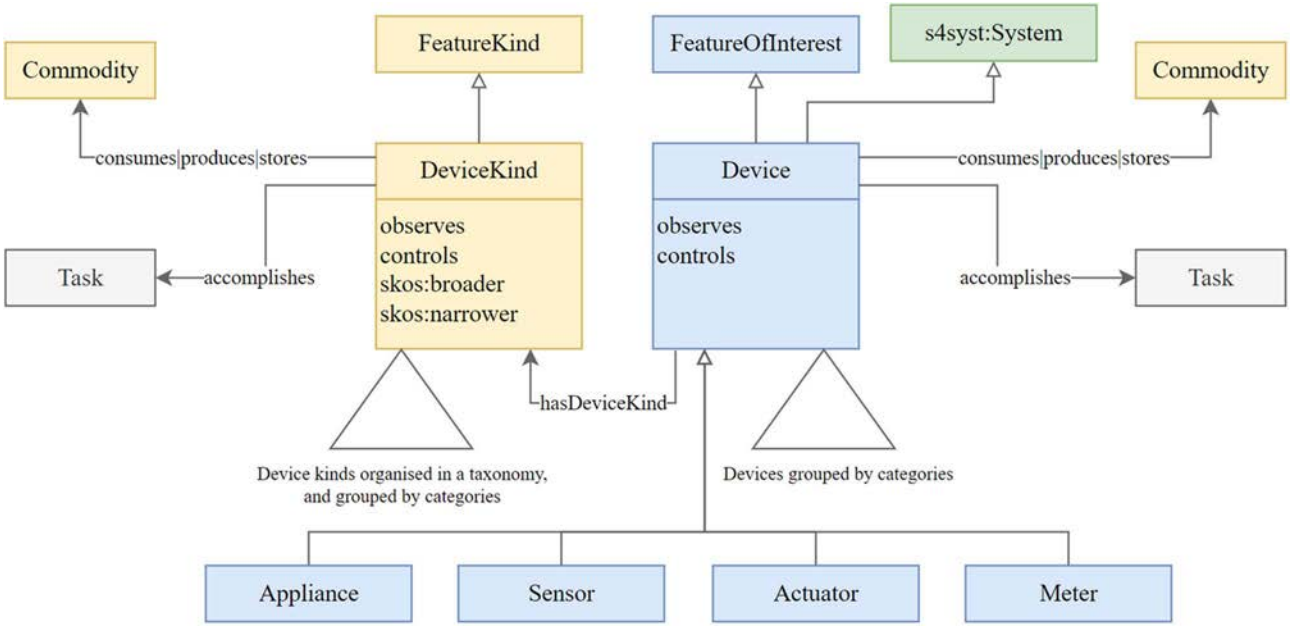


Figure 6: Devices

Class `saref:Device` represents any tangible object designed to accomplish a particular task by performing one or more functions. An instance of `saref:Device` represents one specific real world entity.

EXAMPLE 1: Examples of devices are a light switch, a temperature sensor, an energy meter, a water flow meter, and a laundry dryer. A laundry dryer is designed to dry laundry, and to accomplish this task it has a start/stop function.

EXAMPLE 2:
```
ex:Meter4837QW123 a saref:Device ;
     rdfs:label "Meter 4837QW123"@en ;
     rdfs:comment """The meter that measures the incoming water flow
     of the Computer Science school."""@en .
```

EXAMPLE 3:
```
ex:Dryer354A1E08FA a saref:Device ;
     rdfs:label "Dryer 354A1E08FA"@en ;
     rdfs:comment "The laundry dryer with serial number 354A1E08FA"@en.
```

Devices are also systems (`s4syst:System`, see [5]) and features of interest (`saref:FeatureOfInterest`).

Class `saref:DeviceKind` allows to describe kinds of devices, with common properties and common states having the same value, and with common functions and services. Devices kinds may be used to describe models of devices in online catalogues.

A device can be linked to its device kinds using OP `saref:hasDeviceKind`.

EXAMPLE 4:
```
s4abcd:SmartWaterMeterABC123 a saref:DeviceKind ;
   rdfs:label "Smart Water Meter ABC123"@en ;
   rdfs:comment """The smart water meter ABC123 of manufacturer ABC.
    A device kind."""@en .
ex:Meter4837QW123 saref:hasDeviceKind s4abcd:SmartWaterMeterABC123 .
```

A device can act upon (OP `saref:actsUpon`) features, properties, or states. SAREF Core defines different sub-properties of `saref:actsUpon`:

- `saref:observes` for when a device observes a feature of interest, or property or state of that feature.
- `saref:controls` for when a device controls a feature of interest, or property or state of that feature.

NOTE 2: Property `saref:actsUpon` also applies to functions, commands, and procedure executions.

EXAMPLE 5:
```
s4abcd:AAHeatPumpDryer a saref:DeviceKind ;
    saref:controls s4abcd:DryerRotationalSpeed ;
    saref:controls s4abcd:DryerTemperature ;
    saref:observes s4abcd:LaudryBatchHumidity .
```

As shown in Figure 6, SAREF Core provides some examples of classes of devices including appliances, sensors, actuators, and meters. Their definitions are the following:

- `saref:Appliance`: The class of devices designed to accomplish a particular task for occupant use. It consumes, produces, or stores, some commodity.
- `saref:Sensor`: A device designed to observe and measure one or more properties or states of one or more features of interest.
- `saref:Actuator`: A device designed to control one or more properties or states of one or more features of interest.
- `saref:Meter`: A device designed to observe and additionally do some computation and/or display one or more properties or states of one or more features of interest.

The manufacturer and model of a `saref:DeviceKind` or a `saref:Device` can be statedd using DPs `saref:hasManufacturer` and `saref:hasModel`, respectively.

EXAMPLE 6:
```
ex:Meter4837QW123 a saref:Device ;
    saref:hasManufacturer "ABC"@en ;
  saref:hasModel "4837QW123"@en .
```

5.3 Feature kinds and features of interest

Figure 7 illustrates the main classes and properties for feature kinds and features of interest. SAREF extensions and applications may create, specialize, and categorize feature kinds and features of interest as specified in ETSI TS 103 548 [5].

NOTE 1: Class `saref:FeatureKind` has been introduced in SAREF V3.2.1 for the prototypical description of features of interest that can populate online taxonomies, vocabularies, and catalogues.

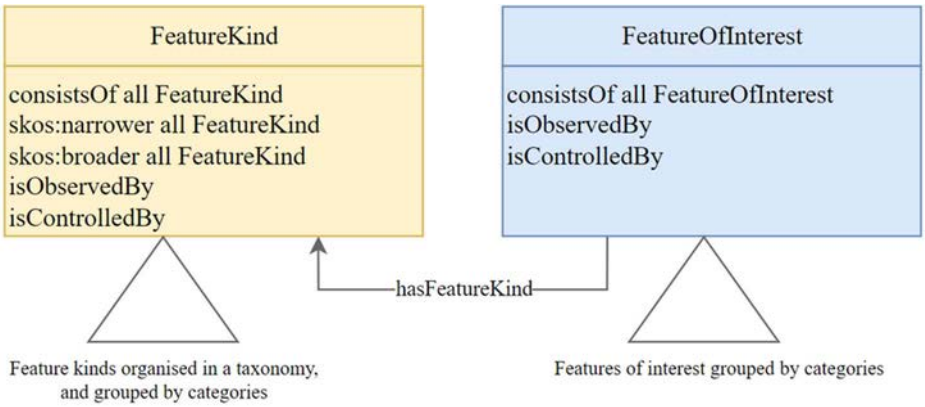


Figure 7: Feature kinds and features of interest

Class `saref:FeatureOfInterest` represents any real world entity from which a property or a state may be acted upon, such as observed and controlled. An instance of `saref:FeatureOfInterest` represents one specific real world entity.

EXAMPLE 1:
```
<etsi_premises/athena> a saref:FeatureOfInterest ;
     rdfs:label "ATHENA amphitheatre"@en ;
     rdfs:comment "The ATHENA amphitheatre in the ETSI premises."@en .
<etsi_premises/athena/window5> a saref:FeatureOfInterest ;
     rdfs:label "ATHENA Window 5"@en ;
     rdfs:comment "Window 5 of ATHENA amphitheatre"@en .
```

Class `saref:Device` is a subclass of `saref:FeatureOfInterest`.

Class `saref:FeatureKind` allows to describe kinds of features of interest, with common properties and common states having the same value. An instance of `saref:FeatureKind` represents an archetype of real world entities, for example to populate product catalogues.

EXAMPLE 2:
```
<1000x2000mmWindowOrientedNorth> a saref:FeatureKind ;
     rdfs:label "1000x2000mm window oriented north"@en ;
     rdfs:comment "The kind of windows with dimensions 1000x2000mm and
     oriented north."@en .
```

Class `saref:DeviceKind` is a subclass of `saref:FeatureKind`.

Feature kinds can be organized in a taxonomy using OPs `skos:narrower` and `skos:broader`.

EXAMPLE 3:
```
<1000x2000mmWindowOrientedNorth> a saref:FeatureKind ;
     skos:broader <1000x2000mmWindow> ;
     skos:broader <WindowOrientedNorth> .
```

A feature of interest can be linked to its kind(s) using OP `saref:hasFeatureKind`. OP `saref:hasDeviceKind` is a sub-property of OP `saref:hasFeatureKind`.

EXAMPLE 4:
```
<etsi_premises/athena/window5>
     saref:hasFeatureKind <1000x2000mmWindowOrientedNorth> .
```

NOTE 2: Feature of interest inherit broader feature kinds.
$$\texttt{saref:hasFeatureKind} \circ \texttt{skos:broader} \sqsubseteq \texttt{saref:hasFeatureKind}$$

EXAMPLE 5:
```
<etsi_premises/athena/window5>
     saref:hasFeatureKind <1000x2000mmWindow> ;
     saref:hasFeatureKind <WindowOrientedNorth> .
```

A feature kind (respectively a feature of interest) may consist of (OP `saref:consistsOf`) other feature kinds (respectively feature kinds or features of interest).

EXAMPLE 6:
```
s4abcd:AAHeatPumpDryer a saref:FeatureKind ;
saref:consistsOf s4abcd:HeatPump , s4abcd:AirCyclingCircuit ,
s4abcd:Dryer .
```

EXAMPLE 7:
```
<etsi_premises/athena> a saref:FeatureOfInterest ;
  saref:consistsOf <etsi_premises/stage> ;
  saref:consistsOf <etsi_premises/athena/window5> .
```

Whenever appropriate, SAREF extensions and applications should also use the classes from SAREF4SYST [5] to specify if a `saref:FeatureKind` or a `saref:FeatureOfInterest` is a system (`s4syst:System`), a Connection between systems (`s4syst:Connection`), or a connection point of a system (`s4syst:ConnectionPoint`).

The model and the manufacturer of a `saref:FeatureKind` or a `saref:FeatureOfInterest` can be explicited using DPs `saref:hasModel` and `saref:hasManufacturer`, respectively.

EXAMPLE 8:
```
<ball-bearing-xsd215sd7f> a saref:FeatureOfInterest , s4syst:System ;
  saref:hasManufacturer "Company X"@en ;
  saref:hasModel "6000-2RZ1 "@en .
```

5.4 Tasks

Class `saref:Task` represents goals for which a device is designed, from a user perspective.

SAREF extensions and applications may create, specialize, and categorize tasks as specified in ETSI TS 103 548 [5], and illustrated on Figure 8.

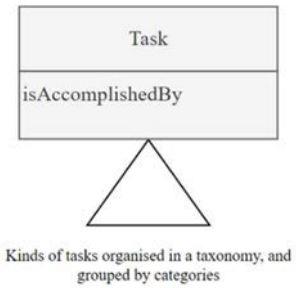


Figure 8: Taxonomy of tasks, grouped by categories

Tasks can be organized in a taxonomy using OPs `skos:narrower` and `skos:broader`.

EXAMPLE 1: Example of tasks are cleaning, drying, and lighting.

```
saref:Drying a saref:Task ;
     rdfs:label "Drying"@en ;
     rdfs:comment "A type of task for which a device is designed."@en .
```

Device kinds and devices can be linked to the one or more tasks they are designed to accomplish with OP `saref:accomplishes`.

EXAMPLE 2:

```
s4abcd:AAHeatPumpDryer a saref:FeatureKind ;
   saref:accomplishes s4abcd:DryingLaundry .
```

EXAMPLE 3:

```
ex:switch21354 a saref:Device ;
   saref:hasDeviceKind s4abcd:Switch ;
   saref:accomplishes s4abcd:Lighting .
```

NOTE 1: Property `saref:accomplishes` can also apply to other classes such as functions, commands, and procedure executions.

NOTE 2: External ontologies can be used to further describe tasks.

5.5 Commodities

Class `saref:Commodity` represents marketable items which may be supplied without qualitative differentiation. Commodities may be consumed, produced, or stored, by some feature of interest or device.

SAREF extensions and applications may create, specialize, and categorize commodities as specified in ETSI TS 103 548 [5], and illustrated on Figure 9.

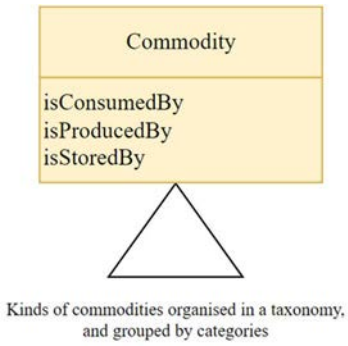


Figure 9: Taxonomy of commodities, grouped by categories

Commodities can be organized in a taxonomy using OPs `skos:narrower` and `skos:broader`.

NOTE: Concepts from existing code lists, vocabularies, and taxonomies, may be used as instances of `saref:Commodity`, for example from the ISDA taxonomy of commodities [i.23], the Standard International Energy product Classification (SIEC) [i.24], the IRENA Energy Taxonomy [i.25] or Wikidata [i.26].

EXAMPLE 1:
```
ex:Electricity a ex:EnergyCommodity ;
     rdfs:label "Electricity"@en ;
     rdfs:comment "The electricity energy commodity."@en ;
     rdfs:isDefinedBy "The IDSA taxonomy of energy commodities" .
```

A feature kind, feature of interest, or device, can consume (OP `saref:consumes`), produce (OP `saref:produces`), or store (OP `saref:stores`), a certain commodity.

EXAMPLE 2:
```
s4abcd:AAHeatPumpDryer a saref:FeatureKind ;
    saref:consumes ex:Electricity ;
    saref:produces <http://www.wikidata.org/entity/Q7892> .
```

5.6 Properties, properties of interest, and property values

5.6.1 Introduction

In SAREF, properties refer to the identifiable qualities of features of interest that can be acted upon by devices, such as observed or controlled. While properties can apply to different features of interest, properties of interest are specific to a feature of interest. Property values describe the value for a property.

NOTE: SAREF V3.2.1 introduced the class `saref:PropertyOfInterest` for identifying the association between a feature of interest and a property in some smart application. Considering `saref:Property` was mostly instantiated in online taxonomies and vocabularies, it was decided to keep this identifier for the generic class (instead of introducing some class like "PropertyKind"), and introduce `saref:PropertyOfInterest`.

Figure 10 illustrates the main classes and properties for describing properties, properties of interest, and property values.

SAREF extensions and applications may create, specialize, and categorize properties as specified in ETSI TS 103 548 [5].

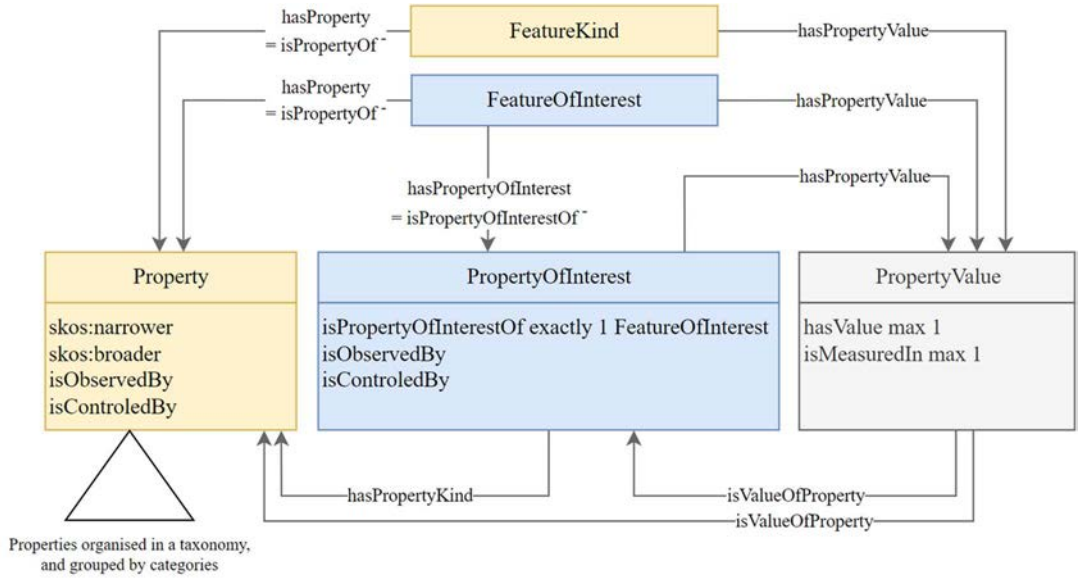


Figure 10: Properties, properties of interest, and property values

5.6.2 Properties

An instance of `saref:Property` can apply to different features of interest.

EXAMPLE 1: Air temperature, pressure, luminosity, etc. are all properties.

```
s4abcd:Temperature a saref:Property ;
   rdfs:label "Temperature"@en ;
   rdfs:comment "The temperature property kind."@en .
s4abcd:Pressure a saref:Property ;
   rdfs:label "Pressure"@en ;
   rdfs:comment "The pressure property kind."@en .
s4abcd:Luminosity a saref:Property ;
   rdfs:label "Temperature"@en ;
   rdfs:comment "The luminosity property kind."@en .
```

NOTE 1: Until SAREF V3.1.1 [i.7], there was an ambiguity between whether properties should be specific or generic to features of interest. This ambiguity has been solved in SAREF V3.2.1 [i.8], and the new modelling choice has been enforced in SAREF V4.1.1 (the present document).

Properties can be organized in a taxonomy using properties `skos:narrower` and `skos:broader`.

NOTE 2: Concepts from existing code lists, vocabularies, and taxonomies, may be used as instances of `saref:Property`.

EXAMPLE 2: Two examples using the QUDT Quantity Kind vocabulary [i.27], and the British Oceanographic Data Centre Parameter Usage Vocabulary [i.30].

```
<https://qudt.org/2.1/vocab/quantitykind/ActiveEnergy>
   a saref:Property ;
   rdfs:label "Active Energy"@en ;
   rdfs:comment """Active Energy" is the electrical energy
transformable into some other form of energy."""@en .

<http://vocab.nerc.ac.uk/collection/P01/current/CDTSZZ01/>
   a saref:Property ;
   skos:prefLabel """Absolute temperature standard deviation of the
atmosphere by dry bulb thermometer"""@en
```

The OP `saref:hasProperty` may be used to link a feature kind or feature of interest to its properties. Its inverse is `saref:isPropertyOf`.

EXAMPLE 3:
```
s4abcd:AAHeatPumpDryer a saref:FeatureKind ;
   saref:hasProperty s4abcd:DryerRotationalSpeed ;
   saref:hasProperty s4abcd:DryerTemperature .
```

NOTE 3: Feature kinds inherit the properties of their broader feature kinds.
```
skos:broader o saref:hasProperty ⊑ saref:hasProperty
```

NOTE 4: Features of interest inherit the properties of their feature kinds.
```
saref:hasFeatureKind o saref:hasProperty ⊑ saref:hasProperty
```

5.6.3 Properties of interest

An instance of `saref:PropertyOfInterest` is specific to a feature of interest. It is inherent to and cannot exist without that feature of interest.

EXAMPLE 1: The air temperature of the atmosphere sample at a certain location and altitude, the received signal strength indicator of a wireless IoT connection, the luminosity of the ETSI ATHENA amphitheatre.

The OP `saref:hasPropertyOfInterest` may be used to link a feature of interest to its properties of interest. Its inverse is `saref:isPropertyOfInterestOf` and is functional.

A property of interest is the property of (OP `saref:isPropertyOfInterestOf`) exactly one feature of interest.

NOTE 1: Properties of interest need not always be explicited. It depends on the use case. Typically, properties of interest are useful in applications, where the association between a feature of interest and a property (i.e. the property of interest) needs to be identified and related to other entities.

Given a property of interest belongs to exactly one feature of interest, it is recommended that its identifier consists of the identifier of the feature of interest, followed by character '#' and a fragment identifier. The fragment identifier part of the IRI of a property of interest should not contain "property".

A property of interest can be linked to its kind(s) using OP `saref:hasPropertyKind`.

EXAMPLE 2: The luminosity of the ETSI ATHENA amphitheatre belongs to the ETSI ATHENA amphitheatre.
```
<etsi_premises/athena#luminosity> a saref:PropertyOfInterest ;
   saref:hasPropertyKind s4abcd:Luminosity ;
   saref:isPropertyOfInterestOf <etsi_premises/athena> ;
   rdfs:comment "The luminosity of amphitheatre ATHENA"@en .
```

NOTE 2: Properties of interest inherit broader properties.
```
saref:hasPropertyKind o skos:broader ⊑ saref:hasPropertyKind
```

NOTE 3: Features of interest inherit the property kinds of their properties of interest.
```
saref:hasPropertyOfInterest o saref:hasPropertyKind ⊑ saref:hasProperty
```

EXAMPLE 3: The ETSI ATHENA amphitheatre has a property luminosity.
```
<etsi_premises/athena> a saref:FeatureOfInterest ;
saref:hasPropertyOfInterest <etsi_premises/athena#luminosity> ;
saref:hasProperty saref:Luminosity .
```

5.6.4 Property Values

Class `saref:PropertyValue` describes the value for a property. The property value is optionally linked to its value expressed as an RDF literal (DP `saref:hasValue`), optionally to the unit of measurement (OP `saref:isMeasuredIn`), and optionally to the properties or properties of interest it is a value of (OP `saref:isValueOfProperty`).

EXAMPLE 1:
```
[] a saref:PropertyValue ;
     saref:hasValue 22.7 ;
     saref:isMeasuredIn <https://qudt.org/2.1/vocab/unit/DEG_C> .
```

EXAMPLE 2:
```
[] a saref:PropertyValue ;
   a <http://www.ontology-of-units-of-measure.org/resource/om-
  2/Temperature> ;
   saref:hasValue 22.7 ;
   saref:isMeasuredIn <http://www.ontology-of-units-of-
   measure.org/resource/om-2/degreeCelsius> .
```

The range of `saref:isMeasuredIn` is defined as `saref:UnitOfMeasure`.

NOTE 1: Concepts from existing code lists, vocabularies, and taxonomies, may be used as instances of `saref:UnitOfMeasure`. For example the QUDT Unit vocabulary [i.28] or the OM vocabulary [i.29].

The OP `saref:hasPropertyValue` links a feature kind, a feature of interest, or a property of interest, to a property value.

EXAMPLE 3:
```
<etsi_premises/athena#size> a saref:PropertyOfInterest ;
   saref:isPropertyOf <etsi_premises/athena> ;
   saref:hasPropertyValue [
      a saref:PropertyValue ;
      saref:hasValue 105.0 ;
      saref:isMeasuredIn <https://qudt.org/2.1/vocab/unit/M2> ] .
```

NOTE 2: The property values are inherited in the hierarchy of feature kinds. This enables to incrementally construct prototypical descriptions of features of interest.
`skos:broader saref:hasPropertyValue ⊑ saref:hasPropertyValue`

NOTE 3: A feature of interest does not inherit the property values of its kinds. There may be multiple reasons why the property value of a feature of interest is different from that of its prototypical descriptions. For example, it may be caused by a defect, a deterioration, or a customization.

The OP `saref:isValueOfProperty` links a property value to the properties and properties of interest it is a value of.

EXAMPLE 4:
```
<1000x2000mmWindowOrientedNorth> a saref:FeatureKind ;
   saref:hasProperty <WindowArea> ;
   saref:hasPropertyValue [
      a saref:PropertyValue ;
      saref:hasValue 2.0 ;
      saref:isMeasuredIn <https://qudt.org/2.1/vocab/unit/M2> ;
      saref:isValueOfProperty <WindowArea> ] .
```

NOTE 4: A property value about a property of interest is also a property value of its property kinds.
`saref:isValueOfProperty o saref:hasPropertyKind ⊑ saref:isValueOfProperty`

NOTE 5: `saref:hasPropertyValue` and `saref:isValueOfProperty` are not inverse properties.

5.7 States, states of interest, and state values

5.7.1 Introduction

In SAREF, states refer to the identifiable conditions that features of interest are or may be in, and that can be acted upon by devices, such as observed and controlled. While states can apply to different features of interest, states of interest are specific to a feature of interest. State values describe the value for a state.

NOTE 1: SAREF V3.2.1 introduced the class `saref:StateOfInterest` for identifying the association between a feature of interest and a state in some smart application. Considering `saref:State` was mostly instantiated in online taxonomies and vocabularies, it was decided to keep this identifier for the generic class (instead of introducing some class like "StateKind"), and introduce `saref:StateOfInterest`.

Figure 11 illustrates the main classes and properties for describing states, states of interest, and state values.

SAREF extensions and applications may create, specialize, and categorize states as specified in ETSI TS 103 548 [5].

NOTE 2: States and Properties are parallel, distinct, but not disjoint. Users can chose whichever seems to best fit their needs. Clause 5.14 discusses Composite Properties, that help to combine properties, states and features of interest in one composite property.

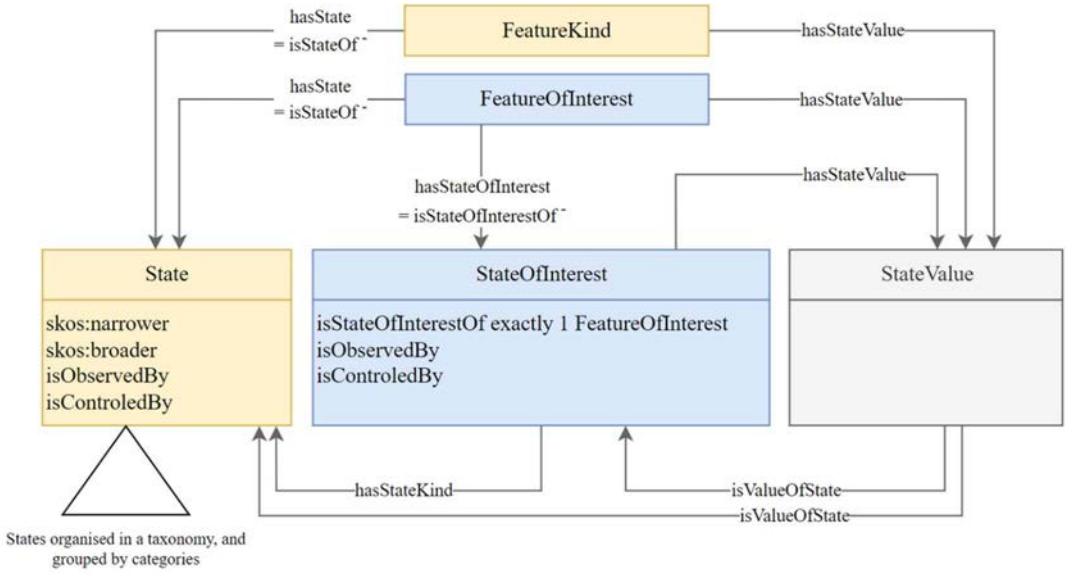


Figure 11: States and states of interest

5.7.2 States

An instance of `saref:State` can apply to different features of interest.

EXAMPLE 1: A switch can be found in the `saref:OnOffState`.

```
ex:OnOffState a saref:State .

ex:Switch a saref:FeatureKind ;
saref:hasState ex:OnOffState .
```

NOTE 1: SAREF is not restricted to binary states such as the `ex:OnOffState`, but allows to define also enumerated states.

States can be organized in a taxonomy using properties `skos:narrower` and `skos:broader`.

NOTE 2: Concepts from existing code lists, vocabularies, and taxonomies, may be used as instances of `saref:State`.

The OP `saref:hasState` may be used to link a feature kind to its states. Its inverse is `saref:isStateOf`.

EXAMPLE 2:

```
s4abcd:TwoButtonsOneWaySwitch a saref:FeatureKind ;
saref:hasState s4abcd:Button1UpDownState , s4abcd:Button2UpDownState .
s4abcd:Button1UpDownState a saref:State ;
skos:broader s4abcd:ButtonUpDownState .
s4abcd:Button2UpDownState a saref:State ;
skos:broader s4abcd:ButtonUpDownState .
```

NOTE 3: Feature kinds inherit the states of their broader feature kinds.

```
skos:broader o saref:hasState ⊑ saref:hasState
```

NOTE 4: Features of interest inherit the states of their feature kinds.

```
saref:hasFeatureKind o saref:hasState ⊑ saref:hasState
```

5.7.3 States of Interest

An instance of `saref:StateOfInterest` is specific to a feature of interest. It is inherent to and cannot exist without that feature of interest.

The OP `saref:hasStateOfInterest` may be used to link a feature of interest to its states of interest.
Its inverse is `saref:isStateOfInterestOf` and is functional.

A state of interest is the state of (OP `saref:isStateOfInterestOf`) exactly one feature of interest.

NOTE 1: States of interest need not always be explicited. It depends on the use case. Typically, states of interest are useful in applications where the association between a feature of interest and a state (i.e. the state of interest) needs to be identified and related to other entities.

Given a state of interest belongs to exactly one feature of interest, it is recommended that its identifier consists of the identifier of the feature of interest, followed by character '#' and a fragment identifier. The fragment identifier part of the IRI of a state of interest should not contain "state".

EXAMPLE: Switch `<switch_sdf5ze4fz3>` has two up/down states of interest.

```
<switch_sdf5ze4fz3> a saref:Device ;
   saref:hasDeviceKind s4abcd:TwoButtonsOneWaySwitch ;
   saref:hasStateOfInterest <switch_sdf5ze4fz3#btn1> ;
   saref:hasStateOfInterest <switch_sdf5ze4fz3#btn2> .
<switch_sdf5ze4fz3#btn1> a saref:StateOfInterest ;
   saref:hasStateKind s4abcd:Button1UpDownState .
<switch_sdf5ze4fz3#btn2> a saref:StateOfInterest ;
   saref:hasStateKind s4abcd:Button2UpDownState .
```

A state of interest can be linked to its kind(s) using OP `saref:hasStateKind`.

NOTE 2: States of interest inherit broader states.

```
saref:hasStateKind o skos:broader ⊑ saref:hasStateKind
```

NOTE 3: Features of interest inherit the state kinds of their states of interest.

```
saref:hasStateOfInterest o saref:hasStateKind ⊑ saref:hasState
```

5.7.4 State Values

Class `saref:StateValue` describes the value for a state. The state value is optionally linked the states or states of interest it is a value of (OP `saref:isValueOfState`).

EXAMPLE 1:

```
saref:OnState a saref:StateValue ;
        saref:isValueOfState saref:OnOffState .
    saref:OffState a saref:StateValue ;
        saref:isValueOfState saref:OnOffState.
```

The OP `saref:hasStateValue` links a feature kind, a feature of interest, or a state of interest, to a state value.

EXAMPLE 2:

```
<switch_sdf5ze4fz3#btn1> a saref:StateOfInterest ;
    saref:isStateOf <switch_sdf5ze4fz3> ;
    saref:hasStateValue saref:OnState .
```

NOTE 1: The state values are inherited in the hierarchy of feature kinds. This enables to incrementally construct prototypical descriptions of features of interest.

```
skos:broader saref:hasStateValue ⊑ saref:hasStateValue
```

NOTE 2: A feature of interest does not inherit the state values of its kinds. There may be multiple reasons why the state value of a feature of interest is different from that of its prototypical descriptions. For example, it may be caused by a defect, a deterioration, or a customization.

The OP `saref:isValueOfState` links a state value to the states and states of interest it is a value of.

NOTE 3: A state value about a state of interest is also a state value of its state kinds.
`saref:isValueOfState o saref:hasStateKind ⊑ saref:isValueOfState`

NOTE 4: `saref:hasStateValue` and `saref:isValueOfState` are not inverse properties.

5.8 Functions and functions of interest

5.8.1 Introduction

In SAREF, functions are logical groups of commands that devices support to accomplish their tasks. Function can act upon (OP `saref:actsUpon` and its sub-properties) features, properties, or states. While functions are independent of any devices, functions of interest are functions actually supported by a device.

NOTE: SAREF V3.2.1 introduced the class `saref:FunctionOfInterest` for specifying which command is actually exposed, and which actual property of interest or state of interest it acts upon. Considering `saref:Function` was mostly instantiated in online taxonomies and vocabularies, it was decided to keep this identifier for the generic class (instead of introducing some class like "FunctionKind"), and introduce `saref:FunctionOfInterest`.

Figure 12 illustrates the main classes and properties for describing functions and functions of interest.

SAREF extensions and applications may create, specialize, and categorize functions as specified in ETSI TS 103 548 [5].

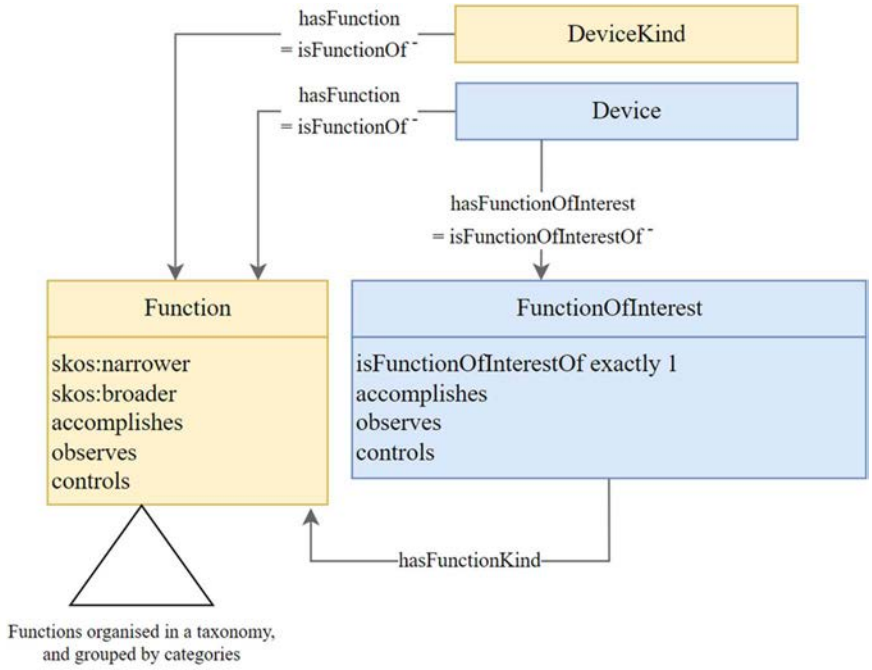


Figure 12: Functions and functions of interest

5.8.2 Functions

An instance of `saref:Function` can apply to different devices.

EXAMPLE 1: To accomplish the task of controlling the light, a smart light switch may have a function for turning on and off the light, and another to set the luminosity of the light.

EXAMPLE 2: To accomplish the task of sensing the temperature, a temperature sensor may have a sensing function.

EXAMPLE 3: To accomplish the task of washing clothes, a washing machine may have a function for washing.

Functions can be organized in a taxonomy using properties `skos:narrower` and `skos:broader`.

NOTE 1: Concepts from existing code lists, vocabularies, and taxonomies, may be used as instances of `saref:Function`.

Kinds of devices can be defined from pre-defined building blocks, based on the functions they have.

EXAMPLE 4:
```
s4abcd:AAHeatPumpDryer a saref:FeatureKind ;
    saref:hasFunction a4abcd:WashingFunction .
```

The OP `saref:hasFunction` may be used to link a feature kind or device to its functions. Its inverse is `saref:isFunctionOf`.

NOTE 2: Feature kinds inherit the functions of their broader feature kinds.
```
skos:broader o saref:hasFunction ⊑ saref:hasFunction
```

NOTE 3: Devices inherit the functions of their device kinds.
```
saref:hasDeviceKind o saref:hasFunction ⊑ saref:hasFunction
```

NOTE 4: No just devices can have functions. OP `saref:hasFunction` can link other entities to functions. For example, software applications may have functions.

5.8.3 Functions of Interest

An instance of `saref:FunctionOfInterest` is supported by exactly one device.

The OP `saref:hasFunctionOfInterest` may be used to link a device to its functions of interest.
Its inverse is `saref:isFunctionOfInterestOf` and is functional.

A function of interest is the function of (OP `saref:isFunctionOfInterestOf`) exactly one device.

NOTE 1: Functions of interest need not always be explicited. It depends on the use case. Typically, functions of interest are useful to specify which command is actually exposed, and which actual property of interest or state of interest it acts upon.

A function of interest can be linked to its kind(s) using OP `saref:hasFunctionKind`.

NOTE 2: Functions of interest inherit broader functions.
```
saref:hasFunctionKind o skos:broader ⊑ saref:hasFunctionKind
```

NOTE 3: Devices inherit the function kinds of their functions of interest.
```
saref:hasFunctionOfInterest o saref:hasFunctionKind ⊑ saref:hasFunction
```

5.9 Commands and commands of interest

5.9.1 Introduction

In SAREF, commands represent the lowest-level directives a device supports and exposes to some network. Commands can act upon (OP `saref:actsUpon` and its sub-properties) features, properties, or states. While commands are independent of any function, commands of interest are commands actually supported by a function of interest.

NOTE: SAREF V3.2.1 introduced the class `saref:CommandOfInterest` for specifying which actual property of interest or state of interest it acts upon. Considering `saref:Command` was mostly instantiated in online taxonomies and vocabularies, it was decided to keep this identifier for the generic class (instead of introducing some class like "CommandKind"), and introduce `saref:CommandOfInterest`.

Figure 13 illustrates the main classes and properties for describing commands and device commands.

SAREF extensions and applications may create, specialize, and categorize commands as specified in ETSI TS 103 548 [5].

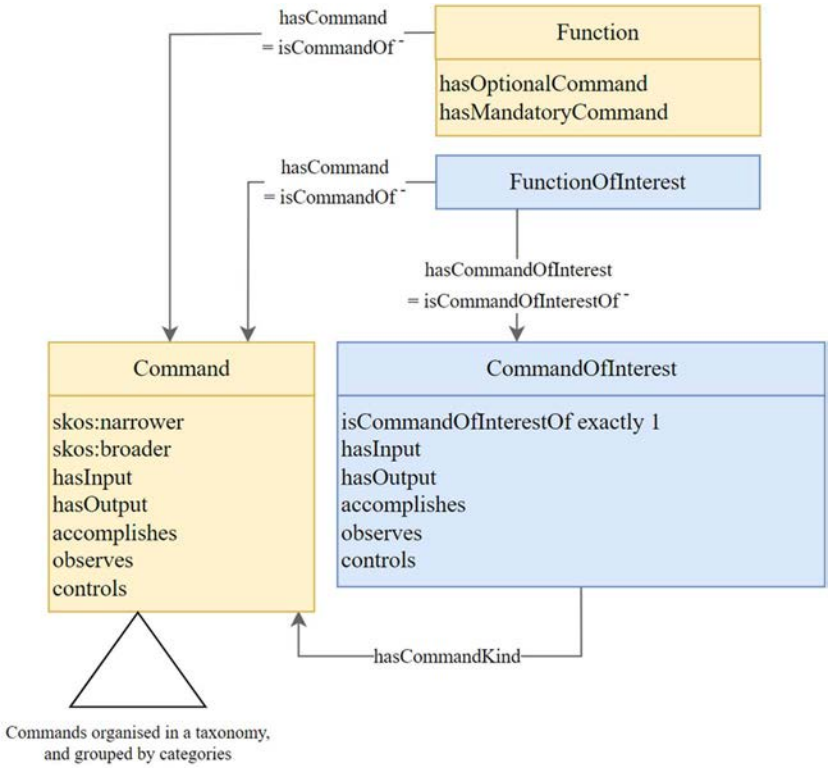


Figure 13: Commands and commands of interest

5.9.2 Commands

An instance of `saref:Command` is independent of any device.

EXAMPLE 1: Observe property, control property, observe state, control state, invoke action, cancel action, turn on or off, change colour, subscribe, publish, etc. are all commands.

Commands can be organized in a taxonomy using OPs `skos:narrower` and `skos:broader`.

Functions can be defined from pre-defined building blocks, based on the commands they have.

The OP `saref:hasCommand` may be used to link a function or function of interest to its commands. Its inverse is `saref:isCommandOf`. SAREF Core defines two sub-properties of `saref:hasCommand`, that only apply to functions:

- `saref:hasMandatoryCommand` for when the command is mandatory to the function;
- `saref:hasOptionalCommand` for when the command is optional to the function.

NOTE 1: Functions inherit the mandatory commands of their broader functions.
`skos:broader o saref:hasMandatoryCommand ⊑ saref:hasMandatoryCommand`

NOTE 2: Functions of interest inherit the mandatory commands of their function kinds.
`saref:hasFunctionKind o saref:hasMandatoryCommand ⊑ saref:hasCommand`

A command may be described in terms of its input parameters using OP `saref:hasInput`. Typically, input parameters are feature kinds, properties, or states.

EXAMPLE 2: Different complementary commands can be defined for controlling a light. Turn on or off the light based on a desired state value, toggle the light status of a specific light, set the luminosity level with a transition time, set the default transition time.

A command may be described in terms of its outputs using OP `saref:hasOutput`. Typically, outputs are properties or states.

EXAMPLE 3: Different complementary commands can be defined for observing a smart home. Observe the temperature will output the indoor temperature property value; observe status of the entry door will output an open/close state.

A command may be described in terms of the properties or states it acts upon, such as observe, or control.

5.9.3 Commands of Interest

A `saref:CommandOfInterest` is a directive actually supported by a device and exposed to some network.

Like for commands, commands of interest may be described in terms of their input parameters, outputs, and of which properties or states they act upon.

The OP `saref:hasCommandOfInterest` may be used to link a function of interest to its command of interest. Its inverse is `saref:isCommandOfInterestOf` and is functional.

A command of interest is the command of (OP `saref:isCommandOfInterestOf`) exactly one function.

NOTE 1: Commands of interest need not always be explicited. It depends on the use case. Typically, commands of interest are useful to specify the actual property of interest or state of interest that is expected as input parameter, output, or that will be observed or controlled.

EXAMPLE 1: The corridor smart light switch supports a command of kind "turn on/off", which controls the state of the outdoor light.

EXAMPLE 2: The smart fridge supports a command of kind "observes temperature", which observes the temperature inside the fridge.

A command of interest can be linked to its kind(s) using OP `saref:hasCommandKind`.

NOTE 2: Commands of interest inherit broader commands.
`saref:hasCommandKind o skos:broader ⊑ saref:hasCommandKind`

NOTE 3: Devices inherit the commands of their commands of interest.
`saref:hasCommandOfInterest o saref:hasCommandKind ⊑ saref:hasCommand`

5.10 Services and Operations

Figure 14 illustrates the main classes and properties for describing services and operations.

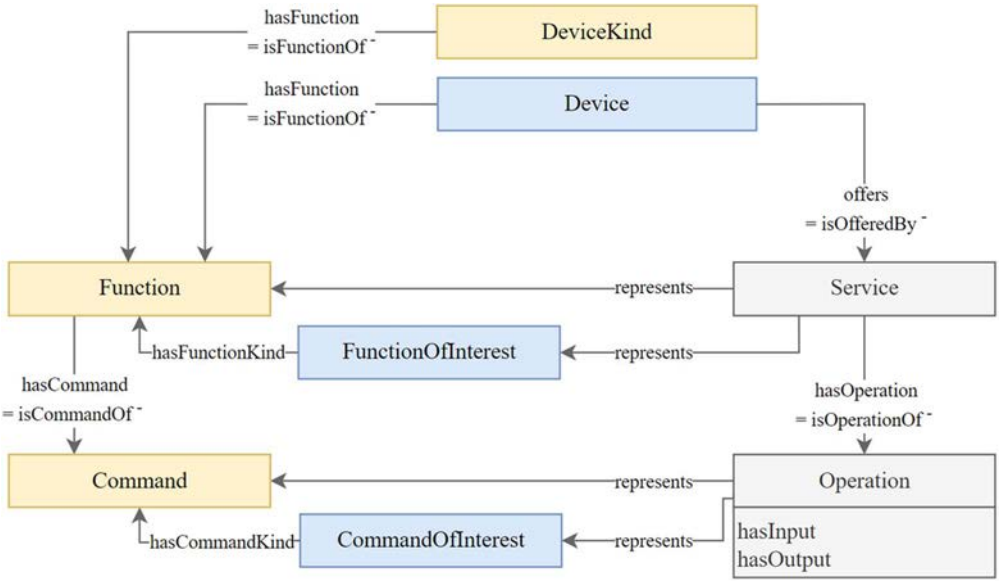


Figure 14: Services and operations

A saref:Service is a digital representation of a function in a network, making it discoverable, registerable and remotely controllable in the network.

OP saref:represents links a service to some function or function of interest it exposes to the network.

A service represents at least one function of interest.

OP saref:offers links a device to a service it exposes to a network. Its inverse if saref:isOfferedBy.

A service is offered by exactly one device.

NOTE 1: Typically, a device connected to a given network offers one service for each of its functions of interest.

EXAMPLE 1: A light switch can offer the service of remotely switching the lights in a home through mobile phone devices that are connected to the local network (ex:SwitchService class). This "remote switching" service represents the ex:OnOffFunction.

A saref:Operation is the means of a service to communicate in a procedure-type manner over the network (i.e. transmit data to/from other devices). It is the -machine interpretable- exposure of a - human understandable - command to a network.

An operation may be described in terms of its inputs and outputs using OP saref:hasInput and saref:hasOutput. Inputs and outputs of operations typically describe the expected schema or shape of network messages.

EXAMPLE 2: To turn on a light, send a CoAP PUT request with CBOR content 0xf5 (true).

OP saref:represents also links an operation to some command or command of interest it exposes to the network.

An operation represents at least one command of interest.

OP saref:hasOperation links a service to its operations. Its inverse is saref:isOperationOf.

An operation belongs to exactly one service.

NOTE 2: Typically, a device connected to a given network offers one service for each of its functions of interest, and each service has one operation per command of interest of the function of interest it represents.

EXAMPLE 3: In the set of operations exposed by a smart light bulb on a given network, one may be dedicated to turn on and off the light and expect a boolean as input. Another one may be dedicated to set the luminosity status and expect a target luminosity level (a byte) and a transition time (encoded on two bytes).

EXAMPLE 4: In the set of operations exposed by a smart washing machine on a given network, one may be dedicated to set the water temperature for the washing cycle, and expected as input a enumerated value. Another one may be dedicated to start, pause, or stop the washing cycle.

NOTE 3: The concept of service is further elaborated in the oneM2M Base Ontology [2], to which the reader is referred in order to model the details of a service that are out of the scope of SAREF.

5.11 Procedure executions and sub-classes

5.11.1 Procedure executions

A saref:ProcedureExecution represents the act of carrying out a procedure.

Figure 15 illustrates the main properties for describing procedure executions.

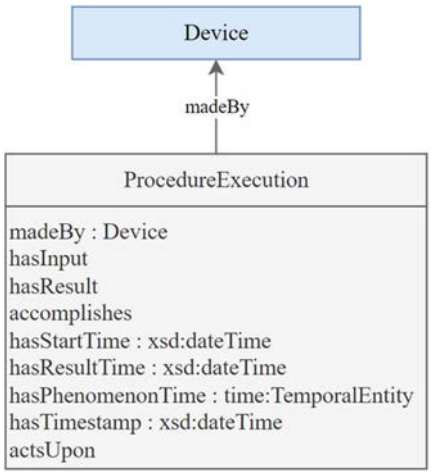


Figure 15: Procedure executions

OP `saref:madeBy` links a procedure execution to the device that made it.

EXAMPLE 1: The observation of the temperature in a room made by a temperature sensor.

OP `saref:madeBy` can link a procedure execution to any entity, not necessarily a tangible device.

EXAMPLE 2: The prediction of the temperature in a room made by a software application.

EXAMPLE 3: A manual measurement of the circumference of a branch of a tree made by a person.

A procedure execution may be linked to its inputs using OP `saref:hasInput`.

A procedure execution may be linked to its result using OP `saref:hasResult`.

DP `saref:hasStartTime` links a procedure execution to the instant of time when it was initiated or tasked, expressed as an `xsd:dateTime` literal.

DP `saref:hasResultTime` links a procedure execution to the instant of time when the procedure is completed, expressed as an `xsd:dateTime` literal.

OP `saref:hasPhenomenonTime` links a procedure execution to the time that the result applies. It may be an interval or an instant, or some other compound temporal entity expressed using OWL Time [8].

When the execution time and the phenomenon time are the same time instants, then DP `saref:hasTimestamp` can be used to simply link a procedure execution to the time of these instants, expressed as an `xsd:dateTime` literal.

Optionally, a procedure execution can act upon (OP `saref:actsUpon`) a feature, property, or state.

5.11.2 Command executions and operation executions

Figure 16 illustrates the main classes and properties for describing command executions and operation executions.

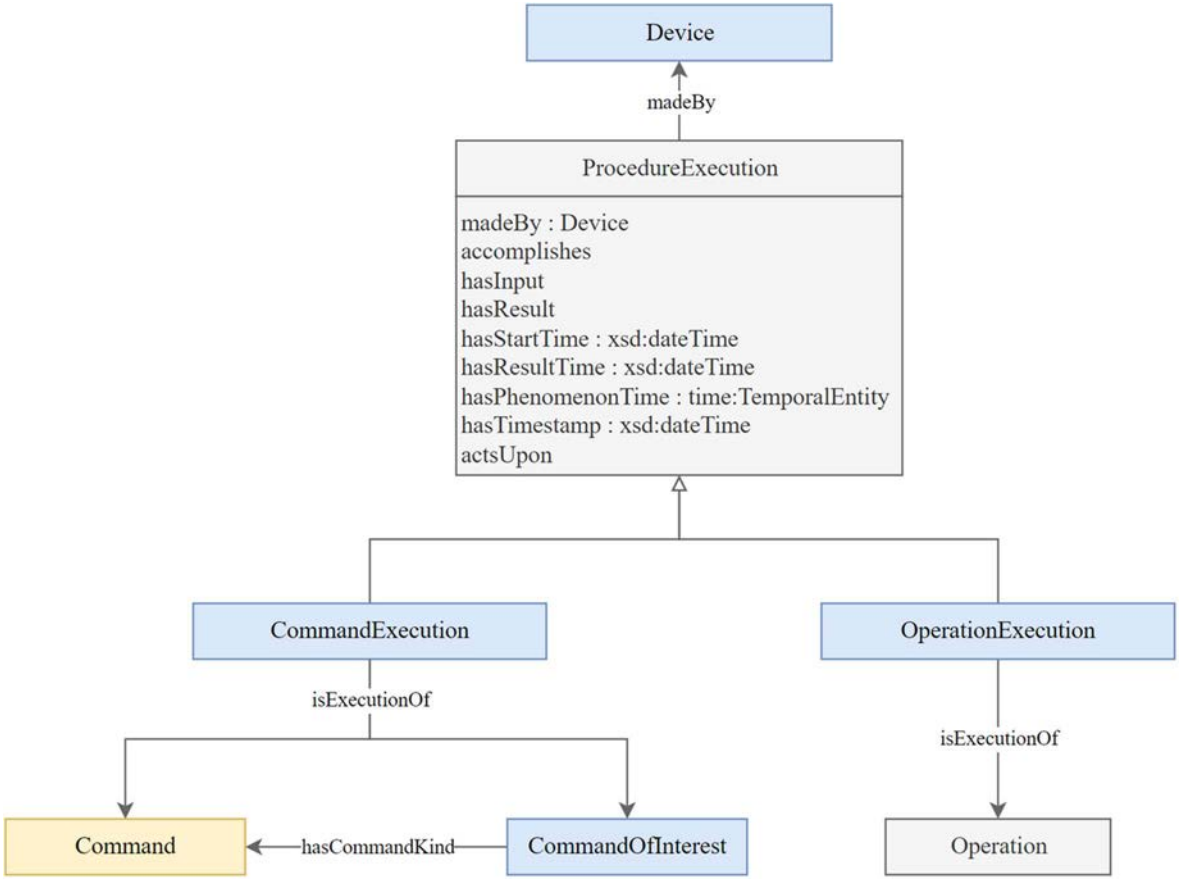


Figure 16: Command executions and operation executions

A `saref:CommandExecution` describes the execution of a command. Typically, its inputs and outputs are human understandable and relate to some feature of interest, such as its state (e.g. `s4abcd:On`), or the value of its temperature (e.g. property value 21,0 °C).

A `saref:OperationExecution` describes the execution of an operation in a network: the - machine interpretable - description of a communication between devices over the network. Typically, its input and result are network messages, that conform to the input and output of the executed operation.

EXAMPLE: A CoAP PUT request with CBOR content 0xf5 (true), a CoAP response with code 2.04 (Changed).

OP `saref:isExecutionOf` links a command execution to the command or command of interest that was executed. It also links an operation execution to the operation that was executed.

NOTE 1: If a command execution is an execution of a command, it is also the execution of its broader commands.
`saref:isExecutionOf o skos:broader ⊑ saref:isExecutionOf`

NOTE 2: If a command execution is an execution of a command of interest, it is also the execution of the command kind of that command of interest.
`saref:isExecutionOf o saref:hasCommandKind ⊑ saref:isExecutionOf`

5.11.3 Observations and Actuations

Figure 17 illustrates the main classes and properties for describing observations and actuations.

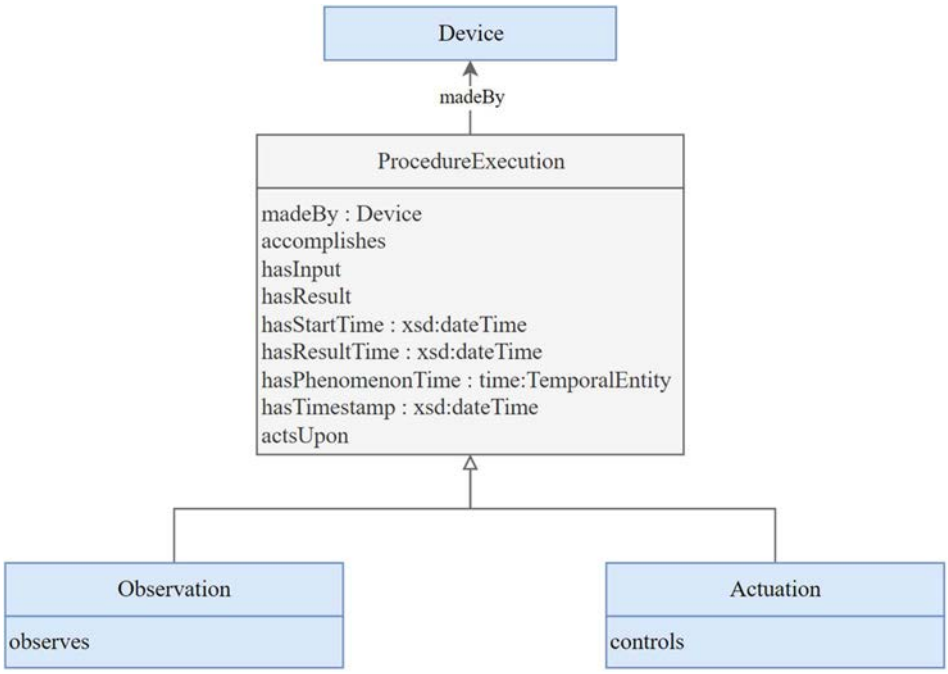


Figure 17: Observations and actuations

A `saref:Observation` is the act of carrying out a procedure to estimate or calculate a value of a property of a feature of interest, or a state of a feature of interest. It links to a sensor to describe what made the observation, and to the observed feature, property, property of interest, state, or state of interest. Typically, its result is a property value or a state value. An observation of a state (OP `saref:observes`) should have a state value as a result (OP `saref:hasResult`). Respectively, an observation of a property should have a property value as a result.

EXAMPLE 1:
```
ex:DTSObservation106 a saref:Observation ;
  saref:madeBy <meter> ;
  saref:hasTimestamp "2023-12-06T21:01:10"^^xsd:dateTime ;
  saref:observes s4watr:Cadmium ;
  saref:hasResult ex:Meter4837QW123Value184 .
ex:Meter4837QW123Value184 a saref:PropertyValue ;
  saref:isValueOfProperty s4watr:Cadmium ;
saref:hasValue 0.005 .
```

EXAMPLE 2:
```
[] a saref:Observation ;
  saref:madeBy <sensor> ;
  saref:hasTimestamp "2023-12-06T21:47:10"^^xsd:dateTime .
  saref:observes <door_zef53ze7> ;
  saref:observes <door_zef53ze7#openclose> ;
  saref:observes saref:OpenClose ;
  saref:hasResult saref:Open .
```

NOTE 1: `saref:Observation` is more general than `saref:Measurement`, as it applies to states in addition to properties. The indirection `saref:hasResult` to a property value improves its semantic correctness. It also improves the alignment with the OGC® and W3C® SOSA/SSN ontology [7]. Therefore, `saref:Measurement` has been deprecated in version V3.2.1 [i.8], and has been deleted in the present document.

A `saref:Actuation` is the act of carrying out a procedure to control the state of the world using an actuator. It links to an actuator to describe what made the actuation, and to the controlled feature, property, property of interest, state, or state of interest. Typically, its input is a property or state value. An actuation of a state (OP `saref:controls`) should have a state value as input (OP `saref:hasInput`). Respectively, an actuation of a property should have a property value as input.

NOTE 2: An observation or an actuation may also be a command execution, if it corresponds to the execution of a directives a device supports and exposes to some network. However, an observation or an actuation should not be an operation execution, as these are intended to be machine interpretable descriptions of network communications.

NOTE 3: It is acceptable that the inputs or results of a command execution are observations or actuations. For example:

- A command execution to aggregate observations will have these observations as input.
- A command execution to retrieve the past five observations will have these observations as output.
- A command execution to plan a series of actuations will have these actuations as input, and potentially also as result if successful.

5.12 Profiles

A device in SAREF can be further characterized by profiles. Figure 18 illustrates the main classes and properties for describing profiles.

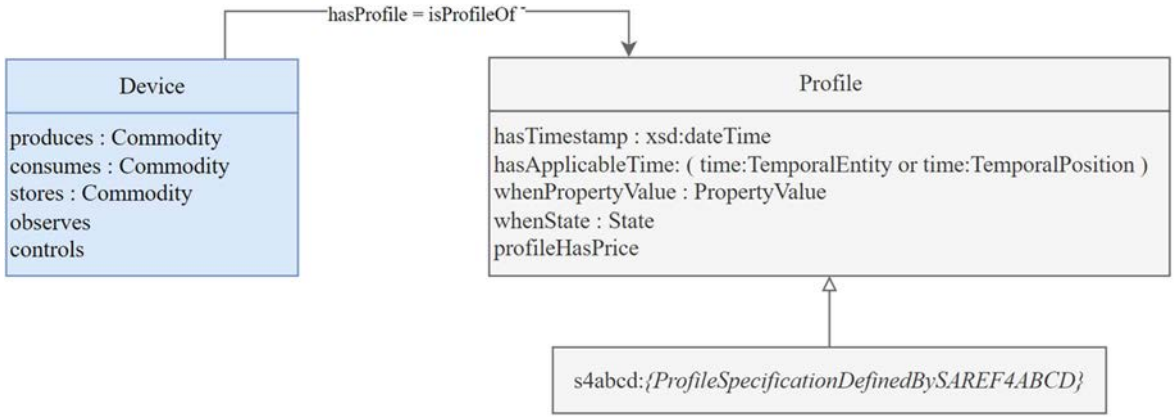


Figure 18: Profiles

A `saref:Profile` describes the money earned (negative values) or paid (positive values) for the use (production or consumption) of a commodity by a device in a certain context.

OP `saref:hasProfile` links a device to its profile. Its inverse is `saref:isProfileOf`. The device should be linked to a certain commodity using OP `saref:isUsedFor` or its sub-properties, and optionally to some property or state using OP `saref:actsUpon` or its sub-properties.

EXAMPLE 1:
```
ex:Device_1 a saref:Device ;
  saref:isUsedFor s4abcd:Electricity ;
  saref:hasProfile ex:FlexibilityProfile ;
  saref:controls <house#temperature> .
ex:FlexibilityProfile a saref:Profile ;
  saref:hasPrice <pricePropertyValue> .
```

The applicable context of a profile can be bound temporally using DP `saref:hasTimestamp` or its subproperties defined by SAREF extensions, or OP `saref:hasApplicableTime` which links to instant or interval or other compound temporal entity expressed using OWL Time [8].

NOTE: `saref:hasApplicableTime` may be applied to other entities, such as functions, commands, or procedure executions.

EXAMPLE 2:
```
ex:FlexibilityProfile
  saref:hasTimestamp "2023-12-15T11:00:00"^^xsd:dateTime .
```

EXAMPLE 3:
```
ex:FlexibilityProfile
   rdfs:comment "applies only on Saturdays"@en ;
   saref:hasApplicableTime [
      a time:DateTimeDescription ;
      time:dayOfWeek time:Saturday ] .
```

The applicable context can be restricted to when the property of a feature of interest has some value (OPs `saref:whenPropertyValue`).

EXAMPLE 4:
```
ex:FlexibilityProfile
   saref:whenPropertyValue [
      a saref:PropertyValue ;
      saref:hasValue 22.0 ;
      saref:isMeasuredIn <http://qudt.org/vocab/unit/DEG_C> ;
      saref:isValueForProperty <house#comfort_temperature_setpoint> ] .
```

The applicable context can be restricted to when a feature of interest has a certain state (OPs `saref:whenState`).

EXAMPLE 5:
```
ex:FlexibilityProfile
   saref:whenState [
      a saref:State ;
      skos:broader s4abcd:ComfortSetpoint ;
      saref:isStateOf <house#temperature_setpoint> ] .
```

OP `saref:profileHasPrice` links a profile to the money earned (negative values) or paid (positive values) for the use (production or consumption) of the commodity by the device.

EXAMPLE 6:
```
ex:FlexibilityProfile
   saref:profileHasPrice [
      a saref:PropertyValue ;
      saref:hasValue 0.2 ;
      saref:isMeasuredIn <http://qudt.org/vocab/currency/EUR> ] .
```

A set of specializations of a Profile is given via the Flexibility Profile defined in the SAREF4ENER extension in ETSI TS 103 410-1 [i.9]. Each Flexibility Profile describes the ways in which a device can regulate its energy consumption and production. Therefore, the Flexibility Profile is a static set of options to choose from and a set of user preferences, instead of a pre-calculated energy usage time series. The details of each Flexibility Profile can be specified using the related extensions.

A specialization of a Profile may additionally relate to other SAREF classes via properties defined in the extensions, including, but not limited to a state, property, property value, function, and command.

EXAMPLE 7:
```
ex:FlexibilityProfile a saref:Profile ;
   s4abcd:{someConditionDefinedBySAREF4ABCD} <conditionSpecification> .
```

5.13 Features of Interest, devices, and spatial objects

An instance may be classified as both `saref:FeatureOfInterest` and `geo:SpatialObject`.

The class `saref:Device` is a sub-class of `geo:Feature` from the GeoSPARQL standard [9], section 6.2.

SAREF application may attach a geometry to a `saref:FeatureOfInterest` or a `saref:Device` using `geo:hasGeometry` or its sub-properties `geo:hasBoundingBox`, `geo:hasCentroid`, `geo:hasDefaultGeometry` [9], section 6.4.

EXAMPLE 1:
```
<etsi_premises> a saref:FeatureOfInterest , geo:Feature ;
   geo:hasCentroid "POINT( 7.052986 43.6169446 )"^^geo:wktLiteral .
```

SAREF application may describe how things are spatially related using different families of topological relations from GeoSPARQL, such as the Simple Features relation family (e.g. `geo:sfWithin`, `geo:sfOverlaps`), the Egenhoer Relation Family (e.g. `geo:ehInside`, `geo:ehOverlap`), the RCC8 Relation Family (e.g. `geo:rcc8tpp`, `geo:rcc8po`) [9], section 7.

EXAMPLE 2: <etsi_premises/athena> a saref:FeatureOfInterest , geo:Feature ;
geo:sfWithin <etsi_premises> .

5.14 Composite Properties and Property Values

The OP `saref:consistsOf` can link a composite property, property of interest, or property value to its individual components.

NOTE 1: A feature kind or feature of interest has the individual components of its composite properties.
`saref:hasProperty o saref:consistsOf ⊑ saref:hasProperty`

Composite Properties can consist of feature kinds, properties, and states. Figure 19 illustrates the local restrictions of `saref:consistsOf` on the class `saref:Property`.

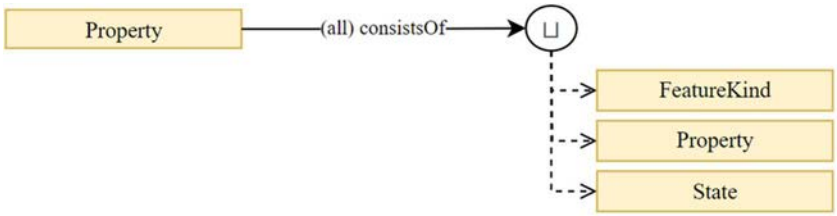


Figure 19: Local restrictions of `saref:consistsOf` on the class of Properties

EXAMPLE 1: A 2D position property observed by a position sensor consists of x and y coordinates.

```
s4abcd:2DPosition_xCoordinate a saref:Property .
s4abcd:2DPosition_yCoordinate a saref:Property .
s4abcd:2DPosition a saref:Property ;
   saref:consistsOf s4abcd:2DPosition_xCoordinate ;
   saref:consistsOf s4abcd:2DPosition_yCoordinate .
```

EXAMPLE 2: A composite presence property observed by a smart presence sensor consists of persons, an occupied/not occupied state, and a temperature property.

```
ex:Person a saref:FeatureKind .
s4abcd:OccupiedOrNotOccupied a saref:State .
s4abcd:Temperature a saref:Property .
s4abcd:CompositePresence a saref:Property ;
   saref:consistsOf ex:Person ;
   saref:consistsOf s4abcd:OccupiedOrNotOccupied ;
   saref:consistsOf s4abcd:Temperature .
```

In addition, properties of interest can be composed of features of interest, properties of interest and states of interest. Figure 20 illustrates the local restrictions of `saref:consistsOf` on the class `saref:PropertyOfInterest`.

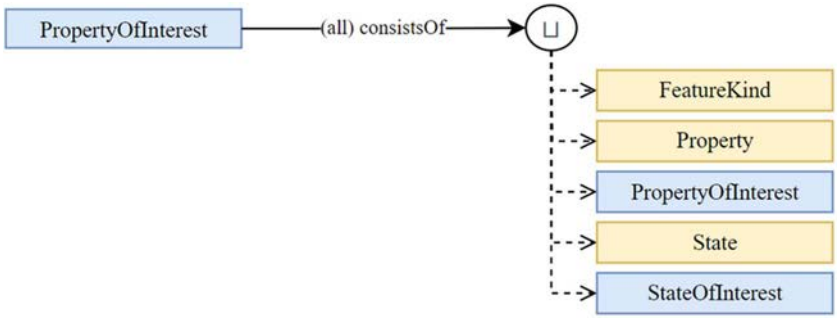


Figure 20: Local restrictions of `saref:consistsOf` on the class of Properties of Interest

NOTE 2: When composed of a property of interest, a property of interest is also composed of the kinds of this property of interest.
`saref:consistsOf o saref:hasPropertyKind ⊑ saref:consistsOf`

NOTE 3: When composed of a state of interest, a property of interest is also composed of the kinds of this state of interest.

`saref:consistsOf o saref:hasStateKind ⊑ saref:consistsOf`

In addition, property values can be composed of property values, and state values. Figure 21 illustrates the local restrictions of `saref:consistsOf` on the class `saref:PropertyValue`.

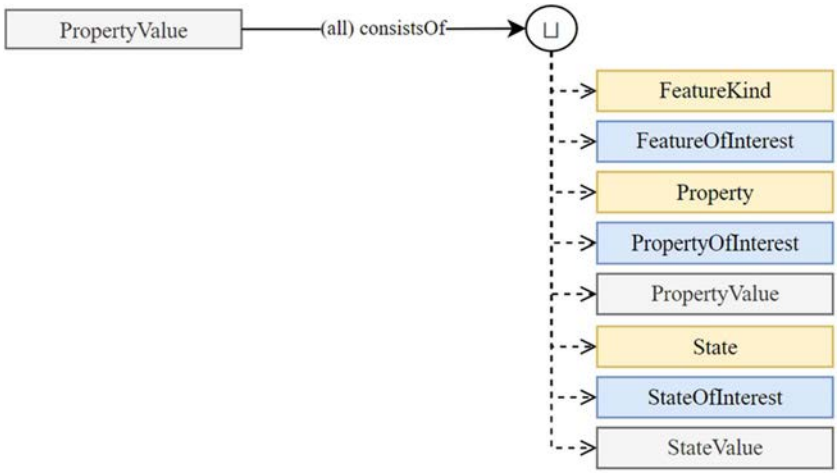


Figure 21: Local restrictions of `saref:consistsOf` on the class of Property Values

EXAMPLE 3: A 2D position value as the result of the observation made by a position sensor.

```
<position_sensor> a s4abcd:PositionSensor ;
   saref:madeExecution <observation> .
<observation> a saref:Observation ;
  saref:observes s4abcd:2DPosition, <room> ;
  saref:hasResult <value> .
<value> a saref:PropertyValue ;
   saref:consistsOf <value_x> , <value_y> .
<value_x> a saref:PropertyValue ;
   saref:isValueOfProperty s4abcd:2DPosition_xCoordinate ;
   saref:hasValue 2 .
<value_y> a saref:PropertyValue ;
   saref:isValueOfProperty s4abcd:2DPosition_yCoordinate ;
   saref:hasValue 3 .
```

EXAMPLE 4: A composite presence value as the result of the observation made by a smart presence sensor.

```
<smart_camera> a s4abcd:SmartPresenceSensor ;
   saref:madeExecution <observation> .
<observation> a saref:Observation ;
  saref:observes s4abcd:CompositePresence, <room> ;
  saref:hasResult <composite_presence_value> .
<composite_presence_value> a saref:PropertyValue ;
   saref:consistsOf <person1>, <person2> ;
   saref:consistsOf s4abcd:Occupied ;
   saref:consistsOf <temperature_value> .
s4abcd:Occupied a saref:StateValue ;
   saref:isValueOfState s4abcd:OccupiedOrNotOccupied ;
<temperature_value> a saref:PropertyValue ;
   saref:hasValue 22.7 ;
   saref:isMeasuredIn <https://qudt.org/2.1/vocab/unit/DEG_C> .
```

NOTE 4: When composed of a property value, a property value is also composed of that value's property.

`saref:consistsOf o saref:isValueOfProperty ⊑ saref:consistsOf`

NOTE 5: When composed of a state value, a property value is also composed of that value's state.

`saref:consistsOf o saref:isValueOfState ⊑ saref:consistsOf`

NOTE 6: When composed of a feature of interest, a property value is also composed of the kind of that feature of interest.

`saref:consistsOf o saref:hasFeatureKind ⊑ saref:consistsOf`

6 Mapping between SAREF and oneM2M Base Ontology

6.1 Introduction

In ETSI TS 118 112 [2], oneM2M has created a base ontology that describes key classes, relations and properties that are relevant for enabling semantic functionalities within oneM2M systems, as well as enabling interoperability between applications and interworking with existing non-oneM2M technologies. The approach is that given a semantic description of instances according to the oneM2M Base Ontology, a oneM2M resource structure can be automatically created.

General oneM2M resources are created for those ontology instances that are related to functions, e.g. creating an Application Entity resource for a device like a washing machine and creating containers, e.g. for storing the status of the washing machine. They will enable application interactions and thus concern dynamic aspects. Other, more static aspects like the manufacturer of a device will be stored in special semantic descriptor resources that are attached to general oneM2M resources, e.g. there is oneM2M resource representing a device which has a semantic descriptor resource attached that contains semantic information related to the device, e.g. the manufacturer. Such a semantic descriptor resource also contains information concerning the relation to other resources, e.g. operations that can be executed.

A two-step approach for the mapping of SAREF instances to oneM2M resources is used. In the first step, key SAREF classes are mapped to oneM2M Base Ontology classes by defining relations between the SAREF and the oneM2M classes. In the second step, instances modelled according to those SAREF classes for which such a definition exists, are also automatically modelled according to the corresponding oneM2M Base Ontology classes. In the second step, the oneM2M instantiation rules are applied to those instances of SAREF classes that are derived from oneM2M classes. Not all SAREF classes can be mapped to base ontology classes as SAREF models certain aspects that are closely related to the smart applications domain and the base ontology is meant to be agnostic to specific application domains. If no equivalent oneM2M Base Ontology classes exist, e.g. for `saref:Commodity`, the respective SAREF instances are stored together with SAREF instances with which they are connected through an object property and which are mapped to the oneM2M Base Ontology.

6.2 Mapping between SAREF and oneM2M Base Ontology

Figure 22 shows the mapping between SAREF and the oneM2M Base Ontology. Relationships based on `owl:equivalentClass` and `owl:equivalentProperty` are considered to link the key classes and object properties of SAREF and the oneM2M Base Ontology. These are needed to be able to apply the oneM2M instantiation rules to the semantic description of entities that are described according to SAREF.

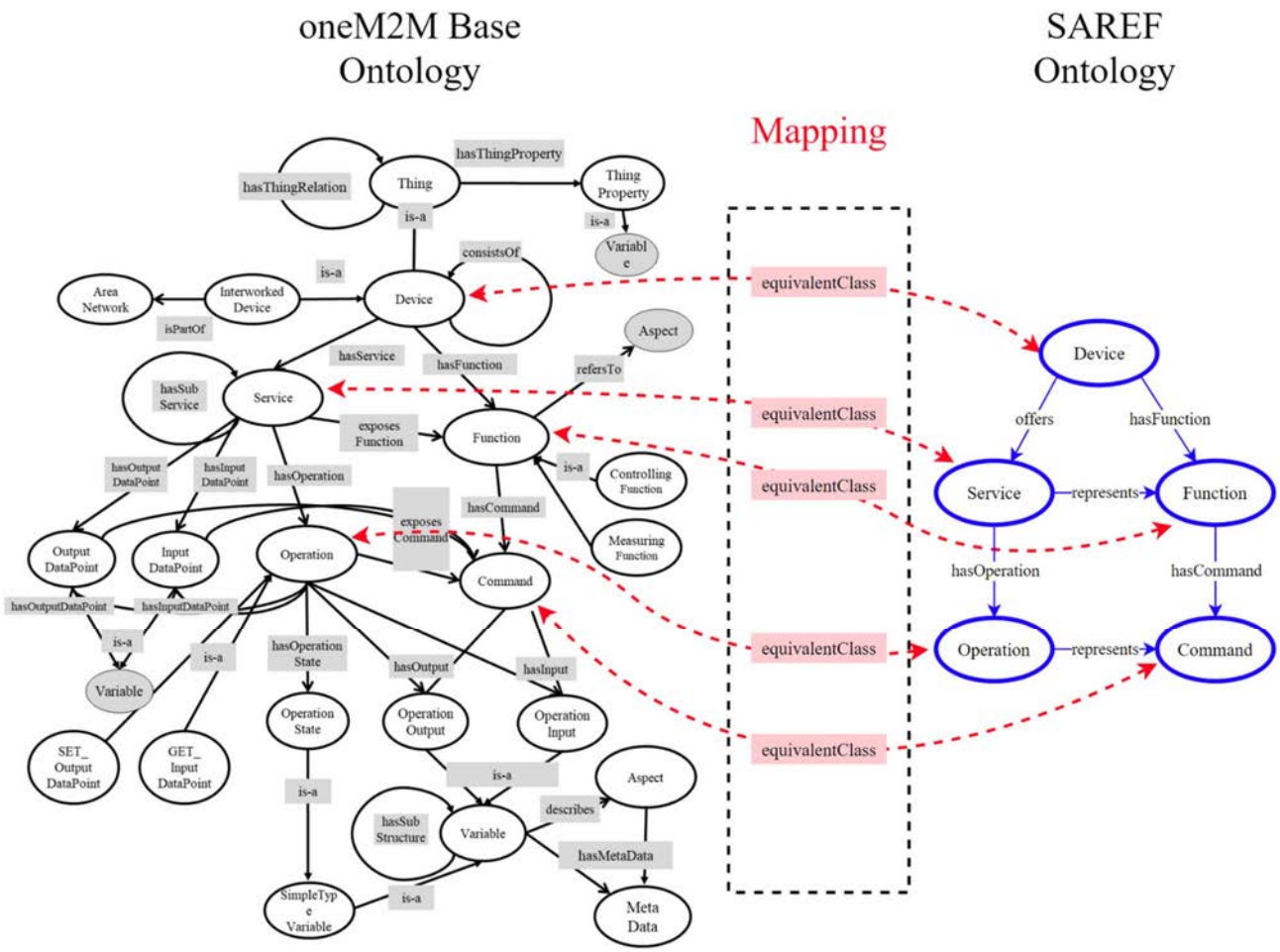


Figure 22: Mapping between SAREF and the oneM2M Base Ontology

Table 2 shows which SAREF class is mapped to which oneM2M class (and vice-versa). As a result, all oneM2M instantiation rules defined for the oneM2M class can also be applied to the instance of the respective SAREF class, and oneM2M instances can be discovered from SAREF when querying for devices, functions, commands and services.

Table 2: Mapping of SAREF classes and oneM2M Base Ontology classes

SAREF	Mapping	oneM2M
saref:Device	Equivalent class of	oneM2M:Device
saref:Service	Equivalent class of	oneM2M:Service
saref:Function	Equivalent class of	oneM2M:Function
saref:Command	Equivalent class of	oneM2M:Command
saref:Service	Equivalent class of	oneM2M:Service
saref:Operation	Equivalent class of	oneM2M:Operation

Table 3 shows which SAREF object property is mapped to which oneM2M object property.

Table 3: Mapping of SAREF object properties and oneM2M Base Ontology object properties

SAREF	Mapping	oneM2M
saref:offers	Equivalent property of	oneM2M:hasService
saref:hasFunction	Equivalent property of	oneM2M:hasFunction
saref:hasCommand	Equivalent property of	oneM2M:hasCommand
saref:represents	Super property of	oneM2M:exposesFunction
saref:consistsOf	Super property of	oneM2M:consistsOf
saref:hasInput	Super property of	oneM2M:hasInput
saref:hasOutput	Super property of	oneM2M:hasOutput

6.3 Instantiation Rules for Creating the oneM2M Resource Structure

The Smart Applications oneM2M Mapping shall follow the instantiation rules defined in clause 7 of ETSI TS 118 112 [2].

Annex A (informative): Change history

Date	Version	Information about changes
March 2017	V2.1.1	• The classes and properties related to how to represent devices in building spaces (such as the saref:BuildingSpace class, saref:BuildingObject class and saref:isLocatedIn property) have been removed from SAREF and incorporated into the SAREF4BLDG extension related to buildings [i.4], including the reuse of the W3C® WGS84 geo positioning vocabulary. • The saref:DeviceCategory and saref:FunctionCategory classes have been removed. Instead, the hierarchy of device categories has been implemented directly as subclasses of the saref:Device class. • The information specific for energy efficiency has been moved to the SAREF4ENER extension. For example, the saref:Profile class has been redefined to accommodate only the properties that are general enough for any type of profile, not only for energy and power. Details on how to specifically model a power profile can be found in the SAREF4ENER extension [i.2]. • The subclasses of the saref:Energy class have been removed (i.e. Average Energy, Maximum Energy, Minimum Energy, Total Energy, HVAC Energy, Hot Water Energy and Lighting Energy). • The saref:Property class has been split into two classes (saref:Property and saref:Measurement), as it is done in the SAREF4ENVI extension in [i.3], in order to properly accommodate the distinction between the concept of property (an observable quality of something) and the concept of measurement (a concrete value observed for a property). • Too restrictive cardinality restrictions have been revised, sometimes making them optional rather than mandatory (better to make properties mandatory in the extensions that specialize SAREF for a specific purpose, rather than restricting SAREF, whose purpose is more general). For example, definitions of units of measurement using an enumeration (owl:oneOf) were too restrictive because they did not allow using other units than those enumerated. Therefore, the individuals of each class are still there, but the owl:oneOf enumeration has been removed. • The saref:UnitOfMeasure subclasses use individuals from the OM ontology for unit of measures (http://www.wurvoc.org/vocabularies/om-1.6/), but this is not the only solution. It has been clarified in the comments that the OM ontology is an example, but other ontologies can be used. • The global restrictions (rdfs:domain and rdfs:range) in object properties have been completely removed not to hinder interoperability. Object properties are now only restricted locally in the classes. • The subclasses of saref:Task have been transformed in individuals. • The saref:hasTask property has been removed from the saref:Profile class to resolve an ambiguity with the saref:Task class.
January 2020	V3.1.1	NOTE: In some cases, references have been added to indicate the issue in the ETSI Forge (https://saref.etsi.org/sources/saref-core/issues) where the change was discussed and implemented. • Added the saref:FeatureOfInterest class and the properties used to relate it to saref:Measurement (saref:hasMeasurement and saref:isMeasurementOf) and to saref:Property (saref:hasProperty and saref:isPropertyOf) (#40). • Added the saref:measurementMadeBy property as inverse of saref:makesMeasurement (#40). • The saref:relatesToProperty and saref:relatesToMeasurement properties are now inverse of each other (#21). • The range of saref:hasValue, which was defined as xsd:float, has been removed to support other datatypes for measurements (#24). • The range of datatype properties, which was defined as xsd:string, has been removed in order to support strings with language tags (rdf:langString). This affects saref:hasDescription, saref:hasManufacturer, saref:hasModel and saref:hasName, which now have implicitly a range of rdfs:Literal (#1). • The saref:hasName property has been removed and the use of rdfs:label is recommended (#3). • The saref:hasDescription property has been deprecated and the use of rdfs:comment is recommended (#6).

Date	Version	Information about changes
		• Too restrictive restrictions have been removed and in some cases the ontology documentation has been updated to reflect this: a saref:Command having at most one saref:hasDescription (#2); a saref:Task being accomplished by at least one saref:Device (#14); a saref:Device having a typical consumption of only energy or power (#19); a saref:Commodity being measured in units of measure (#20); restrictions on the values of saref:accomplishes that only covered specific use cases (#13); universal restrictions on the saref:actsUpon property have been changed to existential ones (#31); restrictions on the commands that a saref:Function may have, since there could be others not included in the defined enumerations (#28); restrictive documentation of saref:EventFunction and saref:SmokeSensor (#18). • Bugs have been fixed: some devices (saref:DoorSwitch, saref:EnergyMeter, saref:LigthSwitch, saref:SmokeSensor, and saref:TemperatureSensor) were defined as subclass of a device and at the same time as consisting of such device (#11); a saref:LightSwitch controls light and does not measure light, since it is an actuator (#12); incorrect documentation of saref:OnState and saref:OffState (#39). • The saref:BuildingRelated, saref:EnergyRelated and saref:FunctionRelated classes have been removed, since the different extensions already classify the device types (#3). • The instances of the subclasses of saref:Command have been removed, since they represent examples (#32). • The instances of saref:UnitOfMeasure have been moved to examples (#22). • The classes related to the SAREF4ENER extension have been removed: saref:Generator, saref:Storage, saref:Load and saref:EnergyMeter (#3). • Some instances of saref:Device have been moved to examples: saref:WashingMachine, saref:LightingDevice, saref:MicroRenewable, saref:Multimedia, and saref:Network (#23). • Added domain and range axioms wherever the definition of a property unambiguously identifies the domain or the range of the property (#15). • The OWL Time ontology is not imported anymore (#36). • The documentation of some ontology terms has been updated for clarification (#5). • Language tags have been added to all labels and comments (#7).
January 2024	V3.2.1	NOTE: Rationale and decisions for many of the changes in V3.2.1 are available in issues in project https://saref.etsi.org/sources/saref-core/issues. • Definition of saref:FeatureOfInterest is clarified, not only properties are measured, but properties and states can be observed and controlled. • Definition of saref:FeatureOfInterest: Clarify it is a specific real world entity. • Definition of saref:FeatureOfInterest: allValuesFrom axioms are unnecessary and have been deleted. • Introduced class saref:FeatureKind • Use SKOS to organize generic classes in taxonomies. • Introduce hasClassKind for Class is Feature, Device, Property, State, Function, Command. • Add local restrictions on consistsOf on feature of interest • Defined domain of saref:hasManufacturer to union of feature kind and feature of interest. • Defined domain of saref:hasModel to union of feature kind and feature of interest. • Most axioms on saref:Device were useless and have been deleted. • definition of saref:Device clarifies it is a real world entity. • saref:Device is explicitly a subclass of saref:FeatureOfInterest and s4syst:System. • saref:actsUpon now holds not only between a command and a state. It generalizes actsUpon, observes, controls. Domain and range are unions of several classes. • Introduced property saref:isActedUponBy. • Introduced property saref:observes, as sub-property of saref:actsUpon. and generalization of measures. • Introduced property saref:controls, as sub-property of saref:actsUpon. and generalization of saref:controlsProperty. • Introduced property saref:isControlledBy. • saref:measuresProperty is deprecated: will be deleted in the next release of SAREF • saref:isMeasuredByDevice is deprecated: will be deleted in the next release of SAREF • saref:isControlledByDevice is deprecated: will be deleted in the next release of SAREF • saref:controlsProperty is deprecated: will be deleted in the next release of SAREF • Definition of saref:Appliance is grounded on tasks and commodities.

Date	Version	Information about changes
		• Definition of Sensor is simplified and grounded on saref:observes. Local restriction on saref:hasFunction is deleted. • Definition of Actuator is simplified and grounded on saref:controls. Local restriction on saref:hasFunction is deleted. • Definition of Meter is simplified and grounded on saref:observes. Local restriction on saref:hasFunction is deleted. • Deprecated saref:HVAC for it is only an example. • Deprecated saref:Switch for it is only an example. • Deprecated saref:DoorSwitch for it is only an example. • Deprecated saref:LightSwitch for it is only an example. • Deprecated saref:SmokeSensor for it is only an example. • Deprecated saref:TemperatureSensor for it is only an example. • The following tasks are deprecated, and may be deleted in the next release of SAREF: saref:Cleaning, saref:Comfort, saref:Drying, saref:EnergyEfficiency, saref:Entertainment, saref:Lighting, saref:MeterReading, saref:Safety, saref:Washing, saref:WellBeing. • Definition of saref:Commodity is simplified. • saref:Electricity is deprecated: will be defined as an instance of saref:Commodity in the next release of SAREF • saref:Gas is deprecated: will be defined as an instance of saref:Commodity in the next release of SAREF • saref:Coal is deprecated: will be defined as an instance of saref:Commodity in the next release of SAREF • saref:Water is deprecated: will be defined as an instance of saref:Commodity in the next release of SAREF • New category of commodities saref:EnergyCommodity • New category of commodities saref:NaturalResourceCommodity • Introduced saref:consumes, saref:isConsumedBy, saref:produces, saref:isProducedBy, saref:stores, saref:isStoredBy. • The definition and scope of usage for saref:Property is clarified. Axiomatization is simplified. • applicability of saref:hasProperty is relaxed to also saref:FeatureKind • Introduced saref:PropertyOfInterest to solve the ambiguity between generic vs specific properties. • Introduced saref:hasPropertyOfInterest and saref:isPropertyOfInterestOf. Cannot use saref:hasProperty due to the combination of cardinality restrictions and sub-property chain axioms makes the ontology exist the OWL 2 DL profile. • Introduced saref:PropertyValue to simplify the association of a measure to a property, without always needing it to be a measurement made by a device. • Clarified definition of saref:hasValue. • Relaxed domain of saref:isMeasuredIn so it can apply to saref:PropertyValue as well as saref:Measurement. The new range will be enforced in the next release of SAREF. • Introduced saref:hasPropertyValue. Introduced saref:isValueOfProperty.
February 2025	V4.1.1	• Deprecated classes and properties have been deleted • Reworded definition of saref:Sensor and saref:Meter • saref:hasValue is now optional on saref:PropertyValue • improved parallel between Property and State descriptions • introduced saref:StateValue • Removed examples of commodity categories • Added notes and examples in multiple sections • Added clause 5.14 about composite properties • Reworked figures throughout the document • Added migration guide from V3.1.1 to V4.1.1 • Updated the oneM2M base ontology to SAREF mapping • Added principle on generic versus specific entities distinction • Added references to EN 303 760

Annex B (informative): Migration guide from SAREF V3.1.1 to SAREF V4.1.1

The present document specifies SAREF V4.1.1, which contains significative changes compared to the previous major release SAREF V3.1.1 [i.7]. One of these changes concerns the `saref:Measurement` class, which in SAREF V4.1.1 has been deleted and replaced by the `saref:Observation` class and the modelling of related properties. This annex provides support for the stakeholders that have applied the `saref:Measurement` modelling pattern in the past according to SAREF V3.1.1, to migrate their existing solutions to SAREF V4.1.1 as specified in the present document. This support is provided by making use of a running example and illustrating, step by step, how to perform the transformation.

NOTE: In order to avoid confusion between graph snippets written with different versions of SAREF, this annex will use the `saref3:` prefix to indicate classes and properties that belong to SAREF V3.1.1 [i.7], and the `saref4:` prefix to indicate classes and properties that belong to SAREF V4.1.1. The `ex:` prefix is used to denote an example instance of SAREF. The `abcd:` prefix is used to indicate a class or OP that does not exist in SAREF and is taken from an external ontology

The following example shows a Turtle snippet about measurements that will be used as starting point for the transformation. It uses SAREF V3.1.1, as indicated by the `saref3:` prefix, and contains two measurements (`ex:meas_dev1_temp0` and `ex:meas_dev1_bat0`) made by the same device (`ex:dev1`) about two different properties, namely the temperature of the living room where the device is located (`ex:prop_dev1_temp_lr`) and the device's own battery level (`ex:prop_dev1_bat_lr`).

EXAMPLE 1:

```
ex:dev1 a saref3:Device ;
   abcd:hasDeviceType "Heat series"^^xsd:string ;
   saref3:hasManufacturer "ACME"^^xsd:string ;
   saref3:hasModel "5000"^^xsd:string ;
   saref3:makesMeasurement ex:meas_dev1_temp0 , ex:meas_dev1_bat0 ;
   saref3:measuresProperty ex:prop_dev1_temp_lr , ex:prop_dev1_bat_lr ;
   s4ener:serialNumber "S4R3F411EXA"^^xsd:string .

ex:foi_living_room a saref3:FeatureOfInterest ;
   saref3:hasProperty ex:prop_dev1_temp_lr , ex:prop_dev1_bat_lr ;
   saref3:hasMeasurement ex:meas_dev1_temp0 , ex:meas_dev1_bat0 .

ex:prop_dev1_temp_lr a saref3:Temperature ;
   saref3:isMeasuredByDevice ex:dev1 ;
   saref3:isPropertyOf ex:foi_living_room ;
   saref3:relatesToMeasurement ex:meas_dev1_temp0 .

ex:prop_dev1_bat_lr a saref3:Energy ;
   saref3:isMeasuredByDevice ex:dev1 ;
   saref3:isPropertyOf ex:foi_living_room ;
   saref3:relatesToMeasurement ex:meas_dev1_bat0 .

ex:meas_dev1_temp0 a saref3:Measurement ;
   saref3:hasTimestamp "2025-01-27T09:03:00"^^xsd:dateTime ;
   saref3:hasValue "22.5"^^xsd:float ;
   saref3:isMeasuredIn om:degreeCelsius ;
   saref3:relatesToProperty ex:prop_dev1_temp_lr ;
   saref3:isMeasurementOf ex:foi_living_room ;
   saref3:measurementMadeBy ex:dev1 .

ex:meas_dev1_bat0 a saref3:Measurement ;
   saref3:hasTimestamp "2025-01-27T09:03:00"^^xsd:dateTime ;
   saref3:hasValue "42"^^xsd:float ;
   saref3:isMeasuredIn om:Percent ;
   saref3:relatesToProperty ex:prop_dev1_bat_lr ;
   saref3:isMeasurementOf ex:foi_living_room .
```

- Device

This step focuses on the initial part of EXAMPLE 1, which provides information about the device, the measurements it makes, and the measured properties. EXAMPLE 2a shows this information in SAREF V3.1.1 (as extracted from EXAMPLE 1), while EXAMPLE 2b shows the target code in SAREF V4.1.1, after the migration.

EXAMPLE 2a:
```
ex:dev1 a saref3:Device ;
   saref3:makesMeasurement ex:meas_dev1_temp0 , ex:meas_dev1_bat0 ;
   saref3:measuresProperty ex:prop_dev1_temp_lr , ex:prop_dev1_bat_lr ;
   s4ener:serialNumber "S4R3F411EXA"^^xsd:string .
```

EXAMPLE 2b:
```
ex:dev1 a saref4:Device ;
   saref4:madeExecution ex:obs_dev1_temp0 , ex:obs_dev1_bat0 ;
   saref4:observes ex:prop_dev1_temp_lr , ex:prop_dev1_bat_lr ;
   s4ener:serialNumber "S4R3F411EXA"^^xsd:string ;
   saref4:hasDeviceKind ex:dk_heat5000 .
```

In the migration, the `saref3:Measurement` class and the related OP `saref3:makesMeasurement` are deleted and replaced in SAREF V4.1.1 by the `saref4:Observation` class and the related OP `saref4:madeExecution`. The OP `saref3:measuresProperty` is deleted and replaced by `saref4:observes`. In SAREF V4.1.1, the OP `saref4:hasDeviceKind` is a newly added relation that allows to provide further information about general categories of devices (`saref4:DeviceKind`), in a catalogue-like fashion. Therefore, the EXAMPLE 2b presents one additional triple compared to EXAMPLE 2a, namely the OP `saref4:hasDeviceKind` that links `ex:dev1` to an instance of the `saref4:DeviceKind` class (`ex:dk_heat5000`).

- Device Kind

With the introduction of the new class `saref4:DeviceKind` in SAREF V4.1.1, the OPs relating to the general device's information in SAREF V3.1.1 (such as the device's manufacturer and model shown in EXAMPLE 3a), are moved to the `saref4:DeviceKind` class, which describes the general information relevant to this *kind* of device (as in a catalogue). EXAMPLE 3b shows the target code of this migration, which also contains the additional OP `saref4:observes`, which relates the `saref4:DeviceKind` to the properties it observes (`om:Temperature` and `abcd:BatteryLevel`).

EXAMPLE 3a:
```
ex:dev1 a saref3:Device ;
   abcd:hasDeviceType "Heat series"^^xsd:string ;
   saref3:hasManufacturer "ACME"^^xsd:string ;
   saref3:hasModel "5000"^^xsd:string ;
```

EXAMPLE 3b:
```
ex:dk_heat5000 a saref4:DeviceKind ;
   abcd:hasDeviceType "Heat series"^^xsd:string ;
   saref4:hasManufacturer "ACME"^^xsd:string ;
   saref4:hasModel "5000"^^xsd:string ;
   saref4:observes om:Temperature , abcd:BatteryLevel .
```

- Feature of Interest and Feature Kind

This step focuses on the part of EXAMPLE 1 that provides information about the context of the device, namely the room in which the device is located (modelled as a feature of interest) and its properties. EXAMPLE 4a shows this information in SAREF V3.1.1 (as extracted from EXAMPLE 1), while EXAMPLE 4b shows the target code in SAREF V4.1.1, after the migration.

EXAMPLE 4a:
```
ex:foi_living_room a saref3:FeatureOfInterest ;
   saref3:hasProperty ex:prop_dev1_temp_lr , ex:prop_dev1_bat_lr ;
   saref3:hasMeasurement ex:meas_dev1_temp0 , ex:meas_dev1_bat0 .
```

EXAMPLE 4b:
```
ex:foi_living_room a saref4:FeatureOfInterest ;
   saref4:hasPropertyOfInterest ex:prop_dev1_temp_lr ;
   saref4:isObservedBy ex:obs_dev1_temp0 ;
   saref4:hasFeatureKind ex:fk_room .

ex:fk_room a saref4:FeatureKind ;
   saref4:hasProperty om:Temperature .
```

```
ex:dev1 saref4:hasPropertyOfInterest ex:prop_dev1_bat ;
   saref4:isObservedBy ex:obs_dev1_bat0 ;
   saref4:hasDeviceKind ex:dk_heat5000 .

ex:dk_heat5000 saref4:hasProperty abcd:BatteryLevel .
```

This example shows that the definition of the living room as feature of interest, is the same in SAREF V3.1.1 and SAREF V4.1.1. However, the subsequent OP `saref3:hasProperty` (in EXAMPLE 4a) is replaced by `saref4:hasPropertyOfInterest` (in EXAMPLE 4b), and `saref3:hasMeasurement` (in EXAMPLE 4a) is replaced by `saref4:isObservedBy` (in EXAMPLE 4b). Additionally, EXAMPLE 4b defines the OP `saref4:hasFeatureKind` to indicate that the considered feature of interest (`ex:foi_living_room`) belongs to a general category (`ex:fk_room`), which is defined as a `saref4:FeatureKind` that has a generic property (`om:Temperature`). In other words, in SAREF V4.1.1 it is possible to generalize that a living room is a type of room (i.e. a feature kind) in which the temperature property can be observed, and this generalization can be reused across different applications that observe the temperature of rooms. EXAMPLE 4b further shows that the device under consideration (`ex:dev1`) has a property of interest (`ex:prop_dev1_bat`) that allows it to make an observation about itself (`ex:obs_dev1_bat0`), i.e. the device becomes its own feature of interest and can measure its own battery level.

- Property and Property of Interest

This step focuses on the part of the running example that describes the properties that are measured and observed, namely the temperature of the room and the battery level of the device. EXAMPLE 5a shows this information in SAREF V3.1.1 (as extracted from EXAMPLE 1), while EXAMPLE 5b shows the target code in SAREF V4.1.1, after the migration.

EXAMPLE 5a:

```
ex:prop_dev1_temp_lr a saref3:Temperature ;
   saref3:isMeasuredByDevice ex:dev1 ;
   saref3:isPropertyOf ex:foi_living_room ;
   saref3:relatesToMeasurement ex:meas_dev1_temp0 .

ex:prop_dev1_bat_lr a saref3:Energy ;
   saref3:isMeasuredByDevice ex:dev1 ;
   saref3:isPropertyOf ex:foi_living_room ;
   saref3:relatesToMeasurement ex:meas_dev1_bat0 .
```

EXAMPLE 5b:

```
ex:prop_dev1_temp_lr a saref4:PropertyOfInterest ;
   saref4:isObservedBy ex:dev1 , ex:obs_dev1_temp0 ;
   saref4:isPropertyOfInterestOf ex:foi_living_room ;
   saref4:hasPropertyKind om:Temperature .

ex:prop_dev1_bat a saref4:PropertyOfInterest ;
   saref4:isObservedBy ex:dev1 , ex:obs_dev1_bat0 ;
   saref4:isPropertyOfInterestOf ex:dev1 ;
   saref4:hasPropertyKind abcd:BatteryLevel .

om:Temperature a saref4:Property ;
   saref4:isPropertyOf ex:kf_room .

abcd:BatteryLevel a saref4:Property ;
   saref4:isPropertyOf ex:dk_heat5000 .
```

EXAMPLE 5a shows two instances (`ex:prop_dev1_temp_lr` and `ex:prop_dev1_bat_lr`) of the `saref3:Temperature` and `saref3:Energy` classes in SAREF V3.1.1. The `saref3:Temperature` and `saref3:Energy` classes (which are subclasses of the `saref3:Property` class that is not shown in the example) have been deleted from SAREF V4.1.1 and replaced in the EXAMPLE 5b by two instances of the `saref4:PropertyOfInterest` class, one to model the property kind `om:Temperature` of the living room, and the other one to model the property kind `abcd:BatteryLevel` of the device. In these instances, the OPs `saref3:isMeasuredByDevice` and `saref3:relatesToMeasurement` are replaced by `saref4:isObservedby`. Moreover, `saref3:isPropertyOf` is replaced by `saref4:isPropertyOfInterestOf`.

- Observation

This step focuses on the replacement of the saref3:Measurement class by the saref4:Observation class, which is a subclass of the saref4:ProcedureExecution class, a new addition in SAREF V4.1.1. In the EXAMPLE 6b, the results of the observations are the value of the room temperature (ex:pv_dev1_temp0) and the value of the device battery level (ex:pv_dev1_bat0). The OPs saref3:relatesToProperty and saref3:isMeasurementOf are replaced by saref4:observes, and saref4:measurementMadyBy is renamed to saref4:madeBy.

EXAMPLE 6a:
```
ex:meas_dev1_temp0 a saref3:Measurement ;
  saref3:hasTimestamp "2025-01-27T09:03:00"^^xsd:dateTime ;
  saref3:relatesToProperty ex:prop_dev1_temp_lr ;
  saref3:isMeasurementOf ex:foi_living_room ;
  saref3:measurementMadeBy ex:dev1 .

ex:meas_dev1_bat0 a saref3:Measurement ;
  saref3:hasTimestamp "2025-01-27T09:03:00"^^xsd:dateTime ;
  saref3:relatesToProperty ex:prop_dev1_bat_lr ;
  saref3:isMeasurementOf ex:foi_living_room .
```

EXAMPLE 6b:
```
ex:obs_dev1_temp0 a saref4:Observation ;
  saref4:hasTimestamp "2025-01-27T09:03:00"^^xsd:dateTime ;
  saref4:hasResult ex:pv_dev1_temp0 ;
  saref4:observes ex:prop_dev1_temp_lr, om:Temperature,
ex:foi_living_room ;
  saref4:madeBy ex:dev1 .

ex:obs_dev1_bat0 a saref4:Observation ;
  saref4:hasTimestamp "2025-01-27T09:03:00"^^xsd:dateTime ;
  saref4:hasResult ex:pv_dev1_bat0 ;
  saref4:observes ex:prop_dev1_bat , abcd:BatteryLevel , ex:dev1 ;
  saref4:madeBy ex:dev1 .
```

- Property Value

The information in EXAMPLE 7a about saref3:hasValue, saref3:isMeasuredIn and saref3:relatesToProperty is all moved under the saref4:PropertyValue in EXAMPLE 7b. The saref4:PropertyValue class is new in SAREF V4.1.1. It contains the value (saref4f:hasValue) and unit of measure (saref4:isMeasuredIn) from the measurement. The OP saref3:relatesToProperty is renamed to saref4:isValueOfProperty. The OP saref4:PropertyValue provides the possibility to give a saref4:Property or saref4:PropertyOfInterest a value which is a static constant instead of a value that has been explicitly observed, as such seen in clause 5.6.4.

EXAMPLE 7a:
```
ex:meas_dev1_temp0 a saref3:Measurement ;
  saref3:hasValue "22.5"^^xsd:float ;
  saref3:isMeasuredIn om:degreeCelsius ;
  saref3:relatesToProperty ex:prop_dev1_temp_lr .

ex:meas_dev1_bat0 a saref3:Measurement ;
  saref3:hasValue "42"^^xsd:float ;
  saref3:isMeasuredIn om:Percent ;
  saref3:relatesToProperty ex:prop_dev1_bat_lr .
```

EXAMPLE 7b:
```
ex:pv_dev1_temp0 a saref4:PropertyValue ;
  saref4:hasValue "22.5"^^xsd:float ;
  saref4:isMeasuredIn om:degreeCelsius ;
  saref4:isValueOfProperty ex:prop_dev1_temp_lr ,
    om:Temperature .

ex:pv_dev1_bat0 a saref4:PropertyValue ;
  saref4:hasValue "42"^^xsd:float ;
  saref4:isMeasuredIn om:Percent ;
  saref4:isValueOfProperty ex:prop_dev1_bat ,
    abcd:BatteryLevel .
```

- Result in SAREF V4.1.1

EXAMPLE 8 shows the final complete turtle snippet that translates the SAREF V3.1.1 measurements show initially in the EXAMPLE 1 into SAREF V4.1.1.

EXAMPLE 8:

```
ex:dev1 a saref4:Device ;
   saref4:hasResult ex:obs_dev1_temp0 ,
      ex:obs_dev1_bat0 ;
   saref4:observes ex:prop_dev1_temp_lr ,
      ex:prop_dev1_bat_lr ;
   s4ener:serialNumber "S4R3F411EXA"^^xsd:string ;
   saref4:hasDeviceKind ex:dk_heat5000 .

ex:dk_heat5000 a saref4:DeviceKind ;
   abcd:hasDeviceType "Heat series"^^xsd:string ;
   saref4:hasManufacturer "ACME"^^xsd:string ;
   saref4:hasModel "5000"^^xsd:string ;
   saref4:observes om:Temperature , abcd:BatteryLevel .

ex:foi_living_room a saref4:FeatureOfInterest ;
   saref4:hasPropertyOfInterest ex:prop_dev1_temp_lr ;
   saref4:isObservedBy ex:obs_dev1_temp0 ;
   saref4:hasFeatureKind ex:fk_room .

ex:fk_room a saref4:FeatureKind ;
saref4:hasProperty om:Temperature .

ex:dev1 saref4:hasPropertyOfInterest ex:prop_dev1_bat ;
   saref4:isObservedBy ex:obs_dev1_bat0 ;
   saref4:hasFeatureKind ex:dk_heat5000 .

ex:dk_heat5000 saref4:hasProperty abcd:BatteryLevel .

ex:prop_dev1_temp_lr a saref4:PropertyOfInterest ;
   saref4:isObservedBy ex:dev1 ,
      ex:obs_dev1_temp0 ;
   saref4:isPropertyOfInterestOf ex:foi_living_room ;
   saref4:hasPropertyKind om:Temperature .

ex:prop_dev1_bat a saref4:PropertyOfInterest ;
   saref4:isObservedBy ex:dev1 ,
      ex:obs_dev1_bat0 ;
   saref4:isPropertyOfInterestOf ex:dev1 ;
   saref4:hasPropertyKind abcd:BatteryLevel .

om:Temperature a saref4:Property ;
   saref4:isPropertyOf ex:kf_room .

abcd:BatteryLevel a saref4:Property ;
   saref4:isPropertyOf ex:dk_heat5000 .

ex:obs_dev1_temp0 a saref4:Observation ;
   saref4:hasTimestamp "2025-01-27T09:03:00"^^xsd:dateTime ;
   saref4:hasResult ex:pv_dev1_temp0 ;
   saref4:observes ex:prop_dev1_temp_lr,
      om:Temperature,

      ex:foi_living_room ;
   saref4:madeBy ex:dev1 .
```

```
ex:obs_dev1_bat0 a saref4:Observation ;
   saref4:hasTimestamp "2025-01-27T09:03:00"^^xsd:dateTime ;
   saref4:hasResult ex:pv_dev1_bat0 ;
   saref4:observes ex:prop_dev1_bat ,
      abcd:BatteryLevel ,
      ex:dev1 ;
   saref4:madeBy ex:dev1 .

ex:pv_dev1_temp0 a saref4:PropertyValue ;
   saref4:hasValue "22.5"^^xsd:float ;
   saref4:isMeasuredIn om:degreeCelsius ;
   saref4:isValueOfProperty ex:prop_dev1_temp_lr , om:Temperature .

ex:pv_dev1_bat0 a saref4:PropertyValue ;
   saref4:hasValue "42"^^xsd:float ;
   saref4:isMeasuredIn om:Percent ;
   saref4:isValueOfProperty ex:prop_dev1_bat , abcd:BatteryLevel .
```

History

Document history		
V1.1.1	November 2015	Publication
V2.1.1	March 2017	Publication
V3.1.1	February 2020	Publication
V3.2.1	January 2024	Publication
V4.1.1	March 2025	Publication