Emerging Topics in Semantic Technologies
E. Demidova, A.J. Zaveri, E. Simperl (Eds.)
ISBN: 978-3-89838-736-1

Modeling Smart Sensors on top of SOSA/SSN and WoT TD with the Semantic Smart Sensor Network (S3N) modular Ontology

Samya SAGAR [a,b], Maxime LEFRANÇOIS [c], Issam REBAI [a], Khemaja MAHA [d], Serge GARLATTI [a], Jamel FEKI [b], and Lionel MÉDINI [e]

[a] *Lab-STICC, IMT Atlantique, Bretagne Loire, F-29238 Brest, France*
[b] *MIRACL Laboratory University of Sfax, Tunisia*
[c] *Mines Saint-Etienne, Univ Lyon, Univ Jean Monnet, IOGS, CNRS, UMR 5516 LHC, Institut Henri Fayol, F.-42023 Saint-Etienne France*
[d] *Prince Research Group, Isitc, University of Sousse Sousse, Tunisia*
[e] *Univ Lyon, Université Lyon 1, LIRIS UMR 5205 CNRS, F-69622, Lyon, France*

Abstract. The joint OGC and W3C standard SOSA/SSN ontology describes sensors, observations, sampling, and actuation. The W3C Thing Description ontology under development in the W3C WoT working group describes things and their interaction patterns. In this paper we are interested in combining these two ontologies for modeling Smart-Sensors. Along with basic sensors, a Smart-Sensor contains a micro-controller that can run different algorithms adapted to the context and a communicating system that exposes the Smart-Sensor on some network. For example, a smart accelerometer can be used to measure cycling cadence, step numbers or a variety of other things. The SOSA/SSN ontology is only able to model partially the adaptation capabilities of Smart-Sensors to different contexts. Thus, we design an SOSA/SSN extension, called the Semantic Smart Sensor Network (S3N) ontology. S3N answers several competency questions such as how to adapt the Smart-Sensor to the current context of use, that is to say selecting the algorithms to provide the right sensors outputs and the micro-controller capabilities.

Keywords. SSN Ontology, Smart-Sensor, Ontology modeling, context adaptation

1. Introduction

Sensors are devices that are sensitive to, detect, measure, convert stimuli from the physical environment into a usable electrical signal. Thanks to the considerable progress in electronics and communication technology, the number of sensors embedded in everyday objects has greatly increased over the past years. Such sensors are typically not de-

ployed on their own, but connected to a micro-controller capable of running algorithms and storing data, and a communication module that can send data or receive commands, using various Internet of Things protocols and data formats. Such compound systems enable the emergence of new types of "Smart-" applications in various domains (e.g., health, military, building, home, city, industry). We therefore refer to these systems using term *Smart-Sensor*. Unlike traditional sensors (called *Basic-Sensors* in the context of this paper), Smart-Sensors are multi-functional: they can be used in multiple contexts, and are able to execute several algorithms. For example, the same Smart-Sensor hosting a 3-Axis accelerometer embedded in some clothes could theoretically compute and broadcast different kind of data depending on the activity performed by a user (running, cycling, skating, sleeping, walking, etc.). Relevant information to select the appropriate algorithm includes computational and storage capabilities, energy concerns, and any contextual information including climate, user preferences, or network environment.

The main contribution of the paper consists of an ontology, called *Semantic Smart Sensor Network* (S3N), designed on top of the Open Geospatial Consortium (OGC) and the World Wide Web (W3C) joint Semantic Sensor Network Ontology (SOSA/SSN) ontology [7,9,8] and the W3C Web of Things Thing Description (WoT TD) ontology. It semantically describes Smart-Sensors, their different computation and communication profiles, and how different algorithms may be selected and loaded, potentially at runtime. To the best of our knowledge, S3N is the first ontology that: (1) Combines the new SOSA/SSN and the TD ontology and formalizes an alignment between them; (2) Adapts the pattern defined in the SSN System Capabilities module to model properties of other things than Systems; (3) Reuses the SEAS innovative publication scheme so as to be published as three ontology modules each defining terms in the same namespace: `http://w3id.org/s3n/`.

We adopt a knowledge representation methodology inspired from that of Noy and McGuinness [18] and apply the following steps according to which the rest of this paper is structured:

Step 1, analyze the domain. Section 2.1 provides background knowledge on Smart-Sensors;

Step 2, develop scenarios. Section 2.2 details three scenarios involving the use of similar smart accelerometers in different contexts;

Step 3, extract competency questions from the scenarios to define the scope of the ontology. Section 2.3 lists competency questions for two phases in the life-cycle of a Smart-Sensor, namely the operational phase and the reconfiguration phase;

Step 4, choose which of the existing ontologies to reuse. Section 3 overviews related existing ontologies, and justifies our choices regarding the ontologies to base our work on: the OGC and W3C SOSA/SSN ontology, and the W3C TD ontology.

Step 5, develop the ontology. These ontologies are combined and extended in Section 4 to form the S3N ontology that is composed of three different ontology modules, each of which answers different sets of competency questions.

Step 6, qualitatively validate the ontology by showing how it answers the competency questions. This step is led throughout Section 4.

2. Background, Scenarios, and Competency Questions

This section provides background in the domain of Smart-Sensors. It first introduces the important definitions for Smart-Sensors and Basic-Sensors. Secondly, three detailed scenarios involving Smart-Sensors are presented. Finally, competency questions for the S3N ontology are deduced from these three scenarios.

2.1. Smart-Sensors vs Basic-Sensors

Several definitions for *sensor* were proposed in the literature. Authors in [3] defines sensors with respect to the activity of observing the physical property (e.g., temperature, depth) of a feature of interest (e.g., a lake) and report observations. In [12], authors define sensors as data sources which produce a sequence of data items over time (a data stream). They conclude that with this definition we already have a very large number of sensors deployed all over the planet. From a wider vision, we could also refer to [17], which describes a sensor like a source producing a value representing a phenomenon in a particular field. In a broader sense, a sensor is a device sensitive to some stimulus from the physical environment, which detects this observed physical data and converts it into a usable signal (electric, radio, light).

In the SOSA/SSN ontology [7], the OGC and the W3C recently agreed on the following definition for Basic-Sensors, which we adopt in this paper: *A [Basic-] Sensor is a Device, agent (including humans), or software (simulation) involved in, or implementing, a Procedure. [Basic-]Sensors respond to a Stimulus, e.g., a change in the environment, or Input data composed from the Results of prior Observations, and generate a Result. [Basic-]Sensors can be hosted by Platforms.*

Referring to the IEEE 1451 standard [5], the author in [5] describes a Smart-Sensor as: *A Smart-Sensor hosts a set of basic sensors together with communicating components, and a specific component, usually a micro-controller, to condition and process the signal by executing embedded functions before transmitting output data to the control network. Some Smart-Sensors can be configured in a custom manner.*

As is, this second definition presents a certain ambiguity when one tries to distinguish a Smart-Sensor from a basic one. In fact, some Basic-Sensors actually do some signal processing (e.g., detection or filtering), have parameter adjustment capabilities (configuration of their refresh frequency), and implement some communication protocols (e.g., I2C). In our opinion, to be considered as Smart-Sensors they should be endowed with functionalities that make them configurable and adaptable to the context of use. For example, the LIS2DH 3-Axis accelerometer sensor measures both the acceleration in m/s^2 along the x, y and z-axis, and the inertia. It implements an algorithm that converts the captured physical quantity into some usable digital information, but one cannot install any another algorithm to adapt its output to a new activity. On the other hand, Smart-Sensor SMSACTI can be programmed for both cycling and running. For most context of use, domain experts define a list of *indicators* which are used to assess the activity: pedaling speed is an indicator for cycling, while the number of steps and the number of consumed calories are indicators for running. Smart-Sensor SMSACTI can then compute values for these indicators from the acceleration measurements using algorithms. So as to enable this, Smart-Sensor SMSACTI contains a LIS2DH sensor and a micro-controller that can be reprogrammed. In this paper, we use the following

Table 1. Main differences assumed between Basic-Sensors and Smart-Sensors

	hosts	implements	makes
Basic-Sensor	-	One top procedure	Observations
Smart-Sensor	some Basic-Sensors, a micro-controller, a communicating component	Several procedures usable in different contexts	Algorithm Executions

definition for the concept of Smart-Sensor, whose differences which that of Basic-Sensor are summarized in Table 1:

The concept of a smart sensor is based on its ability to (1) acquire data thanks to its embedded sensors, (2) process this data thanks to one or more algorithms its micro-controller implements, (3) output and communicate indicator values, (4) be reprogrammable and reconfigurable.

The next section provides three different scenarios involving Smart-Sensors to point out the competency questions required for designing the S3N Ontology later on.

2.2. Scenarios

The aim of our work is to provide semantic representation allowing the interpretation and understanding operations of Smart-Sensors. In this section, we depict three different scenarios that illustrate the need for a Smart-Sensor to be adapted to the context of use. The scenarios are as follows: i) Scenario A: Abdel is a sportsman practicing cycling and running ii) Scenario B: Beth is a professional athlete training for biathlon; iii) Scenario C: Charles is an elderly person requiring constant monitoring. All three scenarios make use of a Smart-Sensor based on LIS2DH 3-Axis Accelerometer sensors.

2.2.1. Scenario A.

The first scenario involves Abdel, a sportsman that practices cycling and running. Abdel bought a sportswear short equipped with this Smart-Sensor. He wants to monitor his activity using a Bluetooth Low Energy connection to his Smart Watch. During a race, he monitors the stride number, an estimation of the running distance and his average stride frequency. When he practices cycling, he monitors the pedalling cadence, the "riding out of the saddle" (or dancing) duration and the road steepness. These six indicators can be computed from the same raw data given by the LIS2DH embedded in the Smart-Sensor. When the sportsman selects the activity he wants to practice, the Smart-Sensor needs to load the relevant algorithms from its persistent storage to its in-memory storage to switch its functioning mode. Now, if Abdel wants to start roller-skating, he may browse the web for an algorithm he can load on his Smart-Sensor to get an estimation of his skating rate. That algorithm can be installed on the Smart-Sensor only if: (1) the inputs of the algorithm can be provided by the LIS2DH, and (2) that the capabilities of the Smart-Sensor's micro-controller are sufficient for executing this algorithm.

2.2.2. Scenario B.

In the second scenario, Beth, a professional athlete wants to train for biathlon. Her technical staff Bob wants to assess her ski technique and her stability during the shooting sessions. For that, the staff equipped Beth with one LIS2DH per limb (arms and legs), connected to a central Smart-Sensor that broadcasts data using CoAP over WiFi. While she is skiing, acceleration data is analyzed to compute the arms and legs movement amplitude, the ski technique that is used, and the tilt and rotation angle of the different limbs. While she is shooting, the acceleration data is analyzed to measure her stability and concentration capacity. Like in the first scenario, the Smart-Sensor has different roles in different biathlon contexts. It needs to detect changes of phases, and adapt the algorithm it uses.

2.2.3. Scenario C.

The last scenario illustrates how a Smart-Sensor can be reused in a different domain. Assume Charles, an elderly person requires constant monitoring. He is equipped with a wristband that hosts the Smart-Sensor. While Charles falls asleep, the Smart-Sensor analyses his quality of sleep and sends data regularly to a local MQTT broker. It detects his body movements and classifies them (abrupt, micro-movements, etc.). When in the middle of the night the Smart-Sensor detects movements that indicate Charles gets out of bed (e.g., to drink or to go to the lavatory), the Smart-Sensor switches in a fall-detection mode. It analyses the accelerometer data and raises an alert to the MQTT broker in case Charles falls. However, when one of the related components of the Smart-Sensor dysfunctions, the Smart-Sensor sends a value of error code, potentially with the specification of the cause. Thus, the caregivers or the emergency services can be notified and react accordingly.

2.3. Competency Questions

Based on these three scenarios, we identify two main phases in the life-cycle of a Smart-Sensor that are relevant for our work, both having a different list of competency questions: The *Operational Phase* and the *Reconfiguration Phase*.

2.3.1. Operational Phase

This phase is the default one where the Smart-Sensor is deployed and its micro-controller executes various algorithms on the Basic-Sensor observation results, producing indicator values.

CQ1: What are the components of the Smart Sensor: its Basic-Sensors, micro-controller, and communicating component?

CQ2: What is the set of algorithms a micro-controller implements?

CQ3: What indicator did the algorithm output a value for?

CQ4: For a given result output by the micro-controller, what algorithm has been used?

CQ5: If an error occurs when executing an algorithm, What is the reason and the origin of that error?

CQ6: How can one access or subscribe to the output of the micro-controller?

2.3.2. Reconfiguration Phase

reconfiguration capability is the reason why Smart-Sensors are "Smart", and adaptable to the planned activity or other contextual informations. The reconfiguration phase consists of choosing and loading some algorithm. The choice of an algorithm is done considering four important things:

1. the context of use (including the activity and other contextual information);
2. the indicator that the algorithm outputs values for (e.g., number of pedaling cycles, stress);
3. make sure that the algorithm inputs are properties that basic sensors observe;
4. make sure that the computation and storage capabilities of the micro-controller are sufficient.

In order for the algorithms to be executed by the Smart-Sensor, we identify the following competency questions:

CQ7: For a given algorithm, what is the context of use?
CQ8: For a given algorithm, what are the indicator it outputs?
CQ9: For a given algorithm, what are the properties it requires as input?
CQ10: What are the capabilities of a micro-controller?
CQ11: For a given algorithm, what are the minimal requirements for the micro-controller to execute it?
CQ12: Can a micro-controller implement a given set of algorithms?
CQ13: How can one change the set of algorithms a micro-controller implements?

Now that we listed the competency questions our ontology needs to cover, the next section aims at choosing among the relevant existing ontologies those that we will reuse.

3. Relevant Existing Ontologies

Some of the competency questions we identified to model Smart-Sensors are already covered by existing ontologies. In this section we overview a short list of such ontologies and justify which of them we reuse. We focus on ontologies designed to model sensors and the IoT, and limit ourselves to those that are standards or will soon be: SOSA/SSN [7,9,8], WoT TD [10], SAREF [4]. We also consider some ontologies that are built on top of them: M3 [6], SEAS [14].

3.1. The new OGC and W3C SOSA/SSN standard ontology

Between 2000 and 2012, several conceptual models have been proposed to model sensors and their observations. Examples include the OGC Sensor Model Language (SensorML) [2], the OGC Observations and Measurements (O&M) [19], OBOE [16], Sensei [1], Seronto [21]. These were reviewed by the W3C Semantic Sensor Network Incubator Group, which in turn developed the SSNX ontology published in 2011 [13] that then became widely reused in other ontologies and implemented in datasets. From this success, the OGC and the W3C started the joint Spatial Data on the Web Working Group

and revisited the SSNX ontology to account for shortcomings of the initial work, account for actuation and sampling in addition to observations, and publish the new version of the ontology as both a W3C recommendation and a OGC implementation standard.

The result of this work has recently been published as a modular ontology named SOSA/SSN, that specifies the semantics of sensors, observations, actuation, and sampling. Given the importance of a joint OGC and W3C standard, we commit ourselves in reusing as much as possible from SOSA/SSN. The main modules of the SSN ontology that are relevant for our work are:

- SOSA (Sensor, Observation, Sampler, and Actuator) is a lightweight core module with few terms and little axiomatization;
- SSN (Semantic Sensor Network) imports SOSA, introduces other terms, and adds more axiomatization to SOSA and SSN terms;
- SSN-System (the SSN System Capability module) imports SSN (and indirectly imports SOSA) and defines additional classes and properties used to model the system capabilities, operating range, and survival range.

These modules define terms under the following namespaces that are shorten with the following prefixes, that we will reuse in the rest of this paper.

```
@prefix sosa: <http://www.w3.org/ns/sosa/> . # for SOSA
@prefix ssn: <http://www.w3.org/ns/ssn/> . # for SSN
@prefix ssn-system: <http://www.w3.org/ns/ssn/systems/> . # SSN-System
```

Obviously, one may reuse the concept sosa:Sensor to describe Basic-Sensors, and as they are often mounted directly on the Smart-Sensor (at least in Scenario A. and C.), a Smart-Sensor will be modeled as a sosa:Platform that also hosts the Micro-Controller and the communicating component. Then as a Micro-Controller implements algorithms which are explicitly mentioned in the definition of sosa:Procedure, we may model them as different sub-types of the class ssn:System which implement different sosa:Procedures.

3.2. The M3 and M3-lite ontologies

The M3 ontology [6] and its light version[1] list different kinds of sensors, along with the typical domain in which they can be used. Although this classification is relevant to use cases where some set of sensors could need to be selected for a given procedure in a given context, we choose not to reuse them for the following reasons: (a) they have not yet been adapted to the new version of SSN, and (b) in this work we are assuming sensors can be used in different applications for different domains.

3.3. The W3C Thing Description ontology

Another standardization group of high interest for our work is the W3C Web of Things working group, whose charter[2] includes the design of *an ontology and a binding to short names for cross-domain metadata for the Web of Things, including the data model ex-*

[1] http://purl.org/iot/vocab/m3-lite#
[2] W3C WoT Charter - https://www.w3.org/2016/12/wot-wg-2016.html

posed to applications, as well as security and communications metadata. [10] is the current specification of the Thing Description ontology. TD is still under development. A Thing is defined in [11] as *the abstraction of a physical or virtual entity that needs to be represented in IoT applications. This entity can be a device, a logical component of a device, a local hardware component, or even a logical entity such as a location (e.g., room or building)*. A td:InteractionPattern represents the specification of a pattern using which one may interact with a Thing's properties, actions, or events. We propose to reuse the TD ontology in our work to cover Competency Questions CQ6 and CQ13. This ontology defines terms under namespace `http://www.w3.org/ns/td#` that is shorten with the prefix `td:`, which we will be using in the rest of this paper.

```
@prefix td: <http://www.w3.org/ns/td#>.
```

We already proposed some alignment between SOSA/SSN and TD in the past,[3] which has been discussed by the WoT group. A td:Thing can be considered as a ssn:System which implements interaction patterns (sosa:Procedures) with some input and output description. TD additionally models the access point where one may trigger executions of these procedures using class td:Link.

```
td:Thing rdfs:subClassOf ssn:System .
td:interaction rdfs:subPropertyOf ssn:implements .
td:InteractionPattern rdfs:subClassOf sosa:Procedure .
td:inputData rdfs:subPropertyOf ssn:hasInput .
td:outputData rdfs:subPropertyOf ssn:hasOutput .
```

A Smart-Thing may be modeled as a td:Thing that provides at least: (a) some property or event interaction pattern which describes how one may read, observe, or subscribe to the algorithms execution results, and (b) some property interaction pattern that describes how one can read or write the set of algorithms that its Micro-Controller implements.

3.4. The ETSI SAREF ontology

The ETSI Smart Appliances Reference (SAREF) ontology [4] focuses on the concept of device, which is defined in the context of the Smart Appliances study as *a tangible object designed to accomplish a particular task in households, common public buildings or offices. In order to accomplish this task, the device performs one or more functions*. Extensions for different verticals are under development for the Energy-, Building-, Environment-, and Industry- domains. SAREF's Sensor concept offers services that represent functions, one of which must be a sensing function with some sensing range. SAREF could be used to answer part of our Competency Questions, but it fails at properly distinguishing between the various components we assume a Smart-Sensor contains. Also, there exists some overlap with the concepts from SOSA/SSN, and proposing a proper alignments should be the focus of a separate work. We therefore choose to stick to SOSA/SSN and TD, and ignore SAREF for the time being.

[3] https://lists.w3.org/Archives/Public/public-wot-ig/2017Jul/0008.html

3.5. The SEAS ontology

The SEAS ontology [14] is a modular and versioned ontology with all the terms it define having the same namespace. It is built on top of SOSA/SSN, and contains as a core four simple ontology patterns to describe physical systems and their connections, value association for their properties, and the activities by which such value association is done. These ontology patterns are then instantiated for different engineering-related verticals. Some modules in SEAS could be of interest to this work such as the seas:CommunicationOntology module, which can be used to describe systems that communicate using different protocols, and their connections. However, in this work and as a first step, we will solely use SOSA/SSN and TD. On the other hand, we will reuse the SEAS proposal for exposing our ontology as a modular ontology while letting the terms it defines have the same namespace.

In the next section we reuse and extend SOSA/SSN and the current TD ontology to model Smart-Sensors, and we use the SEAS innovative publication scheme.

4. Development of the S3N Ontology

This section introduces the Semantic Smart Sensor Network ontology (S3N), built on top of SOSA/SSN and the current TD ontology to model Smart-Sensors and cover the competency questions listed in Section 2.3. Similar to SSN and its different modules, S3N is divided in three modules that semantically describes a specific aspect of Smart-Sensors and answer different sets of competency questions. These three modules can be imported separately or together, and they all define terms in the same namespace `http://w3id.org/s3n/`, that we shorten with `s3n:`. They are published according to the publication and metadata best practices using the SEAS innovative publication scheme [14].

```
@prefix s3n: <http://w3id.org/s3n/>.
```

The next sections describe these three modules in sequence:

Module s3n:S3NCore **(S3N Core),** described in details in Section 4.1, describes the constitution of Smart-Sensors and its adaptability. It imports SSN and covers Competency Questions CQ1-4 and CQ7-9;

Module s3n:S3NProcedure **(S3N Procedure),** described in Section 4.2, describes algorithms' features, and capabilities of a micro-controller. It imports S3N-core and SSN-System, and covers Competency Questions CQ10-12;

Module s3n:S3NThing **(S3N Thing),** described in Section 4.3, describes possible interactions with a Smart-Sensor, imports S3N-core and TD, and covers Competency Questions CQ6 and 13.

4.1. The S3N Core Module

The S3N-core module has URL `http://w3id.org/s3n/S3NCore`, and extends SSN with 7 classes and 3 object properties. We model the Smart-Sensors composition and their adaptability in terms of the algorithms they can execute.

4.1.1. Smart-Sensors Composition.

Obviously no new concept is defined for Basic-Sensors as sosa:Sensor fits perfectly. On the other hand, S3N introduces classes for the other three main concepts:

s3n:MicroController: A MicroController is a compact integrated circuit containing a processor, some memory, and input/output (I/O) peripherals on a single chip, and is designed to govern a specific operation in an embedded system. It implements some Procedures, and makes ProcedureExecutions.

s3n:CommunicatingSystem: A CommunicatingSystem can be used to exchange information with other CommunicatingSystem on some network.

s3n:SmartSensor: A SmartSensor is composed of one or more Sensors together with a MicroController that implements different Procedures, and make Executions of these Procedures on the result of the Observations these Sensors make to output a resulting value for some Indicator. This value may then be communicated by some CommunicatingSystem.

A Smart-Sensor hosts different components, and can therefore be considered as a sosa:Platform. Copying the SOSA axiomatic scheme, S3N asserts that the domain of sosa:hosts also includes s3n:SmartSensor, while its range also includes s3n:MicroController and s3n:CommunicatingSystem. Property sosa:hosts is used when the location of the hosted entity is dependent of that of the hosting entity, but the hosted entity is not necessarily a component of the hosting entity. For example in Scenario B., Beth may host the Smart-Sensor and all four Basic-Sensors, but the Smart-Sensor only hosts the micro-controller and the WiFi communicating component.

```
<sensor1> a sosa:Sensor . <sensor2> a sosa:Sensor .
<sensor3> a sosa:Sensor . <sensor4> a sosa:Sensor .
<micro-controller> a s3n:MicroController .
<communicatingcomp> a s3n:CommunicatingSystem .

<Beth> a foaf:Person , sosa:Platform ;
  sosa:hosts <smartsensor>, <sensor1>, <sensor2>, <sensor3>, <sensor4> .

<smartsensor> a s3n:SmartSensor ;
 sosa:hosts <micro-controller>, <communicatingcomp> .
```

Although the Smart-Sensor does not directly host each of the Basic-Sensors, one may consider they are its components along with the micro-controller and the WiFi communicating component, using property ssn:hasSubSystem. This property cannot be used to link Beth to these electronic devices, unless she is a cyborg.

```
<smartsensor> ssn:hasSubSystem <sensor1>, <sensor2>, <sensor3>, <sensor4>,
     <micro-controller>, <communicatingcomp> .
```

Then taking advantage of the richer axiomatization of SSN, s3n:SmartSensor is defined as a sub-class of both sosa:Platform and ssn:System, while both s3n:MicroController and s3n:CommunicatingSystem are defined as sub-classes of ssn:System. Finally, we model the fact that a s3n:SmartSensor contains at least one of each components. Thus, competency question CQ1 is covered.

4.1.2. Algorithms and their Executions

SOSA/SSN follows similar design patterns for Sensors (that implement Procedures and make Observations), Actuators (that implement Procedures and make Actuations), and Samplers (that implement Procedures and make Samplings). As we introduce new sub-types of ssn:System, it is justified to reuse this pattern and model s3n:MicroControllers make. Unlike for Sensors and Actuators that may implement different kinds of procedures (not only algorithms), s3n:MicroControllers are specifically designed to implement sensing related algorithms as well as adapatation and reconfiguration ones. We thus specialize sosa:Procedure and propose the following pattern instantiation: s3n:MicroControllers implement some s3n:Algorithm and make s3n:AlgorithmExecution activities (property s3n:madeAlgorithmExecution). Parallel to SSN, the S3N ontology specifies that s3n:MicroController only makes algorithm executions, implements at least one thing, and that thing is an Algorithm.

This covers Competency Question CQ2. We choose not to mimic SSN to define an inverse relation named *madeByMicrocontroller* because other systems could make algorithm executions too. Now, SSN contains axiom 1, which implies that any sosa:Observation a sosa:Sensor makes uses all of the sosa:Procedures that sensor implements. In our use cases a s3n:MicroController can implement different algorithms so we do not transpose this axiom to micro-controllers.

$$\text{sosa:madeBySensor} \circ \text{ssn:implements} \sqsubseteq \text{sosa:usedProcedure} \tag{1}$$

So as to cover Competency Question CQ7, we define a property s3n:forContext, which is intended to link an s3n:Algorithm to its context of use. The link between an s3n:AlgorithmExecution and the specific algorithm it used can be made explicit using sosa:usedProcedure, which covers Competency Question CQ4.

Using SSN, one may describe the input and output of an s3n:Algorithm, for instance using instances of ssn:Property (generic as defined in [7, §7.4]), for example `ex:6AxisAcceleration` as the input and `ex:CyclingRate` as the output. As already mentioned in Section 2, this second property is what domain experts we were working with name *indicators*. We therefore define a class s3n:Indicator as a sub-class of ssn:Property. This covers Competency Questions CQ3, CQ8, and CQ9.

```
ex:CyclingRate a s3n:Indicator .
<CyclingAlgorithm1> a s3n:Algorithm ;
  ssn:hasInput ex:6AxisAcceleration ;
  ssn:hasOutput ex:CyclingRate .
```

Finally, the result of an s3n:AlgorithmExecution may be a literal (using sosa:hasSimpleResult) or an instance of sosa:Result (using sosa:hasResult), in which case it may additionally be an instance of s3n:Error and potentially have a s3n:hasCause. This covers Competency Question CQ5.

Below is an example of Smart-Sensor `<smartSensor1>` whose micro-controller made executions of two different algorithms, one resulting in 17.0 Celcius Degrees and the other resulting in an error. We use the cdt:ucum datatype as defined in [15].

```
<avgTemp#24H> a s3n:Algorithm . <avgTemp#1Y> a s3n:Algorithm .
```

```
<smartSensor1> sosa:hasSubSystem <uc1> .
<uc1> a s3n:MicroController ;
  sosa:implements <avgTemp#24H>, <avgTemp#1Y> ;
  sosa:madeAlgorithmExecution <exec1>, <exec2> .

<exec1> a s3n:AlgorithmExecution ;
  sosa:usedProcedure <avgTemp#24H> ;
  sosa:hasSimpleResult "17.0 DEG"^^cdt:ucum .

<exec2> a s3n:AlgorithmExecution ;
  sosa:usedProcedure <avgTemp#1Y> ;
  sosa:hasResult [ a s3n:Error ; s3n:cause <insufficientMem> ] .
```

4.2. The S3N Procedure module

The S3N-procedure module has URL http://w3id.org/s3n/S3NProcedure and extends SSN-System with 7 classes and 2 object properties.

Unlike the old version of SSN that restricted the scope of capability description to the sole class of Sensors, any ssn:System can have its capabilities described in the new SSN Capability module. This module proposes a pre-defined list of system properties that are important for sensors and actuators, and can actually be reused for other types of systems such as s3n:Microcontroller and s3n:CommunicatingSystem. For example, the definition of ssn-system:Frequency may be understood somehow applicable for these systems. The S3N-procedure module leverages this extensibility and adds two new system properties: (1) s3n:Memory for s3n:Microcontrollers: The memory of the micro-controller under the defined Conditions; and (2) s3n:MaximumBandwidth for s3n:CommunicatingDevices: The maximal bandwidth of the communicating device under the defined Conditions. This answers Competency Question CQ10.

The S3N-procedure also reuses the SSN-Systems design pattern to describe procedure properties such as the duration, computational cost, storage cost, etc. of a Procedure under some specified Conditions such as a size of input. This extension is using terms s3n:ProcedureFeature, s3n:hasProcedureFeature, s3n:hasProcedureProperty, and s3n:ProcedureProperty. Procedure property s3n:ComputationalCost is proposed for any procedure, and procedure properties s3n:TimeComplexity and s3n:SpaceComplexity are specifically defined for s3n:Algorithms. This answers Competency Question CQ11. These terms can also be used to model the fact that an Algorithm can have different capabilities depending on the activity it is used for, using ssn-system:inCondition, thus covering possible extensions of Competency Question CQ7.

As for Competency Question CQ12, a SPARQL ASK Query may be written to check if a micro-controller's memory is greater than the sum of the computational costs of each of the procedures one wants to load in it.

4.3. The S3N Thing module

The S3N Thing module has URL http://w3id.org/s3n/S3NThing, imports both S3N Core and TD. It contains the alignments that were proposed between SSN and TD concepts, and a single additional logical axiom that defines s3n:SmartSensor as a sub-class of td:Thing.

That way, the s3n:SmartSensor holds the description of how one may interact with its different components. The following example illustrates the use of the TD ontology on scenario A.: the Smart-Sensor Abdel is using may have (1) a property interaction pattern that describes how one may check the list of algorithms its micro-controller implement; (2) an action interaction pattern for pairing a new bluetooth device to the communicating system; (3) event interaction patterns to receive indicator values and notifications when battery runs low. This therefore answers Competency Questions CQ6 and CQ13. Although TD is still undergoing structural and naming changes, it is likely that its core (things have interaction patterns that are described) will remain almost unchanged, so little work should be needed to update the S3N Thing module in the future. In the current version of TD, (3) may be partially described as follows:

```
<smartsensor01> a s3n:SmartSensor ;
   td:interaction [ a td:Property ;
     rdfs:comment "How to observe the algorithm execution results."@en ;
     td:observable true ;
     td:link [ td:href <wss://192.168.24.75/> ; td:mediaType "application/json" ]
          ] .
```

5. Conclusion

With the miniaturization of electronic components, the augmentation of computational and storage capabilities, and the rise of the Web of Things, sensors tend to become smarter and smarter. Smart-Sensors have the ability to be reprogrammed or reconfigured such that they can be used in multiple contexts with different algorithms. In this paper we adopted a strict knowledge engineering methodology to develop a semantic model for Smart-Sensors. The result, the Semantic Smart Sensor Network (S3N) Ontology, combines the OGC and W3C SOSA/SSN standard and the W3C TD future standard to describe the architecture of Smart-Sensors, the algorithms they can execute and the indicator values they output, the features of these algorithms and the capabilities of the different components, and the patterns one may use to interact with the Smart-Sensors. It answers the 13 competency questions for the operational phase and the reconfiguration phase in the life cycle of Smart-Sensors that were listed in Section 2.3, thus enabling to semantically describe the three scenarios from Section 2.2.

To the best of our knowledge, this work and the S3N ontology is the first that (1) Combines the new SOSA/SSN and the TD ontology and formalizes an alignment between them; (2) Adapts the pattern defined in the SSN System Capabilities module to model properties of other things than Systems; (3) reuses the SEAS innovative publication scheme. The ontology S3N constitutes one of the main core modules of the SMS ontology intended to semantically model the manufacturing and use of smart clothings which are used for several activities as sports, well-being, etc. Details of SMS ontology are described in our previous work [20].

Future work on the S3N content includes the development of some extension related to the communicating component of Smart-Sensors, which would enable semantic description of communication protocol, along with security and privacy concerns. We also plan to keep the S3N ontology up to date with the latest version of the TD ontology, and

implement demonstrators with actual Smart-Sensors. Finally, the modeling that is proposed in S3N can be extended to other types of "Smart" things, such as Smart-Actuators.

Acknowledgments.

This work has been partly funded by the Bpi France SMartSensing project, the ANR 14-CE24-0029 OpenSensingCity project, and a bilateral research convention between ARMINES Fayol and ENGIE R&D.

References

[1] Payam Barnaghi, Stefan Meissner, Mirko Presser, and Klaus Moessner. Sense and sensability: Semantic data modelling for sensor networks. In *Proceedings of the ICT Mobile Summit*, pages 1–9, 2009.

[2] Mike Botts and Alexandre Robin. OGC SensorML: Model and XML Encoding Standard. OGC Encoding Standard, Open Geospatial Consortium, February 04 2014. version 2.0.

[3] Michael Compton, Cory Henson, Laurent Lefort, Holger Neuhaus, and Amit Sheth. A survey of the semantic specification of sensors. In *In 2nd International Semantic Sensor Networks Workshop*, 2009.

[4] Laura Daniele, Frank den Hartog, and Jasper Roes. Created in close interaction with the industry: the smart appliances reference (saref) ontology. In *International Workshop Formal Ontologies Meet Industries*, pages 100–112. Springer, 2015.

[5] Stephane Gervais-Ducouret. Next smart sensors generation. In *Conference: Sensors Applications Symposium*. IEEE, 2011.

[6] Amelie Gyrard, Soumya Kanti Datta, Christian Bonnet, and Karima Boudaoud. Cross-domain internet of things application development: M3 framework and evaluation. In *Future Internet of Things and Cloud (FiCloud), 2015 3rd International Conference on*, pages 9–16. IEEE, 2015.

[7] Armin Haller, Krzysztof Janowicz, Simon J D Cox, Danh Le Phuoc, Kerry Taylor, and Maxime Lefrançois. Semantic Sensor Network Ontology. W3C Recommendation, W3C, October 19 2017.

[8] Armin Haller, Krzysztof Janowicz, Simon J.D. Cox, Maxime Lefrançois, Kerry Taylor, Danh Le Phuoc, Josh Lieberman, Raúl García-Castro, Rob Atkinson, and Claus Stadler. Sosa: A lightweight ontology for sensors, observations, samples, and actuators. *Semantic Web Journal*, 2018.

[9] Krzysztof Janowicz, Armin Haller, Simon J.D. Cox, Danh Le Phuoc, and Maxime Lefrançois. Sosa: A lightweight ontology for sensors, observations, samples, and actuators. *Journal of Web Semantics*, 2018.

[10] Sebastian Kaebisch and Takuki Kamiya. Web of Things (WoT) Thing Description. First Public Working Draft, W3C, September 14 2017.

[11] Kazuo Kajimoto, Matthias Kovatsch, and Uday Davuluru. Web of Things (WoT) Architecture. First Public Working Draft, W3C, September 14 2017.

[12] Danh Le-Phuoc and Manfred Hauswirth. Linked open data in sensor data mashups. In *Proceedings of the 2Nd International Conference on Semantic Sensor Networks - Volume 522*, SSN'09, pages 1–16, Aachen, Germany, Germany, 2009. CEUR-WS.org.

[13] Laurent Lefort, Cory Henson, and Kerry Taylor. Semantic Sensor Network XG Final Report. W3C Incubator Group Report, W3C, June 28 2011.

[14] Maxime Lefrançois. Planned ETSI SAREF Extensions based on the W3C&OGC SOSA/SSN-compatible SEAS Ontology Patterns. In *Proceedings of Workshop on Semantic Interoperability and Standardization in the IoT, SIS-IoT,*, July 2017.

[15] Maxime Lefrançois and Antoine Zimmermann. The unified code for units of measure in RDF: cdt:ucum and other UCUM datatypes. In *Proc. Extended Semantic Web Conference (ESWC'18)*, Heraklion, Crete, Greece, June 2018.

[16] Joshua Madina, Shawn Bowers, Mark Schildhauer, Sergeui Krivovc, Dean Pennington, and Ferdinando Villa. An ontology for describing and synthesizing ecological observation data. *Ecological Informatics*, 2(3):279–296, 2007.

[17] Holger Neuhaus and Michael Compton. The semantic sensor network ontology: A generic language to describe sensor assets, 2009.

[18] Natasha F. Noy and Deborah L. McGuinness. Ontology development 101 : A guide to creating your first ontology : Knowldege systems laboratory, stanford university. *Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880*, 2001.

[19] Hajo Rijgersberg, Mark van Assem, and Jan Top. Ontology of units of measure and related concepts. *Semantic Web*, 4(1):3–13, 2013.

[20] Samya Sagar, Issam Rebai, Maha Khemaja, and Jamel Feki. Ontologie modulaire pour la fabrication et l'exploitation de vêtements intelligents dédiés au sport. In Catherine Roussey, editor, *IC 2017 : 28ème Journées francophones d'Ingénierie des Connaissances*, pages 139–144, Caen, 2017.

[21] D.C. van der Werf, M. Adamescu, M. Ayromlou, N. Bertrand, J. Borovec, H. Boussard, C. Cazacu, T. Van Daele, S. Datcu, M. Frenzel, V. Hammen, H. Karasti, M. Kertesz, P. Kuitunen, M. Lane, J. Lieskovsky, B. Magagna, J. Peterseil, S. Rennie, H. Schentz, K. Schleidt, and L. Tuominen. Seronto a socio-ecological research and observation ontology: the core ontology. Alter-Net Deliverable 4.I6.D2, 2009.