# Intégration de SOSA/SSN, SAREF, et TD dans l'ontologie CoSWoT

Maxime Lefrançois, Catherine Roussey, Fatma-Zohra Hannou, Victor Charpenay

# Intégration de SOSA/SSN, SAREF, et TD dans l'ontologie CoSWoT

M. Lefrançois[1], C. Roussey[2], F.-Z. Hannou[3], V. Charpenay[1]

[1] Mines Saint-Etienne, Univ Clermont Auvergne, INP Clermont Auvergne,
CNRS, UMR 6158 LIMOS, Saint-Étienne, France
[2] MISTEA, University of Montpellier, INRAE
Institut Agro, Montpellier, France
[3] EDF R&D, Palaiseau, France

maxime.lefrancois@emse.fr, catherine.roussey@inrae.fr, fatma-zohra.hannou@edf.fr,
victor.charpenay@emse.fr

## Résumé

*Cet article décrit l'ontologie CoSWoT construite au cours du projet « Constrained Semantic Web of Things » à l'aide de la méthodologie ACIMOV [13]. Cette ontologie réconcilie trois ontologies de référence du Web des Objets : « Semantic Sensor Network » (SSN/SOSA) [11] et « Thing Description » (TD) [3] du W3C, « Smart Applications REFerence » (SAREF) de ETSI [9], à travers la proposition d'un nouveau patron de conception architectural « Kinds of X and X of Interest ». Son objectif est de permettre à des dispositifs distribués sur le Web d'échanger des données sans ambiguïté. Nous illustrons les éléments de cette ontologie par un cas d'usage en bâtiment intelligent.*

## Mots-clés

*Web des Objets Sémantique, ontologie modulaire, description de service, capteur, actionneur, raisonnement distribué, patron de conception d'ontologies.*

## Abstract

*This article describes the CoSWoT ontology built during the "Constrained Semantic Web of Things" project using the ACIMOV [13] methodology. This ontology reconciles three reference ontologies from the Web of Things world : "Semantic Sensor Network" (SSN/SOSA) [11] and "Thing Description" (TD) [3] of the W3C, "Smart Applications REFerence" (SAREF) from ETSI [9], through the definition of a new ontology design pattern "Kinds of X and X of Interest". Coswot's goal is to enable devices distributed on the Web to exchange data unambiguously. We will illustrate the elements of this ontology with a smart building use case.*

## Keywords

*Semantic Web of Objects, ontology, service description, sensor, actuator, distributed reasoning, design pattern.*

## 1 Introduction

Le projet Constrained Semantic Web of Things (CoSWoT) [1] s'attaque aux verrous liés à l'utilisation du Web sémantique dans les objets contraints de l'Internet des Objets, en particulier ceux liés à l'interopérabilité sémantique et au raisonnement distribué.

Le projet CoSWoT a pour objectif de proposer une architecture logicielle distribuée et embarquée sur des dispositifs contraints du Web des Objets (WoT). Cette architecture a deux caractéristiques principales : (1) elle utilisera des modèles de connaissances à base de graphes pour déclarer la logique applicative des dispositifs et la sémantique des messages échangés ; (2) les dispositifs auront des capacités de raisonnement afin de répartir les tâches de traitement entre eux. Le développement d'applications WoT sera simplifié : notre plateforme permettra le développement et l'exécution d'applications intelligentes et décentralisées du WoT malgré l'hétérogénéité des dispositifs connectés. La plateforme proposée sera testée sur plusieurs cas d'utilisation dans le bâtiment intelligent et en agriculture numérique.

La contribution principale présentée dans cet article est l'ontologie CoSWoT, qui comprend les connaissances communes à tous les composants de la plateforme CoSWoT et les connaissances spécifiques à certains domaines d'application. Le développement de l'ontologie CoSWoT vise à satisfaire les objectifs suivants :

- **O1** L'ontologie doit être alignée à des ontologies de référence du WoT et doit réutiliser des ontologies de référence des domaines visés ;
- **O2** L'ontologie doit être modulaire, avec des modules qui couvrent les connaissances communes à tous les composants de la plateforme WoT ;
- **O3** L'ontologie doit avoir une structure homogène et prédictible, de sorte que des concepts similaires définis dans des domaines différents soient décrits de la même manière ;
- **O4** Des sous-ensembles de l'ontologie CoSWoT, ou *vues*, devraient pouvoir être embarquées dans des dispositifs avec des contraintes de ressources, pour manipuler, raisonner et échanger des données de capteurs pour une application spécifique.

Nous illustrons le développement de cette ontologie pour une application du domaine du bâtiment intelligent, qui

1. https://coswot.gitlab.io/

porte sur le bâtiment Espace Fauriel (EF) de Mines Saint-Étienne [8]. Notre exemple implique plusieurs dispositifs déployés dans le bureau 429 :

— un radiateur avec un thermomètre, piloté à distance ;
— un noeud près de la fenêtre avec un thermomètre, un capteur qui évalue le taux de dioxyde de carbone de l'air (capteur CO2), un bras actionnable permettant d'ouvrir la fenêtre et de donner son statut, des capacités de communications, de calcul et de raisonnement ;
— un noeud près du tableau avec un thermomètre, un capteur de CO2 et des capacités de communication ;
— un noeud près de la porte avec un thermomètre, un capteur de CO2 et des capacités de communication.

Dans notre exemple, les capteurs de CO2 mesurent toutes les 5 secondes, les noeuds envoient leurs mesures au noeud de la fenêtre pour calculer la mesure maximale de concentration de CO2. A partir de cette mesure, ce noeud déduira la qualité de l'air du bureau. L'objectif est d'automatiser l'ouverture de la fenêtre en fonction de la qualité de l'air.
Le reste de l'article est organisé ainsi : la section 2 donne un aperçu des évolutions récentes des ontologies de référence du domaine du WoT. La section 3 propose un patron de conception architectural qui permet de mitiger certains problèmes d'hétérogénéité identifiés dans ces ontologies. La section 4 présente la méthode de développement de l'ontologie CoSWoT pour satisfaire les objectifs **O1** à **O4**. La section 5 donne un aperçu de l'ontologie CoSWoT et présente quelques modules illustrés sur notre exemple. La section 6 conclue notre article.

# 2 Évolution des ontologies du WoT

Avec l'émergence du Semantic Web of Things (SWoT), les standards incluent des ontologies, de façon à promouvoir l'utilisation de quelques ontologies de référence, plutôt que d'observer un morcellement et la création de nouvelles ontologies avec chaque nouveau projet. Les organismes de standardisation qui contribuent aux ontologies du SWoT sont notamment : (1) Le World Wide Web Consortium (W3C), organisme international de standardisation du Web et du Web sémantique. (2) L'European Telecommunication Standards Institute (ETSI), organisme européen de standardisation pour les télécommunications. (3) oneM2M, consortium international rassemblant des organismes de standardisation (dont l'ETSI), des organismes de recherche et des industriels autour d'un standard pour l'Internet of Things (IoT). Nous avons présenté à IC 2019 un positionnement sur le Web Sémantique des Objets, dans lequel nous discutions des évolutions récentes des ontologies standardisées pour le WoT [19, Section 2]. Nous reprenons cette section ici et présentons une vision actualisée de leur évolution.

## 2.1 OGC&W3C SOSA/SSN

Le groupe d'incubation *Semantic Sensor Network* du W3C a publié en 2011 un rapport analysant les différents modèles conceptuels existants pour décrire les capteurs et leurs observations, ainsi qu'une proposition d'ontologie SSNX [18]. L'ontologie SSNX a été massivement réutilisée par d'autres ontologies et jeux de données. Parallèlement, l'Open Geospatial Consortium (OGC) publiait le modèle UML Observations and Measurements (O&M) V2.0 [4]. En 2017, le groupe de travail Spatial Data on the Web Working Group (SDW-WG) commun aux organismes de standardisation OGC et W3C a publié une mise à jour de cette ontologie en 2017, nommée **SOSA/SSN** [14, 12]. Celle-ci spécifie la sémantique des capteurs et actionneurs, entre autres. Elle permet de décrire notamment les capteurs, les propriétés des entités d'intérêt qu'ils observent, les observations qu'ils font et le résultat de ces observations. De manière analogue, elle permet de décrire les actionneurs, les propriétés des entités d'intérêt sur lesquelles ils peuvent agir, et les actionnements qu'ils font et le résultat de ces actionnements. SSN se compose d'un noyau léger appelé SOSA (Sensor, Observation, Sample, and Actuator), d'un module d'extension plus expressif SSN, d'un module SSN-Systems séparé pour les capacités du système et d'un ensemble de modules d'alignement avec d'autres ontologies. Le travail sur SOSA/SSN s'est poursuivi du côté du W3C avec des propositions d'extension pour les collections d'observation, et à l'OGC avec la publication récente d'une nouvelle version V3.0 du standards O&M (maintenant : *Observations, measurements and samples*). Dans la foulée, le groupe SDW-WG a planifié le développement d'une nouvelle version de l'ontologie SOSA/SSN pour 2024 [2].

## 2.2 W3C Thing Description

Le W3C comprend un autre groupe de travail qui contribue directement au SWoT : le W3C Web of Things working group. Un *Thing* est défini dans [17] comme l'abstraction d'une entité (virtuelle ou physique) pouvant être manipulée par une application IoT, *e.g.* un objet, un service, ou une entité logique telle qu'une pièce ou un bâtiment. L'ontologie **Thing Description** [15] décrit les *affordances d'interaction* des objets, c'est à dire les fonctionnalités qu'on peut invoquer pour manipuler son état, ou déclencher un processus. TD peut être utilisé conjointement avec l'ontologie des contrôles hypermédias HCTL [1] et le vocabulaire RDF pour JSON Schema [2] pour décrire précisément comment formuler une requête (HTTP, CoAP, MQTT, ...) pour interagir avec l'objet [16]. Un *Servient* –contraction de *Server* et de *Client*– est une suite logicielle qui expose ou consomme des Things.

## 2.3 SAREF

Le développement de l'ontologie Smart Applications Reference (**SAREF**) [3] [5, 9], a débuté en 2014/2015 avec une étude demandée par la Commission européenne pour comparer et aligner les modèles de connaissances en lien avec l'IoT, afin de limiter la fragmentation de ce marché. SAREF consiste aujourd'hui en un module noyau, et une dizaine d'extensions pour des domaines d'application comme l'énergie, les bâtiments, ou l'agriculture. Initialement centrée sur le concept de *Device* en tant qu'objet physique rem-

2. https://github.com/w3c/sdw-sosa-ssn/
3. https://saref.etsi.org/

plissant une ou plusieurs fonctions, l'ontologie SAREF a été progressivement amélioré au fil des versions. La version V3.2.1 de SAREF Core [7] a été publiée en janvier 2024, et démontre une volonté forte de convergence vers SOSA/SSN. Par exemple, la classe *Measurement* est déprécié en faveur de la classe *Observation*, reprise de SOSA/SSN. Des choix de modélisation de SAREF sont suggérés au groupe de travaille SDW-WG, et certains ont déjà été intégrés comme la classe *ProcedureExecution* qui généralise les observations et actuations, et la classe *PropertyOfInterest* qui décrit les qualités observables intrinsèquement liées à une entité d'intérêt. De plus, SAREF définit un ensemble de patrons d'ontologie de référence [6] qui contraignent son usage. Ces patrons indiquent quelles sont les classes et propriétés qui peuvent ou ne peuvent peut être spécialisées, et dans quel contexte. L'année 2024 devrait voir la publication d'une nouvelle version majeure (V4.1.1) de SAREF Core où les concepts dépréciés dans V3.2.1 seront supprimés, et une nouvelle version majeure (V2.1.1) de chacune des extensions de domaines conforme à la nouvelle version de SAREF Core et aux différents patrons d'ontologie de référence [4].

# 3 Positionnement pour CoSWoT

## 3.1 Convergence et influence

L'évolution parallèle et actuelle de SOSA/SSN, TD, et SAREF avec des gouvernances distinctes pose des défi concernant l'objectif **O1** pour l'ontologie CoSWoT. Nous contribuons à chacune de ces ontologies, mais il est difficile d'anticiper leur trajectoire d'évolution, du fait qu'elles résultent de décisions plus ou moins consensuelles dans leurs communautés respectives. On note néanmoins un désir de convergence, et des questions de modélisation similaires (voir [10, Section 7]). Par exemple, la question de savoir si une instance de la classe *Property* doit être générique (ex. température ambiante) ou spécifique à une entité d'intérêt (ex. la température du bureau 429) a été longtemps débattue. Il existe des usages pour les deux façons de modéliser [5]. La décision a été prise dans SAREF de restreindre l'emploi de *Property* seulement pour les propriété génériques, et d'introduire *PropertyOfInterest* pour décrire les propriétés spécifiques à une entité d'intérêt. SOSA/SSN a adopté ce choix également. De même, certains utilisateurs décrivent avec *sensor* des types de capteurs (ex. DHT22), alors que d'autres décrivent des instances (ex. le capteur DHT22 dans une pièce). SAREF introduit *FeatureKind* pour décrire des archétypes d'entités d'intérêt, mais pas la sous-classe *DeviceKind* ou *SensorKind*. Le nouveau choix de modélisation est discuté dans SOSA/SSN [6]. Ainsi, parfois le spécifique est défini par le suffixe « OfInterest », parfois le générique est défini par le suffixe « Kind », rendant difficile la compréhension de ces ontologies par un nouvel utilisateur. De plus, la compatibilité arrière est systématiquement rompue avec les usagers minoritaires d'un choix de modélisation. Enfin, il n'est pas encore clair quand et comment ces classes doivent être spécialisées. Doit-on privilégier la définition d'une instance de « Kind », ou d'une sous-classe de « OfInterest » ? Ces entités doivent-elles être définies par paire ? Doit-on utiliser le *punning* systématiquement ? [7] Est-ce pertinent de définir des sous-classes de la classe « Kind » ? Dans l'ontologie CoSWoT, nous proposons un patron de conception architectural unifié « *Kinds of X and X of Interest* », qui mitige ces problèmes et apporte des réponses à ces questions.

## 3.2 Le patron de conception architectural « *Kinds of X and X of Interest* »

L'ontologie CoSWoT prend comme référence les ontologies SOSA/SSN, SAREF et TD. Les classes importantes de SOSA/SSN et SAREF (FeatureOfInterest, Device, Sensor, Property, State, Observation, ...) sont redéfinies dans l'espace de nom de CoSWoT `https://w3id.org/coswot/`. Pour satisfaire l'objectif **O3** nous proposons une modélisation légèrement différente, en suivant systématiquement un patron de conception que l'on nomme « *Kinds of X and X of Interest* », illustré dans la figure 1.

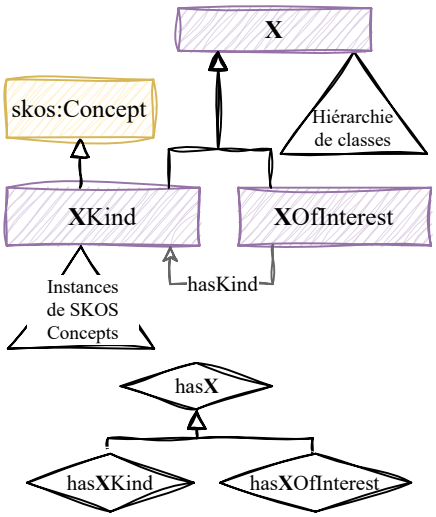


FIGURE 1 – Patron de conception architectural « *Kinds of X and X of Interest* », et l'extension « *has X* »

La classe **X** (ex. *Feature*, *Property*, *Device*) représente la classe des entités dont le type est indéterminé (ni générique, ni spécifique). Elle peut être spécialisée dans une hiérarchie de classe. La classe **X** est équivalente à l'union disjointe de **X**Kind et **X**OfInterest, qui représentent la classe des archétypes de **X**, et des **X** spécifiques, respectivement. **X**Kind est une sous-classe de *skos:Concept*, et les instances qui peuplent cette classe sont organisées en un modèle SKOS à l'aide des propriétés *skos:broader* et *skos:narrower* par exemple. Des restrictions locales sur **X**Kind forcent les concepts plus spécifiques et plus génériques à également être de type **X**Kind.

4. `https://portal.etsi.org/tb.aspx?tbid=726`
5. Voir sdw-sosa-ssn#106
6. Voir sdw-sosa-ssn#107, sdw-sosa-ssn#209
7. *punning* en OWL 2 fait référence à l'utilisation d'une même IRI pour identifier une classe, une propriété, et/ou un individu, sans que ces entités n'aient formellement de lien.

La propriété hasKind lie une entités spécifique à son type. Une restriction locale sur **X**OfInterest force tout objet de la propriété hasKind à être de type **X**Kind. De plus, hasKind est sous-propriété de la chaîne hasKind o skos:broader, de sorte qu'une instance de **X**OfInterest hérite des **X**Kind plus génériques. La propriété hasKind étant « non-simple » [8] de par cet axiome, elle ne doit pas faire l'objet de restrictions de cardinalité. On peut néanmoins ajouter une restriction locale existentielle **X**OfInterest $\sqsubseteq \exists$hasKind.**X**Kind, et une restriction universelle **X**Kind $\sqsubseteq \forall$hasKind.$\bot$. Cette dernière restriction interdit l'utilisation de hasKind sur les instance de type **X**Kind.

Lorsqu'une sous-classe de **X** est définie, il est possible, mais pas obligatoire, d'utiliser le même patron de conception architectural. Par exemple, nous aurons *Sensor*, *SensorKind*, et *SensorOfInterest*, qui seront respectivement des sous-classes de *Device*, *DeviceKind*, et *DeviceOfInterest*.

Finalement, il est commun de devoir modéliser qu'une instance du patron pour **Y** fait référence à une instance du patron pour **X**. Un **X**OfInterest sera spécifique à un unique **Y**OfInterest. Par exemple, une propriété d'intérêt est spécifique à une entité d'intérêt. Le patron propose donc une extension « *has X* » qui définit six propriétés has**X**, is**X**Of, has**X**Kind, is**X**KindOf, has**X**OfInterest, et is**X**OfInterestOf, cette dernière étant fonctionnelle. Trois axiomes de type subPropertyChainOf permettent d'inférer les types :

has**X**Kind $\sqsubseteq$ skos:broader o has**X**Kind

has**X**Kind $\sqsubseteq$ hasKind o has**X**Kind

has**X**Kind $\sqsubseteq$ has**X**OfInterest o hasKind

Même si has**X**Kind est non-simple, la classe **X**OfInterest peut recevoir une restriction local de cardinalité =1 sur is**X**OfInterestOf sans que l'ontologie ne viole les contraintes du profile OWL2 DL.

Ce patron facilite la séparation des préoccupations : les développeurs d'extensions de CoSWoT pourront privilégier la définition de sous-classes de **X**; les développeurs de taxonomies, thésaurus, et catalogues en ligne pourront privilégier l'instanciation des classes **X**Kind; les développeurs d'applications pourront privilégier l'instanciation des classes **X**OfInterest.

# 4 Méthode de développement

Le développement de l'ontologie CoSWoT est réalisé en application de la méthodologie d'ingénierie d'ontologies ACIMOV (Agile Continuous Integration for Modular Ontologies and Vocabularies) [13]. ACIMOV offre aux équipes de développement d'ontologie un cadre agile, et un ensemble de ressources permettant d'accompagner l'entièreté du cycle de vie de l'ontologie, telles que des outils de validation, ou de développement/intégration continus. ACIMOV offre un support à (1) la réutilisation des ontologies de référence, (2) la modularité, (3) la collaboration.

8. Les propriétés non-simple font l'objet de restrictions globales pour satisfaire le profile OWL2 DL

**Réutilisation des ontologies de référence.** Dans CoSWoT, les ontologies SAREF, SOSA/SSN et TD ont été identifiées comme répondant aux besoins exprimés dans les cas d'usages. Elles représentent des références dans le domaine du WoT, ayant un maintien régulier, une documentation riche, et bénéficiant d'une large adoption de la communauté du domaine (besoin **O1**).

**Modularité.** CoSWoT est conçue comme une ontologie modulaire pour permettre une gestion efficace de la complexité du domaine du WoT, et des domaines d'applications sous-jacents. Cette modularité est d'abord horizontale : à l'image des ontologies de référence réutilisées, des modules (appelés modelets dans ACIMOV) sont créés pour capturer des représentations de connaissances complémentaires, chacun conçu autour d'un concept clé, et défini pour répondre à des questions de compétences issues des cas d'application CoSWoT (besoin **O2**). D'autres part, une hiérarchie des modelets est défini; des modelets core sont créés pour consolider les connaissances du domaine du WoT génériques/communes à plusieurs domaines d'applications, et sont par la suite spécialisés selon les spécificités des applications (besoin **O3**). Cette modularité permet aussi de faciliter la réutilisation de certaines parties de l'ontologie, identifiées comme pertinentes pour un cas d'usage particulier à travers la création de vues (besoin **O4**).

Les développeurs d'ontologie constituent un « *backlog* » de modelet, qu'ils prennent en charge de manière parallèle afin d'optimiser le processus de développement.

**Collaboration.** comme introduit plus tôt, le périmètre du projet CoSWoT s'étend à deux domaines d'applications (agriculture numérique et bâtiment intelligent). Les experts de ces domaines ainsi que les « *product owners* » en charge de développement des cas d'usages sont impliqués dans le processus de développement de l'ontologie CoSWoT. En application de ACIMOV, la stratégie d'acquisition des connaissances démarre via la collecte des spécifications des cas d'usage par les experts de domaines, et se poursuit tout au long du cycle de vie de l'ontologie, en affinant les besoins grâce à des réunion récurrentes permettant de valider les modules ontologiques et d'assurer une intégration régulière des feedbacks des différentes parties prenantes de CoSWoT. La collaboration concerne aussi l'équipe de développement d'ontologies qui organise des sessions de revue régulières pour valider les modelets développés.

# 5 Ontologie CoSWoT Core

## 5.1 Aperçu

Le noyau de l'ontologie CoSWoT est compatible OWL2 DL et définit 61 classes, 69 propriétés objet, et 6 propriétés de données. Il contient 14 modelets, 8 instances du patron « *Kinds of X and X of Interest* » dont 4 fois avec l'extension « *has X* ». La figure 2 illustre les relations d'import entre ces modelets. Chaque modelet est accessible à son URL, et la fusion de tous ces modelets est accessible avec négociation de contenu à l'URL `https://w3id.org/coswot/core/`. Chaque terme est défini dans l'espace de nom coswot: (enregistré sur le

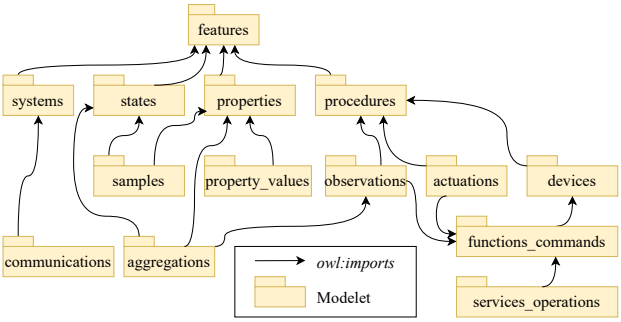


FIGURE 2 – Aperçu des dépendances entre les modelets de l'ontologie CoSWoT

service prefix.cc), et un HTTP GET à son URL redirige vers le modelets qui le définit. Dans le reste de cette section, nous présentons quelques-uns de ces modelets.

## 5.2 Features

Le modelet **coswot:core/features** décrit les entités représentant toute entité du monde réel ayant une propriété ou un état qui sera observé ou contrôlé. Une instance de *coswot:Feature* est soit une instance de *coswot:FeatureOfInterest*, c'est à dire une entité spécifique du monde réel, soit une instance de *coswot:FeatureKind*, c'est à dire un archétype d'entité.

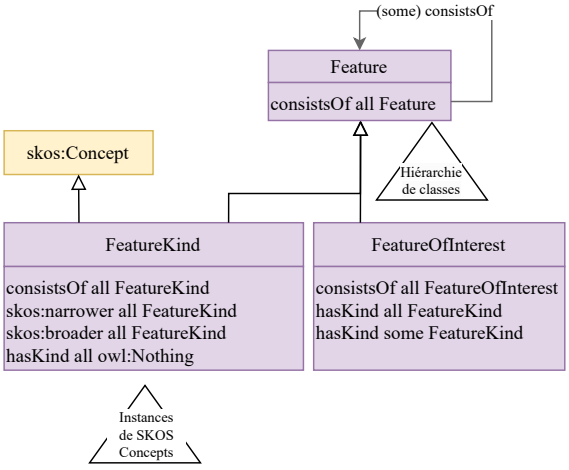


FIGURE 3 – Diagramme Chowlk du modelet Features. NOTE : les diagrammes peuvent être téléchargés et visualisés avec l'application draw.io : `https://drive.usercontent.google.com/uc?id=1YLwbnZobNIHKCXJfScjohD-5jGThZfpf&export=download`

Ainsi dans notre exemple, l'entité objet de l'étude est le bureau 429 du bâtiment EF. Comme le montre la figure 4, la représentation du bureau est une instance de la classe *coswot:FeatureOfInterest*, car cette ressource identifie un objet unique du monde réel. Elle est aussi instance de la classe *coswot:OfficeRoom* sous classe *coswot:Feature*. La ressource indique que ce bureau peut aussi être classé dans la catégorie bureau de 20 m2.

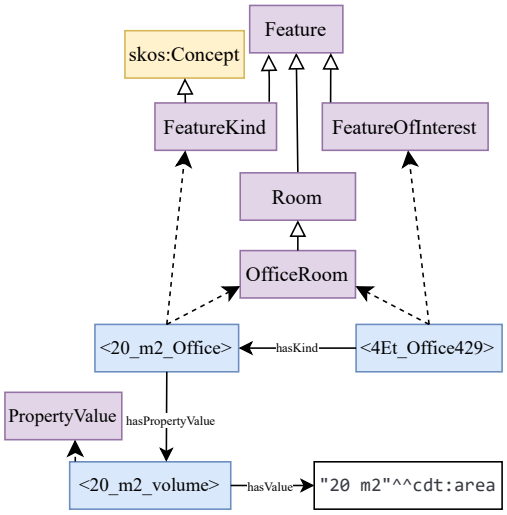


FIGURE 4 – Diagramme Chowlk du bureau 429

## 5.3 Devices

Le modelet **coswot:core/devices** décrit les appareils ou dispositifs, dont les capteurs et les actionneurs. Comme les appareils sont des entités du monde réel, il spécialise le modelet Features. Une représentation d'un appareil est une instance de *coswot:Device*, sous classe de *coswot:Feature*. Cette représentation est soit une instance de *coswot:DeviceOfInterest*, c'est à dire un appareil identifié du monde réel, soit une instance de *coswot:DeviceKind*, c'est à dire un archétype d'appareils.

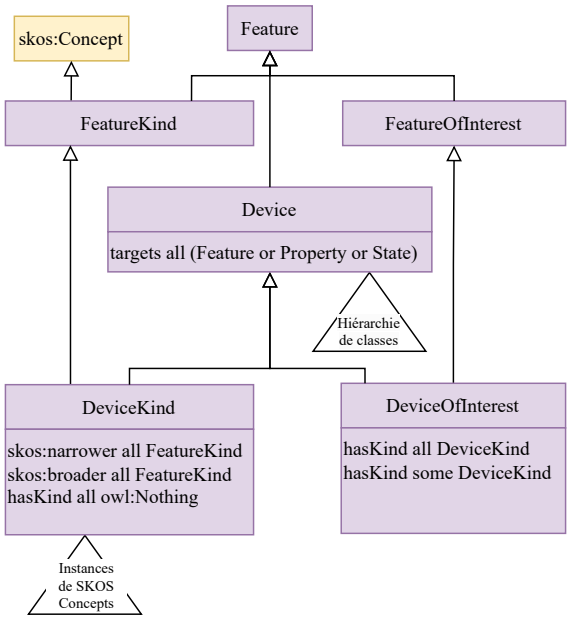


FIGURE 5 – Diagramme Chowlk du modelet Devices

Ainsi dans notre exemple, le bureau 429 est équipé de plusieurs appareils : le radiateur, la fenêtre à ouverture automatique, les capteurs, etc. La figure 6 présente le servient de la fenêtre qui contient deux capteurs. Ainsi, les représentations du servient, des deux capteurs sont des ins-

tances de *coswot:DeviceOfInterest* car ces ressources identifient un objet unique du monde réel. Elles sont aussi instances de sous classes *coswot:Device* : *coswot:Servient*, *coswot:Thermometer*, *coswot:GazSensor*. La ressource représentant le thermomètre indique que ce capteur à une précision de 0.1%.

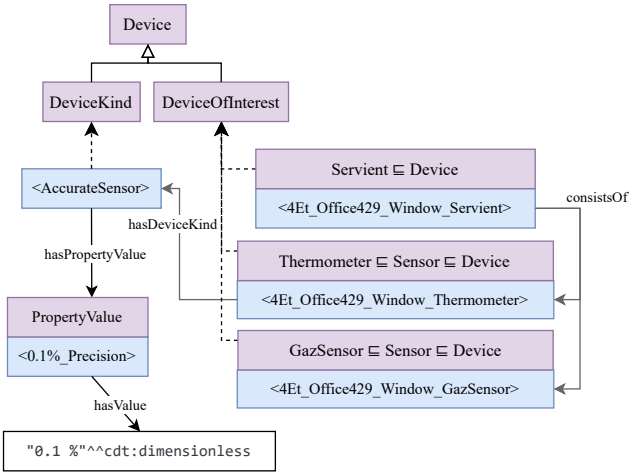


FIGURE 6 – Diagramme Chowlk du servient de la fenêtre

## 5.4 Properties

Le modelet **coswot:core/properties** décrit les propriétés des entités. Les propriétés font référence aux qualités identifiables des entités que des dispositifs peuvent observer. Une instance de *coswot:Property* est soit une instance de *coswot:PropertyOfInterest*, c'est à dire une caractéristique spécifique à une entité identifiée du monde réel, soit une instance de *coswot:PropertyKind*, c'est à dire un archétype de propriété qui peut s'appliquer à différentes entités.

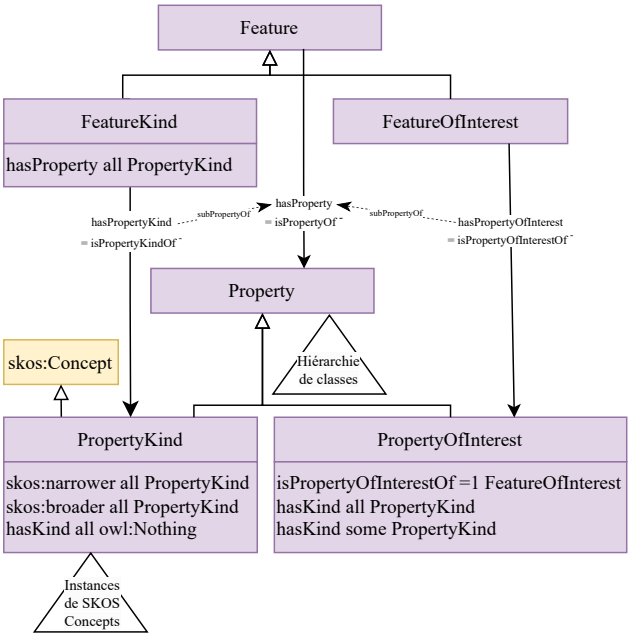


FIGURE 7 – Diagramme Chowlk du modelet Properties

Ainsi dans notre exemple, la propriété que nous modélisons est la température de l'air du bureau 429 du bâtiment EF. Comme le montre la figure 8, cette ressource est une instance de la classe *coswot:PropertyOfInterest* car elle qualifie un objet unique du monde réel. Elle est de type *coswot:AirTemperature* qui est une instance de *coswot:Property* que l'on pourrait trouver dans une taxonomie en ligne. La figure 8 indique les différentes cibles (« target ») du thermomètre.

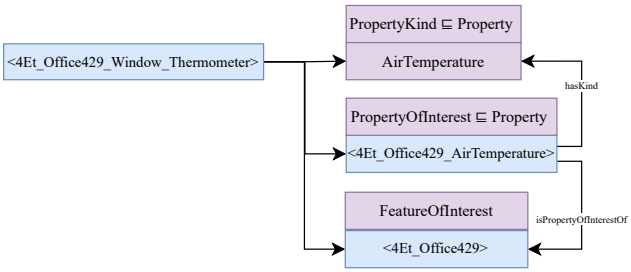


FIGURE 8 – Diagramme Chowlk de la témpérature de l'air du bureau 429

## 5.5 Property Values

Le modelet **coswot:core/property_values** décrit comment indiquer la valeur d'une grandeur physique. Il spécifie également comment associer une grandeur physique à une propriété d'une entité. Une valeur de propriété peut être utilisée pour définir des hiérarchies SKOS dans les archétypes. Une instance de *coswot:PropertyValue* est liée à une valeur et à une unité. Nous recommandons d'utiliser *cdt:ucum* pour associer directement l'unité à la valeur.

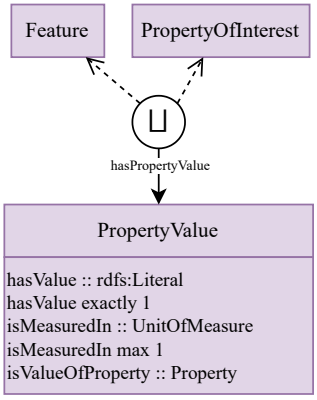


FIGURE 9 – Diagramme Chowlk du modelet Property values

Ainsi, dans nos exemples précédents nous avons utilisé plusieurs fois des instances de *coswot:PropertyValue*, pour définir l'archétype des bureaux de 20 m2 4, l'archétype des capteurs de précision 11.

## 5.6 Samples

Le modelet **coswot:core/samples** décrit les échantillons des entités d'intérêt c'est à dire les parties de l'entité d'intérêt (en général des prélèvements) sur lesquelles les observa-

tions seront effectuées. Une instance de *coswot:Sample* est forcément une instance de *coswot:FeatureOfInterest*. Dit autrement, un échantillon est forcément un objet identifié du monde réel.

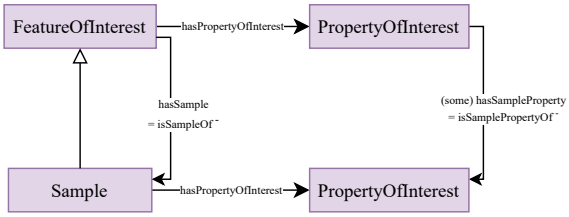


FIGURE 10 – Diagramme Chowlk du modelet Samples

Ainsi dans notre exemple, les capteurs qui mesurent la température de l'air ou la concentration de CO2 contenue dans l'air, n'observent que l'air situé dans leur environnement et non pas l'intégralité de l'air dans le bureau 429. Par convention des constructeurs de capteurs, l'échantillon d'air observé correspond à un volume d'air d'1 m3 situé autour de la sonde [9]. Comme le montre la figure 11, un échantillon est créé pour chaque capteur, instance de la classe *coswot:Sample* avec ses propriétés associées.


FIGURE 11 – Diagramme Chowlk d'un échantillon d'air observé par un capteur de CO2 dans le bureau 429

## 5.7 Procedures

Le modelet **coswot:core/procedures** décrit les procédures et leurs exécutions. Les exécutions de procédures généralisent les observations, les prédictions, et les actuations. Il s'agit d'un patron général représentant toute exécution d'une procédure définie qu'elle soit exécutée par un appareil ou par un humain. Une instance de *coswot:ProcedureExecution* a une date de début et une date de fin. Elle est liée à l'entité *coswot:FeatureOfInterest* qui l'exécute et à sa procédure *coswot:Procedure*. Une procédure peut prendre la forme d'un algorithme pour un appareil, ou d'un guide méthodologique pour un humain.

## 5.8 Observations

Le modelet **coswot:core/observations** définit les capteurs et décrit l'exécution de procédures d'observations. Il spécialise le modelet procédure et fonctions_commandes. Une instance de *coswot:Observation* doit forcément observer au

9. Nous pourrions construire un archétype de capteur ayant un échantillon d'1 m3 qui serait associé à tous les capteurs du bâtiment, ce qui n'est pas très discriminant

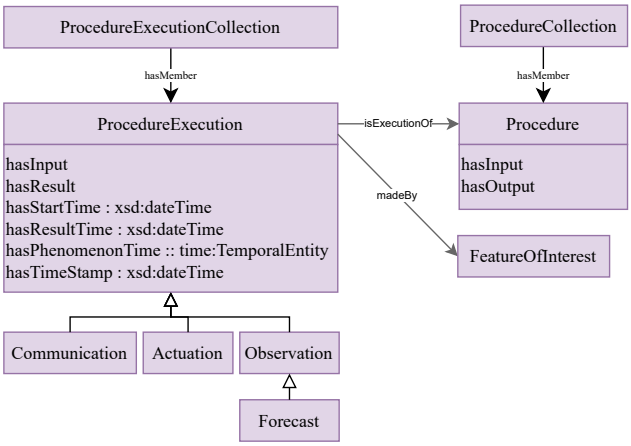


FIGURE 12 – Diagramme Chowlk du modelet Procedures

moins une chose que ce soit une propriété *coswot:Property*, une entité *coswot:Feature* ou un état *coswot:State*.

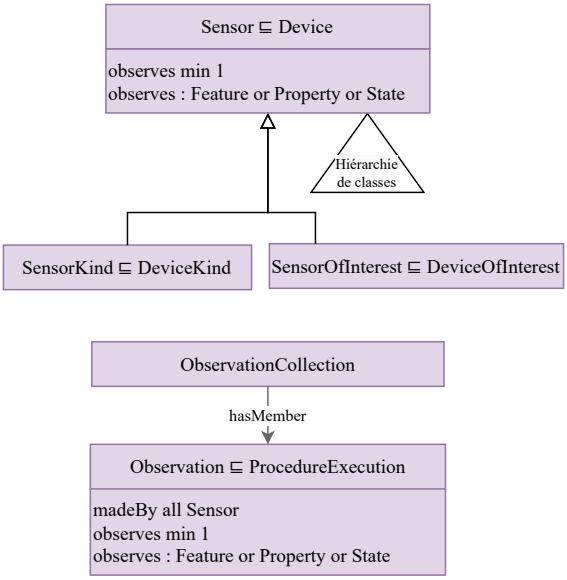


FIGURE 13 – Diagramme Chowlk du modelet Observations

Nous prenons l'exemple de la figure 14 qui décrit la mesure effectuée par le capteur de CO2 de la fenêtre. La représentation du capteur est instance des classes *coswot:DeviceOfInterest* et *coswot:GazSensor*. Cette mesure porte sur l'échantillon d'air localisé autour de la fenêtre, dont la représentation est instance de la classe *coswot:Sample*. cet échantillon a une propriété identifiée, instance des classes *coswot:PropertyOfInterest* et *coswot:CarbonDioxydeConcentrationInAir*. Cette mesure a pour résultat 1500 ppm. Il s'agit d'une mesure instantanée effectuée le 6 juillet 2021 à 09 :30 et 02 secondes.

## 5.9 Aggregation

Le modelet **coswot:core/aggregations** décrit les agrégations qui sont une spécialisation des observations. Une instance de *coswot:Aggregation* prend en entrée un ensemble d'observations représentée par une instance de la classe

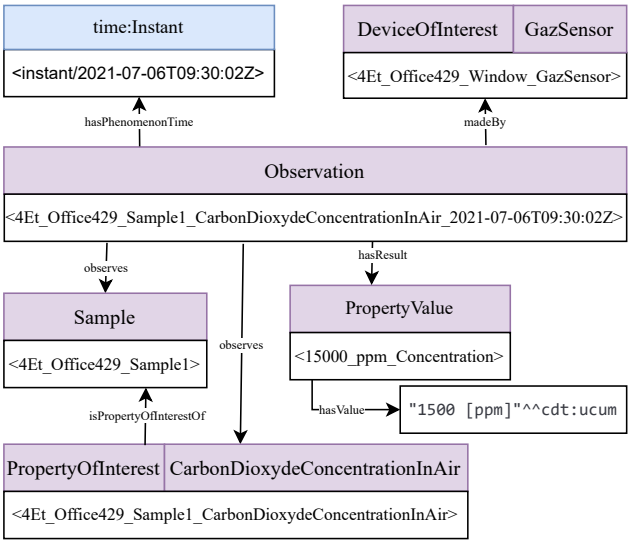


FIGURE 14 – Diagramme Chowlk d'une mesure du capteur de CO2 de la fenêtre

*coswot:ObservationCollection*, et produit un résultat, représenté par une instance de *coswot:PropertyValue*.

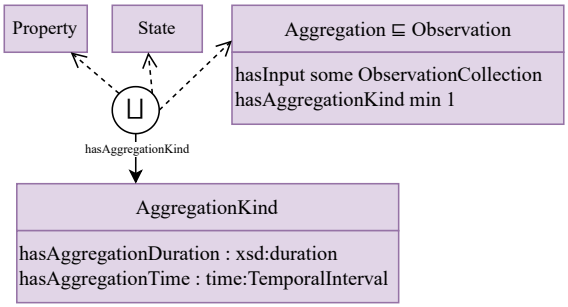


FIGURE 15 – Diagramme Chowlk du modelet aggregations

Dans notre exemple représenté sur la figure 16, notre application agrège les mesures instantanées des 3 capteurs de CO2. Ces capteurs mesurent tous à une fréquence de 5 secondes et les servients de la porte et du tableau envoient instantanément leur mesure au servient de la fenêtre. Ainsi, le servient de la fenêtre agrège les mesures produites dans une fenêtre temporelle de 5 secondes. Cette agrégation prend en entrée les trois mesures de capteurs de CO2 regroupées dans une collection d'observations non représentées dans le diagramme. Cette agrégation observe le bureau 429 et plus particulièrement la propriété identifiée, instance des classes *coswot:PropertyOfInterest* et *coswot:CarbonDioxydeConcentrationInAir*.

## 5.10 Communication

Le modelet **coswot:core/communications** vise à décrire les communications, et les systèmes de communications. Il spécialise le modelet procedures. Les servients de l'architecture CoSWoT représentent des systèmes de communications qui s'échangent des messages, chacun représentant l'identifiant d'un graphe nommé, comportant l'information

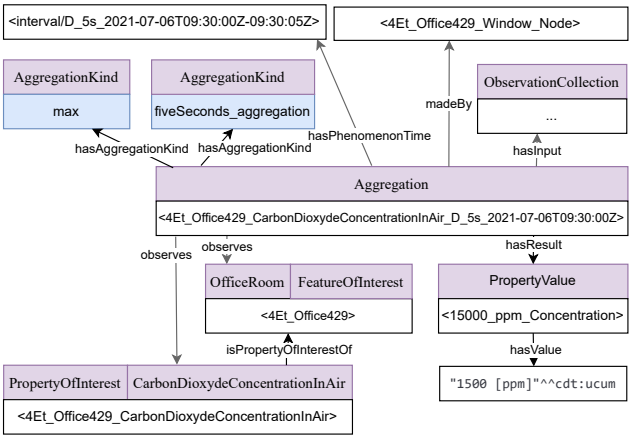


FIGURE 16 – Diagramme Chowlk de l'agrégation des mesures des capteurs de CO2 du bureau 429

échangée.

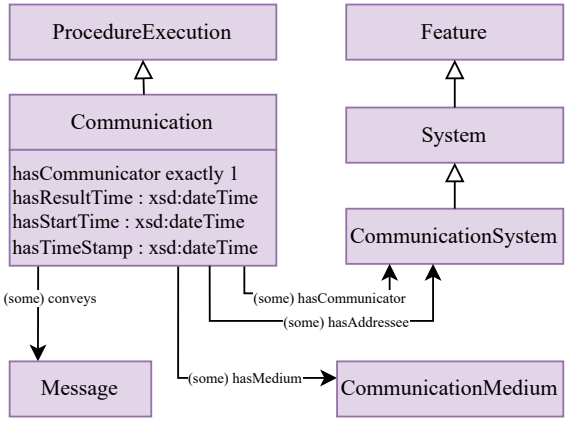


FIGURE 17 – Diagramme Chowlk du modelet communications

Dans notre exemple de la figure 18, le servient de la porte envoie l'identifiant du graphe nommé contenant ces deux dernières mesures : mesure de concentration de CO2 et température. Cette communication a deux destinataires, le servient du radiateur et le servient de la fenêtre. Le communication se fait sur le réseau eduroam.

# 6 Conclusion

Cet article présente l'ontologie CoSWoT, qui vise à améliorer l'interopérabilité sémantique dans le domaine du Web des Objets, en considérant le contexte des applications constituées d'objets distribués sur le Web. Dans ce cadre, la conception de l'ontologie CoSWoT réutilise les ontologies de références du domaine du WoT : SSN/SOSA, TD, et SAREF. Le développement de l'ontologie CoSWoT prend appui sur une méthodologie agile d'ingénierie d'ontologies, ACIMOV, pour permettre la réutilisation des ontologies références, tout en garantissant le respect des principes de modularité, d'agilité, et bénéficier d'outils d'intégration et de développement continus. L'un des objectifs de l'ontolo-

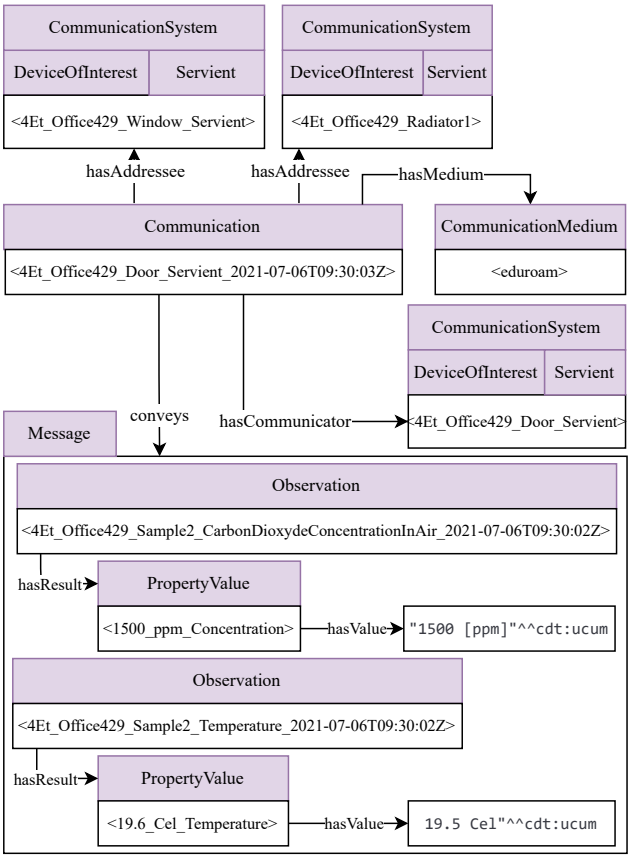


FIGURE 18 – Diagramme Chowlk de la communication entre deux servients

gie CoSWoT est d'intégrer les ontologies références, tout en réconciliant la définition des concepts clés du domaines. Ce travail a donnée lieu au patron de conception architectural « *Kinds of X and X of Interest* », qui différencie les concepts clés de ces ontologies (ex. *FeatureOfInterest*, *Device*,..), en distinguant leurs instances spécifiques de celles génériques. L'usage de ce patron a pour but de clarifier la représentation des concepts et de faciliter l'usage des propriétés associées. Sur la base de ce patron, CoSWoT est conçue comme un ensemble de modelets (modules) core regroupant les concepts WoT communs à tous les domaines d'applications verticaux. Des vues de l'ontologie sont par la suite réalisées en spécialisant une partie des modelets, pour permettre de répondre aux besoins du domaine d'application objet de l'étude. Cela a notamment été illustré dans l'article sur le cas du bâtiment intelligent EF.

Le travail sur l'ontologie CoSWoT se poursuit pour concrétiser son usage dans plus de cas d'usage (issus de l'agriculture numériques par exemple). Il permettrait à terme de formuler des recommandations quant à l'intégration des spécificités des objets contraints dans les ontologies de référence, renforçant l'interopérabilité sémantique au delà des architectures centralisées.

# Remerciements

Le projet CoSWoT est financé par l'agence nationale de la recherche sous la référence ANR-19-CE23-0012.

# Références

[1] V. Charpenay and M. Kovatsch. Hypermedia Controls Ontology, Editor draft, 10 May 2023. W3c working group draft, W3C, May 2023.

[2] V. Charpenay, M. Lefrançois, and M. Poveda Villalón. JSON Schema in RDF, Editor draft, 10 May 2023. W3c working group draft, W3C, May 2023.

[3] V. Charpenay, M. Lefrançois, M. Poveda Villalón, and S. Käbisch. Thing Description (TD) Ontology, Editor draft, 10 May 2023. W3c working group draft, W3C, May 2023.

[4] Simon Cox. Observations and Measurements. OGC project document 10-025r1, OGC, March 2011.

[5] Laura Daniele, Frank den Hartog, and Jasper Roes. Created in close interaction with the industry : the smart appliances reference (saref) ontology. In *FOMI 2015*, pages 100–112. Springer, 2015.

[6] ETSI TC SmartM2M. SmartM2M ; SAREF reference ontology patterns. Technical Specification ETSI TS 103 548 V1.2.1, ETSI, January 2024.

[7] ETSI TC SmartM2M. SmartM2M ; Smart Applications ; Reference Ontology and oneM2M Mapping. Technical Specification ETSI TS 103 264 V3.2.1, ETSI, January 2024.

[8] Isaac Fatokun, Arun Raveendran Nair Sheela, Thamer Mecharnia, Maxime Lefrançois, Victor Charpenay, Fabien Badeig, and Antoine Zimmermann. Modular knowledge integration for smart building digital twins. In *LDAC'23*, 2023.

[9] Raúl García-Castro, Maxime Lefrançois, María Poveda-Villalón, and Laura Daniele. The ETSI SAREF ontology for smart applications : a long path of development and evolution. In *Energy Smart Appliances : Applications, Methodologies, and Challenges*. Wiley, 2023.

[10] 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.

[11] Armin Haller, Krzysztof Janowicz, Simon JD Cox, Maxime Lefrançois, Kerry Taylor, Danh Le Phuoc, Joshua Lieberman, Raúl García-Castro, Rob Atkinson, and Claus Stadler. The sosa/ssn ontology : a joint w3c and ogc standard specifying the semantics of sensors, observations, actuation, and sampling. *Semantic Web-Interoperability, Usability, Applicability an IOS Press Journal*, 56, 2019.

[12] 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.

[13] Fatma-Zohra Hannou, Victor Charpenay, Maxime Lefrançois, Catherine Roussey, Antoine Zimmermann, and Fabien Gandon. The ACIMOV Methodology : Agile and Continuous Integration for Modular Ontologies and Vocabularies. In *2nd Workshop on Modular Knowledge associated with FOIS 2023*, 2023.

[14] 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.

[15] Sebastian Kaebisch, Takuki Kamiya, Michael McCool, and Victor Charpenay. Web of Things (WoT) Thing Description. Candidate Recommendation, W3C, May 16 2019.

[16] M. Koster and E. Korkan. Web of Things (WoT) Binding Templates. W3c working group note, W3C, January 2020.

[17] Ryuichiand Kovatsch, Matthiasand Matsukura, Michael Lagally, Toru Kawaguchi, Kunihiko Toumura, and Kazuo Kajimoto. Web of Things (WoT) Architecture. Candidate Recommendation, W3C, May 16 2019.

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

[19] Nicolas Seydoux, Maxime Lefrançois, and Lionel Médini. Positionnement sur le Web Sémantique des Objets. In *IC 2019*, pages 8–25, July 2019.