Project Acronym:	**BIMERR**
Project Full Title:	**BIM-based holistic tools for Energy-driven Renovation of existing Residences**
Grant Agreement:	**820621**
Project Duration:	**42 months**

DELIVERABLE D4.2
BIMERR Ontology & Data Model 1

Deliverable Status:	**Draft**
File Name:	**D4.2 v0.13 final.docx**
Due Date:	**31/03/2020 (M15)**
Submission Date:	**26/03/2020 (M15)**
Task Leader:	**UPM (T4.2)**

Dissemination level	
Public	X
Confidential, only for members of the Consortium (including the Commission Services)	

This project has received funding from the European Union's Horizon 2020 Research and innovation programme under Grant Agreement n°820621

The BIMERR project consortium is composed of:		
FIT	Fraunhofer Gesellschaft Zur Foerderung Der Angewandten Forschung E.V.	Germany
CERTH	Ethniko Kentro Erevnas Kai Technologikis Anaptyxis	Greece
UPM	Universidad Politécnica De Madrid	Spain
UBITECH	Ubitech Limited	Cyprus
SUITE5	Suite5 Data Intelligence Solutions Limited	Cyprus
HYPERTECH	Hypertech (Chaipertek) Anonymos Viomichaniki Emporiki Etaireia Pliroforikis Kai Neon Technologion	Greece
MERIT	Merit Consulting House Sprl	Belgium
XYLEM	Xylem Science And Technology Management Gmbh	Austria
CONKAT	Anonymos Etaireia Kataskevon Technikon Ergon, Emporikon Viomichanikonkai Nautiliakon Epicheiriseon Kon'kat	Greece
BOC	Boc Asset Management Gmbh	Austria
BX	Budimex Sa	Poland
UOP	University Of Peloponnese	Greece
UOE	University of Edinburgh	United Kingdom
NT	Novitech As	Slovakia
FER	Ferrovial Agroman S.A	Spain

Disclaimer

BIMERR project has received funding from the European Union's Horizon 2020 Research and innovation programme under Grant Agreement n°820621. The sole responsibility for the content of this publication lies with the authors. It does not necessarily reflect the opinion of the European Commission (EC). EC is not liable for any use that may be made of the information contained therein.

AUTHORS LIST

Leading Author (Editor)			
Surname	**First Name**	**Beneficiary**	**Contact email**
Chávez	Serge	UPM	schavez@delicias.dia.fi.upm.es

Co-authors (in alphabetic order)				
#	**Surname**	**First Name**	**Beneficiary**	**Contact email**
1	Bosché	Frédéric	UoE	f.bosche@ed.ac.uk
2	Bountouni	Nefeli	Suite5	nefeli@suite5.eu
3	Fenz	Stefan	Xylem	fenz@xylem-technologies.com
4	García-Castro	Raúl	UPM	rgarcia@fi.upm.es
5	Giannakis	Giorgos	Hypertech	g.giannakis@hypertech.gr
6	González-Gerpe	Salvador	UPM	sgonzalez@delicias.dia.fi.upm.es
7	Kollmer	Stephan	FIT	stephan.kollmer@izb.fraunhofer.de
8	Kousouris	Spiros	Suite5	spiros@suite5.eu
9	Ntalaperas	Dimitris	Ubitech	dntalaperas@ubitech.eu
10	Thanos	Tsakiris	CERTH	atsakir@iti.gr
11	Lampathaki	Fenareti	Suite5	fenareti@suite5.eu
12	Poveda-Villalón	María	UPM	mpoveda@fi.upm.es
13	Valero	Enrique	UoE	e.valero@ed.ac.uk
14	Vergeti	Danai	Ubitech	vergetid@ubitech.eu
15	Tavakolizadeh	Farshid	FIT	farshid.tavakolizadeh@fit.fraunhofer.de

REVIEWERS LIST

List of Reviewers (in alphabetic order)				
#	**Surname**	**First Name**	**Beneficiary**	**Contact email**
1	Fenz	Stefan	Xylem	fenz@xylem-technologies.com
2	Bosché	Frédéric	UOE	f.bosche@ed.ac.uk

REVISION CONTROL

Version	Author	Date	Status
0.01	UPM	08-12-2019	Added ToC
0.02	UPM	02-01-2020	Added methodology draft
0.03	UPM	20-02-2020	Added technology Revised methodology
0.04	Suite5, UPM, UoE, Xylem, Hypertech, CERTH, Ubitech, FIT	01-03-2020	Added first version of section 3
0.05	Suite5, UPM, UoE, Xylem, Hypertech, CERTH, Ubitech, FIT	04-03-2020	Revised version of section 3
0.06	Suite5, UPM, UoE, Xylem, Hypertech, CERTH, Ubitech, FIT	06-03-2020	Pre-internal review
0.07	UPM, Suite5	09-03-2020	Added data model to ontology converter description.
0.08	UPM, Suite5	11-03-2020	Added ontology and data model alignment section
0.09	UPM	11-03-2020	Submission to internal review
0.10	Xylem	16-03-2020	Internal review
0.11	UOE	17-03-2020	Internal review
0.12	Suite5	18-03-2020	General review
0.13	UPM	19-09-2020	Final version addressing internal review comments

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

Acronym	Meaning
RenoDSS	Renovation Decision Support System
PWMA	Project Workflow Management
PRUBS	Profile Resident Usage of Building System
BICA	Building Information Collection Application
ARIFBA	Augmented Reality enabled In-situ Building Feature Annotation
BIF	BIMERR Interoperability Framework
HVAC	Heating, Ventilating and Air Condition
IFC	Industry Foundation Classes
JSON	JavaScript Object Notation
ORSD	Ontology Requirement Specification Document
OWL	Web Ontology Language
obXML	Occupant Behaviour XML
XML	eXtensible Markup Language
SenML	Sensor Measurement List
EPW	Energy Plus Weather
BIM	Building Information Modelling
IoT	Internet of Things
KPI	Key Performance Indicator
UML	Unified Modelling Language
URI	Uniform Resource Identifier
IDF	Intermediate Data Format
H&S	Health and Security
WGS	World Geodetic System
XSD	XML Schema Definition
LOT	Linked Open Terms
JSON-LD	JavaScript Object Notation for Linked Data
TTL	Terse RDF Triple Language
RDF	Resource Description Framework
N3	Notation 3 Serialization
BO2DM	BIMERR Ontology to Data Model

EXECUTIVE SUMMARY

This deliverable aims at describing the methodology followed for the design, implementation and publication of the BIMERR ontology and data model. To this direction, the document starts with a detailed description of the different phases carried out during the ontological development process. Note that due to the project's complexity and number of domains, it was necessary to adapt existing methodologies.

The document also covers the alignment and translation from the BIMERR ontology to a JSON serialized data model. First, the BIMERR data model structure is explained highlighting how both conceptualizations are related. The initial assessment allows to identify which component need to be incorporated within the ontology that will help during the transformation process. The section also includes the set of assumptions and rules to follow for an unambiguous translation.

Every subdomain of the BIMERR ontology is described in detail, indicating its requirements and the current model description. This process has been conducted taking as input the set of ontologies and data models identified in D4.1 "Report on Semantic Alignment & Linking of EEB-related Ontologies". The domains covered by the current BIMERR ontology network and described in this document are four, namely: Occupancy Profile, Sensor Data, Key Performance Indicator and Weather. The current ontologies are available online in different formats in the BIMERR ontology network portal[1]. At this point, it is worth mentioning that ontology development is an iterative process and that new concepts and relations can be discovered as new needs appear from the different BIMERR applications. Therefore, such ontologies might evolve during next months in parallel with the definition and implementation of the pending ontologies.

The document finally summarizes the status of every domain module, and the actions that need to be carried out in order to finally develop the complete BIMERR model. The results of the continuous activity in T4.2, that is the final ontologies and data model, will be reported in D4.3 in M30.

[1] https://bimerr.iot.linkeddata.es/

1. INTRODUCTION

The BIMERR interoperability approach relies on a common data model and an ontology to be shared and understood by all components connected to the BIMERRR Interoperability Framework (BIF). More precisely, the common data model is derived as a JSON version from the BIMERR ontology. Information technologies took the term "ontology" from philosophy and redefined it as "formal, explicit specifications of a shared conceptualization [Studer et. a., 1998]. In the case of the BIMERR ontologies, they will be formalized following Description Logics and implemented in the W3C Web Ontology Language standard OWL.[2] As there is a broad number of domains to be considered in the BIMERR conceptualization, the ontology to be developed will be organized as a modular ontology network. Some modules, or parts of them, will be aligned with existing standards or well-known ontologies that will be reused whenever possible. A number of models and ontologies to be reused were already analyzed in D4.1 [Tiwari et.al. 2019].

The goal of this deliverable is to describe (a) the methodology, and the technological support, used for the ontology development process; (b) the data model generation, lifecycle and alignment with the ontologies; and (c) the current status of the BIMERR ontologies developed for the first release. This decision has been made by the BIMERRR consortium during the M12 plenary meeting to follow an iterative and evolving process in which the ontology and data model development would be aligned with the readiness of both the data to be shared and the applications to use such data through the BIF. Finally, this document presents some conclusions about the steps followed and proposes next steps.

For the sake of understandability, readers not familiar with the concept of ontologies in computer science and the web are referred to "Ontology Development 101: A Guide to Creating Your First Ontology"[3] as basic reference for this topic. In addition, deliverables D3.1 "Stakeholder requirements for the BIMERR system" [Tsoulos et.al. 2019] and D4.1 represent the supporting material and previous references useful for following this

[2] https://www.w3.org/TR/owl-ref/

[3] http://www.ksl.stanford.edu/people/dlm/papers/ontology101/ontology101-noy-mcguinness.html

document. As a reminder from D4.1 to the reader, an overview of the relation between tasks from WP3 and WP4 in relation to the ontology and the data model definition is depicted in Figure 1. As it can be observed, in T4.1"Analysis of EEB-related Ontologies and their Semantic Links" the status of the architecture (at the moment of developing T4.1) and the BIMERR uses cases were used to define the ontology scope and an initial set of high level data requirements. In addition, during T4.1, the initial set of requirements were matched to the list of ontologies and resources gathered from D3.2 "Survey of data models, ontologies and standards in the wider Energy Efficient Buildings domain" [Poveda-Villalón et.al., 2019b] to identify potential resources to be reused during the ontology and data model development. Such preliminary exploratory work is taken as input in T4.2 development. Along this document it will be shown that some of the resources analysed in D4.1 are directly reused in BIMERR ontologies (for example the SAREF family of ontologies) or need additional transformations to be taken as input for the BIMERR model (as the obXML standard).

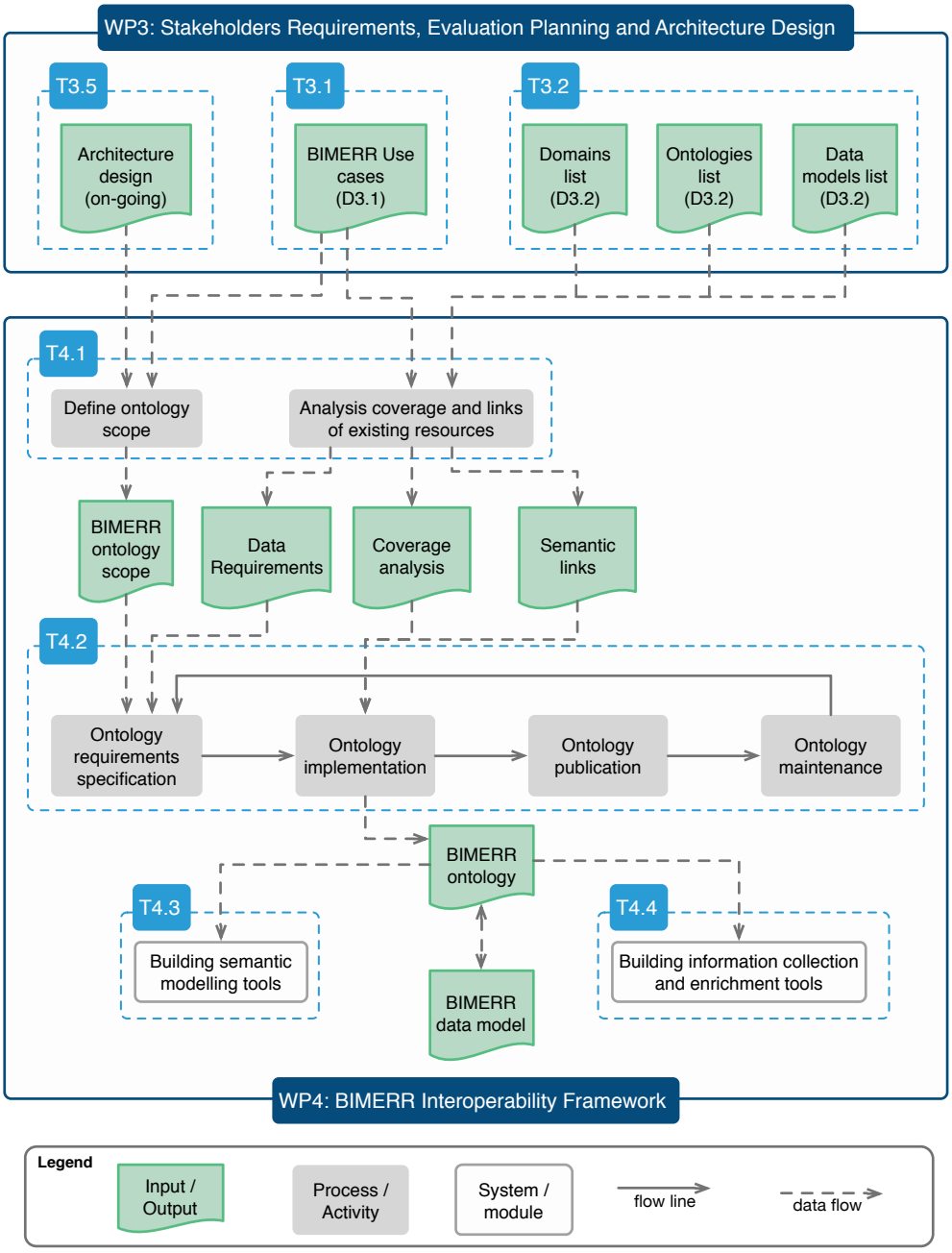

Figure 1. Overview of the relation between WP3 and WP4 tasks related to the ontology and data model definition. Taken from BIMERR D4.1

2. ONTOLOGY DEFINITION METHODOLOGY

This section presents the proposed methodology for ontology development and data model generation to be used within the BIMERR data interoperability approach. This methodology is based on the LOT methodology[4] [Poveda-Villalón et. al., 2019a] which was originally proposed in [Poveda-Villalón, 2012] and [García-Castro et. al., 2017]. It is worth mentioning that some parts of the methodology have been adapted or specialized for the BIMERR project needs and will be documented in the corresponding sections.

The BIMERR ontology is planned to be developed as a network of ontologies or ontology modules that are linked through the use of common concepts and relations. Every domain will be implemented following the workflow shown in Figure 2. Figure 2 also shows the actors involved in each activity and the artefacts generated as output. For example, in the "Ontological requirements specification" activity, the involved actors are the ontology users (for example software developers), domain experts and ontological engineers, meanwhile the main input are documents containing the data sharing needs represented in the form of Competency Questions [Grüninger and Fox, 1995] and tables that will be later detailed.

For each activity presented in Figure 2, the methodology proposes a set of sub-activities and the related technological systems that are available to support some of the activities to be carried out. Each activity will be presented in detail in the following sub-sections.

[4] https://lot.linkeddata.es/

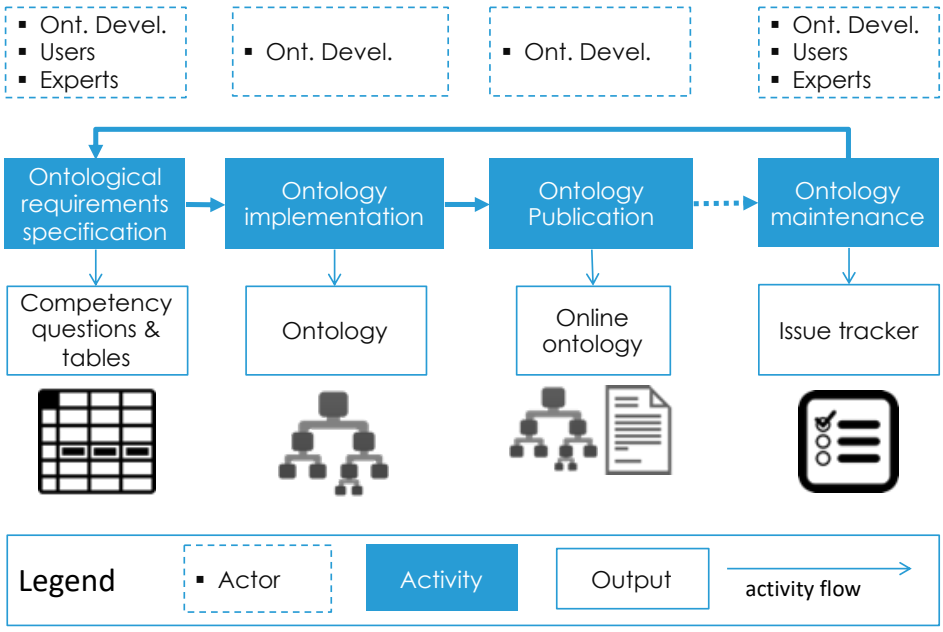

Figure 2. General overview of the ontology development process.

2.1 ONTOLOGY REQUIREMENT SPECIFICATION

The aim of the requirements specification process is to state why the ontology is being built and to identify and define the requirements the ontology should fulfill. In this step, involvement and commitment from experts in the specific domain at hand is required to generate the appropriate industry perspective and knowledge. The activities proposed for the ontology requirement specification process are shown in Figure 3.

For the case of BIMERR, a top-down approach for identifying domains and extracting the requirements has been followed, aligning the activities presented in Figure 3 with the project's lifecycle. In this sense, the process followed in the top-down approach is:

1. Identification of main domains related to the project scope and data to be shared. These domains were proposed in D3.2 [Poveda-Villalón et.al., 2019b] together with a list of related data models, ontologies and standardization bodies. The list of domains identified in D3.2 is: Building, Materials, Energy consumption, Usage patterns and habits, Weather, Reality capture, and Geographical domain.
2. The list of domains to be considered were refined during T4.1, in which the use cases defined for BIMERR in D3.1 [Tsoulos et.al., 2019] were analysed from the point of view of data exchange needs. From this activity a new domain was

identified, namely "Project Management". This activity aligns the "**Use case specification**" and the "**Purpose and scope identification**" activities shown in Figure 3 with the project's lifecycle and reports. In addition, in D4.1, a set of 48 data exchange requirements were identified that served as basis for the detailed requirements to be generated during T4.2.

3. Finally, during T4.2 the data exchange requirements were analysed at the application input/output level in order to define specific low-level requirements to generate the Ontology Requirement Specification Document (ORSD). The process is explained in the rest of this section.

This top-down approach has been useful not only for identifying domains in early stages of the project, but also to identify potential ontologies to be reused and non-ontological resources and standards to be transformed. More precisely, in this document it will be shown that some ontologies already listed in D3.2 were reused like the SAREF and some SAREF extensions and non-ontological resources were transformed as the obXML model.

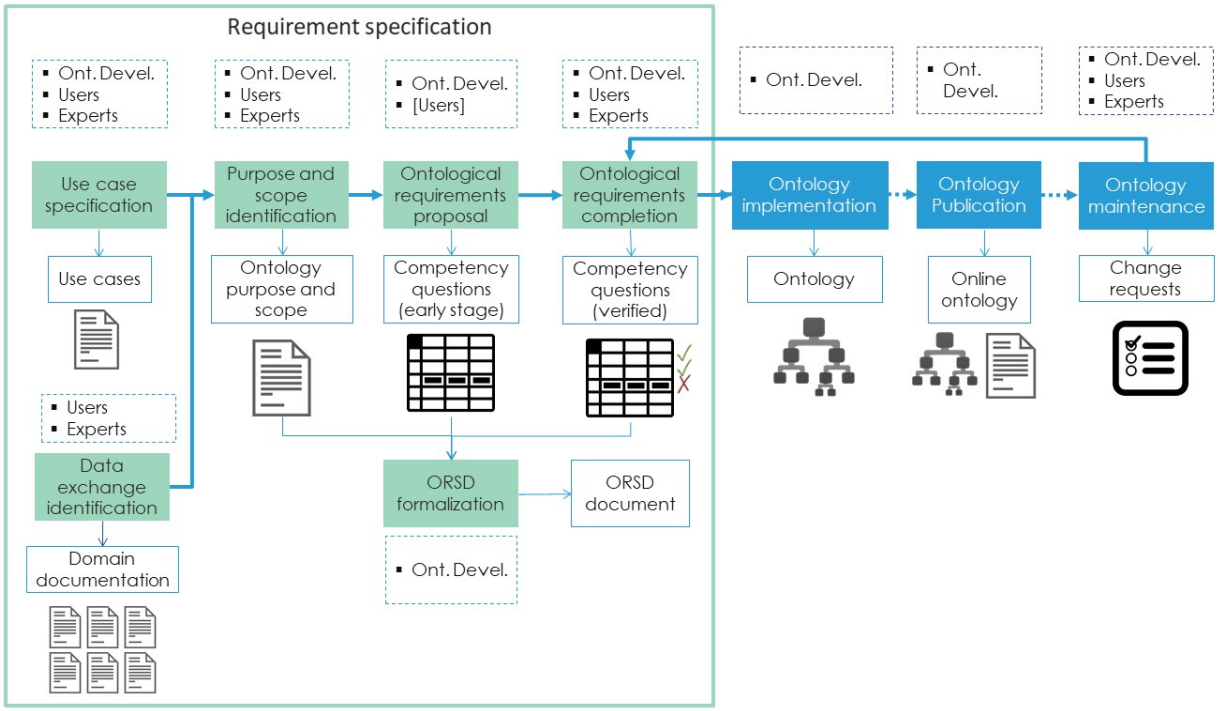

Figure 3. Workflow proposed for ontology requirement specification

The goal of the "**data exchange identification**" activity is to provide the ontology development team with the necessary documentation about the domain to be modeled.

In this case, the documentation to be shared might correspond to: manuals, API's specifications, datasets, standards, formats, etc. The domain experts are the responsible actors for providing this documentation.

The data exchange is normally done between a couple of systems. However, in BIMERR there are numerous combinations of applications and modules providing and requesting data about a number of domains. Due to this situation, the data exchange identification activity has been articulated around weekly teleconference meetings attended by partners responsible for modules and applications that need to exchange data through BIF and partners involved in the BIF development. Prior to the meetings, the following decisions were taken:

- The data to be modelled by the ontology and therefore the BIMERR data model is restricted to:
    a) the data shared through the BIF
        - The data shared through the BIF should be used by at least one BIMERR component or application other than the one providing the data.
    b) the level of details of the data to be queried by the BIF
        - The level of details of the ontology and the data model depend on the type of information to be shared. In some cases, the data details need to be modelled in order to be queried. In other cases data payloads might be treated as information objects for which the information required in the BIF stands at metadata level.

In order to identify the specific domains finally needed for data exchange, two approaches were followed: 1) identification of data exchange pairs between components in the BIMERR architecture and 2) brainstorming between partners to identify data needs for the applications each one is responsible for and to identify potential providers of the needed data from other applications or components in BIMERR. Even though it is clear for most applications and components which data is requires as input, the connections from and to the BIF in the BIMERR architecture diagram were used to promote dialogue among partners and clarify their implicit and explicit data exchange needs. The meetings and communication between partners led us to the conclusion that the domains shown

in Table 1 are those needed to be exchanged though the BIF. This table also shows the expected data origin and applications using them, as well as existing data models or specifications and formats in which the original data might be represented. For each of the domains appearing in the table, specific calls or conversations have taken place in order to define the level of details needed to be included in the ontology and data model.

Table 1. Relation of data domains to be exchanged though the BIF

Domain	Provided by	Needed by	Model/Spec	Formats
Typical Meteorological Year Weather Data	External	RenoDSS	Energy Plus Weather	epw
Weather Forecasts	External	PWMA		json
Usage Patterns & Habits – Occupant Behavior Modeling	PRUBS	RenoDSS	obXML	xml
KPIs	RenoDSS	PWMA RenoDSS		json
Building (Location)	BICA	ARIBFA		json
Building (Geometry)	Scan-to-BIM	Scan-to-BIM ARIBFA RenoDSS BICA	IFC	ifc
Building (Elements and construction)	Scan-to-BIM ARIBFA External	RenoDSS ARIBFA BICA	IFC	ifc
Building (Materials properties)	External	RenoDSS ARIBFA	DDBB model	Ifc, idf
Building Energy Systems (HVAC & Electric Equipment)	Scan-to-BIM ARIBFA External	RenoDSS ARIBFA BICA	IFC	ifc
HVAC & Electric Equipment properties	External	RenoDSS ARIBFA	DDBB model	Ifc, idf
Annotated information objects	ARIBFA PWMA (and End user apps)	BICA		json
Renovation measures	RenoDSS	PWMA		json
Sensor Data	Middleware	PRUBS ARIBFA	SenML	json
H&S Issues	BICA	BICA PWMA		json

Other domains that appeared during the analysis as part of BIMERR but are not expected to be shared by applications through the BIF are: gateways and sensor registry, project management, renovation process, execution results and occupant profiles. As a consequence, in the planning for the ontology and data model development, these domains are not included, therefore no specific requirements are extracted at the moment.

As mentioned, for the general domains shown in Table 1, the specific data exchange level should be defined, that is the "**ontological requirements proposal**" should be carried out. In general, the ontology development team and ontology users generate a proposal for the ontology requirements following the Competency Questions technique, to be complemented by domain experts and users. However, due to the BIMERR project characteristics (complexity of the domains, original structure of the data, and data model characteristics) this activity has been carried out by asking data providers and consumers to agree on the data to be exchanged and fill in for each case three different templates. Such templates were proposed following the METHONTOLOGY [Fernández-López et. al., 1997] methodology for building ontologies and adapting them according to some BIMERR data model characteristics. More precisely the three templates are tables to represent concepts, attributes and relations, as shown in Figure 4. The templates were shared and filled in within the BIMERR Confluence workspace.

The first table ("Concept List") was used for collecting information about the concepts, alternative identifiers and a short description for each of them to clarify their meaning within the domain. The second table ("Attributes") groups all related attributes under the relevant concepts of the first table, i.e. relations to be established between concepts instantiations and particular data values. Additional information, such as a description, expected value data type, maximum cardinality, whether order information is needed, unit of measures, whether they concern sensitive data, the time zone or related standards, is also collected. The third table ("Relations") gathers all the relations between the concepts. Also, additional information is provided such as a description, maximum cardinality, whether they are symmetric, transitive, related standards, etc.

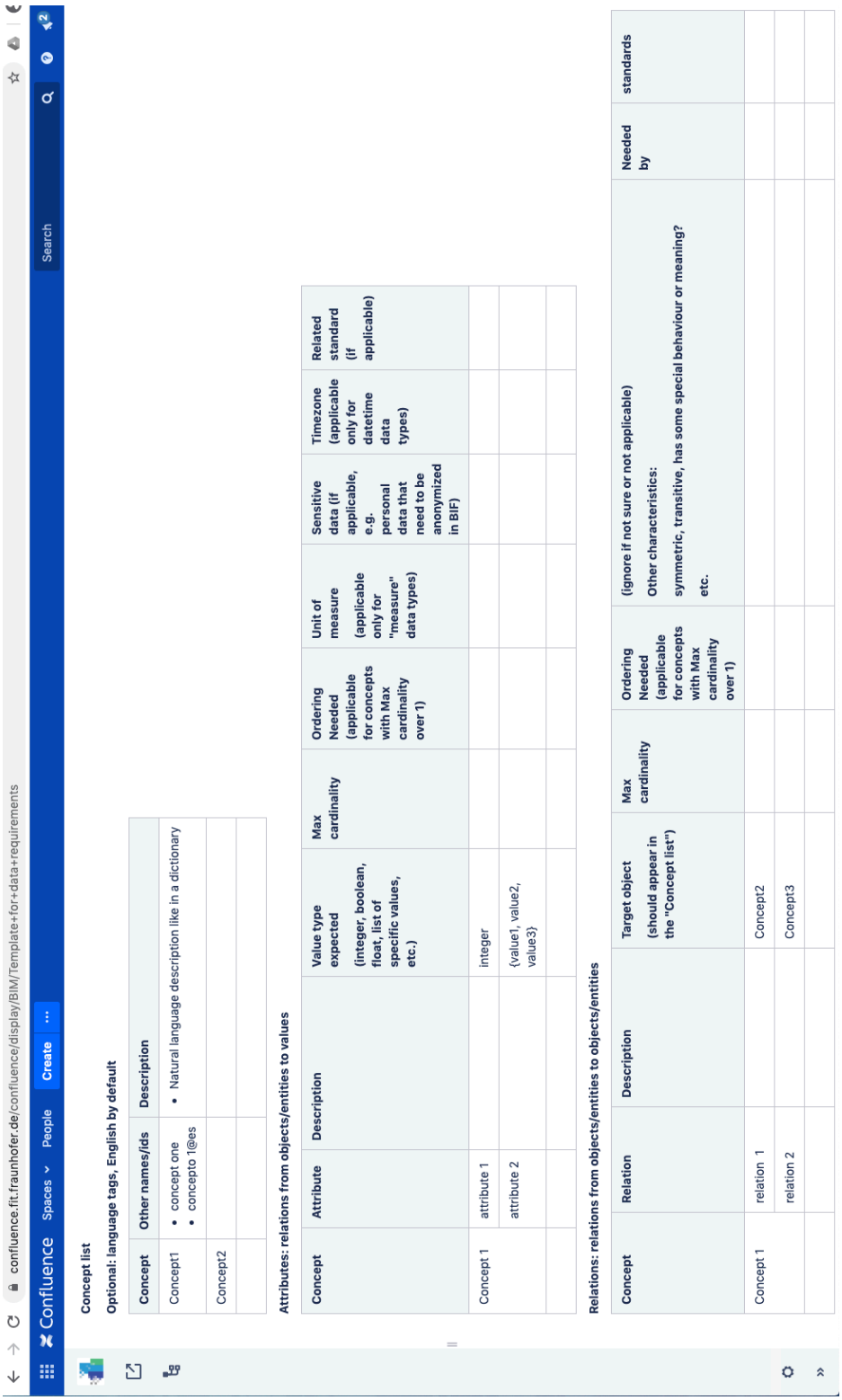

Figure 4. Data low level requirement specification templates.

One page has been created for each domain in Confluence to host the tables for concepts, attributes and relations. Such tables were reviewed by ontology developers with the support of domain experts or ontology users during the "**ontological requirements completion**" in order to check whether the requirements are understandable, complete (for starting the development, as they might evolved during the project lifecycle), realistic, consistent, etc.

Finally, the "**ORSD formalization**" activity is normally carried out in order to produce a document that gathers all ontological requirements which is called "Ontology Requirements Specification Document" (ORSD) [Suárez-Figueroa et. al., 2012]. The ORSD for the BIMERR ontologies is presented in Table 2. It should be mentioned that the original ORSD template includes the functional requirements section that, in the case of the BIMERR project, is tracked for each domain in the Confluence pages and, therefore, is not included in Table 2. However, the requirements used for the currently developed ontologies can be found in Annexes. The original template for the ORSD includes a section for the glossary of terms that is extracted from the competency questions, in order to identify the main terms (that help to separate important terms that should be represented in the ontology, from other words appearing in the competency questions). However, in the case of BIMERR this was not needed, as the technique for writing the requirements following the tables already identifies the terms.

Table 2. ORSD for BIMERR Ontology.

Ontology Requirements Specification Document Template	
1	**Purpose**
	The general goal of the BIMERR ontology and data model is to facilitate data sharing and interoperability among the BIMERR components through the BIMERR Interoperability Framework.
2	**Scope**
	The scope of the BIMERR ontologies is limited to the data shared through the BIF and external data sources needed in the energy efficiency domain and related domains like: KPIs, project management, weather, occupancy behavior, information objects, building geometry, building elements, materials and renovation measurements.
3	**Implementation Language**
	Ontology Web Language
4	**Intended End-Users**

	BIMERR components and application developers *BIMERR end-users and stakeholders*
5	**Intended Uses**
	Data model generation *External data sources integration*
6	**Ontology Requirements**
	a. Non-Functional Requirements
	Annotated in English *Linked to standards when possible* *Open license* *Modular* *Online availability*
	b. Functional Requirements: Groups of Competency Questions
	This section is provided for each specific domain in the confluence pages.

2.1.1 Infrastructure for Ontological Requirements Specification

The main system used for this phase was the BIMERR Confluence website.[5] Confluence is a collaborative environment to organize different aspects of projects. Every partner involved in the project can share relevant information for other teams. In Confluence, different sections/pages can be created to store the requirements associated to the BIMERR ontologies.

For each domain considered in the BIMERR ontology a dedicated Confluence page has been created. Within each page the tables described in Section 1.1.3 are incorporated, so the partner responsible for that domain can fill the information required. Figure 5 shows an overview of the internal organization of the confluence pages.

[5] https://www.atlassian.com/es/software/confluence

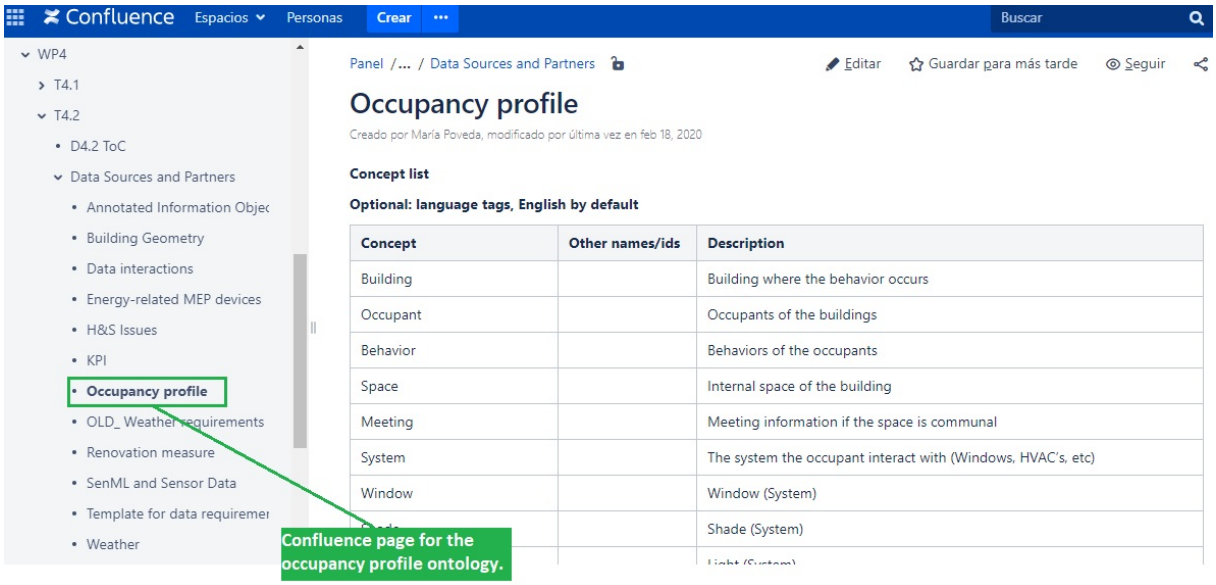

Figure 5. Confluence page for Occupancy Profile data requirements.

2.2 ONTOLOGY IMPLEMENTATION

The aim of the ontology implementation process is to build the ontology using a formal language, based on the ontological requirements identified in the previous steps. After defining the first set of requirements - though modification and addition of requirements is allowed during the development - the ontology implementation phase is carried out through several sprints. The ontology developers schedule and plan the ontology development according to the prioritization of the requirements in the ontology requirements specification process. The ontology development team builds the ontology iteratively, implementing only a certain number of requirements in each iteration. The output of each iteration is a new version of the ontology.

As shown in Figure 6, the ontology implementation activity is divided into four sub activities according to the LOT methodology. It should be mentioned that other activities might be carried out in more complex scenarios, for example, reusing non-ontological resources, reusing ontology design patterns, merging and reengineering ontological resources.

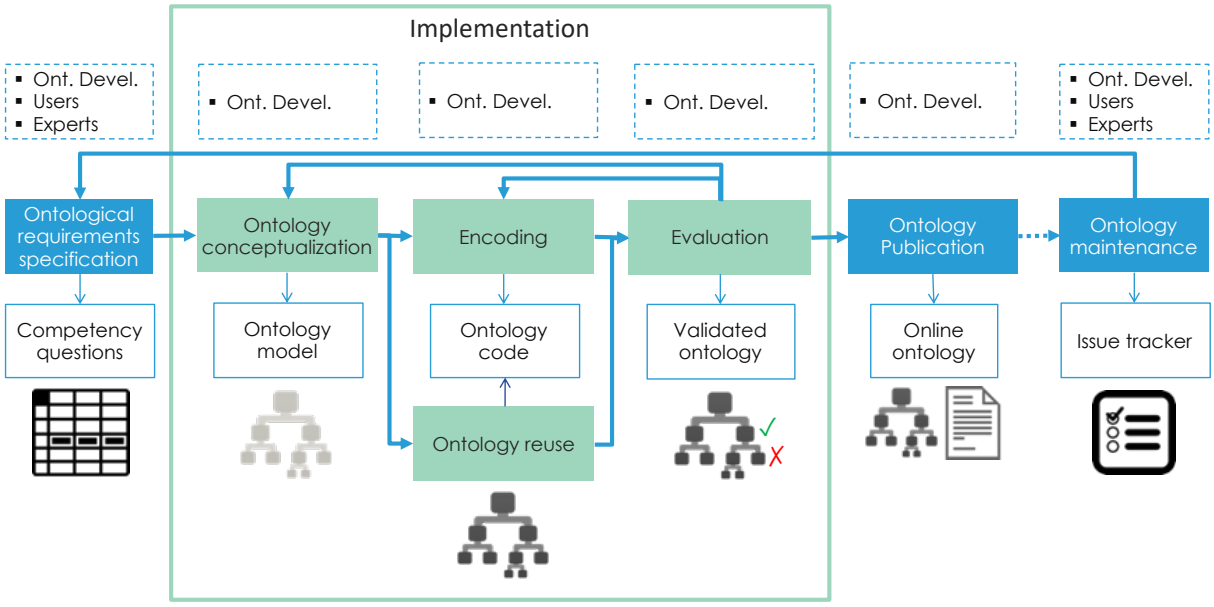

Figure 6. Workflow proposed for ontology implementation

Prior to starting coding the ontology in an ontology implementation language, it is advisable to carry out the "**ontology conceptualization**" activity. The aim of this activity is to build an ontology model from the identified ontological requirements. During the ontology conceptualization, the domain knowledge obtained from the tables of every domain module is organized and structured into a model by the ontology developers. This activity is normally carried out by sketching diagrams, although other techniques might also be used, such as writing the ontology model in a logic language syntax. Due to the fact that the ontology conceptualization is used for communication with domains experts and will also be reused to complete the ontology documentation, the technique used is the generation of diagrams using the UML_onto profile first defined [Hasse et. al., 2009] and later elaborated in [Poveda-Villalón, 2016].

During the "**ontology encoding**" activity, the ontology development team generates computable models in the OWL language from the ontology model. For this activity the developers take as input the ontology conceptualization to obtain a base ontology code which is extended with additional information taken from the ontological requirements.

The ontology code resultant from this activity includes metadata, such as creator, title, publisher, license and version of the ontology.

It is worth noting that during the development of the BIMERR ontologies the following ontology versioning convention has been adopted:

The versioning identification is as similar as possible to the conventions used in software development. In this case, each release follows the pattern major.minor.fix, where each field follows the rules:

- **major:** The field is updated when the ontology covers the complete domain it intends to model. That is, it is a complete product and covers the final goal of the development.
- **minor:** The field is updated when:
    - All the requirements of a subdomain are covered.
    - Documentation is added to the ontology.
- **fix:** The field is updated when:
    - Typos or bugs are corrected in the ontology.

In each iteration, the minor and fix fields might be changed several times.

Each ontology developed under the BIMERR ontology network can follow its individual version status. For example, the Occupancy Profile ontology might be in version v1.0.0 while the Building Geometry ontology might be in version v0.2.3. That is due to the fact that, the network evolution is not managed as a whole, as it is not a particular product but the virtual composition of many ontologies, where each ontology evolves independently.

During the ontology encoding, and in some cases during the ontology conceptualization activity, the "**ontology reuse**" activity might be carried out. In this activity, ontologies as a whole or ontology modules that match the BIMERR requirements might be reused. During this activity the results from D4.1 [Tiwari et.al., 2019] would be taken as input as well as ontology developers' knowledge about existing ontologies.

Before publishing a release version of the ontology, the ontology developers should carry out the "**ontology evaluation**" activity in order to check different aspects:

- The ontology developers guarantee that the ontology does not have syntactic, modeling or semantic errors.

- The ontology developers guarantee that the ontology fulfills the data exchange requirements set for the ontology using the test cases generated in the requirements specification process.

2.2.1 Infrastructure for Ontology Implementation

To support the ontology implementation phase, the ontology developers use several tools to edit, store and evaluate the ontology.

For the conceptualization phase, ontology developers make use of a drawing tool called draw.io,[6] to construct diagrams showing the relationships between concepts and also the attributes attached to them. Draw.io allows the creation of UML like graphical representations of ontologies to support visually the model codification.

The ontology development team uses an ontology editor, such as Protégé,[7] to generate the ontology code. Protégé allows the creation, visualization and manipulation of ontologies in various representation formats.

As solution for ontology storage and version tracking, each BIMERR ontology is stored in a GitHub[8] repository. For each ontology, its GitHub repository includes:

- A folder with the implementation of the ontology.
- A folder with the ontology modeling diagrams.
- A folder with the documentation of the ontology.
- A folder with the tests generated from the ontological requirements.

The ontology developers may use the GitHub or Git proposed workflow to develop the ontology, which can be summarized in the next steps:

- Create a new branch from the master branch.

[6] https://www.draw.io/
[7] https://protege.stanford.edu/
[8] https://github.com/

- Add changes to the ontology and commit them. Each commit has to be associated with a commit message, which is a description explaining why a particular change was made.
- Open a pull request to start a discussion over the changes.
- If the pull request is approved, merge the new branch into the master branch.

The development team uses OnToology [Ahmad et. al., 2018] [9] to generate the documentation and to evaluate the ontology. OnToology, which integrates Widoco [Garijo et. al., 2020],[10] and OOPS! [Poveda-Villalón et. al., 2014][11] generates automatically a folder in the GitHub repository which includes all the resources: diagrams, documentation and evaluation report.

The structure of the folders generated by OnToology for each ontology contained in the GitHub repository is represented in Figure 7:

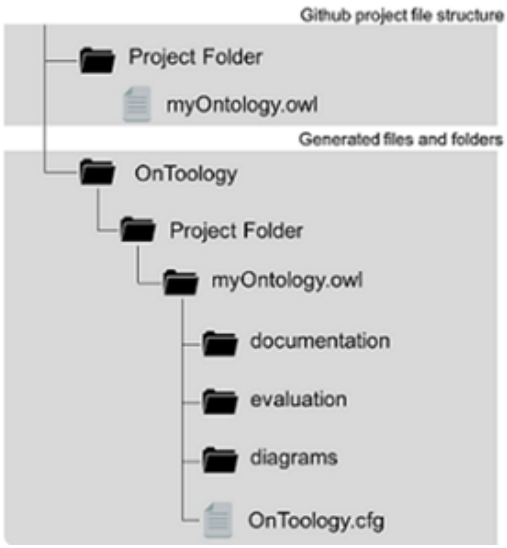

Figure 7. OnToology folder structure

The documentation of the ontology is generated using Widoco, which provides a description of the classes that must be completed by the domain experts. The evaluation is provided by OOPS!, which detects the most common pitfalls that appear when

[9] http://ontoology.linkeddata.es/
[10] https://github.com/dgarijo/Widoco
[11] http://oops.linkeddata.es/index.jsp

developing ontologies. All the resources generated by OnToology are updated whenever there is an alteration in the repository.

2.3 ONTOLOGY PUBLICATION

The aim of the ontology publication process is to provide an online ontology accessible both as a human-readable documentation and a machine-readable file from its URI. Figure 8 shows an overview of the specific steps taken to publish the ontology.

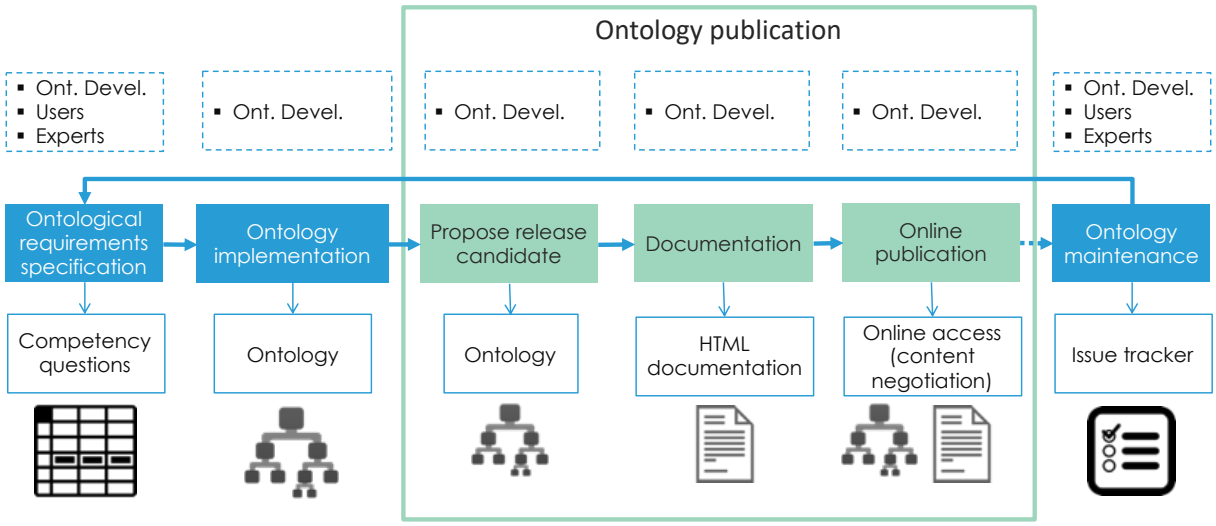

Figure 8. Workflow proposed for ontology publication

Once an ontology version is implemented and evaluated, the ontology development team "**propose a release candidate**" to be published on the Web. It should be considered that:

- In case the ontology fulfills all the requirements of a given subdomain, the ontology development team generates a release version of the ontology, e.g. the BIMERR ontology for release 0.1 is tagged as "release version v0.1.X".
- In case the ontology does not implement all the ontological requirements identified, but only those scheduled for the iteration, the ontology development team generates a pre-release version of the ontology.

The next step is to generate the ontology "**documentation**" taking as input the ontology code generated in the previous activities, the ontology development team in collaboration

with the domain experts generates the ontology documentation of the release candidate. This documentation includes:

- An HTML description of the ontology which describes the classes, properties and data properties of the ontology, and the license URI and title being used. The domain experts have to collaborate with the ontology development team to describe the classes and the properties. This description also includes metadata, such as creator, publisher, creation date, last modification or version number.
- Diagrams which store the graphical representation of the ontology, including taxonomy and class diagrams. When large and detailed models are generated or reused, the diagram would summarize top level entities.

Finally, the "online publication" of the ontology is carried out following the best practices for publishing vocabularies on the Web[12]. That is, the ontology is accessible online via its URI as machine-readable file(s) and a human-readable documentation using content negotiation.

2.3.1 Infrastructure for Ontology Publication

To support the ontology publication phase, the ontology developers publish the ontology online to be accessible to everyone and they also publish the releases in the online BIMERR ontologies portal[13], ensuring the ontology and its documentation is accessible to all users. The portal has different sections to provide different information, namely:

- Ontologies
- Ontology testing (this section is under development for now until the set of requirements for every module becomes stable and we can generate a set of queries to test the representation capabilities of every domain model)

[12] https://www.w3.org/TR/swbp-vocab-pub/
[13] https://bimerr.iot.linkeddata.es/

The Ontologies section is the main section of the portal, which shows the main information about the ontologies created. The section follows a tabular approach which includes:

- Link to the ontology documentation published on the Web, which is generated by Widoco.
    - The ontology URIs follow the pattern: https://bimerr.iot.linkeddata.es/def/XXXX where XXXX identifies the domain for each ontology, for example the URI of the KPI ontology is https://bimerr.iot.linkeddata.es/def/key-performance-indicator
- Ontology description.
- Links to each GitHub repository.
- Links to each GitHub issue tracker.

Figure 9 shows an overview of the information exposed in the BIMERR ontology portal at https://bimerr.iot.linkeddata.es/. A link to the requirements identified by the domain experts will be available once they become more stable.

This section of the portal includes a diagram which reflects the main concepts of each ontology at a general level. Further details are provided in each ontology description.

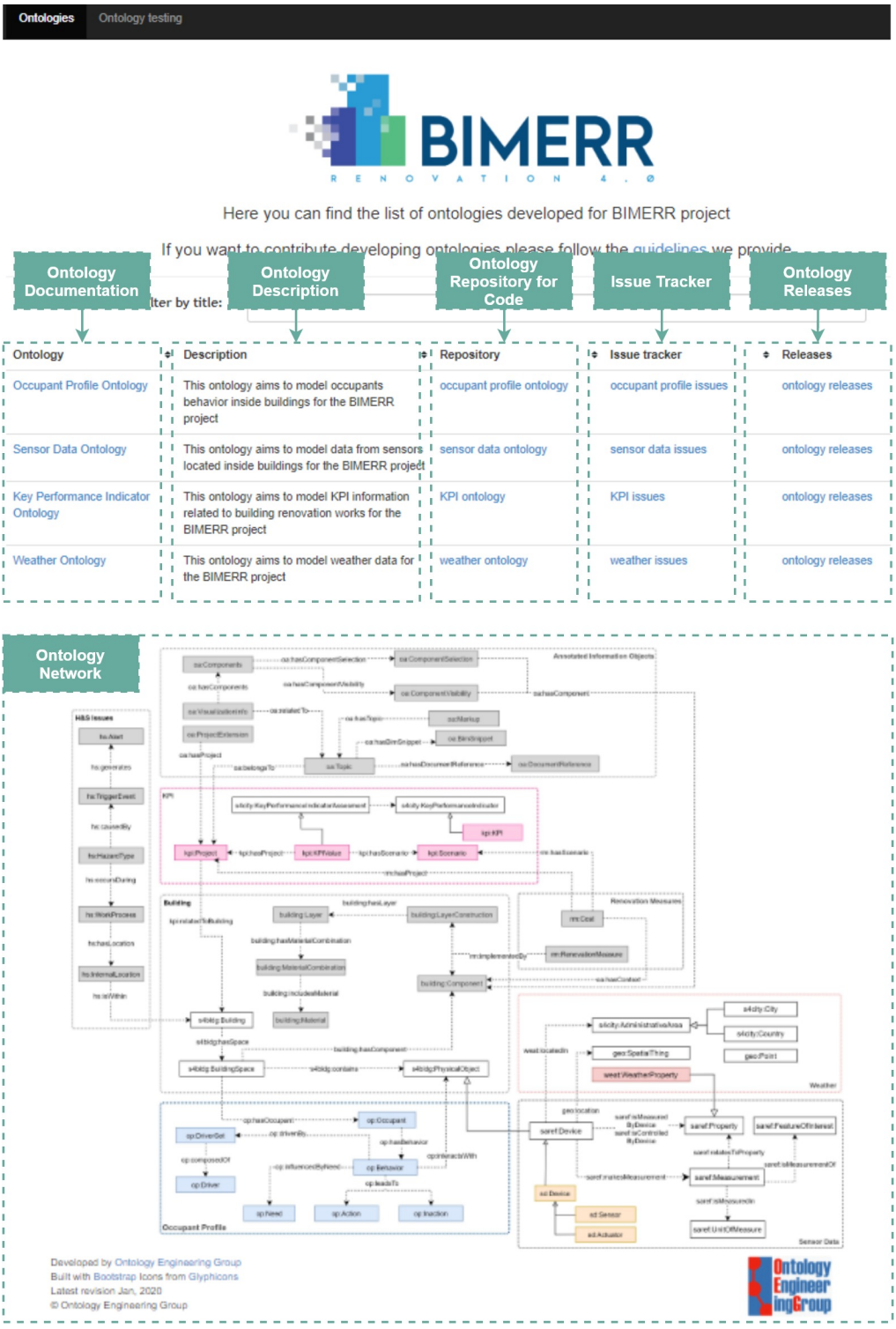

Figure 9. Overview of BIMERR ontology portal

Regarding the guidelines section, the portal also provides the users with a brief overview of the proposed process for developing ontologies and some guidelines which should be followed by the domain experts and ontology developers who wants to contribute. This section includes information about:

- How the repository should be structured
- The tools recommended to be used during the ontology development process
- Ontology versioning
- Issues management

2.4 ONTOLOGY MAINTENANCE

The goal of the ontology maintenance activity is to update the ontology. This may be as a result of the identification of new requirements, the correction of errors or the scheduling of a new iteration for ontology development. During the ontology development process, the domain experts can propose new requirements or improvements over the ontology. If these requirements or improvements are approved by the ontology development team, they are added to the ontology. The whole process is depicted in Figure 10.

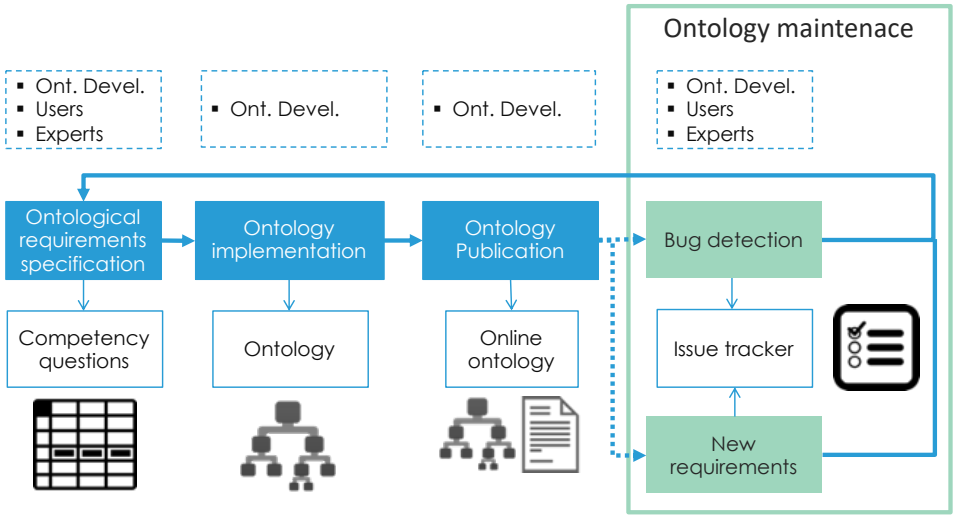

Figure 10. Workflow proposed for ontology maintenance

2.4.1 *Infrastructure for Ontology Maintenance*

To support the maintenance of the ontology, the ontology developers use an issue tracker which manages and maintains the list of issues identified by the domain experts and ontology developers.

All the changes and improvements to the ontology need to be agreed by the whole ontology development team. The GitHub issue tracker is used to discuss improvements and issues about the domains. If domain experts or ontology developers want to add new concepts to the ontology, they have to create a new issue in the GitHub issue tracker, which will be used to start a discussion about the approval of the proposal.

GitHub issues are marked as “open” and “closed”. The issues can also have an assignee, which is the person that is responsible for moving the issue forward. Figure 11 shows the issues section (currently empty) associated to the BIMERR Occupancy Behavior ontology.

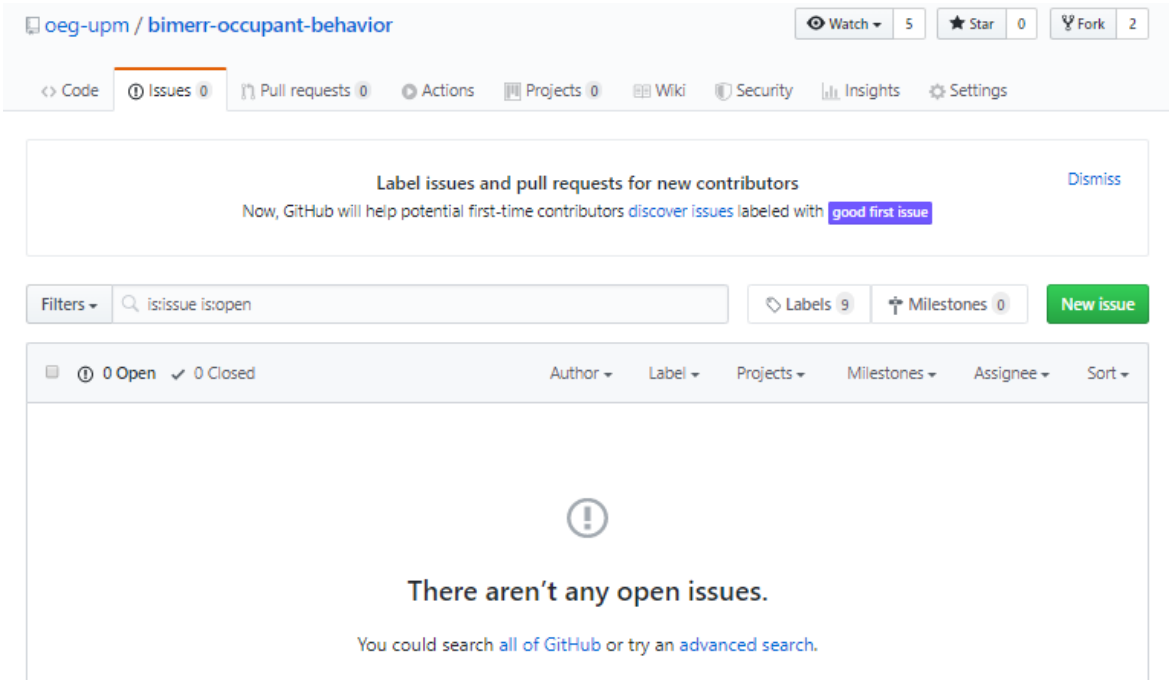

Figure 11. GitHub issues related to the BIMERR ontology

The ontology developers should label each issue according to its topic or status in order to organize the different types of issues. GitHub comes with a few labels by default: bug, duplicate, enhancement, invalid, question, and won’t fix. Those issues that represent new

requirements for the ontology will be labeled by the developers as a "requirement" issue. In addition, ontology developers can create new labels if they think they can improve issue management. The ontology developers are also responsible for closing the issues that have already been addressed.

2.5 TRANSITION FROM ONTOLOGY TO DATA MODEL

2.5.1 Basic Principles for the BIMERR Data Model

In order to address different semantic interoperability challenges for BIM-related data at syntactic, schematic and semantic heterogeneity levels in a performant and efficient manner, a flexible data model is defined based on the BIMERR ontology, reconciling different data standards. Such a data model has been meticulously "designed for change" with the purpose of efficiently managing its whole lifecycle, effectively anticipating its consistent evolution (e.g. how new concepts will be effectively incorporated, without disrupting the existing model, in a way that ensures backward compatibility) and aligning with the ontology as it is unrealistic to consider that any ontology and data model, no matter how well designed, will be inclusive of all the BIM-related data from the whole Architecture, Engineering and Construction (AEC) ecosystem from its early beginning and shall address all future data needs the stakeholders may have.

As depicted in the Table 3, the concepts that are defined in the data model, are annotated with a set of properties to properly provide them with a semantic context and the necessary complementary information to ensure data alignment in the BIMERR Interoperability Framework (BIF).

Table 3. Data model metadata annotations, description and population details

Metadata Title	Description	Population details
definition	A brief overview that acts as an account of a concept's contents.	Automatically extracted from the ontology
type	The data type to which the concept's data comply, e.g. string, integer, boolean, datetime, etc.	Automatically extracted from the ontology
related_terms	A set of related terms (e.g. synonyms) that can be alternatively used to represent the concept.	Manually annotated by the data model admin
standards	A list of standards in which the specific concept is modelled, along with their complementary	Partly extracted from the ontology and

Metadata Title	Description	Population details
	information, i.e. the exact concept used in such a standard, its type (e.g. element, attribute) and its use (required/optional) that applies for attributes.	complemented by the data model admin
date_added	The date when the specific concept was added in the data model.	Used for provenance and alignment with the ontology
date_deprecated	The date when the specific concept became obsolete in the data model.	Used for provenance and alignment with the ontology
version	The version of the model when the concept was last modified.	Used for provenance and alignment with the ontology
children	Grouped information for concepts that are conceptually classified under a concept.	Automatically extracted from the ontology
facet	Complementary information for a concept, essentially grouping the following metadata: cardinalityMax, ordered, sensitive, transformation, measurementType, measurementUnit, timezone	Partly extracted from the ontology and complemented by the data model admin
cardinalityMax	The maximum number of expected appearances of a concept in the data.	Automatically extracted from the ontology
ordered	An indication whether ordering is needed for multiple appearances of the same concept.	Automatically extracted from the ontology
sensitive	An indication whether the specific concept models personal or sensitive data.	Automatically extracted from the ontology
transformation	Information for the transformation rules that are related to a specific applicable standard. It practically contains the function that is required for the transformation and the parameters / concepts that are involved.	Manually annotated by the data model admin
measurementType	An indication of the measurement type that is applicable to a concept, e.g. referring to distance, temperature, etc.	Manually annotated by the data model admin
measurementUnit	The baseline measurement unit for the specific concept and measurement type.	Manually annotated by the data model admin
timezone	The timezone to which the data refer by default.	Manually annotated by the data model admin
codeList	A link to the code list that should be typically used for the data that refer to the specific concept.	Manually annotated by the data model admin
codeType	The type of code list that is applied for the specific concept.	Manually annotated by the data model admin

An indicative extract of the data model (that is expressed in JSON for better manipulation of the necessary custom, additional information) can be found in the next lines:

```
{
 "thermalRequirements": {
    "definition": "The thermal conditions and requirements in a particular space,
spatial zone, zone.",
    "related_terms": [
      "Space Thermal Requirements"
    ],
    "standards": ["ifcXML"],
    "date_added": "15/10/2019",
    "date_deprecated": null,
    "version": 1,
    "children": {
      "requirementsStatus": {
        "definition": "An indication whether the thermal requirements refer to
stated or observed requirements by the occupants.",
        "type": "string",
        "related_terms": [
          "Thermal Requirements Status"
        ],
        "standards": null,
        "date_added": "15/10/2019",
        "date_deprecated": null,
        "version": 1,
        "facet": {
          "cardinalityMax": 1,
          "ordered": false,
          "sensitive": false
        }
      },
      "requirementsIssueTime": {
        "definition": "The time that the thermal requirements report was issued.",
        "type": "datetime",
        "related_terms": [
          "Requirements Issue Datetime"
        ],
        "standards": null,
        "date_added": "15/10/2019",
        "date_deprecated": null,
        "version": 1,
        "facet": {
          "timezone": "UTC",
          "cardinalityMax": 1,
          "ordered": false,
          "sensitive": false
        }
      },
      "requirementsLocation": {
        "definition": "The location that the requirements concern. INDICATIVE
FIELD",
        "type": {
          "$ref": "#/location"
        },
        "related_terms": null,
        "standards": null,
        "date_added": "15/10/2019",
        "date_deprecated": null,
        "version": 1,
        "facet": {
```

```
          "cardinalityMax": 5,
          "ordered": true,
          "sensitive": false
        }
      },
      "meanTemperature": {
        "definition": "Temperature of the space or zone, that is required from
user/designer view point. If no summer or winter space temperature requirements are
given, it applies all year, otherwise for the intermediate period. Provide this
value, if no temperature range (Min-Max) is available.",
        "type": "double",
        "related_terms": [
          "Average Temp",
          "SpaceTemperature"
        ],
        "standards": [
          {
            "ifcXML": "Pset_SpaceThermalRequirements/SpaceTemperature",
            "type": "element"
          }
        ],
        "date_added": "15/10/2019",
        "date_deprecated": null,
        "version": 1,
        "facet": {
          "measurementType": "temperature",
          "measurementUnit": "Celcius",
          "cardinalityMax": 1,
          "ordered": false,
          "sensitive": false
        }
      },
      "ventilationIndicator": {
        "definition": "An indication whether the space is required to have natural
ventilation (TRUE) or mechanical ventilation (FALSE).",
        "type": "boolean",
        "related_terms": ["NaturalVentilation"],
        "standards": [
          {
            "ifcXML": "Pset_SpaceThermalRequirements/NaturalVentilation",
            "type": "element"
          }
        ],
        "date_added": "15/10/2019",
        "date_deprecated": null,
        "version": 1,
        "facet": {
          "cardinalityMax": 1,
          "ordered": false,
          "sensitive": false
        }
      }
    }
  }
```


It needs to be highlighted that the transition from the ontology to the data model is bidirectional and occurs at any time when there are changes from the ontology lifecycle (indicatively, initiated when a new standard or domain is modelled in BIMERR) or the data model lifecycle (initiated when a user in the BIF platform directly adds or suggests a new

concept to represent certain aspects of his/her data). All changes made in either the ontology or the data model are traceable with the help of certain metadata and are viewed under the broader perspective of the evolution strategy for the ontology and the data model. The changes that are performed on the data model and the ontology are classified as major or minor in the "version" metadata, and result into a new version of the data model and the ontology that may be backward compatible (so no action is needed for data that are already collected in the BIMERR Interoperability Framework) or may be non-backward compatible (so certain action for propagating the changes need to be taken).

At high level, all evolution events concern *addition* (of a new concept, a new sub-concept or a new standard), *update* (of metadata or sub-concepts) or *deletion* (of a concept, a sub-concept or their associated metadata) and are practically governed by different actions that are to be taken after an evolution event either automatically or manually (with the intervention of the data model and ontology admins):

- *Propagate action* signifying a backward compatible evolution event that can be adopted without creating any inconsistencies between the data model and the data that are already collected in BIF.
- *Prompt action* to embrace all evolution events that are typically non-backward compatible. Prior to such events being addressed in the data model, their impact on the data that are already checked in needs to be cross-checked and validated as it may require executing certain data transformation functions again (e.g. to rename the properties' titles).
- *Block action*, prohibiting the specific evolution event as it will create major problems on the data that have already populated the BIF platform.

Although the evolution and alignment between the ontology and the data model in BIMERR can be generally handled in a semi-automated manner, all evolution and alignment rules need to be frequently manually checked to avoid any inconsistencies at the model and the ontology levels.

2.5.2 *Infrastructure for the conversion and alignment between the Ontology and the Data Model*

The metadata elements defined in Table 3 that are automatically extracted from the ontology could be classified in two types, namely, those than can be directly extracted from the ontology (for example “description” and “type”,) and those that need a enriched ontology version (for example “sensitive” and “ordered”). This is due to the fact that the ontology language itself does not provide constructs for making explicit such information (for example if ordered is needed for the values of a property), therefore, the ontology should be annotated to keep such information.

A first version of the BIMERR Ontology to Data Model (BO2DM) converter has been developed covering basic transformations and functionalities. BO2DM is able to transform from an OWL coded ontology to a JSON serialized data model. This JSON model follows a particular specification designed for the BIMERR project, which differs from the typical JSON-LD file that off-the-shelf programming library parsers can produce. Libraries such as rdflib[14] for Python, or Jena[15] for Java can load an ontology in any of the common formats such as TTL,[16] N3[17] or RDF/XML[18] and produce a JSON-LD[19] code that have the structure shown in Figure 12.

[14] https://rdflib.readthedocs.io/en/stable/
[15] https://jena.apache.org/
[16] https://www.w3.org/TeamSubmission/turtle/
[17] https://www.w3.org/TeamSubmission/n3/
[18] https://www.w3.org/TR/rdf-syntax-grammar/
[19] https://json-ld.org/

```
{
    "@id": "https://bimerr.iot.linkeddata.es/def/occupancy-profile#leadsTo",
    "@type": [
        "http://www.w3.org/2002/07/owl#ObjectProperty"
    ]
},
{
    "@id": "https://bimerr.iot.linkeddata.es/def/occupancy-profile#Fall",
    "@type": [
        "https://bimerr.iot.linkeddata.es/def/occupancy-profile#Season",
        "http://www.w3.org/2002/07/owl#NamedIndividual"
    ]
},
{
    "@id": "https://bimerr.iot.linkeddata.es/def/occupancy-profile#detailedBy",
    "@type": [
        "http://www.w3.org/2002/07/owl#FunctionalProperty",
        "http://www.w3.org/2002/07/owl#ObjectProperty"
    ]
},
{
    "@id": "https://bimerr.iot.linkeddata.es/def/occupancy-profile#Action",
    "@type": [
        "http://www.w3.org/2002/07/owl#Class"
    ],
    "http://www.w3.org/2000/01/rdf-schema#comment": [
        {
            "@language": "en",
            "@value": "Actions are the interactions with systems or activities that occupants
        }
```


Figure 12. JSON-LD model generated by rdflib parser

This serialization generates a "@id" field for every component of the ontology, such as concepts, relations, and attributes, these identifiers are the URI's of the ontology. Some "@id" also point to blank nodes that correspond to restrictions in OWL language. The "@type" field corresponds to the element role within the ontology implementation. Some common types are `owl:Class`, `owl:ObjectProperty`, `owl:FunctionalProperty`, etc. Each concept or attribute are further detailed by means of `rdfs:comment`, or `rdf:label`, which have as main fields "@language" and "@value" to represent both the language of the comment or label and their string value respectively.

The information provided by common RDF libraries in a JSON-LD structure is not sufficient to give a full representation of the domains needed by the BIF. From the data model code provided above, it can be seen that additional fields are needed such as "standards" or "facet". Besides, the data model needs a nesting of all their children.

At this point it is important to remind that the BIMERR platform will handle two ontological models. The first model will be agnostic to the BIF purposes, just giving the information that describe each domain using OWL constructs without including annotations for

ontology-data model alignment. The second ontology will contain not only the concepts and relations from the previous implementation, but also will include the annotations needed to generate some of the data model elements.

The conversion process is divided into steps. The first one takes the ontology in its original format (e.g. TTL) and produces a JSON-LD version of the file. The transformation is carried out using off-the-shelf parsers incorporated in standard programming libraries such as rdflib for Python. The second step involves the aggregation of information to the extracted JSON-LD in order to achieve the final specification as similar as possible to the data model expected by the BIF.

The second part of the conversion process is the more challenging one and requires a series of additional inputs to get a proper data model.

- **List of root concepts:** The desired JSON file has a set of concepts working as main nodes on the data structure. The rest of the concepts can be organized from those main nodes. From the ontology implementation perspective, it is not possible to know which are the main concepts without those being explicitly indicated in the code. The current version of the converter accepts this set of concepts as a list that will be used later to organize the hierarchy of the dictionary. Another approach is to indicate the root concepts during the implementation phase by means of annotation properties. Which option is better will be analysed in the upcoming version.
- **Children annotations:** Some concepts of the desired data model have children elements as part of their definition. After a comparison between the children of the data models and the elements of the ontology we can conclude that a child element can be either a concept or an attribute. However, it is not clear which attributes can be children elements. To disambiguate this situation, additional annotations, need to be included to tag the attributes that will be in the model as children. During the ontology implementation the annotation property “children” is used for this purpose.

Apart from this list of inputs, the converter makes a set of assumptions in order to simplify the data model generation and get as close as possible to the desired specification. The first design decision is the classification of all concepts at the destination side of an object property as children of the source concept. The second design decision (pattern) states

that if a class concept has a set of subclasses, all those elements are children of that class. However, this point should be analyzed from the data model expected use and, if needed, the converter output for this case should be redesigned.

Following these design principles, the converter performs the parsing process, analyzing every concept, relation and attribute of the JSON-LD file. The parser starts by reading all the concepts of the ontology and adding general attributes such as “standards” or “definition”. The datatype properties marked on the ontology as children are also described using these attributes. The subsequent step is finding the corresponding children for every concept considering the aforementioned rules. At this point, all the concepts are at the same level. The last step consists on the nesting of the corresponding children for every concept, creating a multilevel data structure. In cases where the child of an element is a higher level concept in the hierarchy, the child is described just by referencing the name of it. Figure 13 shows an excerpt of the final data model obtained.

```
"building": {
    "standards": [],
    "data_added": "",
    "date_deprecated": null,
    "version": 1,
    "related_terms": [],
    "facet": {
        "cardinalityMax": 1,
        "ordered": false,
        "sensitive": false
    },
    "children": {
        "space": {
            "standards": [],
            "data_added": "",
            "date_deprecated": null,
            "version": 1,
            "related_terms": [],
            "facet": {
                "cardinalityMax": 1,
                "ordered": false,
                "sensitive": false
            },
            "children": {
                "system": {
                    "children": {
                        "light": {
                            "standards": [],
                            "data_added": "",
                            "date_deprecated": null,
                            "version": 1,
                            "related_terms": [],
                            "facet": {
                                "cardinalityMax": 1,
                                "ordered": false,
                                "sensitive": false
                            },
```


Figure 13. Data model generated BO2DM.

This converter is a work in progress and still needs some adjustments to being able to work properly. One of the main limitations is the capacity to nest over a deep number of levels, which can happen in the case of very large models. Another limitation is that the converter cannot handle inverse relations, which would create infinite loops.

3. CURRENT BIMERR ONTOLOGY AND DATA MODELS

This section describes part of the network ontology developed in this first sprint of the process presented in Section 2. It first provides an overview of the network, and each ontology module is subsequently described in detail in the following sub-sections. It is worth clarifying that in this section the process is mainly explained from the ontologies development point of view. However the results are both the ontologies and the derived data models (after the transformation). That is the reason both artefacts are mentioned in the section. In addition, the data requirements are taken considering the data models needs too.

Table 4 summarizes the current status of each identified domain. Each column indicates the status of each phase within the ontology development process.

Table 4. Summary of domain status.

Domain	Requirements	Conceptualization	Implementation	Publication
Typical Meteorological Year Weather Data	Reviewed	First version	First version	First version
Weather Forecasts	Reviewed	First version	First version	First version
Usage Patterns & Habits – Occupant Behavior Modeling	Reviewed	First version	First version	First version
KPIs	Reviewed	First version	First version	First version
Building (Location)	Not defined	-	-	-
Building (Geometry)	Not defined	-	-	-
Building (Elements and construction)	Not defined	-	-	-
Building (Materials properties)	Not defined	-	-	-
Building Energy Systems (HVAC & Electric Equipment)	Not defined	-	-	-
HVAC & Electric Equipment properties	Not defined	-	-	-

Annotated information objects	Pending of review	To be developed in the next sprint.	To be developed in the next sprint.	To be developed in the next sprint.
Renovation measures	Pending of review	To be developed in the next sprint.	To be developed in the next sprint.	To be developed in the next sprint.
Sensor Data	Reviewed	First version	First version	First version
H&S Issues	Pending of review	To be developed in the next sprint.	To be developed in the next sprint.	To be developed in the next sprint.

The ontology network portal is available online at https://bimerr.iot.linkeddata.es/ (See Figure 9) and it provides all the documentation and code for each BIMERR module.

The ontology network developed for the BIMERR project consists for now of seven ontology modules, where each module models one specific domain. Figure 14, provides a graphical overview of the BIMERR ontology network showing the main concepts defined in each module and a summary of related concepts. In such figure, coloured modules are used to represent currently implemented domains, e.g. the blue boxes correspond to classes of the occupancy profile module implemented in BIMERR. In this sense, classes defined in the Occupancy Profile ontology are painted light blue (e.g. `op:Occupant`), in the Sensor Data ontology in orange (e.g. `sd:Device`), and in the KPI ontology in rose (e.g. `kpi:Project`). Table 5 indicates the list of prefixes and namespaces used for these four BIMERR ontologies implemented in the first sprint.

Table 5. List of published BIMERR ontologies and their corresponding namespaces

Prefix	Ontology namespace
op	https://bimerr.iot.linkeddata.es/def/occupancy-profile#
kpi	https://bimerr.iot.linkeddata.es/def/key-performance-indicator#
weat	https://bimerr.iot.linkeddata.es/def/weather#
sd	https://bimerr.iot.linkeddata.es/def/sensor-data#

White boxes in Figure 14 represent reused terms from existing ontologies, for example in the sensor data module the concept "Device" is taken from the SAREF ontology. The list of prefixes, and corresponding ontologies, reused along the development of the BIMERR

model are listed in Table 6. As it can be observed, some of the ontologies reviewed in previous steps of the project were reused, like SAREF and geo, as they are related to the project domain. Furthermore additional cross-domain ontologies were reused once the requirements and conceptualizations were analyzed in detail, as the case of the SKOS (Simple Knowledge Organization System) and Time ontologies.

Table 6. List of reused ontologies and their corresponding namespaces

Prefix	Ontology namespace
foaf	http://xmlns.com/foaf/0.1/
geo	http://www.w3.org/2003/01/geo/wgs84_pos#
org	http://www.w3.org/ns/org#
xsd	http://www.w3.org/2001/XMLSchema#
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#
saref	https://w3id.org/saref#
s4bldg	https://w3id.org/def/saref4bldg#
s4city	https://w3id.org/def/saref4city#
skos	http://www.w3.org/2004/02/skos/core#
time	http://www.w3.org/2006/time#

Finally, grey modules in Figure 14 represent draft conceptualization of domains pending to be developed. This means that the grey boxes are concepts expected to appear in a module (according to the requirements at this point) but the conceptualization is not detailed and the module is not implemented.

The main hierarchies between concepts are also included. This hierarchies are represented by arrows with white endings (triangles) and can be read as follows: The class in the origin of the arrow is a subclass of the class in the end of the arrow.

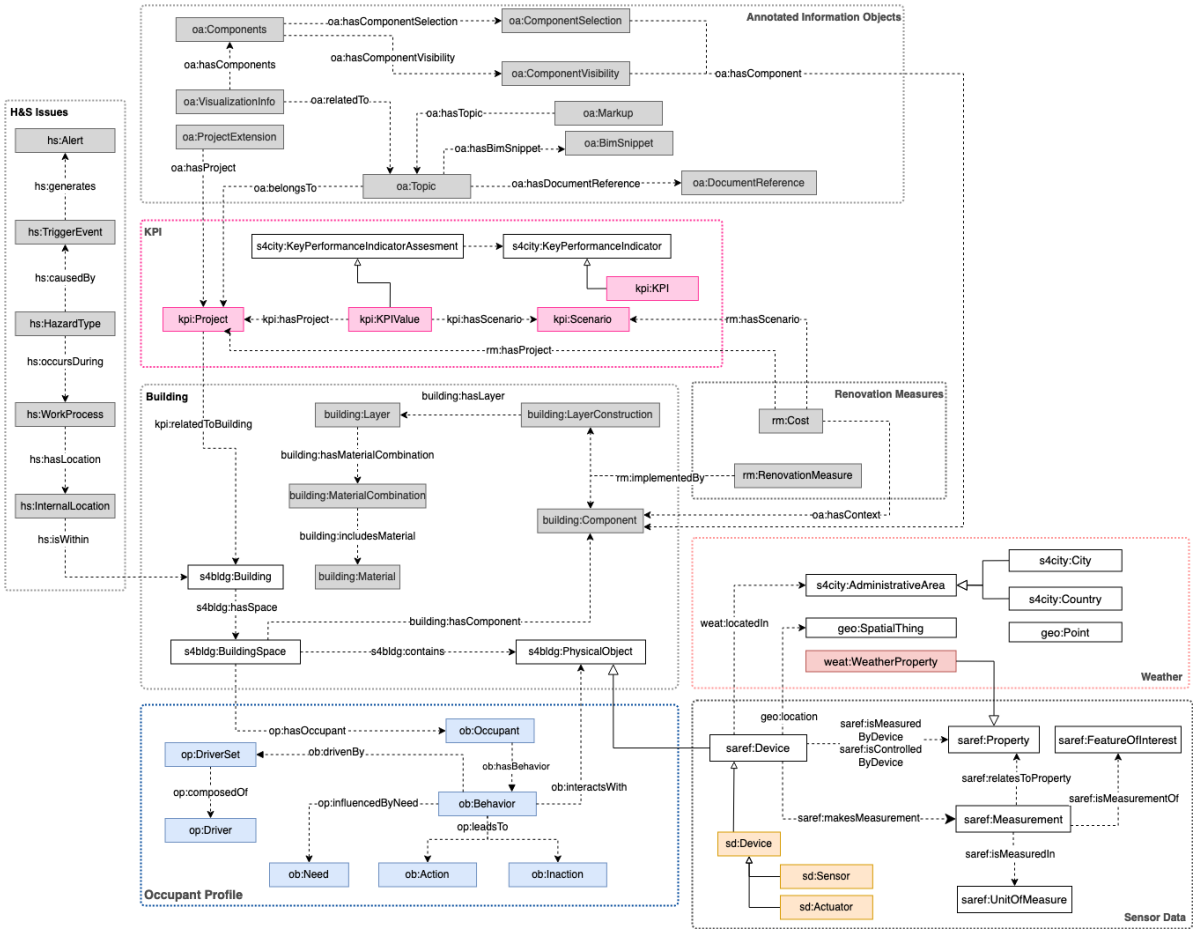

Figure 14. BIMERR ontology network overview.

As shown in Figure 14, the ontology modules have relationships among themselves. First, ad-hoc relationships are defined between terms of different modules, for example the concept `kpi:Project` in the Key Performance Indicator module is related to the `s4bldg:Building` concept in the Building ontology through the property `kpi:relatedToBuilding`.

It is worth noting that complete and up to date documentation of each ontology module is provided online and is accessible through each of the ontologies' URI. In the rest of this document only the main concepts or modeling decisions are detailed.

Apart from the domain-specific ontological requirements, the BIMERR ontologies development is based on the following non-functional requirements:

- Reuse: existing ontologies or standard models shall be reused when possible, thereby increasing interoperability with external systems that might be already using such ontologies. The point is also applied at the meta-level by using standard technologies to implement the ontologies themselves.
- Modularity: the ontology shall be designed as a network in which modules might be interconnected and refer to others.
- Good practices: the ontologies shall be developed following methodologies and best practices commonly used in ontology engineering in order to address ontology development activities such as design, implementation, evaluation, publication, and documentation, among others.

Each ontology module is detailed in the following subsections. In the figures and examples the following graphical conventions are used:

Colored rectangles are used to denote classes created in the ontology being described while white rectangles denote reused classes. For all the entities, it is indicated in which ontology they are defined by the prefix included before their identifier.

Arrows are used to represent properties between classes and to represent some rdf, rdfs and owl constructs, more precisely:

- Plain arrows with white triangles represent the `rdfs:subClassOf` relation between two classes. The origin of the arrows is the class to be declared as subclass of the class at the destination of the arrow.
- Plain arrows between two classes indicate that the object property has declared as domain the class in the origin and as range the class in the destination of the arrow. The identifier of the object property is indicated within the arrow.
- Dashed labeled arrows between two classes indicate that the object property can be instantiated between the classes in the origin and the destination of the arrow. The identifier of the object property is indicated within the arrow.
- Dashed arrows with the identifiers between stereotype signs (i.e., “<< >>”) refer to OWL constructs that are applied to some ontology elements, that is, they can be applied to classes or properties depending on the OWL construct being used.

- Dashed arrows with no identifier are used to represent the `rdf:type` relation, indicating that the element in the origin is an instance of the class in the destination of the arrow.

Datatype properties are denoted by rectangles attached to the classes, in a UML-oriented way. Dashed boxes represent datatype properties that can be applied to the class it is attached to while plain boxes represent that the domain of the datatype property is declared to be the class attached.

Individuals are denoted by rectangles in which the identifier is underlined. Literals are denoted by rectangles in which the value is included between quotation marks.

The representation of additional property axioms (functional, inverse functional, transitive, and symmetric) that are being used in the diagram are shown in the overview ontology legend.

3.1 OCCUPANCY PROFILE MODULE

The Occupancy Profile ontology has been developed to represent people's behavior inside buildings with a focus on the energy impact their actions produce. This information is critical to get more precise building energy profiles, and generate simulation scenarios that fit better with reality. In this sense, the shared conceptualization represented in this ontology focuses on the interaction of occupants with the different systems inside building spaces that could influence their thermal comfort requirements. This ontology has been developed using the Occupancy Behavior XML Schema [Hong et. al., 2019] as a basis, which defines the main. Some additional concepts as well as relations and attributes were added in order to obtain a full semantic representation of the domain.

The current conceptual model defined by the Occupancy Profile ontology is depicted in Figure 15. This model reuses some concepts and properties from the SAREF For Building Ontology (SAREF4BLDG), an extension of the SAREF ontology, that was created mainly to support interoperability between smart appliances in a building context. Some of the most important concepts related to the Occupancy Profile domain are the following:

- **Physical Object:** This concept refers to any object that has a proper space region such as the equipment or mechanisms within the building with which occupants may interact to restore or maintain environmental comfort.
- **Occupant:** This concept represents people that reside within a building or make use of specific spaces within it.
- **Behavior:** This concept is used to define the behaviors that people have inside a space. These behaviors could lead to interactions with systems.
- **Driver:** This entity represents the stimulating factors that provoke energy- related occupant behavior. A driver prompts a building occupant to perform either an action or in-action with a building system, impacting the energy use of a building.
- **Action:** This concept represents the interactions with systems or activities that occupants can perform to achieve environmental comfort.

Additionally, we describe the Building and Building Space concepts. These concepts form part of the building domain, however they are also important within the occupancy profile module.

- **Building:** A building represents a structure that provides shelter for its occupants or contents, and stands in one place.
- **Building Space:** An entity used to define the physical spaces of the building. A building space contains devices or systems.

"Annex 1: Tables of Requirements for the Occupancy Profile Domain" shows the complete list of concepts, relations and attributes taken into consideration during the conceptualization of the occupancy profile domain. These tables are also available on the respective Confluence page.

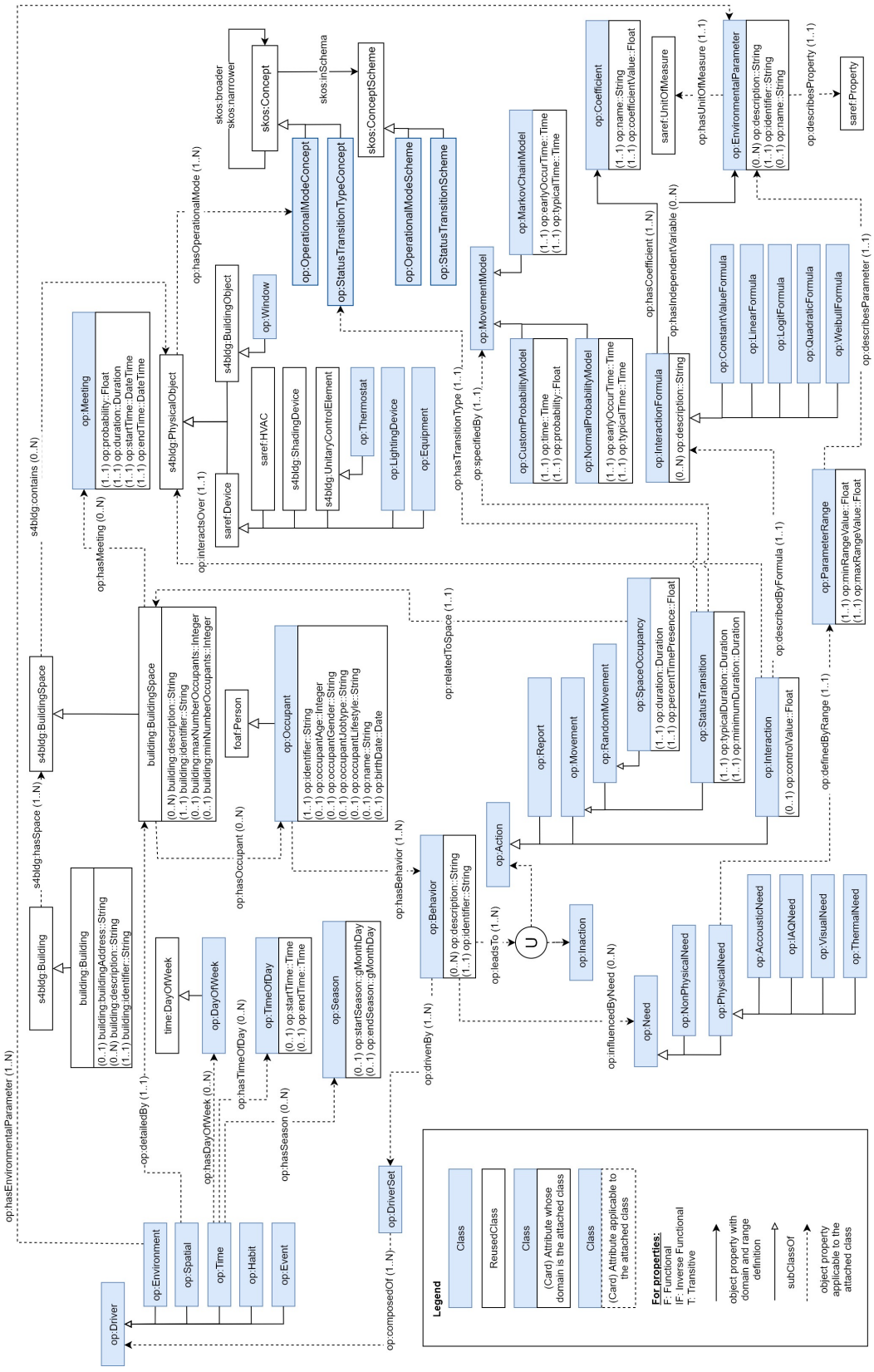

Figure 15. General overview of the Occupancy Profile ontology.

3.1.1 Model Description

The main concepts defined in the ontology, as shown in Figure 15, are: `building:Building`, `building:BuildingSpace`, `op:Occupant`, `op:Behavior`, `op:Driver`, `op:Action and s4bldg:PhysicalObject` previously defined in Section 3.1. In the model conceptualization we can observe that the class `building:Building` can have several spaces, and is related to them through the property `s4bldg:hasSpace`. The `building:BuildingSpace` object can contain several systems that an occupant can interact with. From the obXML schema we have a list of systems that can be accommodated under the class `s4bldg:PhysicalObject`. Equipment such as HVAC's or shading devices can be represented by the classes `saref:HVAC` or `s4bldg:ShadingDevice` respectively. For systems not available in S4BLDG we created new terms, like the concept `op:Window`. Furthermore, the model provides a set of possible control options for these space systems. This enumeration of control options, which include terms like `op:OnOff` or `op:Fixed`, is represent by the class `op:OperationalModeConcept`, subclass of `skos:Concept`. The use of the SKOS model enables a richer expressivity, by allowing the relation of the instances by means of the properties `skos:broader`, `skos:narrower` and others.

Several persons can occupy a specific space, where each occupant can have more than one behavior. Attached to the occupant concept, information about its personal profile can be indicated by means of the attributes `op:occupantAge`, `op:occupantGender`, `op:name`, `op:occupantJobtype`, and `op:occupantLifestyle`.

A behavior could be the responsible for an action, where the concept `op:Action` involves the following subclasses: `op:Report`, `op:Movement` and `op:Interaction`. A person can also decide to not perform any action even though its comfort levels are not satisfied; this scenario is modeled by means of the class `op:Inaction`. The interaction of an occupant with a system can be represented using mathematical models. This is addressed using the property `op:describedByFormula` where the destination is the concept `op:InteractionFormula`, which groups several off-the-shelve mathematical functions. One example of specialization is the concept `op:LinearFormula` which gathers all the linear models that have at least one parameter in its definition.

For a mathematical formula to be properly defined we need a list of its coefficients and independent parameters. In order to model that, the relationships `op:hasCoefficient` and `op:hasIndependentVariable` were created. The first property (`op:hasCoefficient`) allows the connection from a given formula to an instance that will be linked to a certain numerical value and name by means of the data properties `op:coefficientValue` and `op:name` respectively. The second property (`op:hasIndependentVariable`) links a given formula with a parameter that describes a certain environmental factor of the building space.

Additionally, the model should allow to express if the occupant needs are influencing in a certain way its behavior. For that purpose, the property `op:influencedByNeed` was implemented. These needs can be further categorized into Physical Needs, and Non-Physical Needs. Non-physical needs gather together all the possible instances that could not be quantified such as privacy, status, etc. On the other hand, Physical Needs represent comfort requirements that could be defined by fixed limits. To handle this, the concept `op:ParameterRange` was created, which indicates the maximum and minimum value for a specific environmental parameter (`op:EnvironmentalParameter`).

Apart from human-system interactions, there are other kind of occupant activities that are important from the energy consumption perspective. Occupant's movements (`op:Movement`), whether they occur at space or building level, can have an impact over the building energy profile. There are two types of movements covered in the model, random movements inside spaces, represented by the class `op:RandomMovement`, and status transition movements between spaces, represented by the concept `op:StatusTransition`. Status transition events can be approximated by probabilistic models such as the Markov Model, being the connection between both concepts the property `op:specifiedBy`. The particular attributes needed for each stochastic model are attached to their respective class. For instance the `op:NormalProbabilisticModel` requires the attributes `op:typicalTime` and `op:earlyOccurTime` during its instantiation.

Finally, a behavior can be caused by several factors, either internal or external. The object property `op:drivenBy` represents this relation, where the destination of the property is the class `op:DriverSet`, which allows to combine multiple drivers. This model decision

allows to represent time (`op:Time`) as a driver by itself or as complementary information to describe when other type of drivers typically occur.

3.2 SENSOR DATA MODULE

The Sensor Data ontology has been developed to represent measurements generated by sensors located inside building spaces. These devices will be used to estimate building resident energy profiles, by measuring indirectly the interaction between occupants and systems. For the construction of this ontology, the SenML standard was taken as a basis.

The current conceptual model defined by the Sensor Data ontology is depicted in Figure 16. This model reuses concepts and properties from the SAREF ontology which is intended to enable interoperability between different IoT solutions from different manufacturers. Some of the most important concepts taken from SAREF are the following:

- **Property:** An aspect of an entity that can be observable by a sensor.
- **Device:** 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.
- **Measurement:** Represents the measured value made over a property. It is also linked to the unit of measure in which the value is expressed and the timestamp of the measurement.
- **Unit of Measure:** The unit of measure is a standard for measurement of a quantity, such as a Property.

"Annex 2: Tables of Requirements for the Sensor Data Domain" shows the complete list of concepts, relations and attributes taken into consideration during the conceptualization of the sensor data domain. These tables are also available on the respective Confluence page.

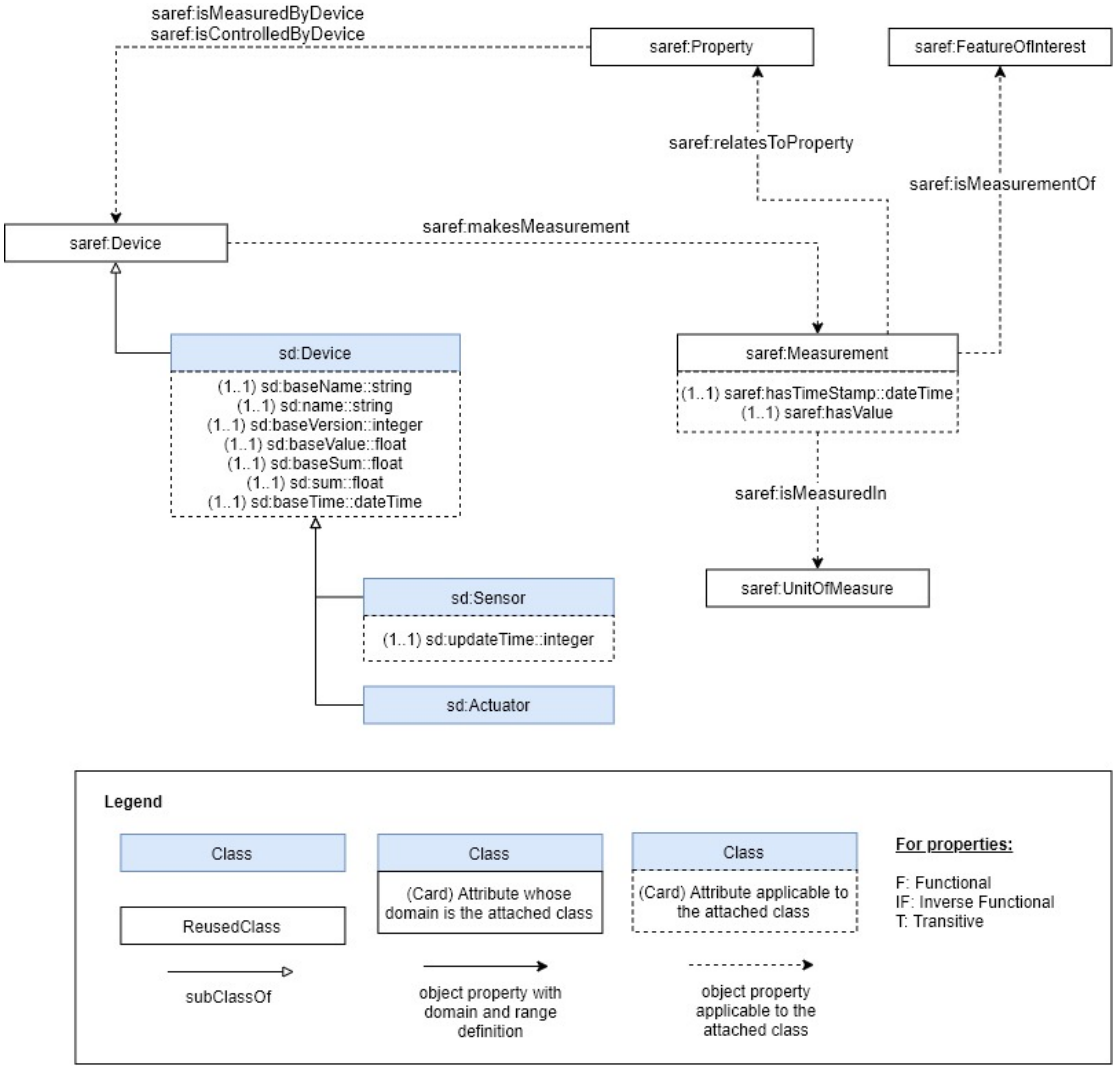

Figure 16. General overview of the Sensor Data ontology.

3.2.1 Model Description

As mentioned previously, the sensor data module has been constructed with the idea of being able to represent data coming from sensors or, if necessary, being able to model commands sent to system actuators. Following this design principle, the `saref:Device` concept was used to represent all the individuals that will perform those functions inside building spaces. According to the SenML standard, a device is related to a record, which could be either a measurement or a command. This relation is modeled by the object property `saref:makesMeasurement` which has as destiny the class `saref:Measurement`, that groups individual records. Attached to this concept there are two attributes that further describe device records. The first one is

saref:hasTimeStamp that connects the concept to a specific relative or absolute time. The second attribute is saref:hasValue that indicates quantitatively the magnitude of the record.

Even though most of the IoT streams coming from the Middleware module in the BIMERR platform will be resolved (i.e. preprocessed), it is recommended to include most of the fields provided by SenML in the model. This may be useful for making more complex queries. In order to model these extra attributes, we need to extend the representation capabilities of saref:Device by means of the concept sd:Device. One of the most relevant attributes attached to this entity is sd:sum which represents the integrated sum of record values over time. This is useful to notice when a sensor has lost communication with the server. Other elements incorporated are the base versions of the attributes described previously such as sd:baseValue, that with saref:hasValue property are added together to get the value of the measurement.

Also, we need to relate the measurement to a specific property of the environment or aspect of an entity. Through the use of the object property saref:relatesToProperty, we can model this connection where the range of the relation is the concept saref:Property, that groups individuals such as temperature, illuminance, or noise level. For a full description of a record we need to indicate the units in which it is expressed. The concept saref:UnitOfMeasure takes responsibility for that function further detailing the concept saref:Measurement throughout the relation saref:isMeasuredIn.

Finally, a record is not only an estimation of a physical property of the world, it also describes a particular thing or context as for instance, the temperature or illuminance within an apartment's room. With this in mind, the property saref:isMeasurementOf is used to link a measurement to the concept saref:FeatureOfInterest, which represents all the entities that are being described or acted over by the devices.

3.3 KEY PERFORMANCE INDICATOR MODULE

The Key Performance Indicator module aims to model the metrics defined at the beginning of building renovation activities to monitor the conformance with typical requirements related to energy efficient buildings. This type of measures serve not only

to get a correct estimation of the advances in the renovation workflow, but also to adjust the planning of the construction activities and improve the communication with stakeholders. The conceptualization proposed also focuses on the relation between KPIs, projects and scenarios. The construction of this ontology has been done using the requirements and needs identified for the PWMA and RenoDSS BIMERR applications.

The up-to-date conceptual model defined for the Key Performance Indicator ontology is illustrated in Figure 17. This model follows good ontology engineering practices, and reuses some terms from existing models, specifically concepts from SAREF for City (SAREF4CITY), which is an extension of the SAREF ontology. Some of the most relevant concepts in the model are defined below:

- **KPI:** A type of performance measurement. KPIs evaluate the success of an organization or of a particular activity in which it engages. (Definition taken from FIWARE)
- **KPI Value:** A quantity assessment of the KPI being evaluated. It also takes into account the time at which the indicator was calculated.
- **Project:** The building renovation project the KPI metric is related to.
- **Scenario:** The particular conditions under which the KPI is being evaluated.

"Annex 3: Tables of Requirements for the Key Performance indicator Domain" shows the complete list of concepts, relations and attributes taken into consideration during the conceptualization of the key performance indicator domain. These tables are also available on the respective confluence page.

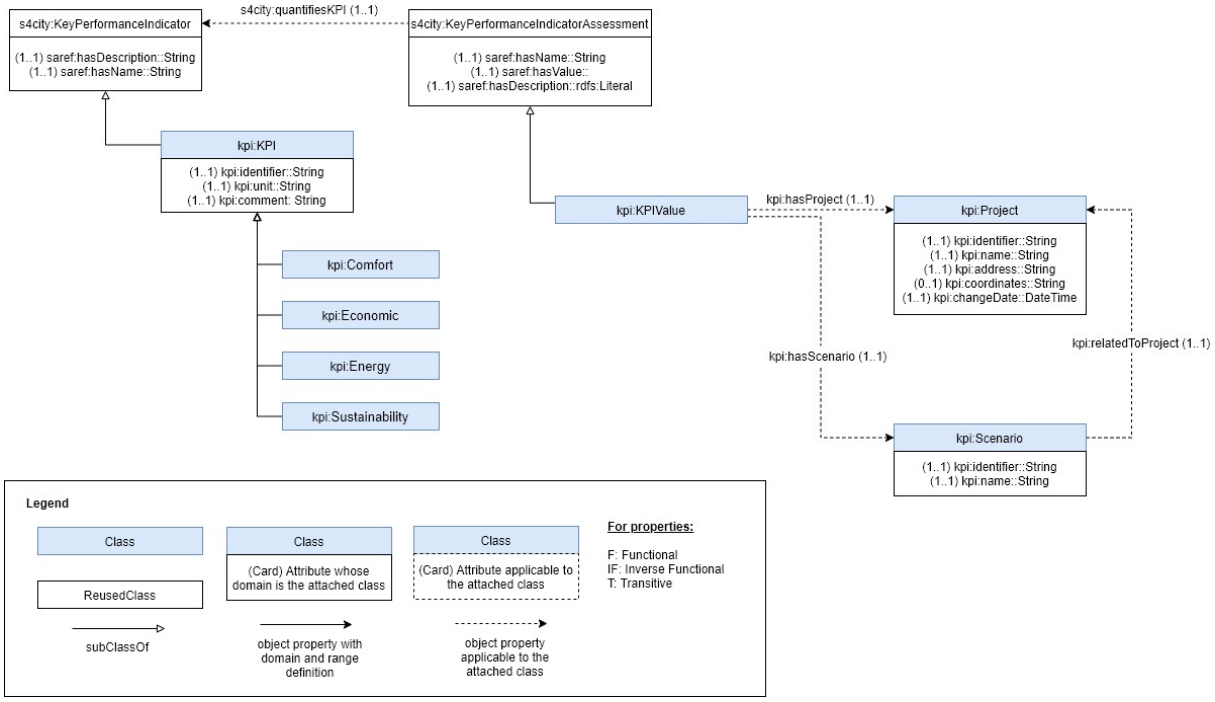

Figure 17. General overview of the Key Performance ontology.

3.3.1 Model Description

The main idea behind this module is to represent some concepts related to the monitoring of project processes and also check if the energy performance indicators established for renovation projects are being satisfied. With this goal in mind, the whole conceptualization circulates around the term `kpi:KPIValue`, that as defined previously in section 3.3, is a detailed specification of a performance indicator. This new concept is an extension of the class `s4city:KeyPerformanceIndicatorAssessment` of SAREF4CITY ontology, which already includes some attributes useful for our purposes such as `saref:hasValue` and `saref:hasDescription`. Those attributes allow us to state a numerical value to the metric being evaluated as well as a detailed description of what is being assessed. KPI values define quantitatively a specific indicator. This relation is modeled by the object property `s4city:quantifiesKPI`, that connects to the `kpi:KPI` concept. This class can be further categorized into four major subclasses: `kpi:Comfort`, `kpi:Economic`, `kpi:Energy`, and `kpi:Sustainability`; each one referring to a different energy aspect of a renovation project, and used to measure the

success of it. In order to reuse some of the attributes and relations previously defined in the SAREF4CITY ontology, we model the `kpi:KPI` concept as a subclass of the entity `s4city:KeyPerformanceIndicator`. In this sense we can reuse the attribute `saref:hasName` to provide a proper name to our KPI individuals. However, the units in which we are expressing the value of performance indicators also need to be provided, we model that by adding the attribute `kpi:unit`[20] to the KPI definition. Additionally, data properties such as `kpi:identifier` and `kpi:comment` are added to achieve a full description of the concept.

Finally, a KPI value exists within the context of a project, which is the activity we are trying to measure the success of. This relation is represented by the property `kpi:hasProject` linking a KPI value to the concept `kpi:Project`. Furthermore, a KPI value is calculated under certain conditions. The conceptualization models this set of conditions by means of the class `kpi:Scenario`. The connection used to link both concepts is the object property `kpi:hasScenario`.

3.4 WEATHER MODULE

The Weather ontology has been developed to define climate properties and locate their measurements. This module is intrinsically linked to the sensor data ontology, because most of the data used in this domain consist of measurements.

The current conceptual model defined by the Weather ontology is depicted in Figure 18. This model reuses concepts and properties from the SAREF, SAREF for City (SAREF4CITY) and WGS84 Geo Positioning (WGS84_POS) ontologies. The first two ontologies have been discussed and reused on the previous domains. The last model (WGS84_POS), is an ontology that provides the Semantic Web community with a namespace for representing latitude, longitude and other information about spatially-located things, using WGS84 as a reference datum.

Some of the most important concepts taken from SAREF4CITY are the following:

[20] It should be mentioned that this model about the unit of KPI might evolve once the data and possible values are further analyzed.

- **Administrative Area**: An administrative division, unit, entity, area or region, also referred to as a subnational entity, constituent unit, or country subdivision, is a portion of a country or other region delineated for the purpose of administration.
- **City**: A city is a large human settlement. A city is distinguished from other human settlements by its relatively great size, but also by its functions and its special symbolic status, which may be conferred by a central authority.
- **Country**: A country is a region that is identified as a distinct national entity in political geography.

Also, some of the most important concepts taken from WGS84_POS are the following:

- **SpatialThing**: Anything with spatial extent, i.e. size, shape, or position, e.g. people, places, bowling balls, as well as abstract areas like cubes.
- **Point**: A point, typically described using a coordinate system relative to Earth, such as WGS84.

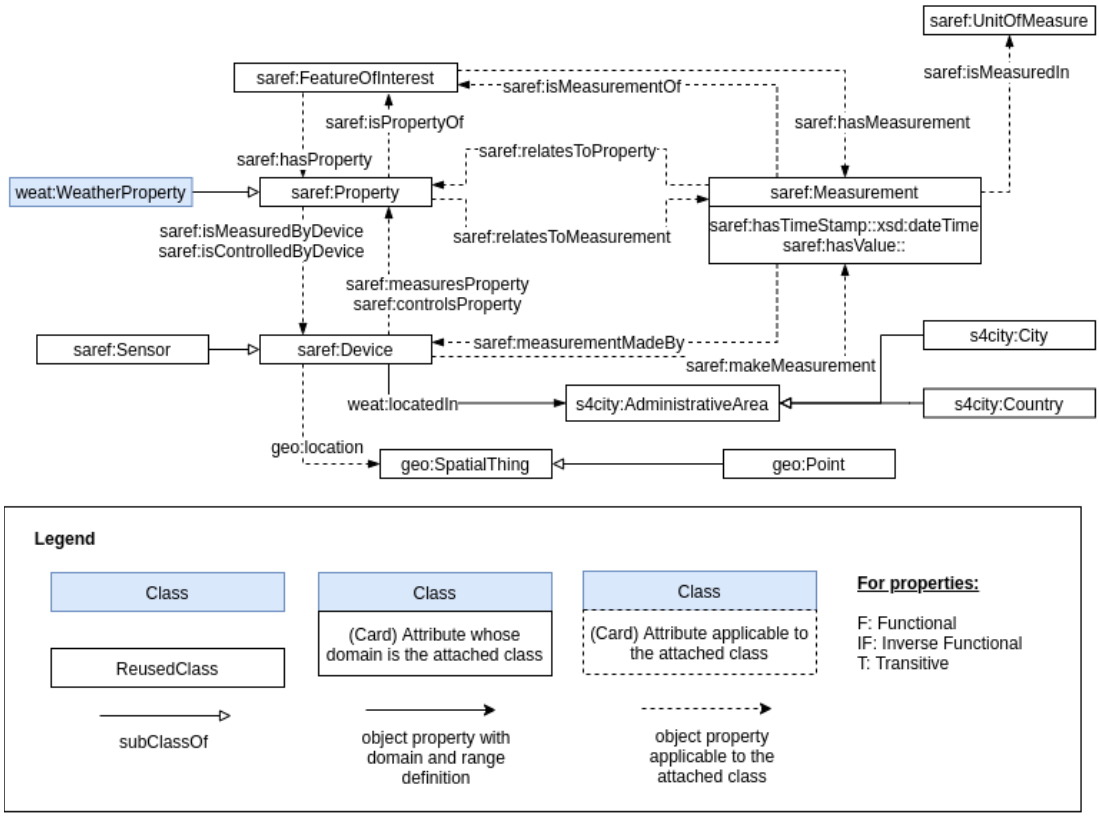

Figure 18. General overview of the Weather ontology.

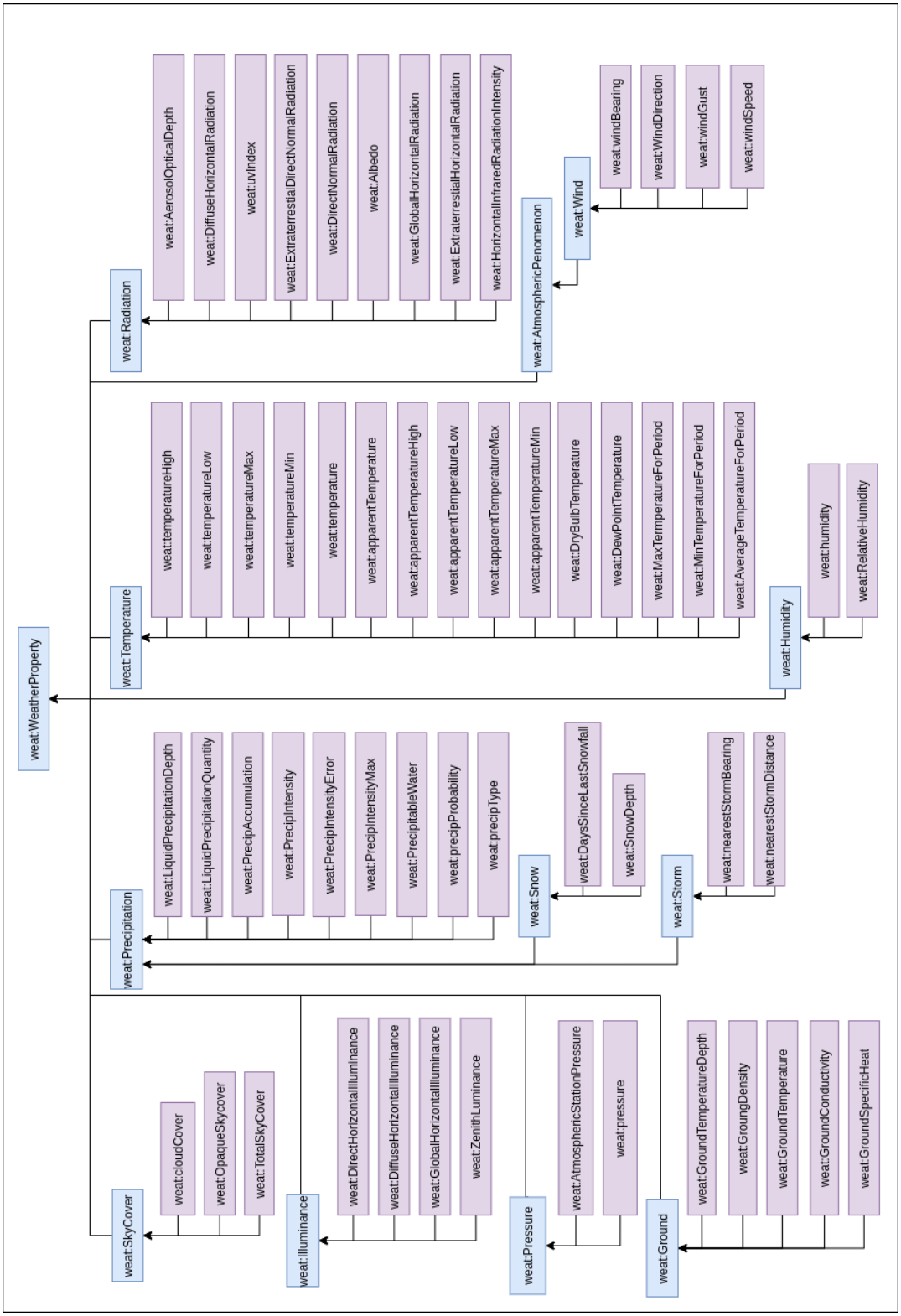

Figure 19. Weather ontology individuals

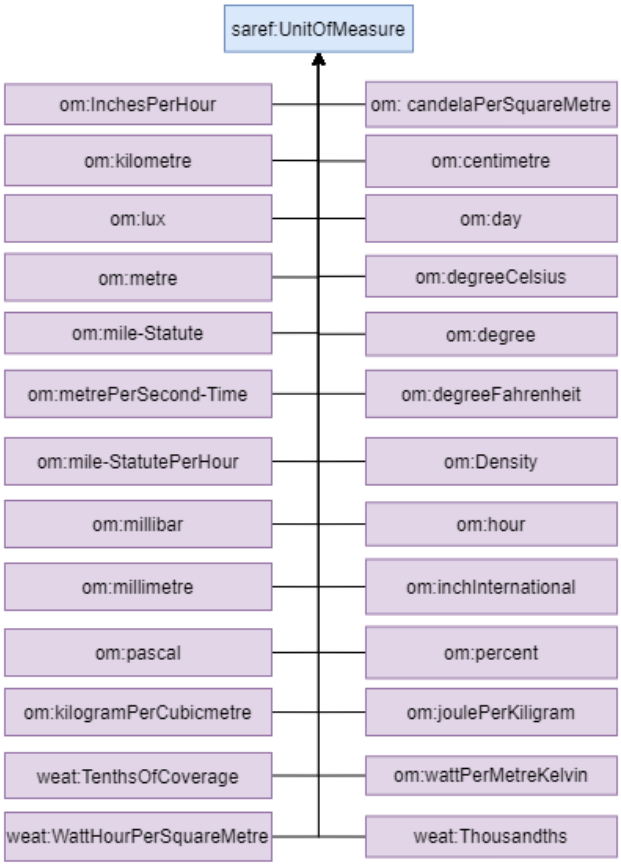

Figure 20. Unit of Measure class individuals.

3.4.1 Model Description

The main concept defined in the ontology, as shown in Figure 18 is `weat:WeatherProperty`. In this model conceptualization we can observe that almost all the model is created by reusing SAREF classes as we mentioned earlier, such as `saref:Device`, `saref:Property`, `saref:FeatureOfInterest`, etc. We used this ontology as reference because the model is applicable to the data needs, it is a standard ontology proposed by the European Telecommunications Standards Institute[21] and there is a family of related ontologies that can be reused for other BIMERR ontology modules.

As it can be seen, the observations or measurements (`saref:Measurement`) are related to the observed property (`saref:Property`), the related feature of interest

[21] https://www.etsi.org/

(saref:FeatureOfInterest), and the unit of measurement (saref:UnitOfMeasure) in which the value is provided and the time stamp in which the measure was taken. The weat:WeatherProperty is defined as a subclass of saref:Property and the weather properties about which there will be measurements in the data are represented as instances of weat:WeatherProperty as shown in Figure 19. Instances of saref:UnitOfMeasure are depicted in Figure 20. The values of the specific measurements would be attached to each instance of saref:Measurement by means of the property saref:hasvalue.

On the other hand, a saref:Device location can be indicated by the property geo:location from the WGS84_POS vocabulary.

SAREF4CITY is used for the class s4city:AdministrativeArea with its subclasses s4city:City and s4city:Country, to locate this saref:Device at a country and city.

CONCLUSIONS

This document detailed the methodological and technological set up to develop the BIMERR ontology and data model in T4.2 activities. The current status and development plans for next iterations have been set. More precisely: there is a first version of requirements for 8 modules (4 pending review), implementation of the first ontology version for 4 ontologies and conceptualization for 6 ontologies (4 first versions already implemented and published). In addition, the first steps towards the data model generation have been established and the lifecycle and interaction between ontologies and data models have been defined.

During the processes and different tasks related to the BIMERR Interoperability Framework it appeared that the well-known Competency Questions technique is not the most suitable one for every project and situation. While it might helpful to identify main concepts and top-level relations, it is not optimal for other use cases. For example, when the interlocutor's profile is closer to the actual data to be modelled in the ontology or data model, more structure artefacts helps to better describe the data exchange requirements. In this sense, the use of adapted templates from METHONTOLOGY has greatly improved the efficiency and communication between ontology and data model developers and the partners involved in the data extraction requirements. In addition, such templates have been designed taking into account the ontology and data model needs, gathering in one place all the information. This feature allowed us to reduce the number of iterations and resources generated during the task, and therefore, the need to constantly maintain the templates.

Regarding the ontology and non-ontological resources reuse, on the one hand, it has been observed that the list and analysis previously done in the project help ontology developers to locate and rank resources of specific domains, like the obXML model to be considered during the implementation. On the other hand, it could be said that part of the ontology reused should be carried out over the concrete data exchange requirements, that is, when specific conceptualizations are reused. In this sense, not only specific resources of the domain at hand are useful, but also general and cross-domain ontologies known by ontology developers.

Also, regarding the data models, even though some data models and sources were identified at the beginning of the project, during the specific data requirements the Energy Plus Weather[22] and darksky[23] has been selected as data sources. This situation, together with the fact that the SAREF ontology provides a network of extensions that integrates an harmonizes different domains, it has been decided to reuse SAREF model for weather information. This decision has been made after check that the SEAS weather ontology does no perfectly fit our needs and some adaptations would be needed and that SAREF is compatible with SSN/SOSA[24] model for the case at hand.

The technological support provided by collaborative environments such as Confluence is useful for keeping track of the discussions together with the documentation generation in a closed environment. However, once the ontology requirements are stable and final it is preferable to publish them together with the resulting ontologies in the portal.

The needed ontology and data model alignments regarding the metadata have been identified in this deliverable as well as the first steps to automatically generate the data model from existing OWL ontologies. This analysis has brought us to the conclusion that the OWL implementation of the ontologies will not be enough to generate a complete data model and that an enriched layer is needed as well. In addition, it should be mentioned that ensuring backwards compatibility and proper evolution of a) ontologies, b) data model and c) existing data remains challenging.

For the next steps, the requirements for the domains of H&S issues, annotated information objects and renovation measures will be reviewed in order to proceed to their conceptualization.

---

[22] https://energyplus.net/weather

[23] https://darksky.net

[24] https://www.w3.org/TR/vocab-ssn/

Special attention will be required with regard the data exchange requirements for the building domain, and related subdomains as geometry and components. While it is a crucial part of the project, the requirements are not yet addressed due to the close relation with IFC management for which discussions are going on within WP4 and also WP3 (architecture). These will affect the level of detail needed from the ontology and data model. More precisely, there are two main decisions to be made in order to establish the level of detail needed by BIF: a) the combination of the IFC functionalities from the BIM manager platform and the BIF and b) the level of detail of the IFC information expected to be queried through the query builder by different BIMERR modules and applications. Once these decisions are taken, the building related information will be prioritized. For the already published ontologies, further validation approaches should be applied, for example, by conducting a coverage analysis of the requirements.

Finally, the ontology annotation layer to produce the enriched version of the BIMERR ontologies for the data model generation should be defined according to the data model needs.

BIBLIOGRAPHY

[Ahmad et. al., 2018] Ahmad, A., Daniel, G., Poveda-Villalón, M., Santana-Pérez, I., Fernández-Izquierdo, A., and Oscar, C.. (2018, July 20). OnToology (Version 1.4). Zenodo. http://doi.org/10.5281/zenodo.1317786

[Fernández-López et. al., 1997] Fernández-López, M., Gómez-Pérez, A., and Juristo, N. (1997). METHONTOLOGY: from ontological art towards ontological engineering. In Proceedings of the Ontological Engineering AAAI97 Spring Symposium Series. American Association for Artificial Intelligence.

[Fernández-López et. al., 1999] Fernández-López, M., Gómez-Pérez, A., Pazos-Sierra, J., and Pazos-Sierra, A. (1999). Building a chemical ontology using methontology and the ontology design environment. IEEE Intelligent Systems, 14(1):37–46.

[García-Castro et. al., 2017] García-Castro, R., Fernández-Izquierdo, A., Heinz, C., Kostelnik, P., Poveda-Villalón, M. and Serena, F. (2017, August 31) Detailed Specification of the Semantic Model, Technical Report, Universidad Politécnica de Madrid (UPM), 2017, VICINITY Project. https://vicinity2020.eu.

[Grüninger and Fox, 1995] Grüninger, M. and Fox, M. (1995). Methodology for the design and evaluation of ontologies. In IJCAI'95, Workshop on Basic Ontological Issues in Knowledge Sharing.

[Hasse et. al., 2009] Haase, P., Brockmans, S., Palma, R., Euzenat, J., d'Aquin, M.: D1.1.2 updated version of the networked ontology model. Tech. rep., Universitat Karlsruhe (2009), neOn Project. http://www. neon-project. org

[Hong et. al., 2019] Hong, T., D'Oca, S., Taylor-Lange, S. and Turner, W. (2015). An ontology to represent energy-related occupant behavior in buildings Part II: Implementation of the DNAS framework using an XML schema. http://dx.doi.org/10.1016/j.buildenv.2015.08.006

[Poveda-Villalón, 2012] Poveda-Villalón, M. A reuse-based lightweight method for developing linked data ontologies and vocabularies (2012, May). In *Extended Semantic Web Conference* (pp. 833-837). Springer, Berlin, Heidelberg. May 2012.

[Poveda-Villalón et. al., 2014] Poveda-Villalón, M., Gómez-Pérez, A., and Suárez-Figueroa, M. C. (2014). OOPS! (ontology pitfall scanner!): An on-line tool for ontology evaluation. International Journal on Semantic Web and Information Systems (IJSWIS), 10(2), 7-34.

[Poveda-Villalón, 2016] Poveda Villalón, M. (2016). Ontology Evaluation: a pitfall-based approach to ontology diagnosis. Thesis (Doctoral), E.T.S. de Ingenieros Informáticos (UPM). https://doi.org/10.20868/UPM.thesis.39448.

[Poveda-Villalón et. al., 2019a] Poveda-Villalón, M., Biliri, E., Bosché, F., Elmasllari, E., Fenz, S., García-Castro, R., Giannakis, G., Kakardakos, T., Katsifaraki, A., Kyprianidis, A., Neubauer, T., Ntalaperas, D., Papapolyzos, T., Parn, E., Rontogianni, E., Tavakolizadeh, F., Tiwari, S., Tsakiris, T., Valero, E., and Vergeti. D. (2019). D3.2 Survey of data models, ontologies and standards in the wider Energy Efficient Buildings domain, Technical Report, Universidad Politécnica de Madrid (UPM), 2019. BIMERR Project. https://bimerr.eu/.

[Poveda-Villalón et. al., 2019b] Poveda-Villalón, M., Fernández-Izquierdo, A., and and García-Castro, R. (2019, January 14). Linked Open Terms (LOT) Methodology (Version 1.0). Zenodo. http://doi.org/10.5281/zenodo.2539305

[SenML, 2018] Jennings, C., Shelby, Z., Arkko, J. (2013) Media Types for Sensor Markup Language (SenML). https://tools.ietf.org/html/rfc8428 (Work in process, Last update: 08-2018)

[Studer et. al., 1998] Studer, R., Benjamins, V.R., and Fensel, D. Knowledge engineering: Principles and methods. Data & Knowledge Engineering 25(1-2) (1998) 161-197

[Suárez-Figueroa et. al., 2012] Suárez-Figueroa, M., Gómez-Pérez, A., and Fernández-López, M. (2012). The NeOn methodology for ontology engineering. In Ontology engineering in a networked world (pp. 9-34). Springer Berlin Heidelberg.

[Tiwari et. al., 2019] Tiwari, S., Alexopoulou, E., Bosché, F., Bountouni, N., Fenz, S., García-Castro, R., Giannakis, G., Kakardakos, T., Koussouris, S., Kyprianidis, A., Neubauer, T., Ntalapera, D., Papapolyzos, T., Parna, E., Prekas, G., Ratajczak-Jeziorska, J., Rontogianni, E., Tsakiris, T., Vafeiadis, G., Valero, E., and Zacharis, E. (2019) Report on Semantic Alignment & Linking of EEB-related ontologies, Technical Report, Universidad Politécnica de Madrid (UPM), 2019. BIMERR Project. https://bimerr.eu/.

[Tsoulos et. al., 2019] Tsoulos, G., Aggelopoulos, K., Athanasiadou, G., Bosché, F., Bouras, T., Demeter, D., Elmasllari, E., Fenz, S., Francesca, E., Hanel, T., Kakardakos, T., Katsifaraki, A., Kousouris, S., Manesis, F., Mojžiš, M., Ntalaperas, D., Papanikolaou, A., Parn, E., Puczyłowska, A., Ratajczak, J., Skiadaresis, G., Tsitsanis, T., Valero, E., Vergeti, D., Walch, M., Zacharis, E., and Zarbouti, D. (2019) Stakeholder requirements for the BIMERR system, Technical Report, Universidad Politécnica de Madrid (UPM), 2019. BIMERR Project. https://bimerr.eu/.

[Garijo et. al., 2020] Garijo, D., Geluk, J., Scharm, M., Ruiz-Iniesta, A., kartgk, Serafini, A., ... Schneider, J. (2020, January 12). dgarijo/Widoco: WIDOCO 1.4.13: Linking evaluation

in documentation (Version v1.4.13). Zenodo. http://doi.org/10.5281/zenodo.3605675

ANNEX 1: TABLES OF REQUIREMENTS FOR THE OCCUPANCY PROFILE DOMAIN

For the sake of readability empty columns have been omitted in the following tables.

Table 7. Table of concepts for the Occupancy Profile domain

Concept	Description
Building	Building where the behavior occurs.
Occupant	Occupants inside a building.
Behavior	Occupant behaviors.
Space	Internal spaces of the building.
Meeting	Meeting information if the space is communal.
System	The systems the occupants interact with.
Window	Class representing window individuals.
Shade	Class representing shade individuals.
Light	Class representing light individuals.
Thermostat	Class representing thermostat individuals.
HVAC	Class representing HVAC individuals.
Driver	Cause that motivates the behavior.
Need	The comfort need of the occupant.
Action	The action the occupant does to satisfy its need.
Driver Time	Type of driver related to time.
Environment	Type of driver related to the environment.
Environment Parameter	Parameters that describe the surrounding environment.
Temperature Parameter	Parameter to represent the temperature within a space.
IAQ Parameter	Parameter to represent the air quality inside a space.
Daylight Parameter	Parameter to represent the amount of day light a space receives.
Illuminance Parameter	Parameter to represent the illuminance inside a space.
Glare Parameter	Parameter to represent the amount of glare inside a space.
Relative Humidity Parameter	Parameter to represent the relative humidity.
Solar Irradiance Parameter	Parameter to represent solar irradiance.
Raining Parameter	Parameter to represent rain levels.
Noise Parameter	Parameter to represent noise levels inside spaces.
Spatial	Type of driver related to the space itself.
Need	Type of driver related to the occupant needs.

Physical Need	Class to represent all the physical needs required by occupants to achieve comfort.
Nonphysical Need	Class to represent internal needs that contribute to achieve comfort such as privacy or status.
Thermal Need	Occupant thermal need.
Acoustic Need	Occupant acoustic need.
Visual Need	Occupant visual need.
IAQ Need	Occupant indoor air quality need.
Interaction	Interaction between an occupant and the system.
Inaction	Class to represent the inaction of an occupant to achieve its comfort needs.
Report	Report of the conditions without an action.
Movement	Occupant movements around a space.
Interaction Formula	Formula to model the probability of interaction between an occupant and a system.
Interaction Coefficient	Coefficient of an interaction formula.
Movement Model	Model to define occupant movements inside a space.
Markov Chain Model	Type of model movement.
Space Occupancy	Occupancy characteristic of a space.
Event	Type of driver related to the events that happen inside a space.
Habit	Type of driver related to occupant habits.

Table 8. Table of relations for the Occupancy Profile domain

Concept	Relation	Target Object
Building	hasSpace	Space
Space	meeting	Meeting
Space	hasSystem	System
Space	usedBy	Occupant
Window	subClassOf	System
Shade	subClassOf	System
Light	subClassOf	System
Thermostat	subClassOf	System
Equipment	subClassOf	System
HVAC	subClassOf	System
Occupant	hasBehavior	Behavior
Behavior	driver	Driver
Behavior	need	Need
Behavior	action	Action

Behavior	interactWith	System
DriverTime	subClassOf	Driver
Environment	subClassOf	Driver
Spatial	subClassOf	Driver
Habit	subClassOf	Driver
Event	subClassOf	Driver
Environment	hasParameter	Environment Parameter
Temperature Parameter	subClassOf	Environment Parameter
IAQ Parameter	subClassOf	Environment Parameter
DaylightFactor Parameter	subClassOf	Environment Parameter
Illuminance Parameter	subClassOf	Environment Parameter
Glare Parameter	subClassOf	Environment Parameter
Relative Humidity Parameter	subClassOf	Environment Parameter
Solar Irradiance Parameter	subClassOf	Environment Parameter

Table 9. Table of attributes for the Occupancy Profile domain

Concept	Attribute	Description	Value Type	Max Cardinality
Building	buildingID	Identification of Building	Integer	1
Building	buildingType	Type of building	{Retail, Gymnasium, Hotel ...}	1
Building	buildingAddress	Address of the building	String	1
Building	description	Description of the building	String	1
Space	spaceID	Identification of the Building Space	Integer	1
Space	spaceType	Type of space	{MeetingRoom, Corridor, Outdoor, ...}	1
Space	description	Description of the space	String	1
Space	maxNumberOccupants	Maximum number of occupants	Integer	1

Space	minNumberOccupants	Minimum number of occupants	Integer	1
Meeting	meetingDuration	Duration of meeting	Integer	1
Meeting	meetingStartTime	Start of the meeting	Date	1
Meeting	meetingEndTime	End of the meeting	Date	1
Meeting	meetingProbability	Probability of meeting occurrence	Float	1
System	systemID	Identification of System	Integer	1
System	description	Description of the system	String	1
Window	windowControlType	Type of control for the window	{operable, fixed}	1
Shade	shadeControlType	Type of control for the shade	{operable, fixed}	1
Light	lightControlType	Type of control for the light	{on/off, dimmable, two step, three step}	1
Thermostat	thermostatControlType	Type of control for the thermostat	{adjustable, none, fixed}	1
HVAC	hvacControlType	Type of control for the HVAC	{central, zonal controllable, zonal fixed}	1
Occupant	occupantID	Identification of the occupant	Integer	1
Occupant	occupantName	Name of the occupant	String	1
Occupant	occupantAge	Age of the occupant	Integer	1
Occupant	occupantGender	Gender of the occupant	String	1
Occupant	occupantLifestyle	Lifestyle of the occupant	String	1
Occupant	occupantJobtype	Jobtype of the occupant	String	1
Behavior	behaviorID	Identification of the behavior	Integer	1
Behavior	description	Desciption of the behavior	String	1
DriverTime	timeOfDay	Time of the day	{morning, noon, evening, midnight}	1

DriverTime	dayOfWeek	Day of the week	{Monday, Tuesday, ...}	1
DriverTime	seasonTime	Season	{summer, winter, fall, spring}	1
Environment Parameter	environmentID	Identification of environment	Integer	1
Environment Parameter	Name	Name of the specific environment	String	1
Environment	environmentUnit			
Driver	driverEventType	Details the circumstances that may be driving occupant actions	String	1
Driver	driverHabit	Personal enumeration traits such as smoking	String	1
Spatial	spatialFunctionalType	Type of space	{residential, office}	1
Spatial	spatialOwnershipType	Type of space	{owned, rented}	1
Spatial	spaceID	Identification of the Building Space	Integer	1
PhysicalNeed	needID	Identification of the need	Integer	1
PhysicalNeed	minConfortValue	Minimum value of the confort range	Float	1
PhysicalNeed	maxConfortValue	Maximum value of the confort range	Float	1
ThermalNeed	comfortOptions	Different type of thermal comfort options	{ISO adaptive comfort standard, ASHRAE adaptive comfort standard, ASHRAE comfort envelop, user-defined comfort envelope}	1
NonPhysicalNeed	nonPhysicalNeedType	Different types of non physical needs such as	String	1

		privacy, view, safety, etc.		
Interaction Formula	description	Description of the formula	String	1
Interaction Parameter	environmentID	Identification of environment (parameter)	Integer	1
SpaceOccupancy	spaceCategory		String	1
SpaceOccupancy	percentTimePresence		Float	1
SpaceOccupancy	duration		Float	1
Event	eventType		String	1
Event	eventTypicalTime		Date	1
Event	eventStartTime		Date	1
Event	eventEndTime		Date	1

ANNEX 2: TABLES OF REQUIREMENTS FOR THE SENSOR DATA DOMAIN

Table 10. Table of concepts for the Sensor Data domain

Concept	Description
Device	A tangible object designed to accomplish a particular task in households, common public buildings or offices.
Sensor	Device that perform the measurement.
Actuator	Device that perform the action in the control loop.

Table 11. Table of relations for the Sensor Data domain

Concept	Relation	Target Object
Sensor	subClassOf	Device
Actuator	subClassOf	Device

Table 12. Table of attributes for the Sensor Data domain

Concept	Attribute	Description	Value Type Expected	Max Cardinality	Unit of Measure
Device	name	Name of the device	string	1	
Device	timeType	Type of time, relative or absolute	{relative, absolute}	1	
Device	timeValue	Time value, when the measurement was taken	float	1	seconds
Device	unit	Unit of the measurement or command	float	1	
Device	value	Value of the measurement or command	Float	1	
Device	sum	Integrated sum of values over time	float		
Device	basName	A base name that is added to the name field.	String		
Device	baseTime	A base time that is added to the time field.	dateTime		seconds

Device	baseSum	A base sum that is added to the sum field	float		
Device	baseValue	A base value that is added to the value field	float		
Device	baseVersion	Version number of the media type format	Integer		
Sensor	updateTime	Period of time that represents the time before a sensor will provide an update reading	float		

ANNEX 3: TABLES OF REQUIREMENTS FOR THE KEY PERFORMANCE INDICATOR DOMAIN

Table 13. Table of concepts for the Key Performance Indicator domain

Concept	Description
KPI	Key performance indicators (KPIs) represent a set of measures focusing on those aspects of organizational performance that are the most critical for the current and future success of the organization.
KPI Value	Represents the concrete value a specific KPI takes after being calculated by an agent in a given time.
Project	A temporary endeavor undertaken to create a unique product or service.
Scenario	A renovation scenario which provides the user with accurate information about energy estimations, cost, and LCA data.

Table 14. Table of relations for the Key Performance Indicator domain

Concept	Relation	Target Object	Max Cardinality
KPI Value	hasKPI	KPI	1
KPI Value	hasProject	Project	1
KPI Value	hasScenario	Scenario	1
Scenario	hasProject	Project	1

Table 15. Table of attributes for the Key Performance Indicator domain

Concept	Attribute	Value Type Expected	Max Cardinality
KPI	ID	string	1
KPI	Name	string	1
KPI	Unit	string	1
KPI	Category	{Economic, energy, sustainability, comfort}	1
KPI	Comment	Float	1
KPI Value	Value	float	*

Project	ProjectID	String	1
Project	Name	String	1
Project	Address	String	1
Project	Coordinates	String	1
Project	changeDate	dateTime	1
Scenario	scenarioID	string	1
Scenario	Name	String	1

ANNEX 4: TABLES OF REQUIREMENTS FOR THE WEATHER DOMAIN

Table 16. Table of concepts for Weather domain (Source: EPW)

Concept	Description
Date	Date when the data was measure. Include details such as year, month, day, hour, minute.
Data Source and Uncertainty Flags	The data and uncertainty flags from various formats (usually show with each field) are consolidated in the E/E+ EPW format
Dry Bulb Temperature	This is the dry bulb temperature in °C at the time indicated
Dew Point Temperature	This is the dew point temperature in °C at the time indicated
Relative Humidity	This is the Relative Humidity in percent at the time indicated
Atmospheric Station Pressure	This is the station pressure in Pa at the time indicated
Extraterrestial Horizontal Radiation	This is the Extraterrestial Horizontal Radiation in Wh/m2
Extraterrestial Direct Normal Radiation	This is the Extraterrestial Direct Normal Radiation Intensity in Wh/m2. (Amount of solar radiation in Wh/m2 received on a surface normal to the rays of the sun at the top of the atmosphere during the number of minutes preceding the time indicated)
Horizontal Infrared Radiation Intensity	This is the Horizontal Infrared Radiation Intensity in Wh/m2.
Global Horizontal Radiation	This is the Global Horizontal Radiation in Wh/m2 (Total amount of direct and diffuse solar radiation in Wh/m2 received on a horizontal surface during the number of minutes preceding the time indicated)
Direct Normal Radiation	This is the Direct Normal Radiation in Wh/m2. (Amount of solar radiation in Wh/m2 received directly from the solar disk on a surface perpendicular to the sin's rays, durong the number of minutes predecing the time indicated)
Diffuse Horizontal Radiation	This is the Diffuse Horizontal Radiation in Wh/m2. (Amount of solar radiation in Wh/m2) received from the sky (excluding the solar disk) on a horizontal surface during the number of minutes preceding the time indicated
Global Horizontal Illuminance	This is the Global Horizontal Illuminance in lux. (Average total amount of direct and diffuse illuminance in hundreds of lux received on a horizontal surface during the number of minutes preceding the time indicated).
Direct Normal Illuminance	This is the Direct Normal Illuminance in lux. (Average amount of illuminance in hundreds of lux received directly from the solar disk on a surface perpendicular to the sun's rays, during the number of minutes preceding the time indicated.)

Diffuse Horizontal Illuminance	This is the Diffuse Horizontal Illuminance in lux. (Average amount of illuminance in hundreds of lux received from the sky (excluding the solar disk) on a horizontal surface during the number of minutes preceding the time indicated.)
Zenith Luminance	This is the Zenith Illuminance in Cd/m2. (Average amount of luminance at the sky's zenith in tens of Cd/m2 during the number of minutes preceding the time indicated.)
Wind Direction	This is the Wind Direction in degrees where the convention is that North = 0.0, East = 90.0, South = 180.0, West = 270.0. (Wind direction in degrees at the time indicated. If calm, direction equals zero.)
Wind Speed	This is the wind speed in m/sec. (Wind speed at time indicated.)
Total Sky Cover	This is the value for total sky cover (tenths of coverage). (i.e. 1 is 1/10 covered. 10 is total coverage). (Amount of sky dome in tenths covered by clouds or obscuring phenomena at the hour indicated at the time indicated.)
Opaque Sky Cover	This is the value for opaque sky cover (tenths of coverage). (i.e. 1 is 1/10 covered. 10 is total coverage). (Amount of sky dome in tenths covered by clouds or obscuring phenomena that prevent observing the sky or higher cloud layers at the time indicated.)
Visibility	This is the value for visibility in km. (Horizontal visibility at the time indicated.)
Ceiling Height	This is the value for ceiling height in m. (77777 is unlimited ceiling height. 88888 is cirroform ceiling.)
Present Weather Observation	Since the primary use of these fields (Present Weather Observation and Present Weather Codes) is for rain/wet surfaces, a missing observation field or a missing weather code implies no rain.
Present Weather Codes	The present weather codes field is assumed to follow the TMY2 conventions for this field
Precipitable Water	This is the value for Precipitable Water in mm. (This is not rain - rain is inferred from the PresWeathObs field but a better result is from the Liquid Precipitation Depth field))
Aerosol Optical Depth	This is the value for Aerosol Optical Depth in thousandths

Snow Depth	This is the value for Snow Depth in cm. This field is used to tell when snow is on the ground and, thus, the ground reflectance may change
Days Since Last Snowfall	This is the value for Days Since Last Snowfall
Albedo	The ratio (unitless) of reflected solar irradiance to global horizontal irradiance
Liquid Precipitation Depth	The amount of liquid precipitation (mm) observed at the indicated time for the period indicated in the liquid precipitation quantity field.
Liquid Precipitation Quantity	The period of accumulation (hr) for the liquid precipitation depth field
Latitude	Latitude where the measure was taken in degree minutes represented in decimal (i.e. 30 minutes is .5)
Longitude	Longitude where the measure was taken in degree minutes represented in decimal (i.e. 30 minutes is .5)
Altitude	Elevation where the measure was taken.
Ground Conductivity	Conductivity of the soil.
Ground Density	Density of the soil.
Ground Specific Heat	Specific heat of the soil.
Ground Temperature	Average ground temperature of period.
Ground Temperature Depth	Depth where the temperature of the ground was taken.

Table 17. Table of concepts for Weather domain (Source: DarkSky API)

Concept	Description
apparentTemperature	The apparent (or “feels like”) temperature in degrees Fahrenheit.
apparentTemperatureHigh	The daytime high apparent temperature.
apparentTemperatureHighTime	The UNIX time representing when the daytime high apparent temperature occurs.
apparentTemperatureLow	The overnight low apparent temperature.
apparentTemperatureLowTime	The UNIX time representing when the overnight low apparent temperature occurs.
apparentTemperatureMax	The maximum apparent temperature during a given date.
apparentTemperatureMaxTime	The UNIX time representing when the maximum apparent temperature during a given date occurs.

apparentTemperatureMin	The minimum apparent temperature during a given date.
apparentTemperatureMinTime	The UNIX time representing when the minimum apparent temperature during a given date occurs.
cloudCover	The percentage of sky occluded by clouds, between 0 and 1, inclusive.
dewPoint	The dew point in degrees Fahrenheit.
humidity	The relative humidity, between 0 and 1, inclusive.
icon	A machine-readable text summary of this data point, suitable for selecting an icon for display. If defined, this property will have one of the following values: clear-day, clear-night, rain, snow, sleet, wind, fog, cloudy, partly-cloudy-day, or partly-cloudy-night. (Developers should ensure that a sensible default is defined, as additional values, such as hail, thunderstorm, or tornado, may be defined in the future.)
moonPhase	The fractional part of the lunation number during the given day: a value of 0 corresponds to a new moon, 0.25 to a first quarter moon, 0.5 to a full moon, and 0.75 to a last quarter moon. (The ranges in between these represent waxing crescent, waxing gibbous, waning gibbous, and waning crescent moons, respectively.)
nearestStormBearing	The approximate direction of the nearest storm in degrees, with true north at 0° and progressing clockwise. (If nearestStormDistance is zero, then this value will not be defined.)
nearestStormDistance	The approximate distance to the nearest storm in miles. (A storm distance of 0 doesn't necessarily refer to a storm at the requested location, but rather a storm in the vicinity of that location.)
ozone	The columnar density of total atmospheric ozone at the given time in Dobson units.
precipAccumulation	The amount of snowfall accumulation expected to occur (over the hour or day, respectively), in inches. (If no snowfall is expected, this property will not be defined.)
precipIntensity	The intensity (in inches of liquid water per hour) of precipitation occurring at the given time. This value is conditional on probability (that is, assuming any precipitation occurs at all).

precipIntensityError	The standard deviation of the distribution of precipIntensity. (We only return this property when the full distribution, and not merely the expected mean, can be estimated with accuracy.)
precipIntensityMax	The maximum value of precipIntensity during a given day.
precipIntensityMaxTime	The UNIX time of when precipIntensityMax occurs during a given day.
precipProbability	The probability of precipitation occurring, between 0 and 1, inclusive.
precipType	The type of precipitation occurring at the given time. If defined, this property will have one of the following values: "rain", "snow", or "sleet" (which refers to each of freezing rain, ice pellets, and “wintery mix”). (If precipIntensity is zero, then this property will not be defined. Additionally, due to the lack of data in our sources, historical precipType information is usually estimated, rather than observed.)
pressure	The sea-level air pressure in millibars.
summary	A human-readable text summary of this data point. (This property has millions of possible values, so don’t use it for automated purposes: use the icon property, instead!)
sunriseTime	The UNIX time of when the sun will rise during a given day.
sunsetTime	The UNIX time of when the sun will set during a given day.
temperature	The air temperature in degrees Fahrenheit.
temperatureHigh	The daytime high temperature.
temperatureHighTime	The UNIX time representing when the daytime high temperature occurs.
temperatureLow	The overnight low temperature.
temperatureLowTime	The UNIX time representing when the overnight low temperature occurs.
temperatureMax	The maximum temperature during a given date.
temperatureMaxTime	The UNIX time representing when the maximum temperature during a given date occurs.
temperatureMin	The minimum temperature during a given date.
temperatureMinTime	The UNIX time representing when the minimum temperature during a given date occurs.

time	The UNIX time at which this data point begins. minutely data point are always aligned to the top of the minute, hourly data point objects to the top of the hour, daily data point objects to midnight of the day, and currently data point object to the point of time provided all according to the local time zone.
uvIndex	The UV index.
uvIndexTime	The UNIX time of when the maximum uvIndex occurs during a given day.
visibility	The average visibility in miles, capped at 10 miles.
windBearing	The direction that the wind is coming from in degrees, with true north at 0° and progressing clockwise. (If windSpeed is zero, then this value will not be defined.)
windGust	The wind gust speed in miles per hour.
windGustTime	The time at which the maximum wind gust speed occurs during the day.
windSpeed	The wind speed in miles per hour.

Table 18. Table of attributes for Weather domain (Source: EPW)

Concept	Attribute	Description	Value Type	Unit of Measure
Dry Bulb Temperature	temperature / dry bulb data	This is the dry bulb temperature in °C at the time indicated	integer	°C
Dew Point Temperature	temperature / dew point data	This is the dew point temperature in °C at the time indicated	integer	°C
Relative Humidity	humidity / relative humidity data	This is the Relative Humidity in percent at the time indicated	integer	%
Atmospheric Station Pressure	pressure / station pressure data	This is the station pressure in Pa at the time indicated	integer	Pa
Extraterrestial Horizontal Radiation	radiation /extraterrestrial horizontal radiation data	This is the Extraterrestial Horizontal Radiation in Wh/m2	integer	Wh/m2
Extraterrestial Direct Normal Radiation	radiation / extraterrestrial direct radiation data	This is the Extraterrestial Direct Normal Radiation Intensity in Wh/m2. (Amount of solar radiation in Wh/m2 received on a surface normal to the rays of the sun at the top of the atmosphere during the	integer	Wh/m2

		number of minutes preceding the time indicated)		
Horizontal Infrared Radiation Intensity	radiation / horizontal infrared radiation data	This is the Horizontal Infrared Radiation Intensity in Wh/m2.	integer	Wh/m2
Global Horizontal Radiation	radiation / global horizontal radiation data	This is the Global Horizontal Radiation in Wh/m2 (Total amount of direct and diffuse solar radiation in Wh/m2 received on a horizontal surface during the number of minutes preceding the time indicated)	integer	Wh/m2
Direct Normal Radiation	radiation / direct radiation data	This is the Direct Normal Radiation in Wh/m2. (Amount of solar radiation in Wh/m2 received directly from the solar disk on a surface perpendicular to the sin's rays, durong the number of minutes predecing the time indicated)	integer	Wh/m2
Diffuse Horizontal Radiation	radiation / diffuse horizontal radiation data	This is the Diffuse Horizontal Radiation in Wh/m2. (Amount of solar radiation in Wh/m2) received from the sky (excluding the solar disk) on a horizontal surface during the number of minutes preceding the time indicated	integer	Wh/m2
Global Horizontal Illuminance	illumninance / global horizontal illuminance data	This is the Global Horizontal Illuminance in lux. (Average total amount of direct and diffuse illuminance in hundreds of lux received on a horizontal surface during the number of minutes preceding the time indicated).	integer	lux
Direct Normal Illuminance	illuminance / direct illuminance data	This is the Direct Normal Illuminance in lux. (Average amount of illuminance in hundreds of lux received directly from the solar disk on a surface perpendicular to the sun's rays, during the number of minutes preceding the time indicated.)	integer	lux
Diffuse Horizontal Illuminance	illuminance / diffuse horizontal illuminance data	This is the Diffuse Horizontal Illuminance in lux. (Average amount of illuminance in hundreds of lux received from the sky (excluding the solar disk) on a horizontal surface during the number of minutes preceding the time indicated.)	integer	lux
Zenith Luminance	illuminance / zenith luminance data	This is the Zenith Illuminance in Cd/m2. (Average amount of luminance at the sky's zenith in tens of Cd/m2 during the number of minutes preceding the time indicated.)	integer	Cd/m2

Wind Direction	wind / wind direction data	This is the Wind Direction in degrees where the convention is that North = 0.0, East = 90.0, South = 180.0, West = 270.0. (Wind direction in degrees at the time indicated. If calm, direction equals zero.)	integer	degrees(°)
Wind Speed	wind / wind speed data	This is the wind speed in m/sec. (Wind speed at time indicated.)	integer	m/sec
Total Sky Cover	Sky cover / total sky cover data	This is the value for total sky cover (tenths of coverage). (i.e. 1 is 1/10 covered. 10 is total coverage). (Amount of sky dome in tenths covered by clouds or obscuring phenomena at the hour indicated at the time indicated.)	integer	tenths of coverage
Opaque Sky Cover	Sky cover / opaque sky cover data	This is the value for opaque sky cover (tenths of coverage). (i.e. 1 is 1/10 covered. 10 is total coverage). (Amount of sky dome in tenths covered by clouds or obscuring phenomena that prevent observing the sky or higher cloud layers at the time indicated.)	integer	tenths of coverage
Visibility	Visibility data	This is the value for visibility in km. (Horizontal visibility at the time indicated.)	integer	km
Ceiling Height	Ceiling height data	This is the value for ceiling height in m. (77777 is unlimited ceiling height. 88888 is cirroform ceiling.)	integer	m
Present Weather Observation	Weather observation data	Since the primary use of these fields (Present Weather Observation and Present Weather Codes) is for rain/wet surfaces, a missing observation field or a missing weather code implies no rain.	string	-
Present Weather Codes	Weather codes data	The present weather codes field is assumed to follow the TMY2 conventions for this field	string	-
Precipitable Water	Precipitation data	This is the value for Precipitable Water in mm. (This is not rain - rain is inferred from the PresWeathObs field but a better result is from the Liquid Precipitation Depth field))	integer	mm
Aerosol Optical Depth	Aerosol depth data	This is the value for Aerosol Optical Depth in thousandths	integer	thousandths
Snow Depth	Snow depth data	This is the value for Snow Depth in cm. This field is used to tell when snow is on the ground and,	integer	cm

		thus, the ground reflectance may change		
Days Since Last Snowfall	Last snowfall data	This is the value for Days Since Last Snowfall	integer	days
Albedo	Albedo data	The ratio (unitless) of reflected solar irradiance to global horizontal irradiance	string	-
Liquid Precipitation Depth	precipitation / liquid precipitation depth data	The amount of liquid precipitation (mm) observed at the indicated time for the period indicated in the liquid precipitation quantity field.	integer	mm
Liquid Precipitation Quantity	precipitation / liquid precipitation quantity data	The period of accumulation (hr) for the liquid precipitation depth field	integer	hours
Latitude	latitude data	Latitude where the measure was taken in degree minutes represented in decimal (i.e. 30 minutes is .5)	integer	-
Longitude	longitude data	Longitude where the measure was taken in degree minutes represented in decimal (i.e. 30 minutes is .5)	integer	-
Altitude	altitude data	Elevation where the measure was taken.	integer	-
Ground Conductivity	ground/ground conductivity data	Conductivity of the soil.	integer	-
Ground Density	ground/ground density data	Density of the soil.	integer	-
Ground Specific Heat	ground/ground specific heat	Specific heat of the soil.	integer	-
Ground Temperature	ground/ground temperature	Average ground temperature of period.	integer	-
Ground Temperature Depth	ground/ground temperature depth	Depth where the temperature of the ground was taken.	integer	-

Table 19. Table of attributes for Weather domain (Source: DarkSky API)

Concept	Attribute	Description	Value Type	Unit of Measure
apparentTemperature	appTemp	The apparent (or "feels like") temperature in degrees Fahrenheit.	integer	Fahrenheit
apparentTemperature High	appTempHigh	The daytime high apparent temperature.	integer	Fahrenheit

apparentTemperatureHighTime	appTempHTime	The UNIX time representing when the daytime high apparent temperature occurs.	integer	-
apparentTemperatureLow	appTempLow	The overnight low apparent temperature.	integer	Fahrenheit
apparentTemperatureLowTime	appTempLTime	The UNIX time representing when the overnight low apparent temperature occurs.	integer	-
apparentTemperatureMax	appTempMax	The maximum apparent temperature during a given date.	integer	Fahrenheit
apparentTemperatureMaxTime	appTempMaxTime	The UNIX time representing when the maximum apparent temperature during a given date occurs.	integer	-
apparentTemperatureMin	appTempMin	The minimum apparent temperature during a given date.	integer	Fahrenheit
apparentTemperatureMinTime	appTempMinTime	The UNIX time representing when the minimum apparent temperature during a given date occurs.	integer	-
cloudCover	cloudCover	The percentage of sky occluded by clouds, between 0 and 1, inclusive.	float	-
dewPoint	dewPointDSA	The dew point in degrees Fahrenheit.	integer	Fahrenheit
humidity	humidityDSA	The relative humidity, between 0 and 1, inclusive.	float	-
icon	icon	A machine-readable text summary of this data point, suitable for selecting an icon for display. If defined, this property will have one of the following values: clear-day, clear-night, rain, snow, sleet, wind, fog, cloudy, partly-cloudy-day, or partly-cloudy-night. (Developers should ensure that a sensible default is defined, as additional values, such as hail, thunderstorm, or tornado, may be defined in the future.)	string	-

moonPhase	moonPhase	The fractional part of the lunation number during the given day: a value of 0 corresponds to a new moon, 0.25 to a first quarter moon, 0.5 to a full moon, and 0.75 to a last quarter moon. (The ranges in between these represent waxing crescent, waxing gibbous, waning gibbous, and waning crescent moons, respectively.)	integer	-
nearestStormBearing	nearestStormB	The approximate direction of the nearest storm in degrees, with true north at 0° and progressing clockwise. (If nearestStormDistance is zero, then this value will not be defined.)	integer	degrees
nearestStormDistance	nearestStormD	The approximate distance to the nearest storm in miles. (A storm distance of 0 doesn't necessarily refer to a storm at the requested location, but rather a storm in the vicinity of that location.)	integer	miles
ozone	ozone	The columnar density of total atmospheric ozone at the given time in Dobson units.	integer	m/V
precipAccumulation	precipAcc	The amount of snowfall accumulation expected to occur (over the hour or day, respectively), in inches. (If no snowfall is expected, this property will not be defined.)	integer	inches
precipIntensity	precipInt	The intensity (in inches of liquid water per hour) of precipitation occurring at the given time. This value is conditional on probability (that is, assuming any precipitation occurs at all).	integer	inches/h
precipIntensityError	precipIntError	The standard deviation of the distribution of precipIntensity. (We only return this property when the full distribution, and not merely the expected mean, can be estimated with accuracy.)	integer	-
precipIntensityMax	precipIntMax	The maximum value of precipIntensity during a given day.	integer	-
precipIntensityMaxTime	precipIntMaxTime	The UNIX time of when precipIntensityMax occurs during a given day.	integer	-
precipProbability	precipProb	The probability of precipitation occurring, between 0 and 1, inclusive.	integer	-

precipType	precipType	The type of precipitation occurring at the given time. If defined, this property will have one of the following values: "rain", "snow", or "sleet" (which refers to each of freezing rain, ice pellets, and “wintery mix”). (If precipIntensity is zero, then this property will not be defined. Additionally, due to the lack of data in our sources, historical precipType information is usually estimated, rather than observed.)	integer	-
pressure	pressure	The sea-level air pressure in millibars.	integer	milibar
summary	summary	A human-readable text summary of this data point. (This property has millions of possible values, so don’t use it for automated purposes: use the icon property, instead!)	string	-
sunriseTime	sunriseTime	The UNIX time of when the sun will rise during a given day.	integer	-
sunsetTime	sunsetTime	The UNIX time of when the sun will set during a given day.	integer	-
temperature	T	The air temperature in degrees Fahrenheit.	integer	Fahrenheit
temperatureHigh	THigh	The daytime high temperature.	integer	Fahrenheit
temperatureHighTime	THighTime	The UNIX time representing when the daytime high temperature occurs.	integer	-
temperatureLow	Tlow	The overnight low temperature.	integer	Fahrenheit
temperatureLowTime	TLowTime	The UNIX time representing when the overnight low temperature occurs.	integer	-
temperatureMax	Tmax	The maximum temperature during a given date.	integer	Fahrenheit
temperatureMaxTime	TMaxTime	The UNIX time representing when the maximum temperature during a given date occurs.	integer	-
temperatureMin	Tmin	The minimum temperature during a given date.	integer	Fahrenheit
temperatureMinTime	TMinTime	The UNIX time representing when the minimum temperature during a given date occurs.	integer	-

time	time	The UNIX time at which this data point begins. minutely data point are always aligned to the top of the minute, hourly data point objects to the top of the hour, daily data point objects to midnight of the day, and currently data point object to the point of time provided all according to the local time zone.	integer	-
uvIndex	uvIndex	The UV index.	integer	-
uvIndexTime	uvIndexTime	The UNIX time of when the maximum uvIndex occurs during a given day.	integer	-
visibility	visibility	The average visibility in miles, capped at 10 miles.	integer	miles
windBearing	windB	The direction that the wind is coming from in degrees, with true north at 0° and progressing clockwise. (If windSpeed is zero, then this value will not be defined.)	integer	degrees
windGust	windG	The wind gust speed in miles per hour.	integer	miles/h
windGustTime	windGTime	The time at which the maximum wind gust speed occurs during the day.	integer	-
windSpeed	windS	The wind speed in miles per hour.	integer	miles/h