Informe Técnico / Technical Report

O4OA Specification

Beatriz F. Martins, Renata Guizzardi, and Óscar Pastor López

Ref. #:	**PROS-TR-2023-I**
Title:	**O4OA Specification**
Author (s):	**Beatriz Franco Martins, Renata Guizzardi, and Óscar Pastor**
Corresponding autor (s):	**bmartins@pros.upv.es, r.guizzardi@utwente.nl, opastor@dsic.upv.es**
Document version number:	**1**
Final version:	**-**
Release date:	**-**
Keywords:	**Ontology Classification, Ontology Networks, Ontology Engineering, Ontology, FAIR Principles**

O4OA Specification

Beatriz Franco Martins[1] [0000−0001−9190−1047], Renata Guizzardi[2], and Oscar Pastor[1] [0000−0002−1320−8471]

[1] Valencian Research Institute for Artificial Intelligence (VRAIN),
Universitat Politècnica de València,
Camino de Vera s/n, 46022 Valencia, Spain,
[2] University of Twente, Twente, Nederland,
bmartins@pros.upv.es, rguizzardi@utwente.ne, opastor@dsic.upv.es

This document is the reference ontology specification for the Ontology for Ontological Analysis (O4OA) version 2.6.

Keywords: Ontology Characterization, Ontology Classification, Ontology Networks, Ontology Engineering, Ontology, FAIR Principles, Ontological Analysis

1 Introduction

The **Ontology for Ontological Analysis** (**O4OA**) describes the (meta)characteristics an ontology can have. In this context, we deal with two perspectives, they are: the ontological perspective which is supported by a set of well-known classification approaches, and considers ontologies as artifacts expressed through some language; and the domain perspective which deals with a semantic consensual agreement about the terms and its definitions (concepts) present in ontologies, as well as its sources of information (norms, standards, etc.). This document is the ontology specification for the O4OA (Reference Ontology) version 2.6.

2 Ontological Background

The O4OA covers the (meta)characteristics of ontological artifacts as well its domain aspects. The O4OA has a reference model written in OntoUML [8] that is grounded over UFO [40]. The FAIR Principles drive the covering (meta) characteristics related to both, data and the ontologies that map these data. Indeed, O4OA is FAIRness, i.e. it encompasses FAIR in itself [60]. The O4OA also has an implemented version through a NoSQL database, and the data are managed with a REST API we developed [58].

3 The Reference Ontology

This section summarizes O4OA reference version; however, there is a repository with the complete ontology [3]

[3] https://bfmartins.gitlab.io/o4oa/content/o4oa.html

3.1 Related Ontologies

Table 1 presents the ontologies related to O4OA reference model:

Table 1: RO4OA related ontologies.

Ontology	Relation	Compatibility
UFO	Ontological Grounding through OntoUML.	High
UFO-A	Structural aspects of grounding.	High
UFO-B	Dynamic aspects (Ontology Versions) of grounding.	High
UFO-C	Social aspects (Ontological Commitment) of grounding.	High
MongoO4OA	O4OA implemented model in MongoDB.	High

3.2 Philosophical Base

Ontologies as artifacts are intended to describe and computationally implement conceptualizations designed to clarify some domain of real-world knowledge. We advocate that this activity should be done judiciously and through the use of the best practices of *Ontology Engineering*. This includes the encompassing of the *FAIR Principles* [77,53] for both the data and the ontological artifacts (reference and operational ontologies) itself. In summary, there must be *FAIRness* to data regarding the expectations of Domain Specialists as well as ontologies that describe these data regarding the expectations of Ontology Engineers [61].

As stakeholders of the *Ontology Engineering process*, *Domain Specialists* and *Ontology Engineers* must have clear, homogeneous, and unambiguous concerning conceptualizations they work in. However, they have different concerning and viewpoints, i.e. different perspectives about the conceptualization. The goal of Ontology Engineers is to produce models with the aim of achieving the best possible approximation of a real-world domain; while Domain Specialists want conceptualizations to produce practical results given a certain set of requirements [60]. Figure 1 summarizes this problem.

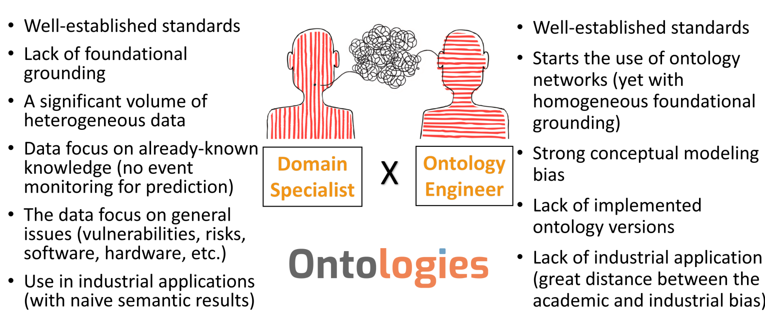

Fig. 1: Different perspectives of stakeholders about ontologies.

Ontological analyzes are a good practice to ensure that ontologies comply with the knowledge clarification requirement about a domain, but in themselves, these analyses do not guarantee FAIRness. To achieve the FAIR Principles, it is necessary to have a common agreement among stakeholders within the Software Engineering process. Moreover, the defined and adopted ontological commitment in the process must be traceable and reproducible. In this sense, O4OA is (Meta)Ontology to describe Ontologies because it models the foundational and domain-related concepts and relationships that are necessary to facilitate the process of Ontological Analysis. The goal (purpose) of O4OA is to clarify and homogenize the necessary (meta)ontological requirements, data, and characteristics to help stakeholders achieve awareness and common sense about conceptualizations [60].

The three pillars of O4OA are:

Classification: Classifies ontologies according to well-known classifications (level of application, level of generality, level of formalization, and level of axiomatization) and established within Ontology Engineering [59,61].

Characterization: Characterizes ontologies by establishing relationships between ontological (meta)characteristics such as their classification, language, representation, purpose, accessibility, copyright, reuse, and implementation, among others [60].

Discrimination: Provides a cloud of concepts that goes beyond the concepts adopted inside an ontology because it brings to light different elements of standardization and policies concerning the Domain Specialists' perspective. Moreover, this brings enlighten possible related ontologies, opening the domains' boxes and their ontologies, in which conceptualizations compound ontology networks [59,69].

All these three pillars are present in O4OA meanwhile being supported by the FAIR Principles.

3.3 Ontological Commitment

Among the definitions of what is an ontology, such as the works [30,12,71,20,39] the ontology definition evolve. According to an initial definition provided the work [31], the Ontological Commitment is a guarantee of completeness about some conceptualization, i.e. they are agreements to use the shared vocabulary in a coherent and consistent manner. Meanwhile, the work [35] formalize this notion by providing the notion that artifacts are ontological committed when there is map between a logical language and a set of structures. These definitions are not antagonistic because human communication requires both syntactic and semantic approaches, the first one is discretionary and the second meaningful. In this sense, syntax and semantic are "sides of the same coin". O4OA comply with a set of notions and best-practices to support its pillars (Classification, Characterization, and Discrimination), all grounded over UFO.

Classification: Among the existing dimensions and approaches to classify ontologies we adopted the most well-established and comprehensive used by the Ontology Engineering community, which are four levels:

Level of applicability: Identifying if there is a Reference Ontology supporting its Operational Version proposed in [42].

Level of generality: Classifying the ontology among Domain, Task, Application, or Core Ontologies (that are more general Domain Ontologies) plus identifying if there is any grounding provided by an Upper Ontology (also called Foundational Ontology). This dimension uses the proposals [32,76].

Level of formalization: Identifying impact of design decisions over the ontology, this include (meta)characteristics related with representation, language limitations, among others. The works [27] and [74] support this classification.

Level of axiomatization: Identifying axiomatic weight that the formalization provides in the ontology. The work [26] supports this classification, besides it is directly linked with the level of formalization classification we adopt.

Figure 2 illustrates these classifications levels.

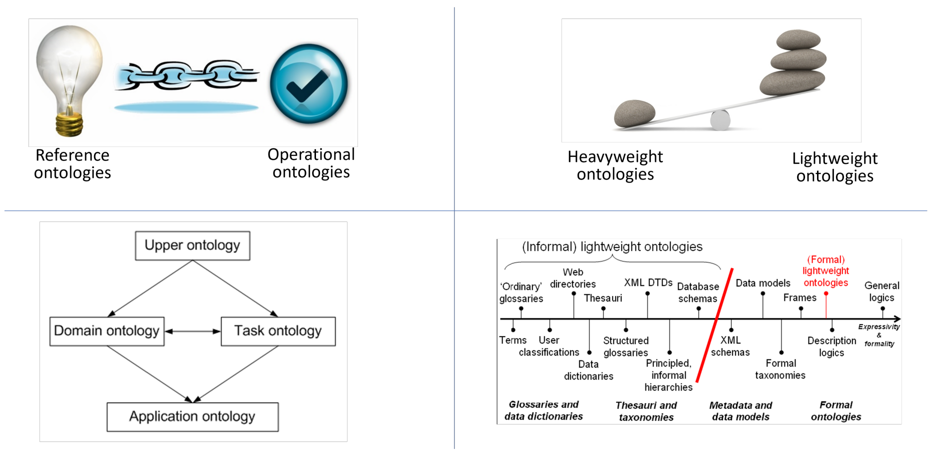

Fig. 2: Ontology classification that O4OA commits with [32,76,27,75,26].

Ontology Characterization: Using the classification and other intrinsic (meta)characteristics of the ontologies it is possible to deal with their relational (meta)characteristics. Table 2 depicts the main relational aspects O4OA deal with to characterize ontologies.

Domain Characterization: From the definitions that exist in ontologies and confronting them with the formed cloud of concepts, it is possible to discriminate (meta)characteristic about the domain conceptualization. A cloud of concepts is supported by well-known standards (sources of documentation) about knowledge domains, besides it exists independently of the ontologies. This approach provides independence over domain concepts, fomenting reasoning about conceptualizations.

Table 2: Ontology characterization commitment.

Relation	Description	Formalism
Ontology Characterization		
drives	When the language of the ontology is specified through an Ontology-Driven Metamodel	Relational Dependence
drives'	When the language of the ontology is specified through an Ontology-Driven Foundational Metamodel	Relational Dependence
groundedOver	When the ontology concepts are grounded over a Foundational ontology.	Specialization
	When the language of the ontology is specified through a Foundational-Ontology-Driven Metamodel	Relational Dependence
subOntology	When an ontology is composed by two or more ontologies	Part/Whole (functional whole)
implementationFor	When an Operational Ontology is defined by a Reference Ontology	Relational Dependence
reuses	When ontologies have concept overlapping	Relational dependence
specifies	When there is a formal specification for a language	Relational dependence
specifies'	When there is a formal specification for an ontology-driven language	Relational dependence
moreFormalThan	When an ontology is more formal than another	Comparison
moreExpressiveThan	When an ontology is more expressive than another	Comparison
moreAxiomatizedThan	When an ontology is more axiomatized than another	Comparison
formedBy	When an ontology a member of an ontology network	Part/Whole (collective)
Ontology (Meta)characterization		
exteriorizes	The language patterns and anti-patterns characteristics that normalize or digress types of ontologies	Relational dependence
carves	The language appropriateness characteristics that handle types of ontologies	Relational dependence
nandles	The language appropriateness over types of ontologies	Characterization
normalizes	The language design-patterns of types of ontologies	Characterization
digresses	The language anti-patterns of types of ontologies	Characterization
moreAppropriateThan	When languages fit to describe certain type of ontology more than another	Comparison
instanceOf	When ontologies are instances of a type of ontology	Instantiation

Table 3: Domain characterization commitment.

Relation	Description	Formalism
Domain Characterization		
describes	The definition provided by a source (documented) that supports a concept	Relational Dependence
formedBy	The set of concepts that form a cloud of concepts	Part/Whole (collective)
contains	Concepts definitions that compound versions of ontologies	Part/Whole (part-wood)
usedToImplement	The representation language that is used to express a conceptualization, denoting a domain description (ontology version)	Relational Dependence
usedToImplement	The implementation language that is used to design an ontology, denoting a ontology schema (ontology version)	Relational Dependence

Table 3 depicts the main relational aspects O4OA deal with conceptualizations and the cloud of concept.

3.4 Competence Questions

Table 4 shows the Competence Questions (CQs) that O4OA must answer.

Table 4: Competence Questions.

Ref.	Competence Question
CQ1	*How to conceptually characterize an ontology (as an artifact)?*
CQ1.1	What is the application level of an ontology?
CQ1.2	What is the generality level of an ontology?
CQ1.3	What is the formalization level of an ontology?
CQ1.4	What is the axiomatization level of an ontology?
CQ1.5	Does an ontology be a well-grounded conceptualization?
CQ1.6	Does an ontology be a well-defined conceptualization?
CQ1.7	Which (meta)characteristics of an ontology interfere with its characterization?
CQ1.8	How do the languages in which ontologies are represented or implemented interfere with their conceptualization?
CQ2	*How to conceptually characterize the domain cloud of concepts of an ontology?*
CQ2.1	What is a concept when it is represented in the context of one or more conceptualizations?
CQ2.2	Which information sources support the conceptualization of a domain?
CQ2.3	Which are the terms and their definitions (cloud of concepts) belonging to the conceptualization of a domain?
CQ2.4	Which sources provide the same (equal, equivalent, similar) definition for a particular term, in the face of one or more conceptualizations?
CQ2.5	Which are the terms (or synonyms) have the same definition, in the face of one or more conceptualizations?
CQ2.6	What are the different definitions for a term in the cloud of concepts, in the face of one or more conceptualizations?
CQ2.7	What are the different terms for a definition in the cloud of concepts, in the face of one or more conceptualizations?
CQ2.8	Which is the distribution of terms in the cloud of concepts, in the face of one or more conceptualizations?
CQ3	*How to conceptually characterize ontology networks (its ontologies as a whole)?*
CQ3.1	What is an ontology grounded over another ontology?
CQ3.2	What are sub-ontologies as parts of another ontology?
CQ3.3	What is the reuse of ontologies?
CQ3.4	What is a language that is ontology-driven?
CQ3.5	What is an ontology implemented?

3.5 Concepts Definition

Table 5 shows the definitions for O4OA concepts.

Table 5: O4OA concepts definition.

Concept	Definition	Stereotype
Abstract (Syntax) Language	The role of a metamodel when it is used to define syntax to a language. The set of available graphic modeling primitives forms the lexical layer and the language abstract syntax is typically defined in terms of a metamodel [42]. A representation of the valid expressions of a domain-specific language (to determine the abstract syntax of the language) [15].	*<<role>>*
Abstract Ontology-Driven Language	The abstract languages (*Abstract (Syntax) Language*) whose metamodels correspond to metamodels driven by ontologies (Ontology-Driven Metamodel), i.e. metamodels constrained by ontologies.	*<<role>>*
ALC	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as ALC [4,3].	*<<subkind>>*
Application Ontology	Application ontologies describe concepts depending both on a particular domain and task, which are often specializations of both the related ontologies. These concepts often correspond to roles played by domain entities while performing a certain activity, like replaceable unit or spare component [32].	*<<subkind>>*
Appropriateness	The intrinsic characteristics present in certain types of languages make them ideal for representing phenomena through an ontology by stakeholders. A language ideal to represent phenomena in a given domain if the metamodel of this language is isomorphic to the ideal ontology of that domain, and the language only has as valid specifications those whose logical models are exactly the logical models of the ideal ontology [42,44]. The Domain Appropriateness of a language is a measure of the suitability of a language to model phenomena in a given domain, or in other words, of its truthfulness of a language to a given domain in reality. Comprehensibility appropriateness refers to how easy is for a user a given language to recognize what that language's constructs mean in terms of domain concepts and, how easy is to understand, communicate and reason with the specifications produced in that language [42].	*<<relator>>*
Atomic Ontology	An ontology that is standalone, i.e. an ontology without sub-ontologies.	*<<subkind>>*
Axiomatization Level	Classify ontologies by identifying axiomatic weight that the formalization provides in the ontology. The work [26] supports this classification, besides it is directly linked with the level of formalization classification we adopt.	*<<category>>*
Behaviour	The intrinsic behavioral characteristics present in certain types of ontologies (artifacts) make them an artifacts to express/describe behaviors. They are used to represent a domain description behavior or to produce behavior in an ontology schema. As it is an element of characterization of ontology types, it denotes the (meta)characteristics present in ontologies (instances of ontology types). The notions of Ontological Commitment [35,44,42] and Ontological Level [33,34] define the behavior described in representations.	*<<mode>>*
Cloud of Concepts	Cloud of Concepts are brainstormed collections of Concepts available in ontologies, independently of their definitions. In other words, a concept (in a cloud) participates in ontologies and may have different definitions (meanings). Indeed, a cloud of concepts is supported by well-known standards (sources of documentation) about knowledge domains, but there are no guarantees that these sources always provide the same (or similar) meaning for a concept. This approach provides independence over domain concepts, fomenting reasoning about conceptualizations.	*<<collective>>*
Composite Ontology	An ontology that is composed of two or more ontologies.	*<<subkind>>*
Computational Language	Languages that directly or indirectly (by steps) can be assembled, compiled, or interpreted by machines, i.e., machine-readable languages.	*<<subkind>>*

– continued on next page

Table 5 continued from previous page

Concept	Definition	Stereotype
Concept	A concept, as an ability (or a *domain abstraction* [42]), relies on the notion of agents possesses moments (mental and cognitive) that regard the capacity of some properties of certain individuals to refer to possible situations of reality [43]. Already, a concept, as an abstract object, refers to the symbolism these agents assign to a *Concept* as an ability; i.e., in this view concepts are the meanings [66] - (containing) modes of presentation, in the Fregean's sense [79]. O4OA names a Concept as an abstract object that contains moments because a definition is associated with a term to refer to this concept as an ability. Note that a concept (<<*mode*>> - ability) is externally dependent on a *Source* (<<*role*>>) to provide a Concept Definition; moreover, it has an *Existential Dependence* on a *Concept* (<<*role*>> - abstract object). This is in line with *Relator Pattern* that bears the notion of *Relational Moments* of UFO [23].	<<*role*>>
Concept Definition	A concept definition is a conceptualization unit clearly expressed that is part of a common sense. In other words, concepts definitions are the building blocks that are elicited in ontologies (as artifacts). Moreover, they are also building blocks for ontological analysis due to they are traceable elements. Formally in O4OA a same concept can be associated with several definitions (equal, similar, divergent or partially divergent) just as the same definition can be given to several concepts (synonymous or not). This allow stakeholders to identify domain conceptual convergences and divergences (and reason about) in ontologies (both in conceptualizations and in implementations).	<<*relator*>>
Conceptualization	The work [42] clarifies two distinct, but correlated, notions about what is a conceptualization. The first (adopted in this definition), is the notion that ontologies provide stakeholders a set of language modeling primitives (i.e. *Concept* – abstract objects or <<*role*>>) that can directly express relevant domain concepts (as abilities or <<*mode*>>), comprising the called *domain conceptualization*. In this case, the elements constituting a conceptualization of a given domain are used to articulate abstractions of a particular state of affairs in reality. And the second is that conceptualizations and abstractions are immaterial entities that only exist in the mind of the user or a community of users of a language [42]. In this case, a conceptualization refers to (mental and cognitive) moments composed of concepts that are also (mental and cognitive) moments; both, are <<*mode*>>. Therefore, O4OA deals with conceptualization as the <<*role*>> of an ontology in representing a domain. Note, this notion is composed by the notion of a conceptualization as <<*mode*>> and externally depends on the *Representation Language*. Indeed, this is an OntoUML pattern described in [23].	<<*role*>>
Concrete Ontology-Driven Language	The "subset" of concrete languages (Concrete (Syntax) Language) that mediates ontology-driven language specifications (Ontology-Driven Language Specifications), i.e., languages whose metamodels are constrained by ontologies.	<<*role*>>
Concrete (Syntax) Language	Concrete syntax is crucial to language design, and it deserves to be a separate element within the language description. Furthermore, the language description should at least contain a mapping from concrete to abstract syntax, and preferably also from abstract to concrete syntax [5]. The concrete syntax is defined by a set of display schemes. A display scheme is attached to each metaclass of the metamodel. Although schemes have a formal structure and can be processed by tools, the syntax definition they provide is nevertheless easily accessible by humans [22].	<<*role*>>
Copyright	The ontology version legal rights. If a person or an organization holds the copyright on a piece of writing, music, etc., they are the only people who have the legal right to publish, broadcast, perform it, etc., and other people must ask their permission to use it or any part of it [1].	<<*mode*>>
Core Ontology	Core ontologies occupy an intermediate position between the superior (or foundation) ontologies and domain ontologies. In general, core ontologies rely on foundation ontologies to add real world semantics to conceptual models, avoiding ambiguities and making them more independent of the domain [76,70].	<<*subkind*>>

– continued on next page

Table 5 continued from previous page

Concept	Definition	Stereotype
`Data Dictionaries`	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as Data Dictionaries. A data dictionary is a centralized repository of information about data such as meaning, relationships to other data, origin, usage, and format. It assists management, database administrators, system analysts, and application programmers in planning, controlling, and evaluating the collections, storage and use of data [**?**].	*<<subkind>>*
`Data Models`	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as Data Models. Types of ontologies represented through Data Models [56,26]. See specifications [4] [5].	*<<subkind>>*
`DB Schemas`	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as Data Base schema. Types of ontologies represented as data base schema (DB Schemas) [74,56,26].	*<<subkind>>*
`Degree of Axiomatization`	The linear dimension (min = 1, max = 100) to estimate and compare axiomatization of an ontology.	*<<integer dimension>>*
`Degree of Formalization`	The linear dimension (min = 1, max = 100) to estimate and compare formalization of an ontology.	*<<integer dimension>>*
`Degree of Expressiveness`	The linear dimension (min = 1, max = 100) to estimate and compare expressiveness of an ontology.	*<<integer dimension>>*
`Delimitation`	The role of intrinsic structural delimitation characteristics present in certain types of ontologies (artifacts) makes them ideal for representing some real-world part of a domain through certain types of language.	*<<role>>*
`Description Logic`	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as Description Logic (DL) [4,3]. Types of ontologies represented in Description Logic [4] based-on languages, such as OWL-DL [56,26]. Note that DL is more expressive than Propositional Logic but lass than General Logic [16]. See OWL-DL specification at [6].	*<<subkind>>*
`Direct Ontological Foundation`	The role of a foundational ontology in grounding non-foundational ontologies that occur when this grounding is made by relations and specializations of general concepts present in the foundational ontology and transmitted (Liskov Principle) to its specializations.	*<<role>>*
`Documents`	According SABiO methodology, the process of knowledge acquisition depends on the domain experts involvement and the a set of consolidated bibliographic material (Documents – including digital versions), such as classical books, international standards, glossaries, lexicons, classification schemes, and reference models [19].	*<<category>>*
`Domain Description`	The best possible description of a domain as an artifact, having a certain ontological commitment and produced through some representation language. A graphical model is a key instrument for supporting communication, meaning negotiation and consensus establishment with domain experts. For building reference domain ontologies, highly-expressive languages should be used to create strongly axiomatized ontologies that approximate as well as possible to the ideal ontology of the domain [19].	*<<relator>>*
`Domain Ontology`	Domain ontologies describe the vocabulary related to a generic domain (like medicine, or automobiles) [32].	*<<subkind>>*
`Encyclopedia`	A book or set of books giving information about all areas of knowledge or about different areas of one particular subject, usually arranged in alphabetical order; a similar collection of information in digital form [1].	*<<kind>>*
`Expressiveness Level`	Classify ontologies based on the degree of expressiveness of the language used to describe them. The work [26] supports this classification, besides it is directly linked with the level of formalization classification we adopt. Indeed it is a bi-dimensional classification.	*<<category>>*
`F-Logic`	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as an ontology schema made with the Frame Logic (F-logic) [50].	*<<subkind>>*

– continued on next page

[4] https://www.omg.org/spec/UML/2.5.1/About-UML
[5] https://www.iso.org/obp/ui/#iso:std:iso:10303:-1:ed-2:v1:en
[6] https://www.w3.org/ns/owl-profile/data/DL

Table 5 continued from previous page

Concept	Definition	Stereotype
(Formal) Lightweight Ontologies	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as (Formal) Lightweight Ontologies. A (formal) lightweight ontology is a triple $O = hN,E,Ci$, where N is a finite set of nodes, E is a set of edges on N, such that $< N,E >$ is a rooted tree, and C is a finite set of concepts expressed in a formal language F, such that for any node $n_i \in N$, there is one and only one concept $c_i \in C$, and, if n_i is the parent node for n_j, then $c_j \sqsubseteq c_i$ [26].	*<<subkind>>*
First-order Logic	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as a FOL description.	*<<subkind>>*
Formal Models (Ontology)	Formal ontologies are rigorously formalized logical theories [74,26]. A formal ontology is a formal description of objects, properties of objects, and relations among objects. This provides the language that will be used to express the definitions and constraints in the axioms. This language must provide the necessary terminology to restate the informal competency questions [75].	*<<category>>*
Formal Ontologies	Ontologies represented through formal ontology-languages [56,26]. Includes: Frames (OKBC); Lightweight Ontologies; Logic Programming (F-Logic); Description Logic (OWL-DL); General Logic.	*<<category>>*
Formal Taxonomes	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as Formal Taxonomies. Beyond informal "$is-a''$ hierarchies, we move to formal "$is-a''$ hierarchies. These include strict subclass relationships. In these systems if A is a superclass of B, then if an object is a subclass of B, it is necessarily the case that it is a subclass of A as well. Similarly, for formal instance relationships, if A"$is-a''$ superclass of B, then if an object is an $instanceofB$, then it is necessarily the case that it is $aninstanceofA$ as well [56,26]. Formal "$is-a''$ hierarchies. In these systems, if B is a subclass of A and an object is an $instanceofB$, then the object is an $instanceofA$. Strict subclass hierarchies are necessary to exploit inheritance. Formal "$is-a''$ hierarchies that include instances of the domain [28].	*<<subkind>>*
Formalization Level	Classify ontologies by identifying impact of design decisions over the ontology, this include (meta)characteristics related with representation, language limitations, among others. The works [27] and [74] support this classification.	*<<category>>*
Foundational Ontology	Foundational (also known as Top-level or Upper) ontologies describe very general concepts like space, time, matter, object, event, action, etc., which are independent of a particular problem or domain: it seems therefore reasonable, at least in theory, to have unified top-level ontologies for large communities of users [32].	*<<subkind>>*
Foundational Ontology-Driven Metamodel	Definitions provided by foundational ontologies define the real-world semantics of foundational ontology-driven languages [8,42,36].	*<<relator>>*
Frames	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as frames. A frame is a complex data structure for representing a stereotypical situation [64]. The concept of a frame system has gained ground as a basic mechanism for representing knowledge. The fundamental idea is very simple: frames are the basic objects of a system, they represent real-world concepts and phenomena; frames can be given named attributes, slots, and slots can be assigned values. Inheritance allows slot values to be used as defaults [55]. The fundamental idea of a frame system is rather simple: A frame represents an object or a concept. Attached to the frame is a collection of attributes (slots), potentially having types (or value restrictions) and potentially filled initially with values. When a frame is being used the values of slots can be altered to make the frame corresponds to the particular situation at hand [56]. Frames. The ontology includes classes and their properties, which can be inherited by classes of the lower levels of the formal is-a taxonomy. In our example, a travel has a unique departure date and an arrival date, a company name and at most one price for a single fare with the company. All these attributes are inherited by the sub-classes of the concept travel [28].	*<<subkind>>*

– continued on next page

Table 5 continued from previous page

Concept	**Definition**	**Stereotype**
`Functional Language`	Higher-order functions and recursion are the basic ingredients of this stateless computational model. The programming languages which presuppose this model are called functional languages and the paradigm that results from this is called the functional programming paradigm [24].	*<<subkind>>*
`General Logic`	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as General Logic. Types of ontologies represented through high-formal languages [56,26]. Includes: First-order Logic; Higher-order Logic; Modal Logic;	*<<subkind>>*
Glossary and Data Dictionaries	A controlled vocabulary is a finite list of terms. Catalogs are an example of this category. Catalogs can provide an unambiguous interpretation of terms – for example, every use of a term (say, "car") will use exactly the same identifier. Another potential ontology specification is a glossary (a list of terms and meanings). The meanings are specified as natural language statements. This provides more information since humans can read the natural language statements. Typically interpretations are not unambiguous and thus these specifications are not adequate for computer agents. They may still be combined with identifiers [56,26]. Includes: Terms; Glossaries (ordinary); Ad-hoc Hierarchies (informal hierarchies); Data Dictionaries.	*<<category>>*
`Graphical Representation`	The intrinsic symbolic graphical characteristics present in certain types of ontologies (artifacts) make them a graphical artifact for representation or implementation.	*<<subkind>>*
`Graphy (Symbolic)`	The intrinsic structural symbolic characteristics present in certain types of ontologies (artifacts) make them able in representing some real-world part of a domain. As it is an element of characterization of ontology types, it denotes the (meta)characteristics present in ontologies (instances of ontology types). The notions of Ontological Commitment [35,44,42] and Ontological Level [33,34] define the graphy (structure) to representations.	*<<mode>>*
`Gray Literature`	Grey literature is information produced outside of traditional publishing and distribution channels, and can include reports, policy literature, working papers, newsletters, government documents, speeches, white papers, urban plans, and so on.	*<<kind>>*
Heavyweight	Ontologies that also contain axioms [27]. Heavyweight ontologies add axioms and constraints to lightweight ontologies [28].	*<<category>>*
`Higher-order Logic`	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as a HOL description.	*<<subkind>>*
Indirect Ontological Foundation	A modeling language based on a foundational ontology can allow for the representation of much more elaborated and semantically precise structures [41]. In order for a modeling language to meet the requirements of expressiveness, clarity and truthfulness in representing the subject domain at hand, it must be an ontologically well-founded language in a strong ontological sense, i.e., it must be a language whose modeling primitives are derived from a proper foundational ontology [8,42,36].	*<<role>>*
Informal (Lightweight) Ontology	An informal ontology is expressed loosely in natural language or in a restricted and structured form of natural language greatly increasing clarity by reducing ambiguity [75].	*<<category>>*
`Informal Taxonomies`	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as Informal Taxonomies [6].	*<<subkind>>*
Implementation	An implementation is a conceptualization (see *Conceptualization*) as the role of a computational version of an ontology representing a domain [19]. Note, this notion also is composed of the notion of a conceptualization as mode (likewise *Conceptualization*), but it externally depends on the *Implementation Language*. Indeed, this is an OntoUML pattern described in [23].	*<<role>>*
Implementation Language	The role of a language when it is used to implement ontologies (*Ontology Schema*).	*<<role>>*
Intended Behavior	The role intended of behavioral characteristics present in certain types of ontologies makes them susceptible to depict semantic patterns.	*<<role>>*
`Knowledge Representation Language`	Proposed in the work [10], KRL is a frame based language with some additional features [64].	*<<subkind>>*
`Knowledge Graphs`	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as Knowledge Graphs.	*<<subkind>>*

– continued on next page

Table 5 continued from previous page

Concept	Definition	Stereotype
Language	Language is a convention that arise from the human capacity of construct a communication channel, i.e. a system of distinct signs corresponding to distinct ideas. Language, as defined, is homogeneous because in the system of signs in which the only essential thing is the union of meanings and sound-images, and in which both parts of the sign are psychological. Language is a concrete thing since its linguistic signs, though basically psychological, are not abstractions. Signs are associations bearing the stamp of collective approval and which added together constitute language-are realities that have their seat in the brain. Besides, linguistic signs are tangible; it is possible to reduce them to conventional written symbols. The very possibility of putting the things that relate to language into graphic form allows dictionaries and grammars to represent it accurately, for language is a storehouse of sound-images, and writing is the tangible form of those images [68].	*<<kind>>*
Language *Specification*	The syntax can and should be divided into two: concrete syntax and abstract syntax, because the superficial structure of a monogram might be completely different from the underlying structure [51]. A modeling language is usually defined in three major steps. The first one is to define concepts of the language, i.e. its vocabulary and taxonomy, as captured by its abstract syntax. Then, its semantics should be described in such a form that the concepts are clearly understood by the users of the language. Finally, it is necessary to precisely describe the notation, as captured by its concrete syntax [22]. The clear separation between abstract and concrete syntax is a technique to cope with the complexity of real-world language definitions since it allows defining the language concepts independently of their representation. For language designers, it is of primary importance to agree on language concepts and on the semantics of these concepts [22].	*<<relator>>*
Language Type	Types of Languages whose instances have intrinsic characteristics that make them be recognized as languages.	*<<type>>*
Lightweight	Ontologies where concepts (described by their attributes and are organized in taxonomies using only the subclass-of relationship), relations and functions, and possibly instances are the only components that are represented [27]. Lightweight ontologies include concepts, concept taxonomies, relationships between concepts, and properties that describe concepts [28]. A (formal) lightweight ontology is a triple $O = hN,E,Ci$, where N is a finite set of nodes, E is a set of edges on N, such that $< N,E >$ is a rooted tree, and C is a finite set of concepts expressed in a formal language F, such that for any node $n_i \in N$, there is one and only one concept $c_i \in C$, and, if n_i is the parent node for n_j, then $c_j \sqsubseteq c_i$ [26]. Taxonomies, thesauri, business catalogs, faceted classifications, web directories, and user classifications are examples of informal prototypes of formal lightweight ontologies. Hereinafter, we will refer to them as (informal) lightweight ontologies [26].	*<<category>>*
Limitation	The intrinsic structural characteristics present in certain types of ontologies (artifacts) make them limited in representing some real-world part of a domain. As it is an element of characterization of ontology types, it denotes the (meta)characteristics present in ontologies (instances of ontology types). The lack of notions of Ontological Commitment [35,44,42] and Ontological Level [33,34] is a limitation on providing meaning to representations.	*<<mode>>*
Lisp	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as an ontology schema made with the Lisp [29]. See specification at [7], [8].	*<<subkind>>*
Logic Programming	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as Logic Programming. The second major paradigm comes from the tradition of logic programming (LP) [57] with one prominent representative being F-Logic ("Ontologies in F-Logic") [39,2].	*<<subkind>>*
Logical Language	The part of the mathematical language that is used in the Logic.	*<<subkind>>*
Markup Language	A notation for identifying the components of a document to enable each component to be appropriately formatted, displayed or used [62].	*<<subkind>>*

– continued on next page

[7] https://www.iso.org/standard/44338.html
[8] https://www.r6rs.org/

Table 5 continued from previous page

Concept	Definition	Stereotype
Mathematical Language	A set of conventionalized rules based in natural languages and other symbolism used by mathematicians, engineers, and scientists to deal with abstractions over the Mathematics.	*<<subkind>>*
Metadata and Data Models	Ontologies represented as Metadata and Data Models [56,26]. Beyond informal "$is-a''$ hierarchies, we move to formal "$is-a''$ hierarchies. These include strict subclass relationships. In these systems if A is a superclass of B, then if an object is a subclass of B, it is necessarily the case that it is a subclass of A as well. Similarly, for formal instance relationships, if A"$is-a''$ superclass of B, then if an object is an $instanceofB$, then it is necessarily the case that it is an $instanceofA$ as well [56,26]. Includes: DB Schemas [74]; XML Schemas; Data Models (UML, STEP); Formal Taxonomies. See specifications [9], [10], [11], [12].	*<<category>>*
Metamodel	A metamodel is a model that consists of statements about models. Hence, a metamodel is also a model but its universe of discourse is a set of models, namely those models that are of interest to the creator of the metamodel. In the context of information systems, a metamodel contains statements about the constructs used in models about information systems [49]. A metamodel is a model used to model modeling itself. A metamodel is also used to model arbitrary metadata (for example, software configuration or requirements metadata) [63]. A representation of the valid expressions of a domain-specific language (to determine the abstract syntax of the language) [15]. A metamodel is a description of the language's abstract syntax since it defines: (i) a set of constructs selected for the purpose of performing a specific (set of) task(s) and, (ii) a set of well-formedness rules for combining these constructs in order to create grammatically valid models in the language [42].	*<<subkind>>*
Modal Logic	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as a Modal Logic description.	*<<subkind>>*
Model	In general, a model can be considered something that serves to describe real-world simulations by using an equivalent or reduced structure in the physical aspect. In the context of Software Engineering and Model-Driven Engineering, several authors present concepts and perspectives on models. A model is a textual or graphical representation of a design (part) [18]. A model is an abstract and unambiguous conception of something (in the real world) that focuses on specific aspects or elements and abstracts from other elements, based on the purpose for which the model is created [54]. A model is a formal specification of the function, structure and behavior of a system within a given context from a specific point of view [73]. A model of a system is a description or specification of that system and its environment for some certain purpose. A model is often presented as a combination of drawings and text. The text may be in a modeling language or in a natural language [7].	*<<kind>>*
Modeling Language	The sub-types of non-computational languages specifically created to represent a conceptualization (as a *<<mode>>*) as model (`Model`).	*<<subkind>>*
Models	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as models. Note that the concept of what is a Model in O4OA, is also present, which is ontologically different meaning.	*<<subkind>>*
Natural Language	A language that has developed in a natural way by people using it to communicate, rather than an invented language or computer code [1]	*<<subkind>>*
Non-Computational Language	Languages that are not machine-readable languages. Note, that non-computational languages eventually can be processable, however, they are not machine-readable in essence. For instance, computers can process music, but music is not machine-readable, the same happens with natural and mathematical languages.	*<<subkind>>*
Non-Foundational Ontologies	Any ontology that is not a Foundational Ontology according to [32].	*<<subkind>>*

– continued on next page

[9] https://www.w3.org/TR/xmlschema11-1/
[10] https://www.w3.org/TR/xmlschema11-2/
[11] https://www.omg.org/spec/UML/2.5.1/About-UML
[12] https://www.iso.org/obp/ui/#iso:std:iso:10303:-1:ed-2:v1:en

Table 5 continued from previous page

Concept	Definition	Stereotype
`NoSQL DB Shemas`	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as NoSQL Data Base Schemas.	*<<subkind>>*
`Number of Axioms`	Number of axioms an ontology have (can be an estimated or approximation, for the purpose of O4OA). Axioms model sentences that are always true and can be used for several purposes, such as constraining information, verifying correctness, or deducting new information. Axioms are also known as assertions (OML) [27]. In the simplest case, an ontology describes a hierarchy of concepts related by subsumption relationships; in more sophisticated cases, suitable axioms are added in order to express other relationships between concepts and to constrain their intended interpretation [32].	*<<perceivable quality>>*
`Object-Oriented Language`	Object-oriented languages introduced the notions of *classes* and *objects*. *Objects* contain both *state* (values) and *methods* (operations). The main operation provided for objects is sending a message to an object. *Classes* provide both *specification* and *implementation* in formation on objects. Not only are the names and specifications of methods included in classes, but also representation information for the state and methods. Most object-oriented languages provide mechanisms for allowing the programmer to restrict access to the representation of the state or methods of objects from clients or sub-classes in order to support information hiding [14].	*<<subkind>>*
`Ontological Constraints`	An explicit representation of the admissible states of the world through a reference domain ontology (which serves as the semantic foundation for domain-specific languages) [15]. Likewise, a reference to a foundational ontology serves as the semantic foundation for foundational ontology-driven languages, as DSL they are [8,42,36].	*<<role>>*
`Ontological Foundation`	The role of an Foundational Ontology plays when provides Ontological Foundation to Non-Foundational Ontologies. The support (Ontological Foundation) of a Foundational Ontology avoids semantic interoperability problems in more specific ontologies, promoting Ontological Grounding [41].	*<<rolemixin>>*
`Ontological Foundation Constraints`	An explicit representation of the admissible states of the world through a reference foundational ontology (which serves as the semantic foundation for foundational ontology-driven languages) [15,8,42,36]. The specialization of Ontological Constraints.	*<<role>>*
`Ontological Grounding`	There is vast literature advocating that Foundational Ontologies have the important role of providing Ontological Grounding to Non-Foundational Ontologies, such as [17,21,36,42,19]. `Ontological Grounding` express the relation that makes foundational ontologies ground non-foundational ontologies, mainly providing to the lasts the *foundational commitments* of the firsts (aspects of being a *Qua-founded* ontology) [37,38,23].	*<<relator>>*
`Ontological Reference`	A domain description when it is used to support its an operational ontology versions (*`Ontology Schema`*).	*<<role>>*
`Ontological Support`	The more appropriate implementation (Ontology Schema) to guarantee desirable computational properties without compromising the previously (Domain Description) defined ontological commitment. Once a reference ontology is produced, many times we want to get an operational version to be used by computer applications. In order to achieve this operational version, we need to design and implement it in a particular machine-readable ontology language [19].	*<<relator>>*
`Ontology (Artifact)`	O4OA strictly adopts the notion that an ontology is an computational artifact. An explicit specification of a shared conceptualization [30]. A formal specification of a shared conceptualization [12]. A formal, explicit specification of a shared conceptualization [71]. A shared and common understanding of a domain that can be communicated between people and application systems [20]. Note that there is a clear difference among an *ontology* as an ontological artifact and *Ontology* as a branch of study in Philosophy [39].	*<<kind>>*
`Ontology-Constrained Metamodel`	The role of a metamodel when it is used to define syntax to languages that comply with an ontology, i.e., an ontology-driven language.	*<<role>>*

– continued on next page

Table 5 continued from previous page

Concept	Definition	Stereotype
Ontology-Driven *Language Specification*	The language specifications (syntax) express the relation between abstract ontology-driven languages and concrete ontology-driven languages. They necessarily derive from metamodels constrained by ontologies.	*<<relator>>*
Ontology-Driven Metamodel	The rigorous definition of the relation between the abstract syntax and the reference domain ontology (to define the real-world semantics of the language) [15].	*<<relator>>*
Ontology-Driven Modeling Language	The sub-types of languages specifically created to represent an ontology-driven model.	*<<subkind>>*
Ontology Network	An Ontology Network is a collection of ontologies related together through various relationships, such as alignment, modularization, and dependency [72]. Ontology networks are not just a set of isolated ontologies grouped together, merely because they act in a domain subdivided into smaller parts (subdomains). On the contrary, they are not limited to a single domain of knowledge, and neither the ontologies of networks are isolated islands of annotated knowledge. Ontologies can relate in different ways within a network.	*<<collective>>*
Ontology Schema	The best possible machine-readable design of a domain representation, having a certain ontological commitment and produced through some implementation language.	*<<relator>>*
Ontology Type (Artifact Type)	Types of artifacts whose instances have intrinsic characteristics that make them be recognized as ontologies.	*<<type>>*
Ontology Version	Under the Ontology Engineering process and its best practices, ontologies must be traceable artifacts. Versioning is an important key for guaranteeing traceability for artifacts, indeed, it is also a consecrated Software Engineering best practice; thus Ontology Engineers also adopt it. Moreover, according to SABiO methodology this notion is strongly present in the differentiation of Operational from Reference Ontologies, the first one as a Domain Description and the other as an Ontology Schema [19].	*<<category>>*
OntoUML	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as a domain description made with the OntoUML languages. [40,8]. See specification at [13], [14], [15], [16].	*<<subkind>>*
Operational Ontology	An Operational Ontology is the actionable version of a Reference Ontology that uses the more appropriate language intending to guarantee desirable computational properties without compromising the previously defined ontological commitment [42].	*<<subkind>>*
Ordinary Glossaries	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as an Ordinary Glossary. A list of technical or special words, especially those in a particular text, explaining their meanings [1].	*<<subkind>>*
Policy	A plan of action agreed or chosen by a political party, a business, etc. [1].	*<<kind>>*
Procedural Language	Procedural languages are the languages whose correct phrases specify actions [24].	*<<subkind>>*
Programing Language	A programming language is an artificial formalism in which algorithms can be expressed [24]. The programming language outlined has been designed for the succinct description of algorithms. The intended range of algorithms is broad; effective applications include micro-programming, switching theory, operations research information retrieval, sorting theory, structure of compilers, search procedures, and language translation [47].	*<<subkind>>*
Prolog	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as an ontology schema made with the Prolog [57]. See specification at [17].	*<<subkind>>*
Reference Ontology	A Reference Ontology should be a conceptualization constructed to make the best possible description of the domain concerning a certain level of granularity and point of view [42].	*<<subkind>>*

– continued on next page

[13] https://nemo.inf.ufes.br/en/projetos/ontouml/
[14] https://github.com/OntoUML
[15] https://ontouml.org/
[16] https://ontouml.readthedocs.io/en/latest/
[17] https://www.iso.org/standard/21413.html

Table 5 continued from previous page

Concept	Definition	Stereotype
`Representation`	The intrinsic symbolic characteristics present in certain types of ontologies (artifacts) make them an artifact for representation or implementation. They are used to represent a domain description or to produce an ontology schema. As it is an element of characterization of ontology types, it denotes the (meta)characteristics present in ontologies (instances of ontology types). The notions of Ontological Commitment [35,44,42] and Ontological Level [33,34] support and provide meaning to representations.	*<<mode>>*
Representation Anti-Pattern	An anti-pattern is a recurrent error-prone modeling decision [52].	*<<role>>*
Representation Language	The role of a language when it is used to represent ontologies (*Domain Description*).	*<<role>>*
Representation Pattern	A pattern is an abstraction from a concrete form that keeps recurring in specific, non-arbitrary contexts [67]. The use of patterns in Software Engineering arises from the acceptance of the well-known notions – for the language Smalltalk – in the work [25] (whose authors are usually referred to as the Gang of Four or just GoF).	*<<role>>*
Reused Ontology	Ontologies when are reused by other ontologies play the role of reused ontologies. Ontologies accomplish the notion of "R" from the FAIR Principles[77,48] play the role of reused ontologies.	*<<role>>*
Reuser Ontology	Ontologies when reuse other ontologies play the role of reuser ontologies. Ontologies accomplish the notion of "R" from the FAIR Principles[77,48] play the role of reused ontologies.	*<<role>>*
`Reusability`	Reusability in the context of software has many dimensions. At its core, reusability aims for someone to be able to re-use software reproducible as described by [9,53]. The context of this usage can vary and should cover different scenarios: (i) reproducing the same outputs reported by the research supported by the software, (ii) (re)using the code with data other than the test one provided [53]. As an software artifact, ontologies is in line with this notions. Reusability: Characteristics of data and their provenance are described in detail according to domain- relevant community standards, with clear and accessible conditions for use [11]. The notion of reusability adopted requires the control of some constraints, they are: (i) Foundational Ontologies can only reuse from Foundational Ontologies; (ii) Core Ontologies can reuse Core Ontologies; Domain Ontologies can reuse Domain Ontologies; Task Ontologies can reuse Task Ontologies; (iii) Application Ontologies can reuse Application, Domain, and/or Task Ontologies; (iv) Application, Domain, Task Ontologies can reuse Core Ontologies; (v) Last but not least, individually, no ontology can not reuse itself or any of its parents (see subOntologies).	*<<relator>>*
`Scripting Language`	Languages that are usually type-less being used to automate the execution of operations in a runtime environment, to bring new functions to applications, and integrate or communicate complex systems and other computational languages. Scripting languages assume that a collection of useful components already exist in other languages. They are intended not for writing applications from scratch but rather for combining components. Scripting languages are often used to extend the features of components; however, they are rarely used for complex algorithms and data structures, which are usually provided by the components. Scripting languages are sometimes referred to as "glue languages" or "system integration languages" [65].	*<<subkind>>*
`Semantic Anti-Pattern`	The anti-patterns, namely, model structures that, albeit producing syntactically valid conceptual models, are prone to result in unintended domain representations, i.e., Semantic Anti-Patterns. They are configurations that when used in a model will typically cause the set of valid (possible) instances of that model to differ from the set of instances representing the intended state of affairs in that domain [46].	*<<relator>>*

– continued on next page

Table 5 continued from previous page

Concept	Definition	Stereotype
Semantic Pattern	Patterns are structures that produce recurrent and syntactically valid conceptual models. When an ontology is built, the (representation or implementation) language used induces the stakeholders to construct conceptualizations via the combination of existing ontologically motivated semantic patterns. These patterns constitute modeling primitives of a higher granularity when compared to usual language primitives. Besides, these higher-granularity modeling elements can only be combined with each other in a restricted set of ways [45].	*<<relator>>*
SHIF	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as SHIF[4,3].	*<<subkind>>*
SHIQ	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as SHIQ [4,3].	*<<subkind>>*
SHOIN	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as SHOIN[4,3].	*<<subkind>>*
Source	Sources represent any documents used to provide definitions for concepts. Sources depict the commitment adopted about a concept. However, it is important to consider. Sources are basically the bases for ontological commitment. Although domain experts are the main source of knowledge acquisition, information sources ensure the registration and traceability of the agreement made (commitment) among stakeholders. Indeed, the *Knowledge Acquisition Process* [19] requires traceability and agreement. Indeed, the notion of Source is in the *Linguistic Level* [33,34].	*<<rolemixin>>*
SROIQ	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as SROIQ[4,3].	*<<subkind>>*
Standard	A Standard is a widely accepted agreement about a domain of knowledge that provides patterns, rules, and guidelines to be followed by stakeholders. Standards can cover a huge range of activities undertaken by governments, organizations, and people. Standards are the distilled knowledge of domain specialists in their subject matter to support the needs of the organizations they represent, such as manufacturers, sellers, buyers, customers, trade associations, users, or regulators.	*<<kind>>*
Structured Glossaries	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as Structured Glossaries. Another potential ontology specification is a glossary (a list of terms and meanings). The meanings are specified as natural language statements. This provides more information since humans can read the natural language statements. Typically interpretations are not unambiguous and thus these specifications are not adequate for computer agents. They may still be combined with identifiers [56].	*<<subkind>>*
Symbolic Appropriateness	The role of intrinsic structural symbolic characteristics present in certain types of ontologies (artifacts) makes them ideal in representing some real-world part of a domain through certain types of language.	*<<role>>*
Task Ontology	Task ontologies describe a generic task or activity (like diagnosing or selling), by specializing the terms introduced in the top-level ontology [32].	*<<subkind>>*
Technical Language	Refers to types of communication that has technical specialized content, i.e., it is a non-computational domain specific natural language.	*<<subkind>>*
Term (syntax)	A word or phrase used to describe a thing or to express a concept, especially in a particular kind of language or branch of study [1].	*<<kind>>*
Terms	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as a structure of terms. People (and computational agents) typically have some notion or conceptualization of the meaning of terms. Software programs sometimes provide a specification of the inputs and outputs of a program, which could be used as a specification of the program. Similarly ontologies can be used to provide a concrete specification of term names and term meanings. Within this line of thought though – where an ontology is a specification of the conceptualization of a term [56]. Note that the concept of what is a Term in O4OA, is also present, which is ontologically different meaning.	*<<subkind>>*

– continued on next page

Table 5 continued from previous page

Concept	Definition	Stereotype
`Tessauri`	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as a Thesauri. Thesauri provide some additional semantics in their relations between terms. They provide information such as synonym relationships. In many cases their relationships may be interpreted unambiguously by agents. Typically thesauri do not provide an explicit hierarchy (although with narrower and broader term specifications, one could deduce a hierarchy) [56].	*<<subkind>>*
Thesauri and Taxonomies	Thesauri provide some additional semantics in their relations between terms. They provide information such as synonym relationships. In many cases their relationships may be interpreted unambiguously by agents. Typically thesauri do not provide an explicit hierarchy (although with narrower and broader term specifications, one could deduce a hierarchy) [56,26]. Taxonomies are used to organize ontological knowledge using generalization and specialization relationships through which simple and multiple inheritance could be applied [56]. Informal "$is-a''$ hierarchies, taken from specifications of term hierarchies. Such hierarchy is not a strict subclass or "$is-a''$ hierarchy [28]. Includes: Thesauri; Structured Glossaries; XML, DTDs; Informal Hierarchies. See specifications [18] [19].	*<<category>>*
`Textual Representation`	The intrinsic symbolic textual characteristics present in certain types of ontologies (artifacts) make them a textual artifact for representation or implementation.	*<<subkind>>*
`Triple Stores`	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as Triple Stores.	*<<subkind>>*
`User Classifications`	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as User Classifications. Classifications (user or hierarchical classifications) are very easy to be created and maintained by an ordinary user. They represent a very natural way for (natural language) markup of the data classified in them. Moreover, classifications are used pervasively on the web, thus creating the necessary "critical mass" of annotated data. However, because they are described in natural language, classifications cannot be easily embedded in the infrastructure of the Semantic Web [78,26].	*<<subkind>>*
`Version Identification`	A sequence of characters (numbers, signs, and letters) is used to identify ontology versions, usually, stakeholders define a policy to generate this sequence of characters.	*<<nominal quality>>*
Unintended Behavior	The role unintended of behavioral characteristics present in certain types of ontologies makes them susceptible to having semantic anti-patterns.	*<<role>>*
`Web Directories`	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as Web Directories. A web directory, also known as a link directory or a subject directory, provides a layer of categories and sub-categories over the World Wide Web. Each web directory creates its own categories according to primary user groups needs, helping them find websites. Web Directories are classifications on the web [78]. In these hierarchies it is typically the case that an instance of a more specific class is also an instance of the more general class but that is not enforced 100% of the time [26].	*<<subkind>>*
`White Literature`	White literature is the commercially published literature that is as a result of its rationale as print for profit is not gray literature.	*<<kind>>*
Well-defined Ontology	An ontology schema when it has a domain description that provides it a representation as an ontological referential. Well-defined Ontology represent ontologies that participate in a relation of implementation through some ontological support; i.e. there is an `Ontology Schema` (implemented model) that has a correspondent `Domain Description` (a reference model) that supports it. Indeed, SABiO strongly advocate that Reference Ontologies are the base for Operational ones, already considering that the lasts are versions of the firsts [19].	*<<role>>*

– continued on next page

[18] https://www.w3.org/TR/xml/
[19] https://www.w3.org/XML/1998/06/xmlspec-report-19980910.htm

Table 5 continued from previous page		
Concept	**Definition**	**Stereotype**
Well-grounded Ontology	The role of a Non-Foundational Ontology plays when receiving Ontological Foundation from Foundational Ontologies. This notion of well-grounded ontologies (or well-founded ontologies) has benefits in representing some reality phenomena has the support of vast literature, these are some of those works [32,21,36,42,19].	<<*role*>>
XML DTDs	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as a Document Type Declaration (DTD) [13]. See specification at [20]	
XML Schemas	The intrinsic characteristics present in certain types of ontologies (artifacts) makes them recognized as XML Schemas. Types of ontologies represented through XML Schemas [56,26]. See specifications at [21] [22].	<<*subkind*>>

3.6 Relations Definition

Table 6 shows the definitions for O4OA part/whole relations.

Table 6: O4OA part/whole (meronymic) relations definition.

Relation	Related Concepts	Stereotype
contains	Ontology Version — Concept Definition	<<*componentOf*>>
subOntology	Composite Ontology — Ontology (Artifact)	<<*componentOf*>>
formedBy	Ontology Network — Ontology (Artifact)	<<*memberOf*>>
formedBy	Cloud of Concepts — Concept	<<*memberOf*>>

Table 7 shows the definitions for O4OA comparative relations.

Table 7: O4OA comparative relations (type-reflexive) definition.

Relation	Concept	Stereotype
	{*reflexive, asymmetric, transitive, total*}	
/ more appropriate than	Ontology Type (Artifact Type)	<<*comparative*>>
/ more axiomatized than	*Axiomatization Level*	<<*comparative*>>
/ more expressive than	*Expressiveness Level*	<<*comparative*>>
/ more formal than	*Formalization Level*	<<*comparative*>>
/ more isomorphic than	Language Type	<<*comparative*>>

Table 8 shows the definitions for O4OA material relations.

[20] https://www.w3.org/TR/xml/
[21] https://www.w3.org/TR/xmlschema11-1/
[22] https://www.w3.org/TR/xmlschema11-2/

Table 8: O4OA material relations (relational dependencies) definition.

Relation	Related Concepts	Stereotype
	Description	
`usedToRepresent`	*Conceptualization — Representation Language*	<<*material*>>
`usedToImplement`	*Implementation — Implementation Language*	<<*material*>>
`drives`	*Ontological Constraints — Ontology-Constrained Metamodel*	<<*material*>>
`drives'`	*Ontological Foundation Constraints — Ontology-Constrained Metamodel*	<<*material*>>
`specifes`	*Concrete (Syntax) Language — Abstract (Syntax) Language*	<<*material*>>
`specifies'`	*Abstract Ontology-Driven Language — Concrete Ontology-Driven Language*	<<*material*>>
`groundedOver`	*Ontological Foundation — Well-grounded Ontology*	<<*material*>>
`implementationFor`	*Well-defined Ontology — Ontological Reference*	<<*material*>>
`reuse`	*Reuser Ontology — Reused Ontology*	<<*material*>>
`describes`	*Concept — Source*	<<*material*>>
`carves`	*Symbolic Appropriateness — Delimitation*	<<*material*>>
`exteriorizes`	*Representation Anti-Pattern — Behavior Anti-Pattern*	<<*material*>>

Table 9 shows the definitions for O4OA mediation relations.

Table 9: O4OA mediation relations (relational dependencies) definition.

Related Concepts	Stereotype
Behavior Pattern — `Semantical Pattern`	<<*mediation*>>
Representation Pattern — `Semantical Pattern`	<<*mediation*>>
Behavior Anti-Pattern — `Semantical Anti-Pattern`	<<*mediation*>>
Representation Anti-Pattern — `Semantical Anti-Pattern`	<<*mediation*>>
Reuser Ontology — `Reusability`	<<*mediation*>>
Reused Ontology — `Reusability`	<<*mediation*>>
Implementation Language — `Ontology Schema`	<<*mediation*>>
Implementation — `Ontology Schema`	<<*mediation*>>
Conceptualization — `Domain Description`	<<*mediation*>>
Representation Language — `Domain Description`	<<*mediation*>>
Symbolic Appropriateness — `Appropriateness`	<<*mediation*>>
Delimitation — `Appropriateness`	<<*mediation*>>
Abstract (Syntax) Language — `Language Specification`	<<*mediation*>>
Concrete (Syntax) Language — `Language Specification`	<<*mediation*>>
Abstract Ontology-Driven Language — `Ontology-Driven Language Specification`	<<*mediation*>>
Concrete Ontology-Driven Language — `Ontology-Driven Language Specification`	<<*mediation*>>
Source — `Concept Definition`	<<*mediation*>>
Concept — `Concept Definition`	<<*mediation*>>
Ontology-Constrained Metamodel — `Ontology-Driven Metamodel`	<<*mediation*>>
Ontological Constraints — `Ontology-Driven Metamodel`	<<*mediation*>>
Well-defined Ontology — `Ontological Support`	<<*mediation*>>
Ontological Reference — `Ontological Support`	<<*mediation*>>
Ontological Foundation — `Ontological Grounding`	<<*mediation*>>
Well-grounded Ontology — `Ontological Grounding`	<<*mediation*>>
Ontological Foundation Constraints — `Foundational Ontology-Driven Metamodel`	<<*mediation*>>
Ontology-Driven Metamodel — `Foundational Ontology-Driven Metamodel`	<<*mediation*>>

Table 10 shows the definitions for O4OA external dependence relations.

Table 10: O4OA external dependence relations definition.

Related Concepts	Stereotype
Concrete (Syntax) Language — Graphy (Symbolic)	*<<externalDependence>>*
Concrete (Syntax) Language — Limitation	*<<externalDependence>>*
Abstract (Syntax) Language — Representation	*<<externalDependence>>*
Abstract (Syntax) Language — Behaviour	*<<externalDependence>>*
Concrete Ontology-Driven Language — Graphy (Symbolic)	*<<externalDependence>>*
Concrete Ontology-Driven Language — Limitation	*<<externalDependence>>*
Abstract Ontology-Driven Language — Representation	*<<externalDependence>>*
Abstract Ontology-Driven Language — Behaviour	*<<externalDependence>>*

Table 11 shows the definitions for O4OA characterization relations.

Table 11: O4OA characterization relations definition.

Related Concepts	Stereotype
Version Identification — Ontology Version	*<<characterization>>*
Number of Axioms — Ontology Version	*<<characterization>>*
Copyright — *Ontology Version*	*<<characterization>>*
Limitation — *Abstract Ontology-Driven Language*	*<<characterization>>*
Limitation — *Abstract (Syntax) Language*	*<<characterization>>*
Behaviour — *Concrete (Syntax) Language*	*<<characterization>>*
Behaviour — *Concrete Ontology-Driven Language*	*<<characterization>>*
Graphy (Symbolic) — *Abstract (Syntax) Language*	*<<characterization>>*
Graphy (Symbolic) — *Abstract Ontology-Driven Language*	*<<characterization>>*
Representation — *Concrete Ontology-Driven Language*	*<<characterization>>*
Representation — *Concrete (Syntax) Language*	*<<characterization>>*
Formalization Level — Ontology (Artifact)	*<<characterization>>*
Expressiveness Level — Ontology (Artifact)	*<<characterization>>*
Axiomatization Level — Ontology (Artifact)	*<<characterization>>*
Appropriateness — *Implementation Language*	*<<characterization>>*
Appropriateness — *Representation Language*	*<<characterization>>*
Appropriateness — *Language Type*	*<<(meta)characterization>>*
Appropriateness — Ontology Type (Artifact Type)	*<<(meta)characterization>>*
Semantical Pattern — Ontology Type (Artifact Type)	*<<(meta)characterization>>*
Semantical Anti-Pattern — Ontology Type (Artifact Type)	*<<(meta)characterization>>*

Table 12 shows the definitions for O4OA structuration relations.

Table 12: O4OA structuration relations definition.

Related Concepts	Stereotype
Degree of Formalization — *Formalization Level*	*<<structuration>>*
Degree of Expressiveness — *Expressiveness Level*	*<<structuration>>*
Degree of Axiomatization — *Axiomatization Level*	*<<structuration>>*

Table 13 shows the definitions for O4OA instantiation relations.

Table 13: O4OA instantiation relations definition.

Related Concepts	Stereotype
Language — Language Type	*<<instantiation>>*
Ontology (Artifact) — Ontology Type (Artifact Type)	*<<instantiation>>*

Table 14 shows the definitions for O4OA commitment relations.

Table 14: O4OA commitment relations definition.

Related Concepts	Stereotype
Language Specification — Behavior	*<<commitment>>*
Language Specification — Graphy (Symbolic)	*<<commitment>>*
Language Specification — Limitation	*<<commitment>>*
Language Specification — Representation	*<<commitment>>*
Ontology-Driven Language Specification — Behavior	*<<commitment>>*
Ontology-Driven Language Specification — Graphy (Symbolic)	*<<commitment>>*
Ontology-Driven Language Specification — Limitation	*<<commitment>>*
Ontology-Driven Language Specification — Representation	*<<commitment>>*
Ontology Version — Appropriateness	*<<commitment>>*

Table 15 shows the definitions for O4OA specialization relations.

Table 15: O4OA specialization relations definition.

Parent	Generalization Set	Constraint/ Design Pattern
	Atomic Ontology, Composite Ontology	{*disjoint,complete*}
	Operational Ontology, Reference Ontology	{*disjoint,complete*}
Operational Ontology	*Implementation*	*Role Pattern*
Reference Ontology	*Conceptualization*	*Role Pattern*
Ontology (Artifact)	Non-Foundational Ontology, Foundational Ontology	{*disjoint,complete*}
	Reused Ontology, *Reuser Ontology*	{*disjoint,complete*}
	Ontological Constraint	*Role Pattern*
Non-Foundational Ontology	Application Ontology, Task Ontology, Domain Ontology	{*disjoint,complete*}
	Well-Grounded Ontology	*Role Pattern*
Domain Ontology	Core Domain Ontology	*Subkind Pattern*
Foundational Ontology	*Direct Ontological Foundation*	*Role Pattern*
	Ontological Foundation Constraints	*Role Pattern*
Foundational Ontology-Driven Metamodel	*Indirect Ontological Foundation*	
Ontological Foundation	*Direct Ontological Foundation*, *Indirect Ontological Foundation*	*Rolemixin Pattern*
Model	Metamodel	*Subkind Pattern*
Metamodel	*Abstract (Syntax) Language*	*Role Pattern*
	Ontology-Constrained Metamodel	*Role Pattern*
Ontology-Driven Metamodel	*Abstract Ontology-Driven Language*	*Role Pattern*

– continued on next page

Table 15 continued from previous page

Parent	**Generalization Set**	**Constraint/ Design Pattern**
Language	Computational Language, Non-computational Language	{*disjoint,complete*}
	Concrete (Syntax) Language	*Role Pattern*
Computational Language	Knowledge Representation Language, Markup Language, Scripting Language, Programing Language	{*disjoint*}
	Implementation Language	*Role Pattern*
Programing Language	Functional Language, Procedural Language, Object-Oriented Language	{*disjoint*}
Non-computational Language	Mathematical Language, Natural Language, Modeling Language	{*disjoint*}
	Representation Language	*Role Pattern*
Mathematical Language	Logical Language	*Subkind Pattern*
Natural Language	Technical Language	*Subkind Pattern*
Modeling Language	Ontology-Driven Modeling Language	*Subkind Pattern*
Ontology-Driven Modeling Language	*Concrete Ontology-Driven Language*	*Role Pattern*
Ontology Version	Ontology Schema, Domain Description	{*disjoint,complete*} *Category Pattern*
Ontology Schema	*Well-defined Ontology*	*Role Pattern*
Domain Description	*Ontological Reference*	*Role Pattern*
Term	Concept	*Role Pattern*
Documents	Policy, Standard, Gray Literature, White Literature, Encyclopedia	{*disjoint*} *Category Pattern*
	Source	*Rolemixin Pattern*
Limitation	*Delimitation*	*Role Pattern*
Graphy (Symbolic)	*Symbolic Appropriateness*	*Role Pattern*
	Terms, Ordinary Glossaries, User Classifications, Web Directories, Data Dictionaries, Tessauri, Structured Glossaries, Informal Taxonomies, Models, DB Schemas, XML Schemas, Formal Taxonomes, Frames, (Formal) Lightweight Ontologies	{*disjoint*} *Mode Pattern*
Models	Data Models	*Mode Subkind Pattern*
(Formal) Lightweight Ontologies	Knowledge Graphs	*Mode Subkind Pattern*
Knowledge Graphs	Triple Stores, NoSQL DB Schemas	{*disjoint*} *Mode Subkind Pattern*
Behaviour	*Behavior Pattern*, *Behavior Anti-Pattern*	{*disjoint,complete*} *Role Pattern*
Representation	*Representation Pattern*, *Representation Anti-Pattern*	{*disjoint,complete*} *Role Pattern*
	Graphical Representation, Textual Representation	{*complete*} *Mode Subkind Pattern*
	OntoUML, XML DTDs, Description Logic, Logic Programing, General Logic	{*disjoint*} *Mode Subkind Pattern*
Description Logic	ALC, SHOIN, SROIQ, SHIF, SHIQ	{*disjoint*} *Mode Subkind Pattern*
General Logic	First-order Logic, Modal Logic, Higher-order Logic	{*disjoint*} *Mode Subkind Pattern*
Logic Programing	Prolog, Lisp , F-Logic	{*disjoint*} *Mode Subkind Pattern*
Axiomatization Level	*Heavyweight*, *Lightweight*	{*disjoint*} *Category Pattern*
Heavyweight	Logic Programing, General Logic, Description Logic, OntoUML	{*disjoint*} *Category Pattern*
Lightweight	*Glossary and Data Dictionaries*, *Thesauri and Taxonomies*, *Metadata and Data Models*, Frames, (Formal) Lightweight Ontologies	{*disjoint*} *Category Pattern*

– continued on next page

Table 15 continued from previous page

Parent	**Generalization Set**	**Constraint/ Design Pattern**
Glossary and Data Dictionaries	Terms, Ordinary Glossaries, User Classifications, Web Directories, Data Dictionaries	{*disjoint*} *Category Pattern*
Thesauri and Taxonomies	Tesauri, Structured Glossaries, Informal Taxonomies, XML DTDs	{*disjoint*} *Category Pattern*
Metadata and Data Models	DB Schemas, Formal Taxonomes, XML Schemas, Data Models	{*disjoint*} *Category Pattern*
Formalization Level	*Formal Models (Ontology)*, *Informal (Lightweight) Ontology*	{*disjoint*} *Category Pattern*
Formal Models (Ontology)	Formal Taxonomes, XML Schemas, Data Models, *Formal Ontologies*	{*disjoint*} *Category Pattern*
Formal Ontologies	Frames, (Formal) Lightweight Ontologies, Logic Programing, Description Logic, General Logic, OntoUML	{*disjoint*} *Category Pattern*
Informal (Lightweight) Ontology	DB Schemas, *Glossary and Data Dictionaries*, *Thesauri and Taxonomies*	{*disjoint*} *Category Pattern*
Expressiveness Level	*Glossary and Data Dictionaries*, *Metadata and Data Models*, *Formal Ontologies*, *Thesauri and Taxonomies*	{*disjoint*} *Category Pattern*

3.7 Packages and Diagrams

O4OA is organized in packages as below:

- O4OA
  - Domain
    - Cloud
    - Definition
    - Document
  - Ontological
    - Characterization
      - Appropriateness
      - Anti-Pattern
      - Pattern
      - Graphy
      - Representation
    - Classification
      - Application Level
      - Granularity Level
      - Axiomatization Level
      - Formalization Level
    - Implementation
    - Language
    - Ontology
    - Reuse

Figures 3 to 29 present the diagrams of each O4OA package. Note that some of those diagrams present small dimensions due to the space limitations of this document; however, for the interested readers we provide better view in the ontology repository [23];

[23] https://bfmartins.gitlab.io/o4oa/diagrams/index.html

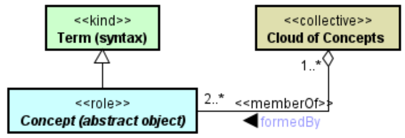

Fig. 3: O4OA `Cloud of Concepts` package content.

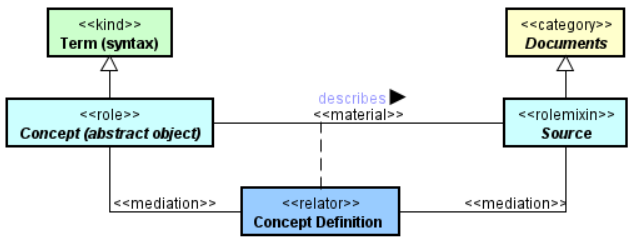

Fig. 4: O4OA `Definition` package content.

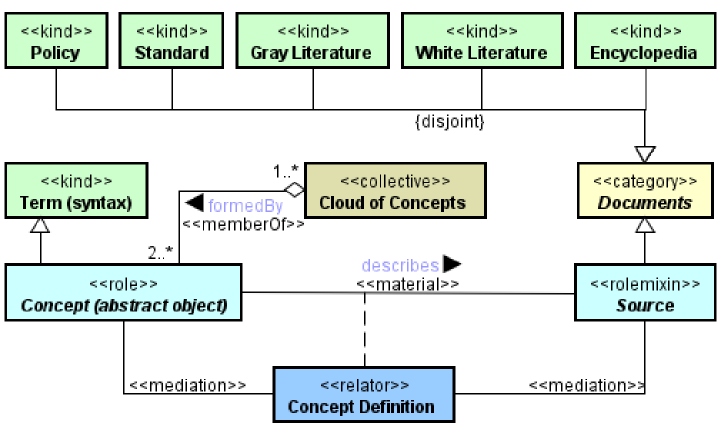

Fig. 5: O4OA `Domain` package content.

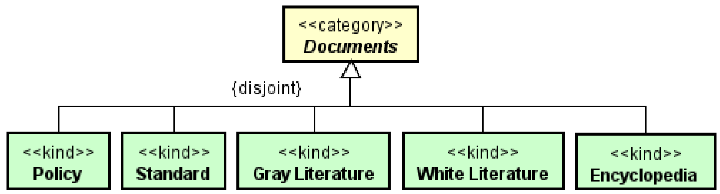

Fig. 6: O4OA `Document` package content.

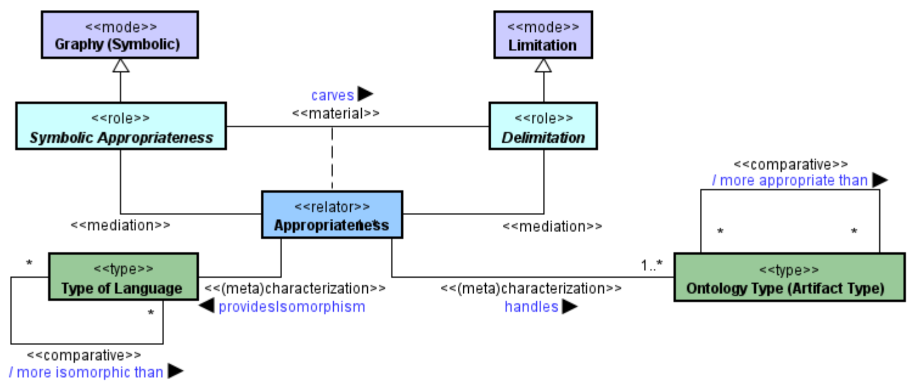

Fig. 7: O4OA `Appropriateness` package content.

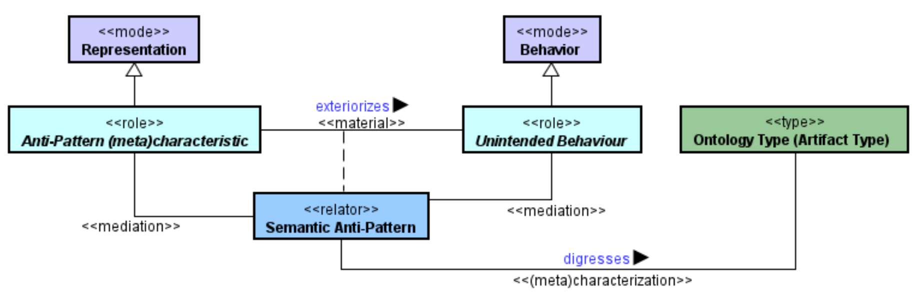

Fig. 8: O4OA `Anti-Pattern` package content.

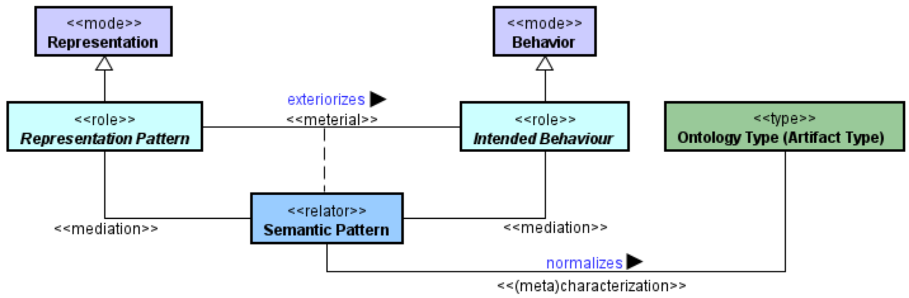

Fig. 9: O4OA `Design Pattern` package content.

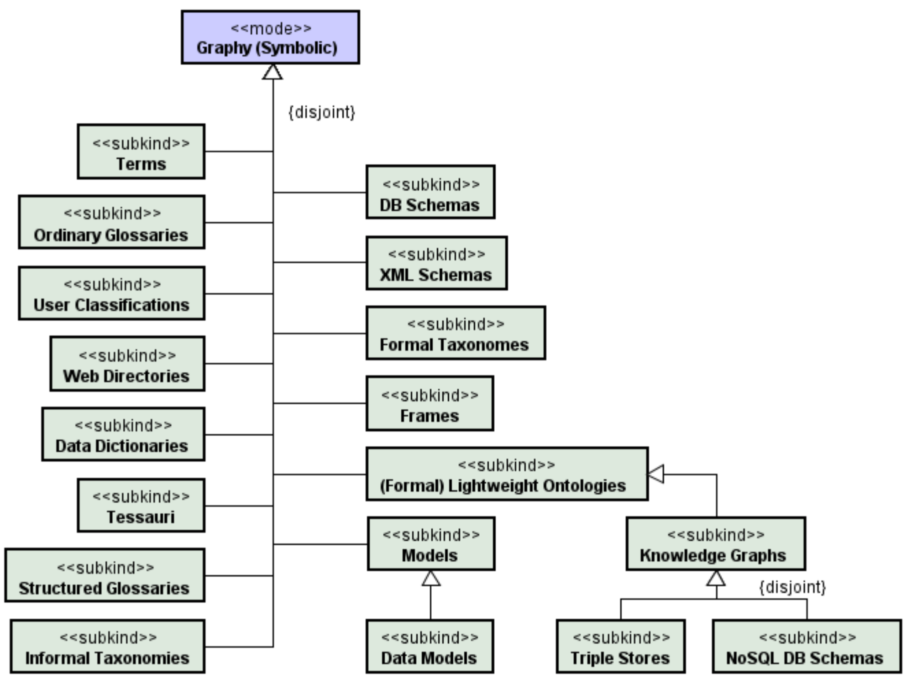

Fig. 10: O4OA `Graphy` package content.

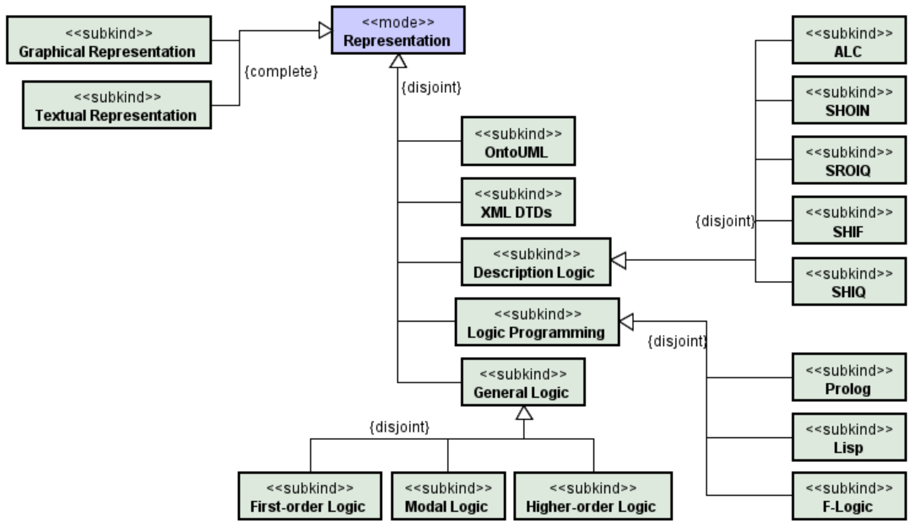

Fig. 11: O4OA `Representation` package content.

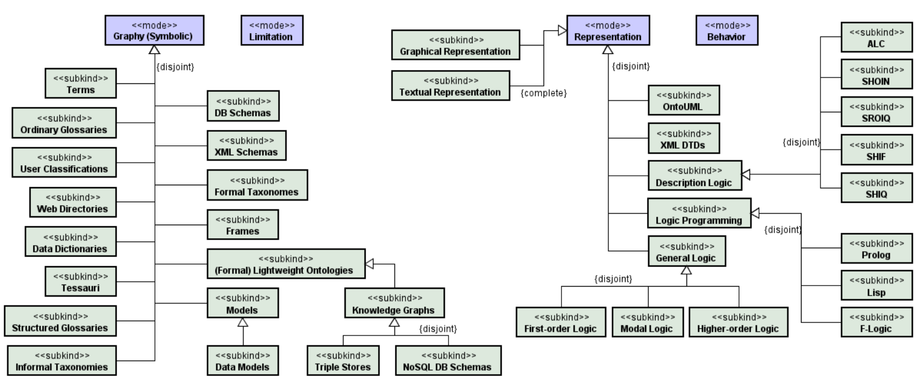

Fig. 12: O4OA `Characterization` (Characteristics) package content.

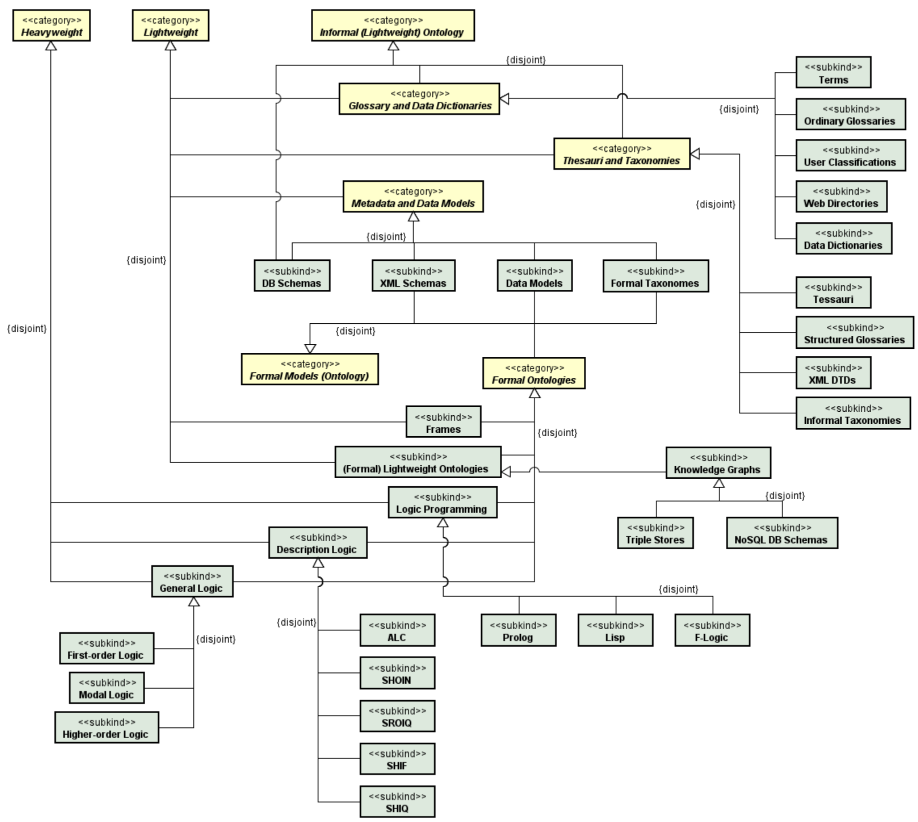

Fig. 13: O4OA `Characterization` package content.

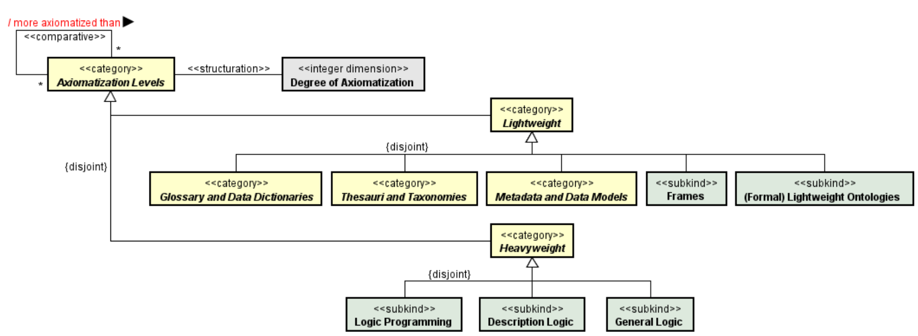

Fig. 14: O4OA `Axiomatization Level` package content.

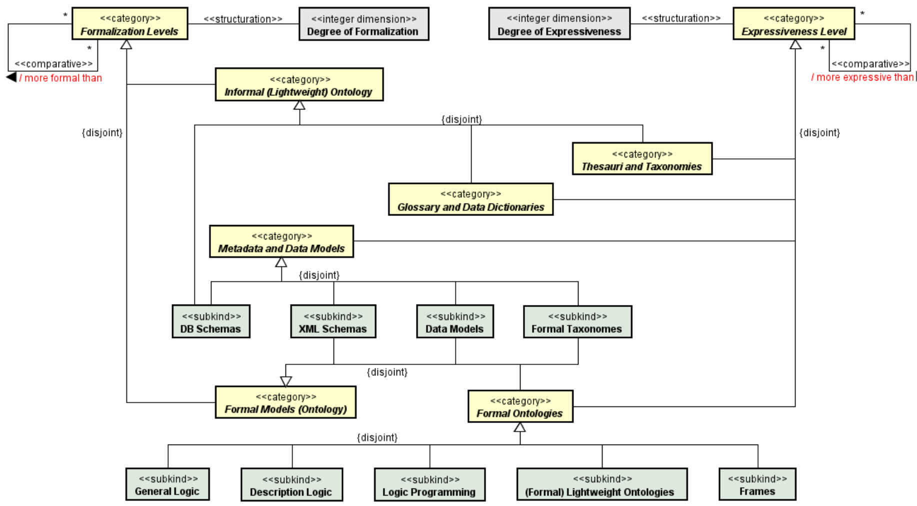

Fig. 15: O4OA `Formalization Level` package content.

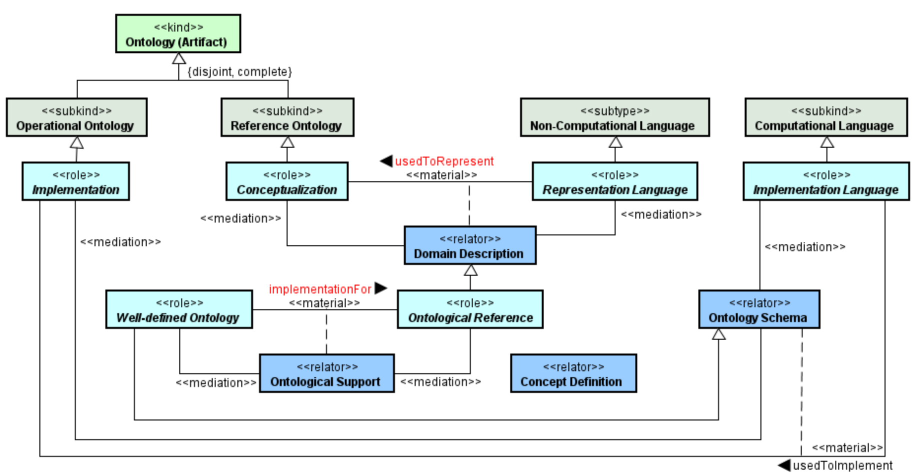

Fig. 16: O4OA `Application Level` package content.

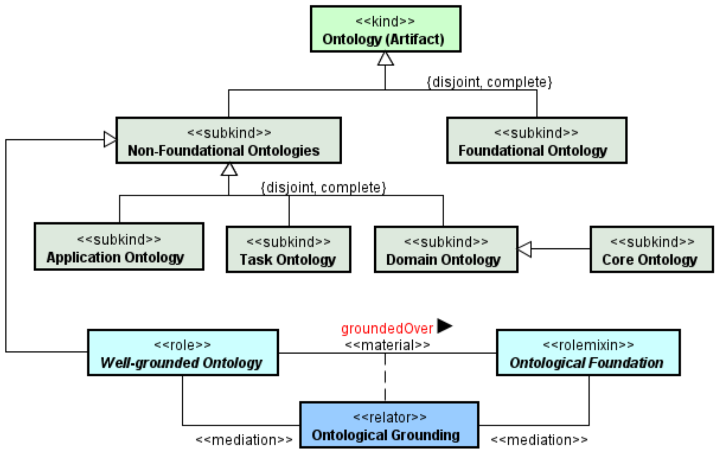

Fig. 17: O4OA `Granularity Level` package content.

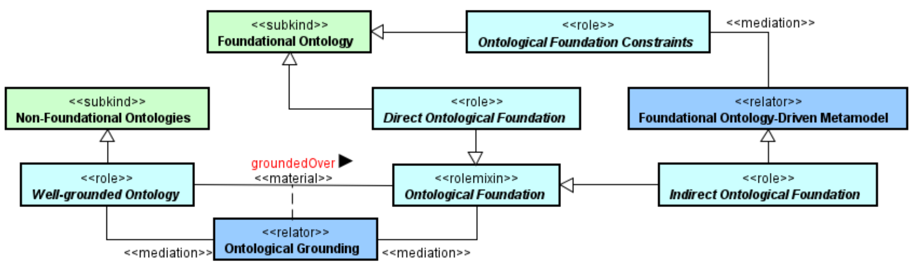

Fig. 18: O4OA `Granularity Level` (Ontological Grounding) package content.

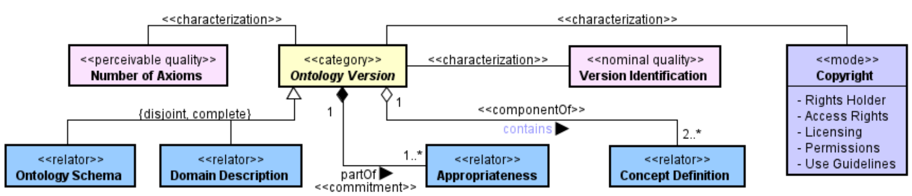

Fig. 19: O4OA `Implementation` package content.

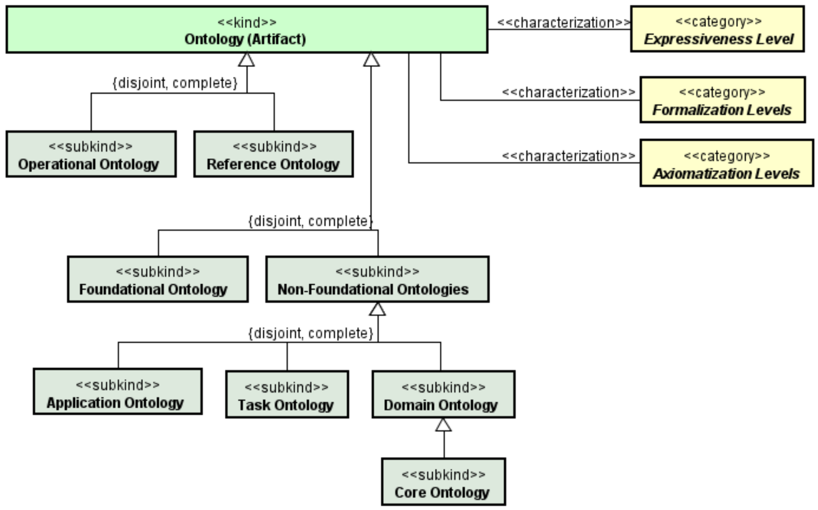

Fig. 20: O4OA `Classification` package content.

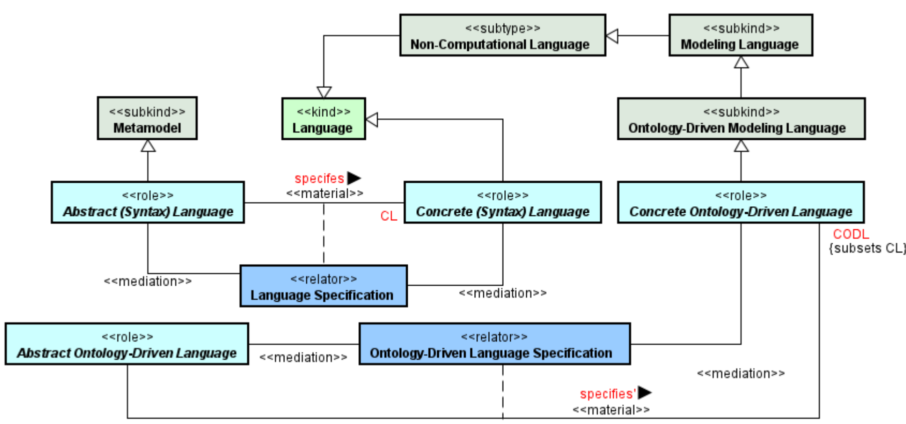

Fig. 21: O4OA `Language` (Ontology-Driven Metamodel) package content.

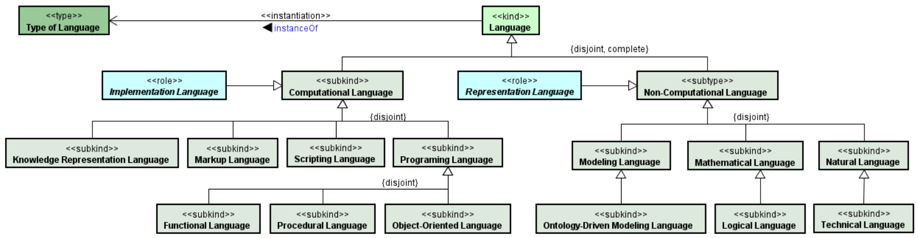

Fig. 22: O4OA Language package content.

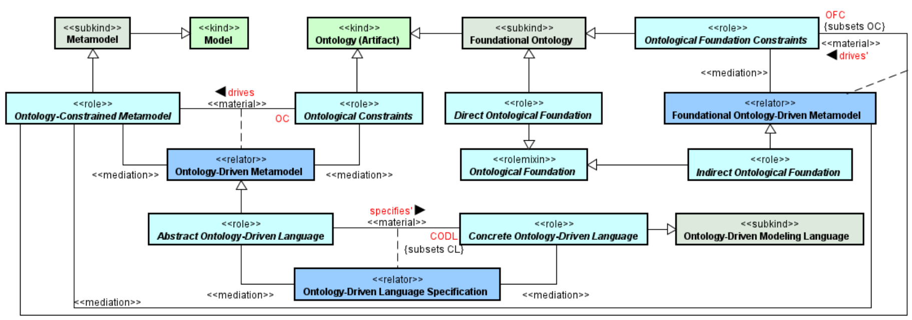

Fig. 23: O4OA Language (Foundational Ontology-Driven Metamodel) package content.

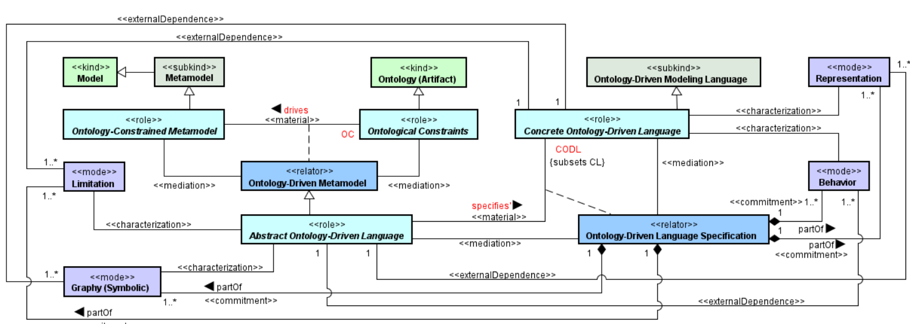

Fig. 24: O4OA Language (Ontology-Driven Specification) package content.

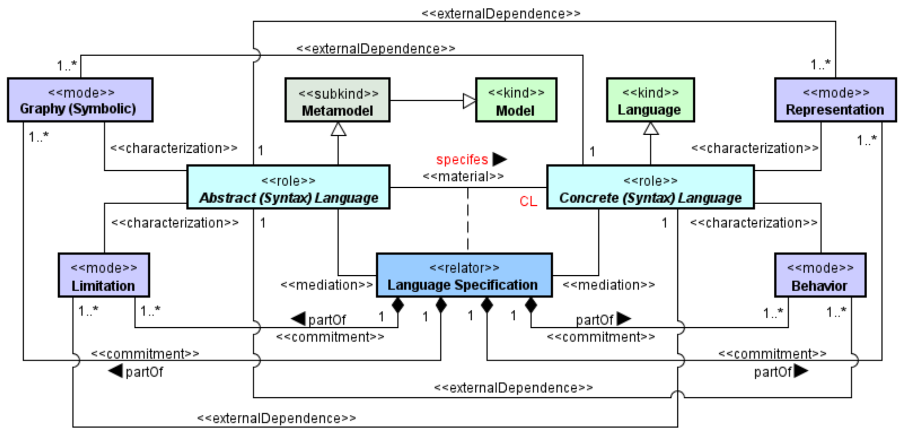

Fig. 25: O4OA `Language` (Specification) package content.

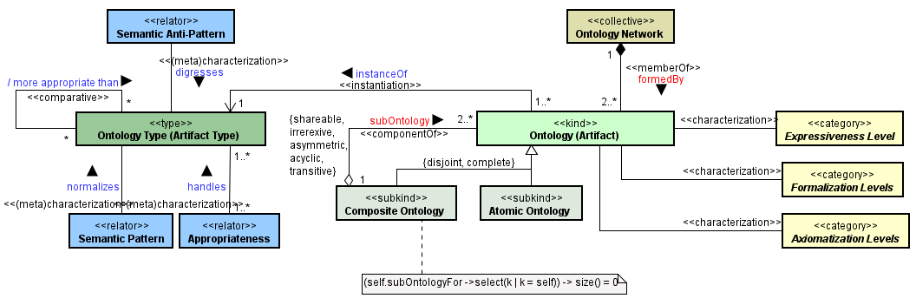

Fig. 26: O4OA `Ontology` package content.

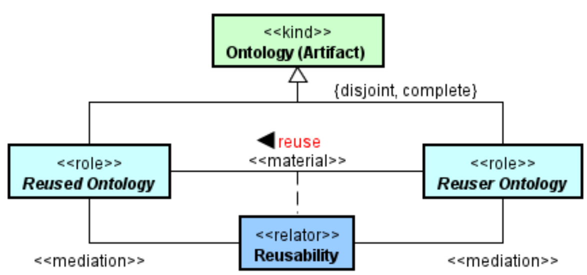

Fig. 27: O4OA `Reuse` package content.

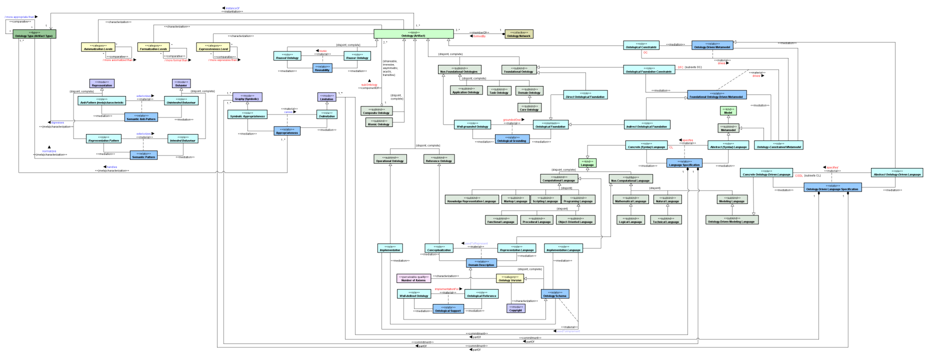

Fig. 28: O4OA `Ontological` package content.

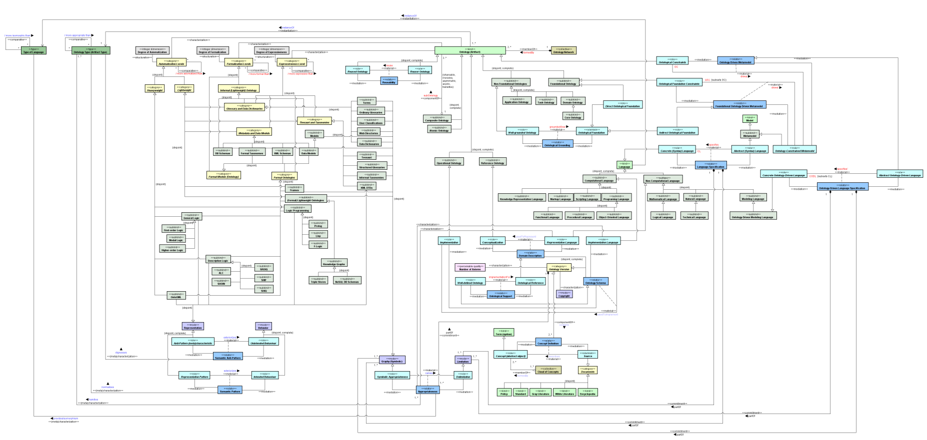

Fig. 29: O4OA root package content.

3.8 Verification

Table 16 shows the results of O4OA coverage regarding the proposed CQs.

Table 16: Results of the O4OA verification.

Ref.	**Concepts and Relations**
CQ1	*How to conceptually characterize an ontology (as an artifact)?*
CQ1.1	The `Application Level` compounds the intrinsic aspects (<<*mode*>>) that characterize Ontologies Types and make explicit the two disjoint ontology <<*subkind*>>, `Reference Ontology`, and `Operational Ontology`. These modes externally depends of ontology roles when ontologies be a `Conceptualization` representing a `Domain Description` or be an `Implementation` of an `Ontology Schema` for a domain.
CQ1.2	The `Generality Level` compounds the intrinsic aspects (<<*mode*>>) that characterize `Ontologies Type` and make explicit these disjoint ontology <<*subkind*>>: `Foundational Ontology`, `Domain Ontology` (including `Core Ontology`), `Task Ontology` and `Application Ontology`.
CQ1.3	The `Formalization Level` of an ontology is a categorization of the possible aspects (<<*mode*>>) that characterize `Ontologies Type`. The `Expressiveness Level` is another category of aspects that characterize types of ontologies meanwhile interfere in the `Formalization Level`.
CQ1.4	The `Axiomatization Level` of an ontology is a categorization of the possible aspects (<<*mode*>>) that characterize `Ontologies Type`. The `Expressiveness Level` is another category of aspects that characterize types of ontologies meanwhile interfere in the `Axiomatization Level`.
CQ1.5	`Well-grounded Ontology` represent ontologies that participate in a relation of grounding through some ontological grounding; i.e. they are grounded over a `Foundational Ontology`.
CQ1.6	`Well-defined Ontology` represents ontologies that participate in a relation of implementation through some ontological support; i.e. there is an `Ontology Schema` (implemented model) that has a correspondent `Domain Description` (reference model) that supports it.
CQ1.7	The intrinsic and relational aspects (characteristics) of ontology types (`Ontology Type`) interfere in ontology characterization, usually these are ontology (meta)characteristics.
CQ1.8	`Ontology Schema` (implemented models) and `Domain Description` (reference models) are mediated by languages; in this case, the roles of languages, `Implementation Language` and `Representation Language` respectively. Therefore, language (meta)characteristics interfere with conceptualizations represented or implemented.
CQ2	*How to conceptually characterize the domain cloud of concepts of an ontology?*
CQ2.1	`Concept Definition` is the <<*relator*>> and its roles (`Concept` and `Source`) using the *Relator Pattern* explain how sources describe concepts in conceptualizations, denoting the relation `describes`.
CQ2.2	`Document` represent the category of consolidated (well-known) bibliographic material (such as standards, policies, and etc.) that may be used as source of information to support the conceptualization.
CQ2.3	The `Cloud of Concepts` collection defines the set of concepts on which it is possible to infer the possible definitions that support one or more ontologies. The relation formed by express which concepts are members of (<<*memberOf*>>) a cloud of concepts collection.
CQ2.4	The <<*relator*>> `Concept Definition` provides all possibilities of sources for conceptualizations since an `Ontology Version` represent conceptualization that contains (<<*componentOf*>>) definitions. Thus the same source can support multiple conceptualizations.
CQ2.5	The <<*relator*>> `Concept Definition` provides all possibilities of concepts (terms) for conceptualizations since an Ontology Version represent conceptualization that contains (<<*componentOf*>>) definitions. Thus the same term can appear multiple conceptualizations.
CQ2.6	A `Concept Definition` may (or not) be a component of an `Ontology Version`; therefore, multiple definitions of the same source may be associated with conceptualizations (represented as `Ontology Version`). These definitions may (or may not) be similar, depending on the commitment adopted in each conceptualization.
CQ2.7	A `Concept Definition` may (or not) be a component of an `Ontology Version`; therefore, multiple definitions of the same `Term` (as `Concept`) may be associated with conceptualizations (represented as `Ontology Versions`). These definitions may (or may not) refer synonymous, depending on the commitment adopted in each conceptualization.
CQ2.8	The part-hood relation contains is between a whole conceptualization (represented as `Ontology Version`) and its parts (`Concept`) delimit the cloud of concepts for this conceptualization.

– continued on next page

Table 16 continued from previous page	
Ref.	**Concepts and Relations**
CQ3	*How to conceptually characterize ontology networks (its ontologies as a whole)?*
CQ3.1	`Ontological Grounding` is the relator and its roles (`Well-grounded Ontology` and `Ontological Foundation`) using the *Relator Pattern* explain how ontologies are grounded, denoting the relation `groundedOver`.
CQ3.2	`Composite Ontology` and `Atomic Ontology` are subkinds of `Ontology` and using the *Weak Supplementation Pattern* describe the notion of Whole/Part of ontologies, denoting the relation `componentOf`.
CQ3.3	`Reusability` is the relator and its roles (`Reused Ontology` and `Reuser Ontology`) using the *Relator Pattern* explain ontologies reuse, denoting the relation `reuse`.
CQ3.4	`Ontology-driven Modeling Language` subtypes of languages which have `Ontology-driven Language Specification`. The relator `Ontology-Driven Metamodel` and its roles (`Abstract Ontology-driven Language` and `Ontological Constraints`) using the *Relator Pattern* foundation explains how ontologies provide constraints that drive languages, denoting the relation `drives`.
CQ3.5	`Ontology Schema` is the relator and its roles (`Implementation Language` and `Implementation`) using the *Relator Pattern* explains how ontologies are implemented, denoting the relation `implementationFor`.

4 Final Considerations

This document presented the O4OA specification regarding its reference ontology version 6.2. However, this document is in its first publication version. Indeed, this specification document refers to a higher O4OA version because it depicts a consensual agreement after many discussions among the project participant stakeholders. Team participation has included:

Ontological Perspective: **O4OA Author:** Beatriz Franco Martins from Universitat Politècnica de València, Spain;

Ontology Domain Specialists: this O4OA specification document authors;

UFO/OntoUML Specialists: the O4OA author and specialist collaborators from other institutions, with special thanks to Giancarlo Guizzardi and Renata Guizzardi from University of Twente, Nederland, Bruno Borlini Duarte from Petróleo Brasileiro SA (Petrobras), Vitória (ES), Brasil, and Christine Griffo from Eurac Research, Free University of Bozen-Bolzano, Itália for fruitful discussions;

Project Management: José Fabián Reyes Román from Valencian Research Institute for Artificial Intelligence (VRAIN), Universitat Politècnica de València, and Ana Cidad Vila from Valencian Graduate School and Research Network of Artificial Intelligence (ValgrAI), Spain;

Project Advisor: Professor Oscar Pastor from PROS Research Center, Universitat Politècnica de València, Spain.

Domain Perspective: Cybersecurity Domain Specialists: Cyber R&D Lab, Israel Team from Accenture LTD; nominally, Ethan Hadar, Gal Engelberg, Moshe Hadad, and Dan Klein;

Cybersecurity Data Research: Undergraduate students of the course "Seguridad Informática y Redes de Datos", and their preceptor Lenin Javier Serrano Gilfrom Ingeniería de Sistemas e Informática, Universidad Pontificia Bolivariana, Colombia;

Literature Review Specialists: José Ignacio Panach from Escola Tècnica Superior d'Enginyeria, Universitat de València, Spain.

Acknowledgments. This work has been developed under the project Digital Knowledge Graph – Adaptable Analytics API with the financial support of Accenture LTD, the Generalitat Valenciana through the CoMoDiD project (CIPROM/2021/023), the Spanish State Research Agency through the DELFOS (PDC2021-121243-I00) and SREC (PID2021-123824OB-I00) projects, MICIN/AEI/10.13039/501 100011033 and co-financed with ERDF and the European Union Next Generation EU/PRTR.

References

1. T.f. hoad. In: The Concise Oxford Dictionary of English Etymology. Oxford University Press, New York (1993)
2. Angele, J., Kifer, M., Lausen, G.: Ontologies in f-logic. In: Handbook on ontologies, pp. 45–70. Springer (2009)
3. Baader, F.: The description logic handbook: Theory, implementation and applications. Cambridge university press (2003)
4. Baader, F., Horrocks, I., Sattler, U.: Description logics. Foundations of Artificial Intelligence **3**, 135–179 (2008)
5. Baar, T.: Correctly defined concrete syntax for visual modeling languages. In: International Conference on Model Driven Engineering Languages and Systems. pp. 111–125. Springer (2006)
6. Bailey, K.D.: Typologies and taxonomies: An introduction to classification techniques, vol. 102. Sage (1994)
7. Belaunde, M., Casanave, C., DSouza, D., Duddy, K., El Kaim, W., Kennedy, A., Frank, W., Frankel, D., Hauch, R., Hendryx, S., et al.: Mda guide version 1.0. 1 (2003)
8. Benevides, A.B., Guizzardi, G.: A model-based tool for conceptual modeling and domain ontology engineering in OntoUML. Enterprise Information Systems pp. 528–538 (2009)
9. Benureau, F.C., Rougier, N.P.: Re-run, repeat, reproduce, reuse, replicate: transforming code into scientific contributions. Frontiers in neuroinformatics **11**, 69 (2018)
10. Bobrow, D.G., Winograd, T.: An overview of krl, a knowledge representation language. Cognitive science **1**(1), 3–46 (1977)
11. Boeckhout, M., Zielhuis, G.A., Bredenoord, A.L.: The fair guiding principles for data stewardship: fair enough? European journal of human genetics **26**(7), 931–936 (2018)
12. Borst, W.N.: Construction of engineering ontologies for knowledge sharing and reuse. (1999)
13. Bray, T., Paoli, J., Sperberg-McQueen, C.M., Maler, E., Yergeau, F.: Extensible markup language (xml) 1.0 (1998)
14. Bruce, K.B.: Foundations of object-oriented languages: types and semantics. MIT press (2002)

15. de Carvalho, V.A., Almeida, J.P.A., Guizzardi, G.: Using reference domain ontologies to define the real-world semantics of domain-specific languages. In: Advanced Information Systems Engineering: 26th International Conference, CAiSE 2014, Thessaloniki, Greece, June 16-20, 2014. Proceedings 26. pp. 488–502. Springer (2014)
16. Dantsin, E., Eiter, T., Gottlob, G., Voronkov, A.: Complexity and expressive power of logic programming. ACM Computing Surveys (CSUR) **33**(3), 374–425 (2001)
17. Degen, W., Heller, B., Herre, H., Smith, B.: Gol: toward an axiomatized upper-level ontology. In: Proceedings of the international conference on Formal Ontology in Information Systems-Volume 2001. pp. 34–46 (2001)
18. Dijkman, R.M.: Consistency in multi-viewpoint architectural design. Citeseer (2006)
19. Falbo, R.D.A.: SABiO : Systematic Approach for Building Ontologies An Overview of SABiO (2011)
20. Fensel, D.: The role of ontologies in information interchange. In: Proceedings of the 2nd International Scientific and Practical Conference on Programming UkrPROG. Citeseer (2000)
21. Fielding, J.M., Simon, J., Ceusters, W., Smith, B.: Ontological theory for ontological engineering: Biomedical systems information integration. In: KR. pp. 114–120 (2004)
22. Fondement, F., Baar, T.: Making metamodels aware of concrete syntax. In: European Conference on Model Driven Architecture-Foundations and Applications. pp. 190–204. Springer (2005)
23. Fonseca, C.M., Porello, D., Guizzardi, G., Almeida, J.P.A., Guarino, N.: Relations in ontology-driven conceptual modeling. In: Conceptual Modeling: 38th International Conference, ER 2019, Salvador, Brazil, November 4–7, 2019, Proceedings 38. pp. 28–42. Springer (2019)
24. Gabbrielli, M., Martini, S.: Programming languages: principles and paradigms. Springer Science & Business Media (2010)
25. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design patterns: elements of reusable object-oriented software. Pearson Deutschland GmbH (1995)
26. Giunchiglia, F., Zaihrayeu, I.: Lightweight ontologies. Tech. rep., University of Trento (2007)
27. Gómez-Pérez, A., Corcho, O.: Ontology languages for the semantic web. IEEE Intelligent systems **17**(1), 54–60 (2002)
28. Gomez-Perez, A., Fernández-López, M., Corcho, O.: (01 2004)
29. Graham, P.: ANSI Common Lisp. Prentice Hall Press, USA (1995)
30. Gruber, T.R., et al.: A translation approach to portable ontology specifications. Knowledge acquisition **5**(2), 199–220 (1993)
31. Gruber, T.: What is an ontology (1993)
32. Guarino, N.: Formal Ontology in Information Systems. In: Proceedings of the 1st International Conference. pp. 6–8. IOS Press, Trento, Italy (June 1998)
33. Guarino, N.: The ontological level. Philosophy and the Cognitive Sciences (1994)
34. Guarino, N.: The ontological level: Revisiting 30 years of knowledge representation. Conceptual modeling: Foundations and applications pp. 52–67 (2009)
35. Guarino, N., Carrara, M., Giaretta, P.: Formalizing ontological commitment. In: AAAI. vol. 94, pp. 560–567 (1994)
36. Guarino, N., Guizzardi, G.: In the defense of ontological foundations for conceptual modeling. Scandinavian Journal of Information Systems **18**(1), 1 (2006)
37. Guarino, N., Guizzardi, G.: "we need to discuss the relationship": Revisiting relationships as modeling constructs. In: International Conference on Advanced Information Systems Engineering. pp. 279–294. Springer (2015)

38. Guarino, N., Guizzardi, G.: Relationships and events: towards a general theory of reification and truthmaking. In: Conference of the Italian Association for Artificial Intelligence. pp. 237–249. Springer (2016)
39. Guarino, N., Oberle, D., Staab, S.: What is an ontology? Handbook on ontologies pp. 1–17 (2009)
40. Guizzardi, G.: Ontological Foundations for Structural Conceptual Models. CTIT, Centre for Telematics and Information Technology (2005)
41. Guizzardi, G.: The role of foundational ontologies for conceptual modeling and domain ontology representation. In: 2006 7th International Baltic conference on databases and information systems. pp. 17–25. IEEE (2006)
42. Guizzardi, G.: On ontology, ontologies, conceptualizations, modeling languages, and (meta) models. Frontiers in artificial intelligence and applications **155**, 18 (2007)
43. Guizzardi, G., de Almeida Falbo, R., Guizzardi, R.S.: Grounding Software Domain Ontologies in the Unified Foundational Ontology (UFO): The case of the ODE Software Process Ontology. In: CIbSE. pp. 127–140 (2008)
44. Guizzardi, G., Ferreira Pires, L., Van Sinderen, M.: An ontology-based approach for evaluating the domain appropriateness and comprehensibility appropriateness of modeling languages. In: International Conference on Model Driven Engineering Languages and Systems. pp. 691–705. Springer (2005)
45. Guizzardi, G., das Graças, A.P., Guizzardi, R.S.: Design patterns and inductive modeling rules to support the construction of ontologically well-founded conceptual models in ontouml. In: Advanced Information Systems Engineering Workshops: CAiSE 2011 International Workshops, London, UK, June 20-24, 2011. Proceedings 23. pp. 402–413. Springer (2011)
46. Guizzardi, G., Sales, T.P.: Detection, simulation and elimination of semantic anti-patterns in ontology-driven conceptual models. In: Conceptual Modeling: 33rd International Conference, ER 2014, Atlanta, GA, USA, October 27-29, 2014. Proceedings 33. pp. 363–376. Springer (2014)
47. Iverson, K.E.: A programming language. In: Proceedings of the May 1-3, 1962, spring joint computer conference. pp. 345–351 (1962)
48. Jacobsen, A., de Miranda Azevedo, R., Juty, N.S., Batista, D., Coles, S.J., Cornet, R., Courtot, M., Crosas, M., Dumontier, M., Evelo, C.T.A., Goble, C.A., Guizzardi, G., Hansen, K.K., Hasnain, A., Hettne, K.M., Heringa, J., Hooft, R.W.W., Imming, M., Jeffery, K.G., Kaliyaperumal, R., Kersloot, M.G., Kirkpatrick, C.R., Kuhn, T., Labastida, I., Magagna, B., McQuilton, P., Meyers, N., Montesanti, A., van Reisen, M., Rocca-Serra, P., Pergl, R., Sansone, S., da Silva Santos, L.O.B., Schneider, J., Strawn, G.O., Thompson, M., Waagmeester, A., Weigel, T., Wilkinson, M.D., Willighagen, E.L., Wittenburg, P., Roos, M., Mons, B., Schultes, E.: FAIR principles: Interpretations and implementation considerations. Data Intell **2**(1-2), 10–29 (2020). https://doi.org/10.1162/dint_r_00024
49. Jeusfeld, M.A.: Metamodel, pp. 1727–1730. Springer US, Boston, MA (2009)
50. Kifer, M., Lausen, G.: F-logic: a higher-order language for reasoning about objects, inheritance, and scheme. In: Proceedings of the 1989 ACM SIGMOD international conference on Management of data. pp. 134–146 (1989)
51. Kleppe, A.: A language description is more than a metamodel. In: Fourth international workshop on software language engineering. vol. 1, pp. 1–4. megaplanet. org (2007)
52. Koenig, A.: Patterns and Antipatterns, p. 383–389. Cambridge University Press, USA (1998)
53. Lamprecht, A.L., Garcia, L., Kuzak, M., Martinez, C., Arcila, R., Martin Del Pico, E., Dominguez Del Angel, V., Van De Sandt, S., Ison, J., Martinez, P.A., et al.: Towards fair principles for research software. Data Science **3**(1), 37–59 (2020)
54. Lankhorst, M.: Foundations, pp. 43–60. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)

55. Lassila, O.: Frames or objects, or both? In: Workshop Notes from the 8th National Conference on AI (AAAI-90): Object-Oriented Programming in AI, Boston (MA) (1990)
56. Lassila, O., McGuinness, D.: The role of frame-based representation on the semantic web. Linköping electronic articles in computer and information science **6**(5), 2001 (2001)
57. Lloyd, J.W.: Foundations of logic programming. Springer Science & Business Media (2012)
58. Martins, B.F., Serrano, L., Reyes, J.F., Panach, J.I., Pastor, O.: Towards the Consolidation of Cybersecurity Standardized Definitions. Tech. Rep. Version 2, Universidad Politecnica de Valencia (2021), http://hdl.handle.net/10251/163895
59. Martins, B.F., Serrano, L., Reyes, J.F., Panach, J.I., Pastor, O., Rochwerger, B.: Conceptual characterization of cybersecurity ontologies. In: 13th IFIP WG 8.1 working conference on the Practice of Enterprise Modelling. pp. 323–338. Springer (2020)
60. Martins, B.F., Reyes Román, J.F., Pastor, O., Hadad, M.: Improving conceptual domain characterization in ontology networks. In: Research Challenges in Information Science: 17th International Conference, RCIS 2023, Corfu, Greece, May 23–26, 2023, Proceedings. pp. 187–202. Springer International Publishing (2023)
61. Martins, B.F., Serrano Gil, L.J., Reyes Román, J.F., Panach, J.I., Pastor, O., Hadad, M., Rochwerger, B.: A framework for conceptual characterization of ontologies and its application in the cybersecurity domain. Software and Systems Modeling **21**(4), 1437–1464 (2022)
62. McDaniel, G.: IBM Dictionary of Computing. McGraw-Hill, Inc., USA (1994)
63. MOF, O.: Omg meta object facility (mof) core specification. version 2.5.1. october 2019 (2019)
64. Mylopoulos, J.: An overview of knowledge representation. ACM SIGART Bulletin (74), 5–12 (1980)
65. Ousterhout, J.K.: Scripting: Higher level programming for the 21st century. Computer **31**(3), 23–30 (1998)
66. Peacocke, C.: A study of concepts. The MIT Press (1992)
67. Riehle, D., Züllighoven, H.: Understanding and using patterns in software development. Tapos **2**(1), 3–13 (1996)
68. Saussure, F.M.: Course in general linguistics. McGraw-Hill Book Company, New York, Toronto, and London (1966)
69. Serrano, L., Martins, B.F., Serrano, J.F., Panach, J.I., Pastor, O.: Una encuesta acerca de la Definición de Conceptos de Ciberseguridad. Tech. rep., Universidad Politecnica de Valencia (2021), https://riunet.upv.es/handle/10251/174756
70. Silva, C., Belo, O.: A core ontology for brazilian higher education institutions. In: CSEDU (2). pp. 377–383 (2018)
71. Studer, R., Benjamins, V.R., Fensel, D.: Knowledge engineering: principles and methods. Data & knowledge engineering **25**(1-2), 161–197 (1998)
72. Suárez-Figueroa, M.C., Gómez-Pérez, A., Motta, E., Gangemi, A.: Introduction: Ontology engineering in a networked world. Springer (2012)
73. Truyen, F.: The fast guide to model driven architecture the basics of model driven architecture. Cephas Consulting Corp (2006)
74. Uschold, M., Gruninger, M.: Ontologies and semantics for seamless connectivity. ACM SIGMod Record **33**(4), 58–64 (2004)
75. Uschold, M., Gruninger, M., et al.: Ontologies: Principles, methods and applications. TECHNICAL REPORT-UNIVERSITY OF EDINBURGH ARTIFICIAL INTELLIGENCE APPLICATIONS INSTITUTE AIAI TR (1996)
76. Van Heijst, G., Schreiber, A.T., Wielinga, B.J.: Using explicit ontologies in kbs development. International journal of human-computer studies pp. 183–292 (1997)

77. Wilkinson, M.D., Dumontier, M., Aalbersberg, I.J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.W., da Silva Santos, L.B., Bourne, P.E., et al.: The fair guiding principles for scientific data management and stewardship. Scientific data **3**(1), 1–9 (2016)
78. Zaihrayeu, I., Sun, L., Giunchiglia, F., Pan, W., Ju, Q., Chi, M., Huang, X.: From web directories to ontologies: Natural language processing challenges. In: International Semantic Web Conference. pp. 623–636. Springer (2007)
79. Zalta, E.N.: Fregean senses, modes of presentation, and concepts. Philosophical Perspectives **15**, 335–359 (2001)