A Multi-level Ontology-based Approach for Descriptors of Catalogued Resources

Vânia Borges [1], Natália Queiroz de Oliveira [1] and Maria Luiza Machado Campos [1]

[1]*Programa de Pós-Graduação em Informática (PPGI) –Universidade Federal do Rio de Janeiro (UFRJ) CEP-21941-901 - Rio de Janeiro – RJ - Brazil*

Abstract

Technological advances and open data initiatives have increased data availability on the Web. These data, published in heterogeneous repositories, represent valuable sources for exploration and decision-making processes. The FAIR principles brought a special focus on the importance of metadata to effectively interoperate and reuse these data (and metadata). However, interoperating metadata associated with the resources catalogued in repositories is still a challenge, as each of them has its own complex architecture. Vocabularies such as DCAT aim to contribute as a step further on this, establishing a common set of elements for catalogue entries. This paper proposes a DCAT ontological analysis, applying multi-level conceptual modeling to treat ambiguities and enrich semantics. The proposed model aims to make the DCAT structures adherent to the reality aspects that need to be represented, establishing ontological commitments in an explicit way, which helps the understanding of those who will use it.

Keywords

Catalogued Resource Descriptors, Multi-Level Modeling, Repositories Interoperability, DCAT.

1. Introduction

Technological advances and open data initiatives have increased data availability from scientific research and government agencies on the Web. These data are organized into datasets and catalogued in institutional or thematic repositories, supported by a diversity of platforms. These platforms employ independent components, in large and growing complex architectures, with little and no easy interaction among themselves, generating data silos [1].

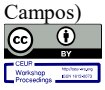
The FAIR principles publication in 2016 highlighted the importance of metadata to promote **F**indability, **A**ccessibility, **I**nteroperability, and **R**euse of data on the Web [2]. In this regard, it is worth emphasizing the evolution of these principles to promote machine-actionable and AI-ready (meta)data, which opens up unprecedented research opportunities and increases reproducibility [3]. These FAIR metadata allow the development of software agents able to act with large volumes of data and accelerate search engine and analysis mechanisms [4].

Aligning metadata elements used by different repositories and software agents is not a trivial task. For interoperability, many repositories adopt protocols such as the Open Archives Initiative Protocol for Metadata Harvesting[2] (OAI-PMH) for open exchange and collection of data and metadata, and yet alternative approaches via Application Programming Interfaces (API) for online access and exchange

Proceedings of the 15th Seminar on Ontology Research in Brazil (ONTOBRAS) and 6th Doctoral and Masters Consortium on Ontologies (WTDO), November 22-25, 2022.

EMAIL: vjborges30@ufrj.br (Vânia Borges); natalia.oliveira@ppgi.com.br (Natália Q. Oliveira); mluiza@ppgi.com.br (Maria Luiza M. Campos)

ORCID: 0000-0002-6717-1168 (Vânia Borges); 0000-0001-8371-142X (Natália Q. Oliveira); 0000-0002-7930-612X (Maria Luiza M. Campos)



[2] https://www.openarchives.org/pmh/

[5]. In general, these protocols and mechanisms for interoperability make use of vocabularies such as Dublin Core (DC)[3] to harvest standardized metadata about datasets.

To improve interoperability between data catalogues on the Web, the Data Catalog Vocabulary (DCAT) has established a lightweight ontology in OWL2[4] to describe catalogued datasets and data services, using the Resource Description Framework[5] (RDF) standard [6]. Aligned with other interoperability protocols, DCAT defines its entities properties (attributes and relations) with DC terms. In addition, following the FAIR principles, it adds PROV ontology[6] (PROV-O) properties to handle provenance.

The DCAT representation, however, can be improved by establishing a well-defined conceptualization of the domain to be treated, in this case, the *domain of the catalogued resource descriptors in Web catalogues*. To promote this conceptualization, domain ontologies built based on foundation ontologies play a special role. They support semantic interoperability at the conceptual level by establishing "contracts" that capture conceptualizations and representations in models or other tools employed to harmonize knowledge [1].

This paper aims to present a well-founded ontological analysis of DCAT, extending its semantics and treating ambiguities. Moreover, the main contribution of this work is to propose a DCAT-based core ontology grounded on a multi-level foundational ontology. This ontology contributes to a consistent conceptualization for resource descriptors on the Web, comprising structured and standardized metadata. It can provide a canonical model for specific implementations, to be adopted by repositories and by Web resource search engines and analysis mechanisms.

This paper is structured as follows: session 2 addresses relevant concepts regarding storage spaces for catalogued resources on the Web and introduces the DCAT vocabulary; session 3 discusses multi-level conceptual modeling and presents the Multi-Level Theory (MLT); session 4 describes the ontological analysis performed and the resulting model; session 5 presents an application of the model, based on a specific repository platform, and session 6 concludes this paper discussing the obtained results and future work.

2. Background: storage spaces and DCAT vocabulary

In the context of open data management infrastructure on the Web, there is a lack of consensus on the meaning and scope of some terms. This section discusses some of them, aiming for a common understanding before delving deeper into the modeling discussions. In addition, in the second subsection, the DCAT vocabulary entities are introduced, and some relevant aspects are detailed.

2.1. Relevant concepts on open data management on the Web

The terms "catalogue" and "repository" have different definitions as Web data management spaces. For this paper, a data catalogue is a curated collection of structured metadata for describing resources and pointing to data resources of interest [7]. It allows users to browse, search and organize these descriptors (metadata) and access the resource [8]. A catalogue usually disposes of metadata for resources stored elsewhere. In this case, metadata for the same resource may be stored in several catalogues [9]. A repository, on the other hand, organizes and manages research data (resources) [9]. In order to do this, besides operating as a catalogue, it supports different forms of storage and organization structures, facilitating the curation of resources, and providing mechanisms to access and manage these resources over time.

Digital repositories built with DSpace [10], a platform for creating open-access digital repositories, exemplify what this paper designates as "repository". DSpace establishes mechanisms for storing, curating, and preserving resources associated with metadata collections that enable their discovery and access. In this case, catalogue functions describe the resources stored in the repository. Examples of catalogues include those created with the FAIR Data Point (FAIR DP) solution. FAIR DP is one of the

[3] https://www.dublincore.org/
[4] https://www.w3.org/TR/owl2-primer/
[5] https://www.w3.org/RDF/
[6] https://www.w3.org/TR/prov-o/

main components of a FAIR Ecosystem. It is a software that works as an infrastructure to provide access to resources through standardized metadata, in strict compliance with the FAIR principles, for humans and machines [4]. Its current version is based on DCAT and includes descriptors for the FAIR DP itself, catalogues, datasets, and distributions [11]. Importantly, there are platforms, such as CKAN[7], that can be used as a repository, as a catalogue, or both at the same time.

Another distinction that deserves attention is "catalogued resource" and "catalogued resource descriptor". The term "catalogued resource" refers to a catalogued object of interest, physical or digital, stored and curated in a repository [12]. The term "catalogued resource descriptor" or "descriptor" refers to a digital object that describes a catalogued resource, contributing to its dissemination. The catalogued resource takes precedence over its descriptor, i.e., the descriptor arises because of it without becoming existentially dependent on it. Hence, a catalogue can allow access to the metadata even when the resource is no longer available.

In order to describe a resource, the descriptor employs properties that can be attributes such as "Title" or relations such as "Publisher", which establishes the agent responsible for the publication. These properties, converted into structured and standardized metadata, constitute the descriptors. Thus, descriptors provide information that facilitate the resource location, access, and understanding, fundamental aspects contributing to its reuse and interoperability. Therefore, considering the FAIR principles, we can assume that an adequately conceptualized set of descriptors, with structured properties associated with standardized vocabularies and ontologies, is essential to describe a FAIR catalogued resource.

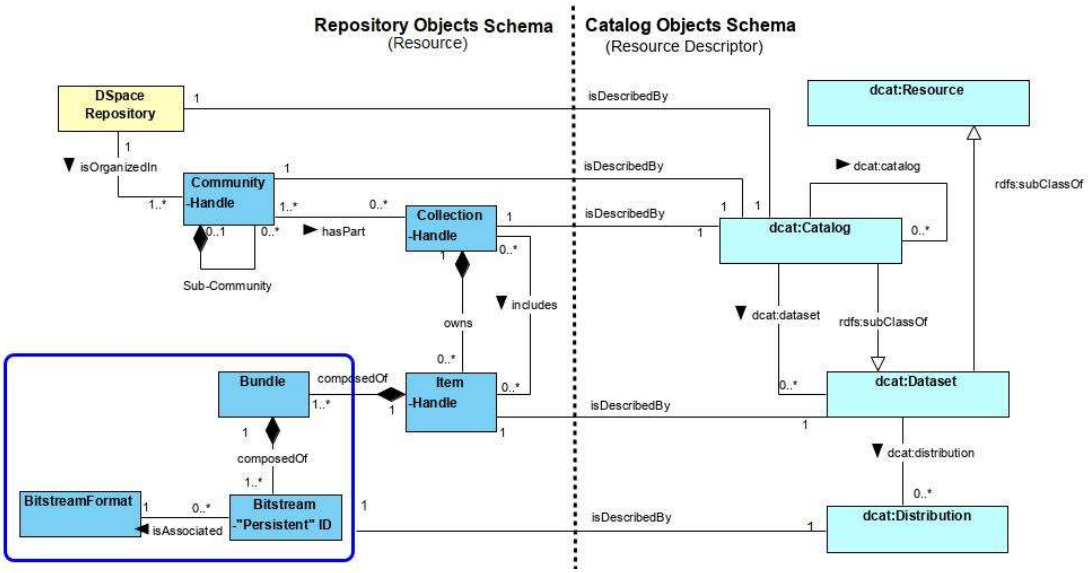


Figure 1: Schematic View for Catalogued Resources and their Descriptors (Resources Metadata), employing an adaptation of the DSpace data model and DCAT Entities.

Figure 1 presents a schematic view illustrating catalogued resources and their potential descriptors. The catalogued resources are represented based on the DSpace data model [10]. These resources are related to their respective descriptors, represented by DCAT entities [6], through the "isDescribedBy" relationship. On the left, we observe the DSpace model base elements, increased by the representation of the repository installation designated "DSpace Repository". In this model, a repository is organized in communities that can aggregate different sub-communities. Each "Community" can have one or more collections, and a "Collection" can be part of several communities. The central element of this model is the "Item". The "Item" belongs to a "Collection", which is responsible for it. Furthermore, an "Item" can be included in different collections. Each item consists of one or more "Bundle" constituted by "Bitstream". The "Bitstream", corresponding to a data file, refers to the actual data stored in their various formats, according to the "BitstreamFormat".

[7] https://ckan.org/

The right side of Figure 1 depicts a subset of DCAT entities, using the prefix "dcat", to describe DSpace catalogued resources. Among them, the *dcat:Resource* entity represents the descriptor of catalogued resources, i.e., a descriptor of resource published or curated by a single agent; the *dcat:Catalog* defines a descriptor for a catalogue, which describes the DSpace repository itself and the different digital organizational structures for items storage; the *dcat:Dataset* describes a dataset, which is defined as a "collection of data, published or curated by an agent, available for access or download in one or more formats or serializations" [6]; and the *dcat:Distribution* describes an accessible form of the dataset, such as a downloadable file.

Particularly, in DCAT, a catalogued resource of dataset type is described by the entities *dcat:Dataset* and *dcat:Distribution*. The *dcat:Dataset* entity describes the dataset as a conceptual entity, while *dcat:Distribution* represents its different serializations for access. Because of this definition, the model in Figure 1 employs *dcat:Dataset* to describe the “Item”, while *dcat:Distribution* describes the set of elements enclosed by the blue square, formed by "Bundle", "Bitstream", and "BitstreamFormat". The *dcat:Distribution* can provide information for these elements, according to the repository rules.

2.2. DCAT Vocabulary

To present the DCAT vocabulary, it is important to understand the term "vocabulary" associated with it. "On the Semantic Web, vocabularies define the concepts and relationships (also referred to as “terms”) used to describe and represent an area of concern" [13]. Vocabularies that classify terms, characterize relationships, and define some constraints for these terms, implemented with a major focus on computational performance, are called lightweight ontologies [14].

DCAT vocabulary is a lightweight ontology implemented in OWL2 to facilitate interoperability between catalogues. It describes catalogued resources, that is, resources published or curated in a repository, and their relationships. Initially created for interoperability among government data repositories [15], it defines concepts using entities and their properties in RDF, describing datasets and data services.

In order to be DCAT-compliant, a catalogue must meet the following requirements [6]: (i) organize access to data into datasets, distributions, and data services; (ii) present a descriptor in RDF for the catalogue itself, the catalogued resources and their available distributions, using DCAT entities and properties; (iii) present consistency with the semantics declared in its specifications.

In addition to the concepts introduced in 2.1, DCAT includes the concepts that follow. The *dcat:DataService* is a descriptor for a data service, which is defined as a "collection of operations accessible to provide access to one or more datasets or data processing functions" [6]. It allows the description of service catalogues, i.e., a catalogue of data service descriptors, providing access to the data assets. The *dcat:CatalogRecord* entity registers the resource descriptor entry in the catalogue, capturing provenance information explicitly, and its use is optional. Finally, there are the entities *dcat:Relationship* and *dcat:Role*. Those entities were defined in version 2.0, allowing users to establish relationships between dataset descriptors and catalogued resource descriptors when these are known but not standardized between DC terms or PROV-O properties.

Given the purpose associated with the concepts *dcat:Relationship* and *dcat:Role*, they can be considered what in multi-level modeling is addressed as a type of types. They allow the creation of types employed in the domain ontologies models, making explicit meaningful relationships between their descriptors. These types are adequately represented by models that adopt more than two levels of stratification, i.e., multi-level models.

3. Multi-level conceptual modeling and Multi-Level Theory (MLT)

Conceptual modeling represents physical and social world aspects, aiming at their understanding and communication among humans [16]. Conceptual models are usually developed by techniques using a strict stratification of entities into two classification levels: a level of types (or classes) and a level of instances (objects). However, there are specific domains where this kind of conventional stratification does not work, and the representation of types of types (or categories of categories) is necessary to model their conceptualization; for these domains, multi-level conceptual modeling is adopted [17].

Different approaches exist to treat Multi-level Modeling (MLM). In this paper we adopt the MLT developed from the Unified Foundational Ontology (UFO) and employed for conceptual modeling of multi-level types [18].

According to Carvalho and Almeida [19], the MLT is a theory that can be considered a foundation ontology for the conceptual modeling of multi-level types. It formally characterizes the nature of the classification levels and defines the relationships between the elements of those levels [19]. To allow the treatment of types at different levels, it admits entities that simultaneously represent a class and an object. In MLM approaches, these entities are called "Clabject" because they present two facets, one acting as an instance to some entity on the level above and another as a class for entities of a level below [20].

Similar to UFO, MLT distinguishes between universals (types) and individuals. "Types are predicative entities that can be applied to a multitude of entities, including types themselves, while individuals are particular entities" [19]. To address the distinction between types, MLT employs the notion of "Type Order" [19] [20]. These orders, equivalent to levels in other approaches, are organized in a stratified manner and represented by logical constants called "Base Types". Relationships are established between the base types and extended to the domain types. MLT defines the primitive "instance of" relation to associate entities to their respective type [19]. The dependency relation "instance of" is employed, as in UML, to indicate that a base type is an instance of another.

Based on this structuring, types whose instances are individuals are called first-order types (1stOT). Types whose instances are first-order types are called second-order types (2ndOT) and so on. Special attention is given to entities that are individuals. Under the theory, the entity must be an instance of the <Individual> base type and have no other instances to be an individual. It presents only the object facet, as adopted by the UML.

The top of Figure 2 depicts the organization of the base types of the MLT, represented by orange rectangles. The representation adopts UML notation. Thus, associations are used to represent relationships between instances and related types. Dashed arrows are used to define dependency relationships between types. The labels of the relations refer to the names of applied predicates.

In this theory, each type in the represented domain is an instance of precisely one of the higher-order base types (<1stOT>, <2ndOT>, and <3rdOT>) and, at the same time, proper specializes a base type at the next level down. Roughly speaking, the proper specialization relation, defined by MLT, guarantees that in type specializations, if a type t1 specializes a type t2, then t1 and t2 are different types, i.e., the instances of the specialized type are a proper subset of the general type. The pattern adopted by MLT is depicted in Figure 2. In this case, "Person" is an instance of <1stOT> and proper specializes <Individual>. The instances of "PersonTypeByGender", i.e., "Woman" and "Man", are specializations of "Person". As it applies to "Person", an instance of <1stOT>, "PersonTypeByGender" is an instance of <2ndOT> and proper specializes <1stOT>.

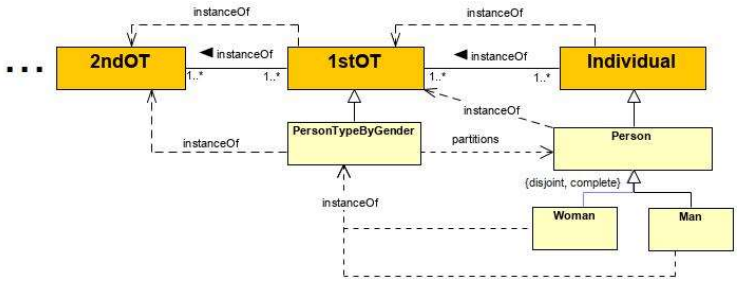


Figure 2: Example of Instantiation and Specialization between domain elements (adapted from [21]).

MLT defines cross-level structural relations between types of adjacent orders. These relations provide support for the analysis of the powertype notions in the literature. It is important to note that for MLT, the instances of powertype are clabjects. The relation named <*isPowertypeOf*> adopts the same concept proposed by Cardelli [22], i.e., powertype establishes sets associated with the base type. The Categorization relation is based on Odell's definition [23], which states that a base type can have more than one powertype. In addition to these notions from the literature, MLT establishes other valuable relations for capturing constraints in multi-level models.

This paper uses the powertype and partition relations for interaction between adjacent order types. According to MLT, type t1 <*isPowertypeOf*> a base type t2 if all instances of t1 specialize t2 and all possible specializations of t2 are instances of t1. In this case, instances of t1 are applicable to instances of t2, but t1 does not define classification criteria. Thus, all specializations of t2, including t2 itself, are instances of t1. A type t1 <*categorizes*> a type t2 if all instances of t1 are proper specializations of t2. To specialize t2, in this case, the instances of t1 apply a classification criterion. Thus, t2 is not an instance of t1. Other specializations of t2 may exist from other classification criteria distinct from the one established by the type t1. A type t1 <*partitions*> t2 if t1 categorizes t2, and each instance of t2 is an instance of exactly one instance of t1. In Figure 2 <PersonTypeByGender> partitions <Person> into <Women> and <Man>. According to the rule, each instance of <Person> is either a <Women> or a <Man>, but not both simultaneously, which establishes that this specialization is disjoint and complete. For more details on MLT other readings are recommended [18] [19].

In addition to organizing types into orders and defining intra-level and cross-level relations, MLT addresses the properties of higher-order types [24]. For MLT, not all properties of higher-order types behave identically. Thus, it employs *regularity properties* to handle what MLM calls deep instantiation. In deep instantiation, the properties of a higher-order type affect entities at lower levels [22]. According to Almeida et al. [24], the values assigned to a *regularity property* of a higher-order type (such as second- and third-order types) affect the intents of instances of these types. Through this property type, invariant aspects are established for instances of a type. For example, you can determine a value or constrain the type of assignment for a property by determining its type(s) or a set of allowed types [22]. Another type of property is the *direct property* [24]. These properties are specific to the higher-order type and are not inherited by its instances. For more details on properties in MLT, refer to [21] [24].

The UFO concepts can be associated with the MLT elements. UFO is a foundational ontology built upon theories from the areas of formal ontology, philosophical logic, philosophy of language, linguistics, and cognitive psychology [25][26]. It establishes a taxonomy to categorize concepts of a domain based on the principles of individuality, rigidity, and dependency, reducing ambiguity, and facilitating understanding.

UFO-MLT is obtained when UFO concepts are associated with MLT elements, respecting axioms, structural relations, and patterns [18][27]. It is an approach to develop domain conceptual models that represent types and types of types, adhering to the rules of a foundational ontology [27]. Thus, the UFO taxonomy provides patterns for types of types in the model. These patterns guide the modeler in defining higher-order types and their relationships.

4. Ontological foundations for descriptors of catalogued resources in repositories

In the context of Information Science, "represent" consists of describing a given information resource according to its features. This description should be sufficient to allow its identification among other resources and its comprehension without requiring access [28]. To provide such a description, DCAT proposes an ontology of descriptors for catalogued resources. Its entities are constituted by metadata from standardized vocabularies that establish the minimum characteristics to describe the resources. This section is responsible for an ontological analysis of these entities, i.e., the associated types/concepts. This analysis emphasizes their ontological nature and establishes the categories of types. Furthermore, as previously mentioned, it provides tools for understanding the parts behavior, employing formal theories such as the theory of essence and identity, parts (mereology), unity and plurality, and dependency [29].

Applying the multi-level foundational ontology to DCAT aims to: (i) allow an analysis of the conceptual nature of terms, identifying their different categories [30]; and (ii) obtain an increase in expressiveness with the treatment of the multiple classification levels, reducing complexity and facilitating understanding [31]. The analysis of the DCAT entities is detailed below. Initially, the study was carried out using the UFO categories. After that, we adopted UFO-MLT for the appropriate treatment of the entities identified as higher-order types. The models presented in this paper were

developed using the Visual Paradigm tool, version 16.3, with the OntoUML plugin[8]. This plugin supports OntoUML modeling by installing the necessary stereotypes and tagged values. In addition, it adds features like a custom user interface, smart diagram coloring, partial and complete model checking, and model transformation [25]. It is worth mentioning that OntoUML is a Conceptual Modeling language whose primitives reflect the ontological micro-theories that comprise the UFO [1][25].

4.1. An ontological analysis of DCAT

Analyzing the DCAT entities with UFO allowed categorizing and differentiating them according to their ontological nature. Figure 3 presents a conceptual model of DCAT with the UFO categorization. Due to space limitations, it covers the DCAT entities but not their properties.

In DCAT, *dcat:Resource* represents the catalogued resource descriptor. It gathers the common properties of different resource descriptors. Because it defines the properties essential to some instances, mainly for dataset descriptors, but incidental to others, it is categorized as a <*Mixin*>. However, when applying DCAT in a specific catalogue context, it is necessary to configure the concepts for that domain. Thus, we consider that *dcat:Resource* should be configured with the catalogue descriptors common (mandatory) properties. In this case, it is categorized as <*Category*>.

A dataset, either a digital or physical object, is described using *dcat:Dataset* and *dcat:Distribution*. The *dcat:Dataset* describes the general characteristics of the object, i.e., a more conceptual view, while *dcat:Distribution* describes the material part, i.e., its different serializations. A *dcat:Dataset* is an independent entity that provides the principle of identity, individuation, and persistence for all its instances, in this case, the descriptors of the conceptual features of a dataset. Considering these characteristics, a *dcat:Dataset* is categorized as a <*Kind*>.

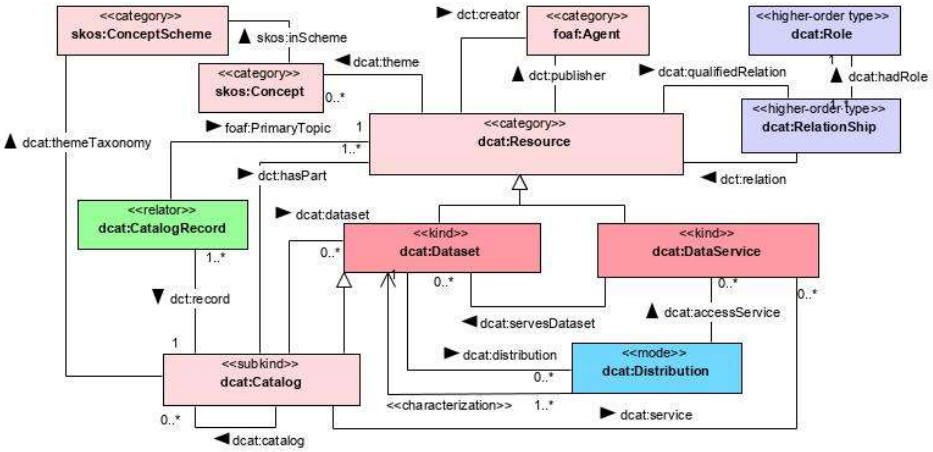


Figure 3: Representation of a DCAT excerpt applying UFO.

A distribution refers to the serialization of the dataset for access or transfer. The *dcat:Distribution* describes these different serializations. In a catalogue, the *dcat:Distribution* instances are existentially dependent on exactly one *dcat:Dataset* instance. Due to this dependency, if a *dcat:Dataset* instance is removed, all its *dcat:Distributions* instances are also removed. Furthermore, the absence of the *dcat:Dataset* instance to describe it makes it impossible to access and even understand the *dcat:Distribution* instances. Without a *dcat:Distribution* instance, essential dataset information is not available. Thus, it is part of the characterization of the dataset. Regarding these characteristics, it is categorized in UFO as a <*Mode*>. Therefore, the *dcat:Distribution* is existentially dependent on a *dcat:Dataset*, contemplating its own <*Moments*>, i.e., the properties that characterize it. Furthermore, as an ontological nature in UFO [25], <*Mode*> provides an identification principle to distinguish instances of distribution descriptors representing different serialization formats or even versions of the catalogued dataset.

[8] https://github.com/OntoUML/ontouml-vp-plugin

According to DCAT, *dcat:Catalog* is a specialization of *dcat:Dataset*. Each instance of *dcat:Catalog* is a set of metadata record that describes catalogued resources, i.e., its instances represent collections of descriptors of datasets and data services and even other catalogues. The latter, here, is understood as organizational structures for datasets in a repository. As a specialization responsible for organizing the repository resource descriptors, *dcat:Catalog* inherits characteristics from *dcat:Dataset* and presents the rigidity property, i.e., it is essential for all its instances. It is then categorized as a <*Subkind*>.

The *dcat:DataService* describes the data services of a repository. Similar to the analysis performed on *dcat:Dataset*, a *dcat:DataService* is an independent entity that describes related operations such as discovery, access, or processing functions on data or related resources. It is categorized as a <*Kind*> that provides an identity principle for its instances.

The *dcat:CatalogRecord* describes registering a specific dataset or data service descriptor in a catalogue. This entity is not mandatory and, when used, works as a log for resource descriptor publications. Analysis of its characteristics allows us to identify that this entity is existentially dependent on the catalogued resource descriptor (*dcat:Resource*) and the catalogue descriptor (*dcat:Catalog*) where the resource was inserted. Based on this analysis, it is categorized as a <*Relator*>, representing the mereological sum of the two entities [32]. Moreover, it originates in the insertion event of a descriptor in the catalogue.

Unlike the other entities that describe the resources catalogued in a repository, the *dcat:Relationship* allows describing qualified relationships between dataset descriptors or from a dataset descriptor to another resource descriptor. It is used when the relationship is known but is not standardized among DC terms or PROV-O properties. The entity employs the *dcat:hadRole* attribute to define the role of the relationship. In *dcat:Resource*, the *dcat:qualifiedRelation* property is responsible for the association to *dcat:Relationship*, identifying the target descriptor of the relation. On the other hand, *dct:relation* property in *dcat:Relationship* associates the source descriptor. This entity defines new types of relationship descriptors and is therefore categorized as a <*Higher-Order Type*>.

The *dcat:Role* is the entity that establishes the role of one resource descriptor concerning another when it is associated with *dcat:Relationship*. In order to qualify relationships, DCAT recommends the adoption of controlled vocabularies such as Geographic Information Metadata[9] (ISO-19115-1), Link Relations by Internet Assigned Numbers Authority[10] (IANA-RELATIONS), DataCite Metadata Schema[11], and others. Since it establishes the role of an additional relationship to be described and handled in a specific catalog, similarly to *dcat:Relationship*, we categorize *dcat:Role* as a <*Higher-Order Type*>. It is important to note that, associated with *prov:Attribution*, *dcat:Role* is also used to describe the function of an agent concerning a resource descriptor, establishing attributions not foreseen in DCAT. This paper focuses on the aspects related to creating new relationship descriptors.

4.2. Categorizing higher-order types

The ontological analysis of DCAT identified the existence of higher-order types. These types provide flexibility to the model, allowing new relationships between types of resource descriptors to be described.

As aforementioned, the entity *dcat:Relationship* holds relationships that cannot be described according to the vocabularies and ontologies standardized by DCAT. When configuring a catalog for a specific domain, the relationship descriptors (instances) must be represented at the same level that details the different descriptors and their properties. Therefore, the entities *dcat:Relationship* and *dcat:Role* should be relocated according to their order as specializations of <1stOT> and instances of <2ndOT>. The appropriate allocation of entities promotes a better understanding of the model.

In order to represent the concept associated with *dcat:Relationship* and establish the relationships between descriptor types, the catalogued resource descriptor type (*:CataloguedResourceDescriptorType*) and the dataset descriptor type (*:DatasetDescriptorType*) were defined. According to MLT, they are also proper specializations of <1stOT> and instances of <2ndOT>.

[9] https://www.iso.org/standard/53798.html
[10] https://www.iana.org/assignments/link-relations/link-relations.xhtml
[11] https://schema.datacite.org/

The former is powertype of the entity *dcat:Resource*. As a result, all the specializations of this entity become instances of the *:CataloguedResourceDescriptorType*, including *dcat:Resource*. It is classified as <*Category*>. The catalogued data descriptor type *(:CataloguedDataDescriptorType*) partitions *dcat:Resource* and has *dcat:Dataset* and *dcat:DataService* as instances. By partitioning an entity categorized as <*Category*> and having <*Kind*> entities as instances, it is classified as <*Kind*>. According to MLT [19], if a type t1 is powertype of a type t2 and a type t3 also categorizes the type t2, then t3 proper specializes t1. Based on this rule, *:CatalogueDataDescriptorType* proper specializes *:CatalogueResourceDescriptorType*.

Specialization between types indicates that all instances of the type will also be instances of the supertype [21]. The *:DatasetDescriptorType* proper specializes *:CataloguedDataDescriptorType* and it is powertype of the *dcat:Dataset* type, with *dcat:Catalog* as its instance. For establishing a subset of catalogued data descriptor types, it is classified as <*Subkind*>.

The *dcat:Relationship* is existentially dependent on the catalogued resource descriptor type (*:CataloguedResourceDescriptorType*) and the catalogued dataset descriptor type (*:DatasetDescriptorType)*. This dependency that binds entities together by mediation relationships characterizes it as a <*Relator*>, and it is twofold. The first dependency refers to the source descriptor type of the relationship. The second refers to the descriptor type related to the first, the target. Importantly, not all resource descriptor types have descriptors for additional relationships. To improve semantics, we add the source descriptor type (*:SourceDescriptorType*) and the target dataset descriptor type (*:TargetDatasetDescriptorType*) as specializations of the catalogued resource descriptor type and the dataset descriptor type, respectively. Furthermore, they identify accidental types that have relationship descriptors. Therefore, *:SourceDescriptorType* is categorized as <*Rolemixin*>, enabling any descriptor type as a relationship source, and *:TargetDatasetDescriptorType* is categorized as <*Role*>. Based on the <*isPowertypeOf*> relation, it establishes as the relationship target the types it instantiates, i.e., *dcat:Dataset* and any of its specializations. Hence, the *dcat:Relationship* maintains DCAT compliance, describing the relationship between dataset descriptors or between a resource descriptor and a dataset descriptor.

The *dcat:Role* establishes the role of one resource descriptor in relation to another when it is associated with *dcat:Relationship*. Its analysis categorizes it as <*Quality*>. As a <*Quality*>, it can be projected into value spaces, establishing domains of standardized values to be adopted. To represent the distinct functions of *dcat:Role*, it was specialized into *:RelationRole* and *:AttributionRole*, each associated with its own value space. These specializations are categorized as <*Subkind*>. In this context, the *:RelationRole* specifies a characterization relationship with *dcat:Relationship*. Thus, instances of *dcat:Relationship* will bear the standardized term associated with it.

Importantly, when expressing in the model that a quality characterizes a type, the instance of the characterized type will be the bearer of this quality instance [32]. Thus, a *dcat:Relationship* instance will have a *dcat:Role* instance associated with it, using the *dcat:hasRole* property. This instance will express the role of a specific relationship type between catalogued resources of a particular domain. When we establish types to be assigned to this property, we define what is called a regularity property in MLT [21]. That is, the property establishes the values to be accepted for its instances.

Figure 4 shows an extract of the MLT-UFO-based DCAT model, employing MLT base types to handle type orders, and UFO categorization. The “instanceOf” edges between domain types and the <1stOT> and <2ndOT> base types have been omitted for legibility reasons. Specializing the <Individual> base type we have *dcat:Resource*, *dcat:Dataset*, *dcat:Distribution*, *dcat:Dataservice*, *dcat:Catalog*, and *dcat:CatalogRecord*. Specializing the <1stOT> base type we have *:CatalogueResourceDescriptorType*, *:CataloguedDataDescriptorType, :DatasetDescriptorType*, *dcat:Role* and their specializations, and *dcat:Relationship*. Entities with highlighted borders were included in the model to enhance semantics.

We observe two important contributions by employing a model that handles types of types. The first is for the catalogue manager, who will have a model for the governance of the descriptor types, as well as their relationships. Through this model, it is possible to define conformity restrictions that users must respect when publishing descriptors in the catalogue. The second refers to the availability of the RDF model associated with the data that provides the agents (humans or machines) the knowledge of the catalogue structure and the standardized terms adopted, contributing to interoperability mechanisms.

Therefore, the second-order domain types establish, together with DCAT concepts, a language structure for understanding the catalogue organization.

As an example, we can cite the *:DatasetDescriptorType*. This type registers the different types of dataset descriptors associated with the catalogue. Thus, it may be considered relevant for a specific domain besides *dcat:Catalog,* a descriptor type that differentiates datasets that handle databases from those that describe data files. The defined types are instances that specialize the *dcat:Dataset* type.

The value spaces for *dcat:Role* also deserve attention. By defining these spaces, the manager can establish vocabularies and ontologies for its catalog, providing compliance rules for new publications. In addition, these standards are available for agent access, contributing to understanding the catalogue structure.

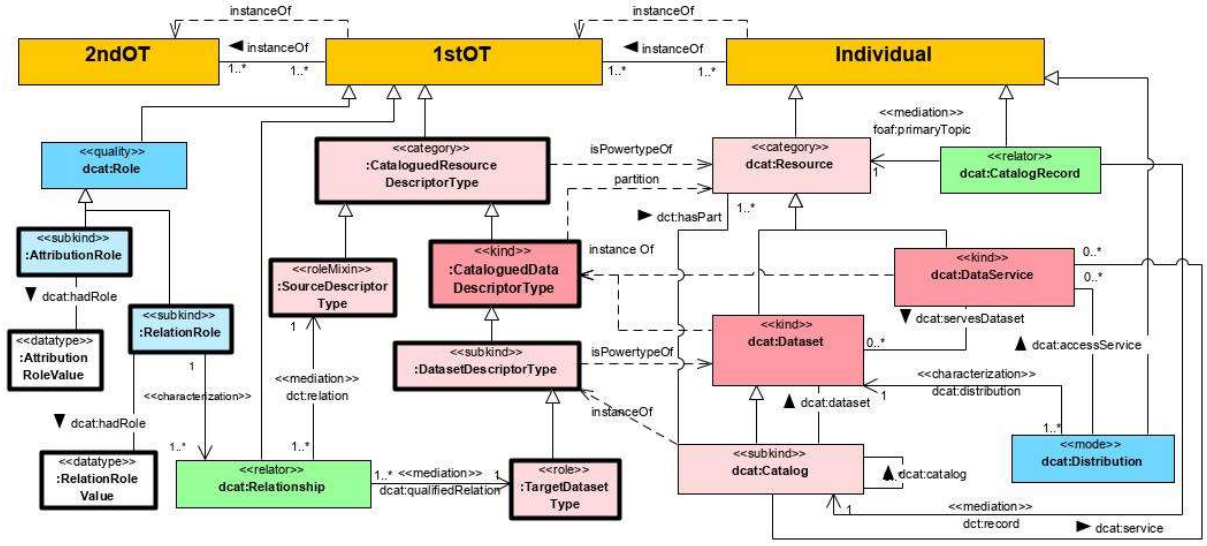


Figure 4: Extract from the Multi-level Model for DCAT.

It should be noted that the associated modeling profile provides additional metaproperties to the DCAT model. Using languages like OntoUML, axioms are generated from these metaproperties and incorporated into the specification results, restricting the interpretation of the terms to the desired reality.

5. Employing MLT-DCAT for a repository

This section provides an overview of applying the MLT-DCAT model to define resource descriptors for a catalogue representing a DSpace repository. Due to space limitations, we will exemplify the treatment for an existing relationship between resources in the repository. In the model in Figure 5, some types and relations have been omitted to emphasize the new domain-specific elements added to the model. These elements have highlighted borders.

Figure 5 presents a relationship descriptor instantiated from the *dcat:Relationship* to fit a specific domain, in this example, to describe the item (dataset) association to its owner catalogue. This association is illustrated in the DSpace model in Figure 1. Thus, the model addresses the *:Relation_isOwner* between the dataset descriptor and the catalogue descriptor that owns it, in compliance with a catalogue that represents catalogued resources in a DSpace repository. This relationship is the bearer of the *IANA_relator:own* term, conforming to the value spaces defined for *:RelationRole*, i.e., terms from the IANA relators and ISO-19115-1 vocabularies.

At the bottom of Figure 5, we find *:Relation_isOwner* as the instance of *dcat_Relationship*. It has the *IANA_relator:own* term associated with *dcat:hadRole* property. In addition, it defines instances of *:SourceDescriptorType* and *:TargetDatasetType* as valid types for range of the *dct:relation* and *dcat:qualifiedRelation* properties, respectively. Finally, for the management of relationship types, *dct:issued* and *dct:modified* have been added as direct properties of the type.

This type is instantiated in the model as :*Relation_isOwner*, a relator that associates a *dcat:Dataset* with its owner collection (*dcat:Catalog*). The relationship is represented between the COVID-19 data

collection descriptor (*ex:COVID_19_Collection_Descriptor*) and COVID-19 item descriptor from Hospital1 (*ex:COVID_19_Hospital1_Item_Descriptor*).

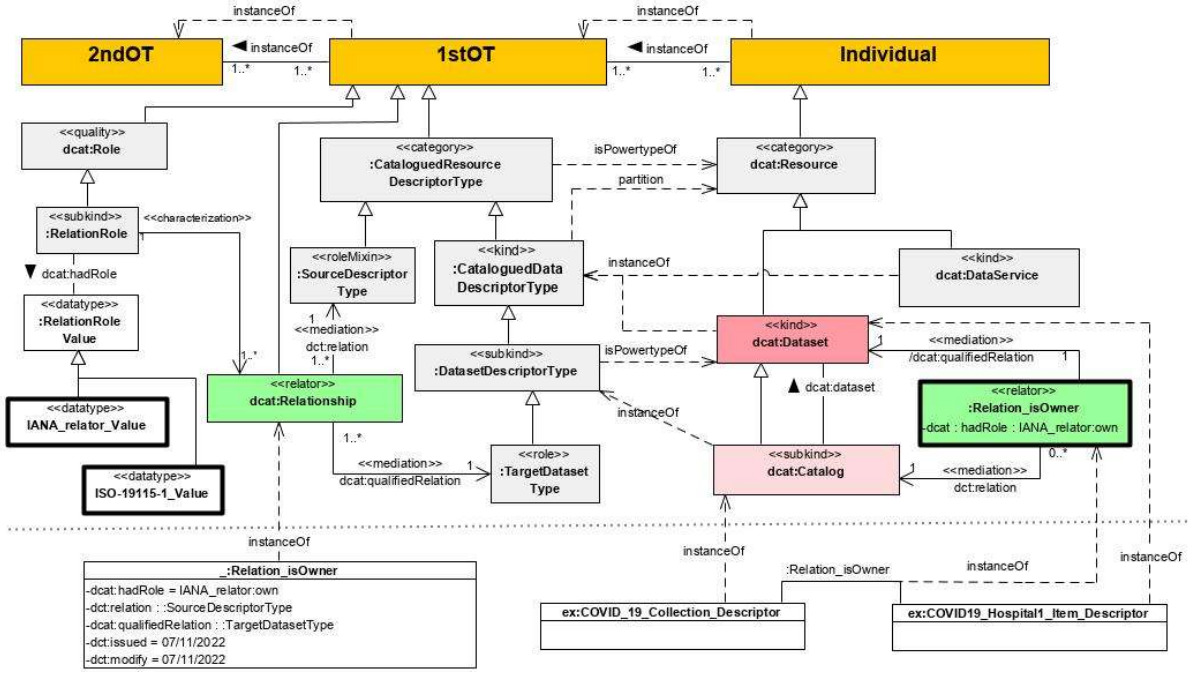


Figure 5: Example of the MLT-UFO-DCAT model representing the Owner relationship for a DSpace repository.

To illustrate the benefits of modeling, based on the contributions presented in the previous section, we provide some competency questions (CQ) answered with the model in Figure5.

- CQ1. Which types of catalogued resource descriptors are handled in the catalogue? It considers the instances of *:CataloguedResourceDescriptorType*.
- CQ2. Which types of descriptors are specific to datasets? It considers the instances of *:DatasetDescriptorType*.
- CQ3. Which descriptor types are related to dataset descriptor types? It refers to the instances of *:SourceDescriptorType*.
- CQ4. Which are the additional relationships employed in the catalogue? They are the instances of *dcat:Relationship*.
- CQ5. Which types of descriptors does each additional relationship associate? They are obtained with the analysis of the pairs instantiated *:SourceDescriptorType* and *:TargetDatasetType* for each instantiated *dcat:Relationship*.
- CQ6. What are the patterns established for the roles played by the additional relationships? It refers to the value spaces established for *:RelationRoleValue*.

The presented CQs assist administrators in managing the catalogue. For agents (human and machine), they allow checking descriptor types and acceptable standards before submission for publication. It could also be applied when publishing a descriptor.

As previously mentioned, the model, in RDF, would make available the structure adopted in the catalogue, as well as the standardized terms (semantic artifacts). In the example of Figure 5, we highlight the standardization for the creation of new relationships, with the definition of value spaces associated with two standardized vocabularies (IANA Relations and ISO-19115-1).

For software agents, the second-order domain types establish, together with the DCAT concepts, a language structure for understanding the catalogue organization, contributing to interoperability mechanisms. These types may also be employed by a software for catalogue management (or a data platform software), allowing the inclusion of new types in a dynamic way. An example would be the addition, by the manager, of two new types of dataset descriptor. The first to handle databases *(:DatabaseDataset*) and the second one, data files (*:DataFileDataset*). As instances of the

:DatasetDescriptorType type, they would specialize *dcat:Dataset*, keeping the rules established by its powertype.

Summarizing this section, modeling with categorized types establishes conformance constraints that domain users must respect. In the ontology-based model of Figure 5, it is possible to explicitly identify the description of a domain specific relationship and the value space employed to compose these relationship descriptors. Furthermore, the adoption of standardized value spaces for relationship roles (i) establishes restrictions to be respected when publishing descriptors, and (ii) clarifies the standards adopted by the catalogue. These aspects contribute to interoperability between repositories by providing means to discover, access, and understand the stored data.

6. Conclusion and future works

As discussed in this paper, the proliferation and use of diverse platforms for digital repositories with differentiated infrastructure for data storage demand special attention to interoperability. To this end, these platforms are highly dependent on metadata to support the discovery, access, and interoperability of their catalogued resources. Analyzing these metadata allows us to identify that they constitute a domain of descriptors that need a consistent conceptualization, explicitly and formally represented, aiming at sharing the concepts involved.

This paper presented an ontological review of DCAT and proposed an extension of its model by applying foundational ontologies. These ontologies clarify the nature of concepts and improve their semantics. As a result, they make explicit the information structures employed for representation.

According to Guizzardi [1], the quality of an information system is associated with how truthful its information structures are concerning the reality aspects it needs to represent. Thus, by applying MLT combined with UFO to DCAT, we add a formal ontological structure that defines aspects independent of their particular nature to its types. These aspects help modelers better understand domain concepts, providing ambiguity handling and defining constraints to better align them to reality.

With its treatment of higher-level types, MLT has provided a new perspective on DCAT concepts, adding types that assist with their understanding. Moreover, the reorganization and the new types can be employed by software for catalogue management and access to their descriptors. According to Kühne [33], if we interpret an ontological metamodel as a domain specific language definition, we can consider it as a linguistic metamodel. In this context, the higher-order types of the proposed model could be treated as a linguistic metamodel that makes explicit the specializations presented in DCAT and possible extensions to a specific domain.

Thus, ontologies based on this canonical model can extend the interoperability mechanisms of digital platforms to repositories, maintaining compliance with protocols such as OAI-PMH and making RDF information available to search engines and retrieval agents. In this way, the improvement of model structures is reflected in the quality of the repository software and interoperability mechanisms that use it, providing: (i) a representation that handles ambiguities; and (ii) management of constraints that promote the proper functioning of solutions.

As future work, we are studying the representation of the attributes and relations associated with DCAT entities with MLT. In parallel, we are considering a treatment to distinguish different forms of dataset representation. This explicit distinction will allow the catalogue to manage datasets with equivalent or non-equivalent distributions, avoiding semantic overload. Thus, the model will contribute to the representation of datasets in institutional repositories working with different communities and forms of data treatment. Furthermore, from the generated model, we intend to implement an operational ontology with gUFO ontology [34], a lightweight implementation of UFO, and to establish a FAIR DP with native conformance.

7. Acknowledgements

This work has been partially supported by CAPES student grants (Process numbers 223038.014313/2020-19 and 88887.613048/2021-00).

8. References

[1] G. Guizzardi. Ontology, ontologies and the "I" of FAIR. Data Intelligence, v. 2, n. 1-2, p. 181-191, 2020. DOI: 10.1162/dint_a_00040.

[2] M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, et al. Comment: The FAIR Guiding Principles for scientific data management and stewardship, Scientific Data, v. 3, n. March, 2016. DOI: 10.1038/sdata.2016.18.

[3] B. Mons. Invest 5% of research funds in ensuring data are reusable. Nature, v. 578, n. 7796, p. 491-491, 2020. DOI: https://doi.org/10.1038/d41586-020-00505-7.

[4] L. O. B. Santos, et al. FAIR data points supporting big data interoperability. Enterprise Interoperability in the Digitized and Networked Factory of the Future. ISTE, London, p. 270-279, 2016.

[5] C. C. Austin, S. Brown, N. Fong, C. Humphrey, A. Leahey, P. Webster. (2016). Research data repositories: review of current features, gap analysis, and recommendations for minimum requirements. *IASSIST Quarterly*, *39*(4), 24-24.

[6] R. Albertoni, D. Browning, S. Cox, A. G. Beltran, A. Perego, P. Winstanley. Data Catalog Vocabulary (DCAT) - Version 2. W3C Recommendation. 4 February 2020. W3C Recommendation. URL: https://www.w3.org/TR/vocab-dcat-2/

[7] H. Sheridan et al. Data Curation through Catalogs: A Repository-Independent Model for Data Discovery. Journal of eScience Librarianship, v. 10, n. 3, p. 4, 2021.

[8] D. Connolly. Catalogs: Resource Description and Discovery. W3C. 24 February 2014. URL: https://www.w3.org/Search/catalogs.html

[9] UMU. Storage, catalogue, repository and archive - what's the difference? Umeå University, [s.d.]. Disponível em: https://www.umu.se/en/library/research-data/specialised-topics/storage-catalogue-repository-and-archive/

[10] T. Donohue. DSpace 7.x Documentation, fev 03, 2022. URL: https://wiki.lyrasis.org/display/DSDOC7x/DSpace+7.x+Documentation.

[11] L. O. Bonino; K. Burger; R. Kaliyaperumal. FAIR DATA POINT – Version 1.1. Jun 29, 2022. URL: https://specs.fairdatapoint.org/.

[12] C. Logoze. The open archives initiative protocol for metadata harvesting. http://www. openarchives. org/OAI/2.0/openarchivesprotocol. htm, 2002.

[13] World Wide Web Consortium (W3C). Semantic Web - Vocabularies. 2015. https://www.w3.org/standards/semanticweb/ontology.

[14] G. Guizzardi. On ontology, ontologies, conceptualizations, modeling languages. In: and (Meta) Models, Frontiers in Artificial Intelligence and Applications, Databases and Information Systems IV, IOS. 2007.

[15] F. Maali, R. Cyganiak, V. Peristeras. Enabling interoperability of government data catalogues. In: International Conference on Electronic Government. Springer, Berlin, Heidelberg, 2010. p. 339-350.

[16] J. Mylopoulos. Conceptual modelling and Telos. Conceptual Modeling, Databases, and Case An integrated view of information systems development., p. 49–68, 1992.

[17] F. Brasileiro, J. P. A. Almeida, V. A. Carvalho., et al. "Applying a Multi-Level Modeling Theory to Assess Taxonomic Hierarchies in Wikidata", p. 975–980, 2016. DOI: 10.1145/2872518.2891117.

[18] V. A. Carvalho. Foundations for Ontology-based Multi-level Conceptual Modeling. Doctoral dissertation - Universidade Federal do Espírito Santo, Brazil, 2016. 36.

[19] V. A. Carvalho; J. P. A. Almeida. Toward a well-founded theory for multi-level conceptual modeling. Software & Systems Modeling, v. 17, n. 1, p. 205-231, 2018. DOI: 10.1007/s10270-016-0538-9.

[20] A. Rossini, J. de Lara, E. Guerra, et al. A comparison of two-level and multi-level modelling for cloud-based applications. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), v. 9153, p. 18–32, 2015. DOI: 10.1007/978-3-319-21151-0_2.

[21] C. M. Fonseca, et al. Multi-level conceptual modeling: Theory, language and application. Data & Knowledge Engineering, v. 134, p. 101894, 2021.
[22] L. Cardelli. Structural subtyping and the notion of power type. In: Proceedings of the 15th ACM SIGPLAN-SIGACT symposium on Principles of programming languages. 1988. p. 70-79.
[23] J. Odell: Power types. In: Journal of Object-Oriented Programing, 7(2), pp. 8-12. 1994.
[24] J. P. A. Almeida, V. A. Carvalho, C. M. Fonseca, G. Guizzardi. (2021). A Note on Properties in Multi-Level Modeling. In 2021 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C) (pp. 497-501). IEEE.
[25] G. Guizzardi, et al. Types and taxonomic structures in conceptual modeling: A novel ontological theory and engineering support. Data & Knowledge Engineering, v. 134, p. 101891, 2021.
[26] G. Guizzardi, et al. UFO: Unified Foundational Ontology. Applied Ontology, n. Preprint, p. 1-44, 2021.
[27] V. A. Carvalho, J. P. A. Almeida, C. M. Fonseca, G. Guizzardi. Multi-level ontology-based conceptual modeling. Data & Knowledge Engineering, v. 109, p. 3-24, 2017.
[28] K. Tomoyose. O Data Catalog Vocabulary (DCAT) para a publicação de dados de pesquisa nos princípios Linked Data. Master's thesis, Universidade Federal de São Carlos, São Carlos, SP, Brazil. 2021. https://repositorio.ufscar.br/handle/ufscar/14116.
[29] J. L. R. Moreira et al. Towards findable, accessible, interoperable and reusable (FAIR) data repositories: improving a data repository to behave as a FAIR data point. Liinc em Revista, v. 15, n. 2, 2019.
[30] V. S. Silva, M. L. M. Campos, et al. An Approach for the Alignment of Biomedical Ontologies based on Foundational Ontologies. Journal of Information and Data Management. October, v. 2, n. 309307, p. 557–572, 2011.
[31] M. Igamberdiev, G. Grossmann, M. Selway, et al. An integrated multi-level modeling approach for industrial-scale data interoperability, Software and Systems Modeling, v. 17, n. 1, p. 269–294, 2018. DOI: 10.1007/s10270-016-0520-6.
[32] C. M. Fonseca, et al. Relations in ontology-driven conceptual modeling. In: International Conference on Conceptual Modeling. Springer, Cham, 2019. p. 28-42.
[33] T. Kühne. Matters of (meta-) modeling. Software & Systems Modeling, v. 5, n. 4, p. 369-385, 2006. DOI:10.1007/s10270-006-0017-9
[34] J. P. A. Almeida, G. Guizzardi, R. A. Falbo, T. P. Sales, gUFO: a lightweight implementation of the Unified Foundational Ontology (UFO), 2019.